TENeT Messaging Specification

Ver. 1.0

2005/03
T-Engine Forum
CONTENTS

1  Introduction-----------------------------------------------------------------------------------------------------------------1
  1.1  Outline of this specification ................................................................................................................................. 1
  1.2  Positioning of these specifications .......................................................................................................................... 1
  1.3  Stipulated range ........................................................................................................................................................ 1
  1.4  Reference stipulations ............................................................................................................................................. 2
  1.5  Notations ................................................................................................................................................................. 2
  1.6  Cast of characters ....................................................................................................................................................... 2
  1.7  Message types ............................................................................................................................................................ 2

2  Data Definitions------------------------------------------------------------------------------------------------------------5
  2.1  Public key certificate .................................................................................................................................................. 5
  2.2  eTRON ID ..................................................................................................................................................................... 6

3  IC Card Functions---------------------------------------------------------------------------------------------------------7
  3.1  Function outline ......................................................................................................................................................... 7
  3.2  IC card data structure ................................................................................................................................................. 7
  3.2.1 Static data/ dynamic data ........................................................................................................................................ 7
  3.2.2 Initial setup data ....................................................................................................................................................... 7
  3.2.3 Folder/ File Structure ............................................................................................................................................. 8
  3.2.4 Folder structure details .......................................................................................................................................... 8
  3.2.5 Folder structure details .......................................................................................................................................... 9
  3.3  Folder and file access privileges ............................................................................................................................... 9
  3.3.1 Folder access privilege ......................................................................................................................................... 9
  3.3.2 File access privileges ........................................................................................................................................... 9
  3.4  Authentication ............................................................................................................................................................... 10
  3.4.1 Authentication mode ............................................................................................................................................. 10
  3.4.2 Owner authentication ....................................................................................................................................... 10
  3.4.3 Local access and remote access ......................................................................................................................... 10
  3.4.4 The relationship between the authentication mode and the message .............................................................................. 11
  3.5  Data domain backup .................................................................................................................................................. 12

4  IC card message processing-------------------------------------------------------------------------------------------- 13
  4.1  Processing branch by message type .......................................................................................................................... 13
  4.2  Respective message processing .................................................................................................................................. 13
  4.2.1 Parameters ............................................................................................................................................................ 13
  4.2.2 Output ...................................................................................................................................................................... 13
  4.2.3 Exceptional Output .............................................................................................................................................. 13
5 Basic message

5.1 Cast of characters

5.2 Input message from the application to the IC card

5.2.1 RequestID

5.2.2 RequestChallenge

5.2.3 Authenticate

5.2.4 CreateFile

5.2.5 DeleteFile

5.2.6 MoveFile

5.2.7 RequestFileInfo

5.2.8 RequestFileList

5.2.9 CreateFolder

5.2.10 DeleteFolder

5.2.11 RequestFolderList

5.2.12 RequestCardInfo

5.2.13 BackUpCard

5.2.14 ReformatCard

5.2.15 RestoreCard

5.3 Output messages from the IC card to the application

5.3.1 DelegatedID

5.3.2 Challenge

5.3.3 AuthMode

5.3.4 SuccessfulFileOperation

5.3.5 SuccessfulFolderOperation

5.3.6 SuccessfulOperation

5.3.7 FileInfo

5.3.8 FileList

5.3.9 FolderList

5.3.10 CardInfo

5.3.11 BackUpInfo

5.3.12 UnsupportedMessage

5.3.13 AccessViolation

5.3.14 ObjectNotFound

5.3.15 IllegalParameters

5.3.16 MemoryOverflow

5.3.17 MaximumNumberOfExceeded

5.3.18 MessageSizeOverflow

5.3.19 ExchangeSuspended

5.3.20 IncompatibleStatus
6 Exchange Control Messages

6.1 Exchange protocol details

6.1.1 Cast of characters

6.1.2 Protocol procedure

6.1.2.1 Main protocol

6.1.2.2 Abort request protocol

6.1.2.3 Resolve request protocol

6.2 State record/ state table

6.3 Inputting messages from the application/ IC card to the IC card

6.3.1 Main protocol messages

6.3.1.1 StartExchange

6.3.1.2 Offer

6.3.1.3 AgreeExchange

6.3.1.4 Agreement

6.3.1.5 ConfirmExchange

6.3.1.6 Confirmation

6.3.1.7 Commitment

6.3.2 Abort request/ resolve request protocol message

6.3.2.1 RecoverExchange

6.3.2.2 ArbitrationRequest

6.3.2.3 Arbitration

6.3.2.4 CancelExchange

6.3.3 Exchange state management messages

6.3.3.1 RequestExgStatusInfo

6.3.3.2 RequestExgStatusList

6.4 Output messages from IC cards to applications

6.4.1 ExchangeCommitted

6.4.2 ExchangAborted

6.4.3 ExgStatusInfo

6.4.4 ExgStatusList

Appendix. APDU Command

A.1 ReqICCID

A.2 Unlock

Bibliography

1 Introduction

These specifications stipulate the message sets for the e^2TP (Extended eTRON Transfer Protocol) with which TENeT (Trusted Environment with Networking eTRON) specification IC cards must be equipped.

1.1 Outline of this specification

These specifications stipulate the TENeT messages, messages sets for the e^2TP messages with which the IC cards must be equipped to safely store and acquire and the authority values.

Copyright © 2005 by T-Engine Forum. All rights reserved
The TENeT specification IC card including the TENeT messages possesses such basic functions as the generating and deleting of folders and authority values, an access control that sets authentication of the access itself and the access privileges for authority values, and an exchange function that safely and impartially transacts authority values.

The exchange function adopts ‘an optimistic exchange protocol [THIF03] for unbiased authority value transactions.’ Optimistic refers to the fact that normally, when an exchange has been completed between two parties, and the exchange is then interrupted, for example due to a communication cut-off, resulting in a state where it loses its own authority value or where the other party’s authority value is not received, it restores it to the state for coordinating with the other exchange party from the above state between the third party organizations, without restarting the exchange with the opposite party.

These functions possessed by the TENeT specification IC card enable it to safely store the authority values and to impartially and safely transact authority values between IC cards.

1.2 Positioning of these specifications

TENeT stipulates the various specifications to enable the mutual operability of the IC cards and application programs that undertake the authority value transactions. The titles of the specifications and their content are as follows.

This specification stipulates the TENeT message specifications.

- TENeT authority value transaction API specification
  - API for the application program to manage and obtain the authority values of the IC cards
- TENeT messaging API specifications
  - API for application programs and IC cards to send and receive e2TP messages
- Extended eTP (e2TP) message specifications
  - Message format for e2TP messages and mapping method for the ISO7816-4APDU
- TENeT message specification [This document]
  - Message set for e2TP messages that must be equipped with TENeT specification IC cards

1.3 Stipulated range

This specification stipulates two types of message sets for the TENeT message.

- Basic message
  - Messages enabling basic functions, the access control function and operating functions, such as data backup.
- Exchange control messages
  - Messages enabling exchange functions for applications to undertake safe and impartial transactions using IC cards.

1.4 Reference stipulations


1.5 Notations

The common notations used in these specifications are defined in Table 1

Table 1 Notations
1.6 Cast of characters

The characters in these specifications are the TENeT IC card, which are the TENeT specification IC card, and application programs (AP) on portable terminals[M2]. The TENeT IC card is inserted into a portable terminal and is operated by the user using a portable terminal APs. When authority value transactions are interrupted, there is a TTP (trusted third party), which serves as the mediator for assuring the compatibility of the authority values between IC cards during a transaction.

TENeT messages are messages sent and received between these characters. The basic messages indicated in section 1.3 are classified into input messages from the AP to the IC card and output messages that notify the processing result of the IC card to the AP. Exchange control messages are classified into input messages from the AP to the IC card and output messages that are sent from the IC card to another IC card or to the AP or the TTP.

1.7 Message types

Basic/exchange messages are input into the DATA section of the e²TP message. The e²TP messages have message types for identifying the respective TENeT messages.

The message types represent major classifications that indicate basic or exchange messages with the upper 1 byte and indicate minor classifications of basic or exchange messages with the lower 1 byte. Within the upper 1 byte, TENeT assigns 00h as the value of the basic message and 01h as the value of the exchange message.

The minor classifications of the lower 1 byte of the TENeT message stipulated in these specifications are indicated in Table 3. For example, the upper 1 byte of the message type for the CreateFile message becomes 00h and the lower 1 byte becomes 40h.

<p>| Table 3 Message lower 1 byte minor classifications of the basic/exchange messages |
|-----------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| Basic minor classification  | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Hexadecimal orthography |
| CreateFile                  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 40h          |
| DeleteFile                  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 1  | 41h          |
| RequestFileInfo            | 0  | 1  | 0  | 0  | 0  | 0  | 1  | 0  | 42h          |
| MoveFile                    | 0  | 1  | 0  | 0  | 0  | 0  | 1  | 1  | 43h          |
| RequestFileList             | 0  | 1  | 0  | 0  | 0  | 1  | 0  | 0  | 44h          |
| CreateFolder                | 0  | 1  | 0  | 0  | 0  | 1  | 0  | 1  | 45h          |
| DeleteFolder                | 0  | 1  | 0  | 0  | 0  | 1  | 1  | 0  | 46h          |
| RequestFolderList           | 0  | 1  | 0  | 0  | 0  | 1  | 1  | 1  | 47h          |
| RequestID                   | 0  | 1  | 0  | 0  | 1  | 0  | 0  | 0  | 48h          |
| BackUpCard                  | 0  | 1  | 0  | 0  | 1  | 0  | 0  | 1  | 49h          |
| RestoreCard                 | 0  | 1  | 0  | 0  | 1  | 0  | 1  | 0  | 4Ah           |
| ReformatCard                | 0  | 1  | 0  | 0  | 1  | 0  | 1  | 1  | 4Bh           |
| RequestCardInfo             | 0  | 1  | 0  | 0  | 1  | 1  | 0  | 0  | 4Ch           |
| RequestChallenge            | 0  | 1  | 0  | 0  | 1  | 1  | 0  | 1  | 4Dh           |</p>
<table>
<thead>
<tr>
<th>Exchange minor classification</th>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
<th>b0</th>
<th>Hexadecimal orthography</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartExchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40h</td>
</tr>
<tr>
<td>Offer</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>21h</td>
</tr>
<tr>
<td>AgreeExchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>42h</td>
</tr>
<tr>
<td>Agreement</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>23h</td>
</tr>
<tr>
<td>ConfirmExchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>44h</td>
</tr>
<tr>
<td>Confirmation</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>65h</td>
</tr>
<tr>
<td>Commitment</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>66h</td>
</tr>
<tr>
<td>RecoverExchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>47h</td>
</tr>
<tr>
<td>ArbitrationRequest</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>28h</td>
</tr>
<tr>
<td>Arbitration</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>49h</td>
</tr>
<tr>
<td>RequestExgStatusInfo</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4Ah</td>
</tr>
<tr>
<td>CancelExchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4Bh</td>
</tr>
<tr>
<td>RequestExgStatusList</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4Ch</td>
</tr>
<tr>
<td>ExchangeCommitted</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2Dh</td>
</tr>
<tr>
<td>ExchangeAborted</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2Eh</td>
</tr>
<tr>
<td>ExgStatusInfo</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2Fh</td>
</tr>
<tr>
<td>ExgStatusList</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>30h</td>
</tr>
<tr>
<td>ExchangeSuspended</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>A8h</td>
</tr>
<tr>
<td>IncompatibleStatus</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>A9h</td>
</tr>
</tbody>
</table>
2 Data Definitions

This chapter stipulates the formats and the values of the data to be used by the TENeT IC card.

2.1 Public key certificate

The TENeT IC card uses the eTron ID that is the 16 byte identifier stipulated in the eTRON/16. See “e2TP Message Specifications [TEE2] for the detailed eTRON ID specifications.

Table 4 indicates the public key certificate possessed by the TENeT IC card. The public key certificate assures that ETRON ID possessed by the TENeT IC card is the eTRON ID correctly issued by the Public Key Certificate Issuance Authority (CA: Certificate Authority). This specification stipulates Ver=02h as the specified value for the public key certificate value. The breakdown of the signature algorithms and the stipulated values that can be used by TENeT are shown in Table 5.

<table>
<thead>
<tr>
<th>#</th>
<th>Symbol</th>
<th>Size (Bytes)</th>
<th>Meaning</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ver</td>
<td>1</td>
<td>Version of the public key certificate</td>
<td>02h</td>
</tr>
<tr>
<td>2</td>
<td>CA_ID</td>
<td>16</td>
<td>eTRON ID of the CA</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Serial_no</td>
<td>4</td>
<td>Certificate serial number</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Time/Start</td>
<td>4</td>
<td>Key usage starting date and time</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Time/End</td>
<td>4</td>
<td>Key usage ending date and time</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>eTRON ID</td>
<td>16</td>
<td>eTRON ID certified by the certificate</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MyKeyVer</td>
<td>1</td>
<td>Public key version</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MyKeyAlgorithm</td>
<td>1</td>
<td>Public key algorithm</td>
<td>See table 5</td>
</tr>
<tr>
<td>9</td>
<td>MyPublicKey</td>
<td>variable</td>
<td>Public key itself</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>SignAlgorithm</td>
<td>1</td>
<td>Signature algorithm allocated by CA</td>
<td>See table 5</td>
</tr>
<tr>
<td>11</td>
<td>Sign</td>
<td>variable</td>
<td>Signature allocated by the CA (signature values #1~#10 according to CA secret key)</td>
<td></td>
</tr>
</tbody>
</table>

Table 5 Signature algorithm

<table>
<thead>
<tr>
<th>#</th>
<th>Symbol</th>
<th>Bit length</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ECDSA (prime field)</td>
<td>160</td>
<td>01h (*1)</td>
</tr>
<tr>
<td>4</td>
<td>ECDSA (2 extensive fields) (RFU)</td>
<td>(RFU)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>RSA</td>
<td>1024</td>
<td>02h</td>
</tr>
<tr>
<td>10</td>
<td>DSA</td>
<td>1024</td>
<td>03h</td>
</tr>
<tr>
<td>12</td>
<td>(RFU)</td>
<td>--</td>
<td>-- h</td>
</tr>
</tbody>
</table>

(*1) ECDSA (prime field) indicates a 160 bit domain parameter in Table 5
### Table 6 Specified value of the domain parameter

<table>
<thead>
<tr>
<th>Item</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elliptical curve a</td>
<td>07 25 46 B5 43 52 34 A4 22 E0 78 96 75 F4 32 C8 94 35 DE 52 42</td>
</tr>
<tr>
<td>Elliptical curve b</td>
<td>00 C9 51 7D 06 D5 24 0D 3C FF 38 C7 4B 20 B6 CD 4D 6F 9D D4 D9</td>
</tr>
<tr>
<td>Base point P.x</td>
<td>07 AF 69 98 95 46 10 3D 79 32 9F CC 3D 74 88 0F 33 BB E8 03 CB</td>
</tr>
<tr>
<td>Base point P.y</td>
<td>01 EC 23 21 1B 59 66 AD EA 1D 3F 87 F7 EA 58 48 AE F0 B7 CA 9F</td>
</tr>
<tr>
<td>Digits of base point P</td>
<td>04 00 00 00 00 00 00 00 01 E6 0F C8 82 1C C7 4D AE AF C1</td>
</tr>
<tr>
<td>Base point Q.x</td>
<td>06 CE 98 1C B0 D2 7B 4B 3A 71 98 F1 A1 5B 53 6D C2 E6 12 29 95</td>
</tr>
<tr>
<td>Base point Q.y</td>
<td>05 0D D9 9B 8F D9 FF 4B 68 F2 80 36 4D 02 49 B7 AF 57 AB 4E 74</td>
</tr>
</tbody>
</table>

#### 2.2 eTRON ID

The eTRON ID is a 16 byte identifier stipulated in the “e^2^TP Message Specification [TEE2].

In TENeT, in the eTRON IDs, the upper 12 bytes are called the ‘domain’ and the lower 4 bytes the ‘port.’ Because TENeT IC cards have ports where the card itself uses port=0000h, the eTRON ID of the TENeT IC card becomes domain|0000h. The TENeT IC card uses this eTRON ID as the identifier when sending and receiving e2TP messages.

TENeT IC cards generate a port in accordance with the request from the AP, and then issue the ‘domain|port’ to the AP as the AP identifier. The port is a unique value within the IC card. The AP uses this identifier as the identifier when sending and receiving e^2^TP messages.
3 IC Card Functions

This chapter explains common characteristics of TENeT IC cards, such as the functions possessed by the TENeT IC card, the data structure within the TENeT IC card and the application authentication method.

3.1 Function outline

The TENeT IC card provides operation and management of such authority values as the generation, deletion and viewing of authority values, owner authentication and access control that protects the authority values and functions for safe and impartial transactions for authority values between IC cards.

The TENeT message specification that applies the above functions to the TENeT IC cards is composed of basic messages and exchange control messages. The respective messages provide the below functions. Detailed explanations of the message operations and parameters are provided in chapters 5 and 6.

- **Basic message**
  - The basic functions enabling basic user operations, such as the generation, deletion and viewing of authority values and folders; the setting of access privileges for authority values and the access control functions that authenticate the access itself to the IC card.
  - Operating functions for actual operation, such as data backup within the card.

- **Exchange control message**
  - Function enabling safe and impartial transactions of authority values between TENeT IC cards.

3.2 IC card data structure

This section explains the data structure of TENeT IC cards

3.2.1 Static data/ dynamic data

The data stored and managed by TENeT IC cards can be static or dynamic.

- **Static data**
  - Static data is written into non-volatile memory, when the IC card is initialized, and not rewritten subsequently.
  - Static data includes data such as initial setup data. Table 7 shows the initial setup data.

- **Dynamic data**
  - Dynamic data is data that may be rewritten following initialization.
  - Dynamic data is data that may be rewritten following initialization.
  - With dynamic data, the atomicity is assured when it is rewritten through the OS function of the IC card.
  - The data size that is assured varies depending on the IC card specification. Folders and files qualify as dynamic data. We will explain the structure of folders and files from section 3.2.3.

3.2.2 Initial setup data

TENeT IC cards store the data shown in Table 7 as initial setup data in non-volatile memory. Table 4 shows a breakdown of the owner certificate. The public key of the CA can be used to verify the legitimacy of the owner certificate.

These specifications do not stipulate the means for setting and changing these initial setup data on the IC card.

<table>
<thead>
<tr>
<th>Data</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public key of the CA</td>
<td>CA public key for owner certificate verification</td>
</tr>
<tr>
<td>ECDSA secret key</td>
<td>Secret key owned by IC card</td>
</tr>
</tbody>
</table>

Table 7 IC card initial setup data

Copyright © 2005 by T-Engine Forum. All rights reserved.
3.2.3 Folder/ File Structure

The characteristics of folders and files, which are dynamic data, are as follows. Diagram 1 shows the relationship between the folders and files.

- Folders
  Folders are a hierarchical structure and there is no root folder. The maximum number of folders depends on the respective IC card packaging.

- Files
  One file corresponds to one authority value. Each file is stored and managed within a folder. The maximum file size within an IC card depends on the respective IC card packaging.

![Diagram 1 Folder/ file structure](image)

3.2.4 Folder structure details

The IC card has the respective folder information shown in Table 8 in each folder. The IC card manages the folder information of the folders within the card as folder lists in a list format.

<table>
<thead>
<tr>
<th>Data</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foldername</td>
<td>Folder name allocated by the IC card owner</td>
</tr>
<tr>
<td>folderACL</td>
<td>Access privilege allocated by the IC card owner</td>
</tr>
<tr>
<td>folderID</td>
<td>Folder identifier assigned by the IC card</td>
</tr>
</tbody>
</table>
3.2.5 Folder structure details

The IC card has the file information shown in Table 9 for each file. The files also have the below characteristics.
- One file corresponds to one type of authority value.
- One authority value of the same type is consolidated into one authority value and the number of files is added.
- Files cannot be falsified, due to the allocating of the eTRON ID for the IC card.
- The creator ID and content of a file are not changed in the distribution process.

<table>
<thead>
<tr>
<th>Data</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileID</td>
<td>The file identifier assigned by the IC card</td>
</tr>
<tr>
<td>fileCnt</td>
<td>Number of files</td>
</tr>
<tr>
<td>fileACL</td>
<td>Access privileges allocated by the file creator</td>
</tr>
<tr>
<td>issuerID</td>
<td>eTRON ID of the file creator</td>
</tr>
<tr>
<td>fileLEN</td>
<td>Data size of the fileDATA</td>
</tr>
<tr>
<td>fileDATA</td>
<td>File content set by the file creator</td>
</tr>
</tbody>
</table>

3.3 Folder and file access privileges

The TENeT IC card sets access privileges for folders and files, to protect the authority values within the card. The TENeT IC card controls access according to the access privileges of the folders and files.

3.3.1 Folder access privileges

Table 8 shows the folder access privileges set for folders. Only the TENeT IC card owner can set them. For example, when the folder access privilege is “101,” a non-owner can also write, transfer and exchange files within the folder, but cannot create them.

Whereas the folder access level mentioned in Table 10 does not apply to the TENeT IC card owner, those indicated in Table 9 do apply.

<table>
<thead>
<tr>
<th>Bit String</th>
<th>Meaning</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Read level permission</td>
<td>Files within the folder can also be read by other than the owner</td>
</tr>
<tr>
<td>010</td>
<td>Create level permission</td>
<td>Files within the folder can also be created by other than the owner</td>
</tr>
<tr>
<td>001</td>
<td>Transfer level permission</td>
<td>Files within the folder can also be transferred and exchanged by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>other than the owner</td>
</tr>
</tbody>
</table>

3.3.2 File access privileges

Table 9 shows the access privileges set for files. Only the file creator can set them. For example, when the file access privilege is “01,” a non-owner can also transfer and exchange files, but cannot copy them. When the file creator operates the file, the access privileges in Table 11 do not apply.
### Table 11  File access privileges

<table>
<thead>
<tr>
<th>Bit String</th>
<th>Meaning</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Copy level permission</td>
<td>File can also be copied by other than the creator</td>
</tr>
<tr>
<td>01</td>
<td>Transfer level permission</td>
<td>File can also be transferred and exchanged by other than the creator</td>
</tr>
</tbody>
</table>

### 3.4 Authentication

#### 3.4.1 Authentication mode

TENeT IC cards have two types of authentication modes, ‘non-authentication mode’ and ‘owner mode.’ When messages are received the control for permitting and not permitting the execution of the concerned messages is by the authentication mode.

The initial state of the TENeT IC card is the non-authentication mode. The TENeT IC card authenticates the authentication requests from the access subject in accordance with the authentication mode specified by the access subject. If the authentication succeeds, the IC card authentication mode shifts to the specified authentication mode. When the authentication fails, the authentication mode within the IC card does not shift.

#### 3.4.2 Owner authentication

The TENeT IC cards authenticates that (the user operating) the AP is the IC card owner by the following method.

First the TENeT IC card receives a RequestChallenge message (section 5.2.2), and then generates a challenge and sends a Challenge message (section 5.3.2) including the challenge.

When the AP receives the Challenge message and acquires the challenge, it inputs into the IC card, the hash value h <challenge|PIN> linking the PIN for the owner authentication of the initial setup data indicated in section 1.9.1 with the challenge and the Authenticate (section 5.2.3) message that utilizes the mode indicating the owner mode for the DATA.

If the challenge and PIN are correctly authenticated, the IC card grants access privileges to the AP of the Authenticate message as the owner mode. When the authentication has failed, it stays in the original non-authentication mode.

The IC card makes the identifier of the transmission source of the Authenticate message and the authentication mode following the shift into a set and then manages multiple sets in a list format. The IC card disregards authentication sets over a fixed number, so as to restrict the resources. The number of sets of an authentication state stored in a list depends on the packaging. When it stores a new authentication set exceeding the upper limit, it disregards the authentication state stored in the list according to the LRU (least recently used). In other words, when the number of authentication sets has exceeded the upper limit, it deletes the set for the authentication state corresponding to the transmission source that has been without a message exchange for the longest time among the stored authentication state sets.

#### 3.4.3 Local access and remote access

The TENeT IC card accepts access from the AP or another TENeT IC card through either local access or remote access.

Local access is the access from the AP within the portable terminal in which the TENeT IC card has been inserted and it judges whether or not it is the access for the AP within that terminal according to the AP...
identifier mentioned in section 2.3.

In other words, when the domain of the transmission source identifier is equivalent to the domain of the TENeT IC card, the TENeT IC card accepts the access from that application as a local access. Conversely, because the domain of the TENeT IC card and AP identifier is different from the TENeT IC card domain, it accepts the access from those devices as remote access.

The TENeT IC card executes all the messages accepted by remote access in the non-authentication mode.

### 3.4.4 The relationship between the authentication mode and the message

When the owner authentication succeeds in the owner mode in accordance with the Authenticate message (section 5.2.3), the local access becomes the access in the owner mode. Table 10 shows the relationship between permitted/ non-permitted execution of the TENeT message.

<table>
<thead>
<tr>
<th>Table 10</th>
<th>Relationship between the respective message and the authentication mode</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Message Classification</strong></td>
<td><strong>Message Designation</strong></td>
</tr>
<tr>
<td>Basic message</td>
<td>CreateFile</td>
</tr>
<tr>
<td></td>
<td>DeleteFile</td>
</tr>
<tr>
<td></td>
<td>RequestFileInfo</td>
</tr>
<tr>
<td></td>
<td>MoveFile</td>
</tr>
<tr>
<td></td>
<td>CreateFolder</td>
</tr>
<tr>
<td></td>
<td>DeleteFolder</td>
</tr>
<tr>
<td></td>
<td>RequestFileList</td>
</tr>
<tr>
<td></td>
<td>RequestFolderList</td>
</tr>
<tr>
<td></td>
<td>RequestID</td>
</tr>
<tr>
<td></td>
<td>BackUpCard</td>
</tr>
<tr>
<td></td>
<td>RestoreCard</td>
</tr>
<tr>
<td></td>
<td>ReformatCard</td>
</tr>
<tr>
<td></td>
<td>RequestCardInfo</td>
</tr>
<tr>
<td></td>
<td>RequestChallenge</td>
</tr>
<tr>
<td></td>
<td>Authenticate</td>
</tr>
<tr>
<td>Exchange message</td>
<td>StartExchange</td>
</tr>
<tr>
<td></td>
<td>AgreeExchange</td>
</tr>
<tr>
<td></td>
<td>ConfirmExchange</td>
</tr>
<tr>
<td></td>
<td>Confirmation</td>
</tr>
<tr>
<td></td>
<td>Commitment</td>
</tr>
<tr>
<td></td>
<td>RecoverExchange</td>
</tr>
<tr>
<td></td>
<td>Arbitration</td>
</tr>
<tr>
<td></td>
<td>RequestExgStatusInfo</td>
</tr>
<tr>
<td></td>
<td>RequestExgStatusList</td>
</tr>
<tr>
<td></td>
<td>CancelExchange</td>
</tr>
</tbody>
</table>

Legend  O: The IC card accepts the message, and then executes the processing
X: The IC card does not accept it and processes it as an error
※: If the condition in the parentheses is met, the IC card executes the message processing
3.5 Data domain backup

The TENeT IC card assures the file domain used to store the file from the memory domain. This size of this file domain can be changed using the Reformat message (section 5.2.14).

When the size of the file area changes, since the file area is momentarily erased, it has a BackUp message (section 5.2.13) that temporarily saves the data within the TENeT IC card to the memory of the portable terminal. When the TENeT IC card accepts the BackUp message, it links random, the generated random number within the card with the backupinfo, which is the data of card to be saved, and then outputs a BackUpInfo message that includes the data (backupinfo| random) PKICC that transferred the signature. random holds it within the card, until the backup is completed.

When the TENeT IC card sends a BackUpInfo message, it changes to the LOCKED state, and prohibits access to the file area during backup by another message.

When re-storing the backupinfo, it inputs the (backupinfo| random) PKICC from the AP to the IC card, using the Restore message. When the IC card has been verified as correct, and when the random number held by it can be crosschecked with ransom show it to be correct, it again stores the backupinfo in the file area, and then completes the backup.
4 IC card message processing

4.1 Processing branch by message type

The TEnet IC card identifies the accepted TEnet message by message type. When a message type is not stipulated in Table 3 of section 1.8, the IC card sends back an UnsupportedMessage message.

When the LEN value indicating the size of the TEnet message that is the DATA of the e2TP message and the size of the actual DATA are different, it outputs an IllegalParameters message.

4.2 Respective message processing

When it does not return any of the error messages mentioned in section 4.1, it performs the processing decided for the respective message from Chapter 5, based on the message type stipulated in Table 3. We will now explain the respective message items.

4.2.1 Parameters

These are the respective values of the TEnet messages. The message values are laid out in the memory area sequentially from the top.

4.2.2 Output

When the TEnet message that has been input is processed normally, it is the TEnet message output by the IC card.

4.2.3 Exceptional Output

This is the TEnet message that is output when the processing of the TEnet message that has been input ends abnormally. TEnet messages output with Exceptional Output vary according to the cause generating the error.
5 Basic message

5.1 Cast of characters

The characters of the basic message are the TEnET IC card and the application program (AP) on the portable terminal that operates the TEnET IC card. The basic messages are classified into input messages, which are the operating commands from the AP to the TEnET IC card, and output messages, which are sent from the TEnET IC card to the AP as the processing result for the input message.

These specifications stipulate the input message from the AP to the TEnET IC card in section 4.2 and the output message from the TEnET IC card to the AP in section 4.3.

5.2 Input message from the application to the IC card

5.2.1 RequestID

Message requesting that the AP issue an ID for the IC card.

The DATA specified by the AP upon generation of this message is empty.

The IC card that accepts the message generates a port, and then generates the identifier AP_ID (domain|port) linked with the domain of the IC card, regardless of the authentication state that is paired with the identifier of the transmission source.

When the above processing ends normally, the IC card sends a DelegatedID with AP_ID set as the value to the transmission source AP.

When the above processing ends abnormally, the IC card sends the below messages to the transmission source AP.

- When the LEN and the DATA size are different, it sends an IllegalParameters message
- When all the ports have been used, it sends a MaximumNumberExceeded message.

Parameters:

| DATA details | None |

Output:

DelegatedID

Data details

<table>
<thead>
<tr>
<th>AP_ID</th>
<th>The issued AP_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16B</td>
</tr>
</tbody>
</table>

Exceptional Output:

<table>
<thead>
<tr>
<th>(Error message)</th>
<th>(Cause)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IllegalParameters</td>
<td>- The LEN is different from the DATA size</td>
</tr>
<tr>
<td>MaximumNumberExceeded</td>
<td>- There is no port space (all the ports are used)</td>
</tr>
</tbody>
</table>
5.2.2 RequestChallenge

Message in which the AP requests the output of a challenge to the IC card. The DATA specified by the AP when this message is input is empty. The IC card that accepted the message generates challengedata only when the eTRON ID domain of the IC card is equivalent to the domain of the transmission source AP.

The IC card sends a Challenge message making the challengedata the value to the transmission source AP, when the above processing ends normally.

The IC card sends the below message to the transmission source AP, when the above processing ends abnormally.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the domain of the transmission source AP and the domain of the eTRON ID of the IC card are different, it sends an AccessViolation message.

Parameters:
- DATA details
  None

Output:
- Challenge
  - DATA details
    - Challenedata    challenge    20B

Exceptional Output:
- AccessViolation  - The IC card is in the LOCKED state/
  the message transmission source is not local access
- IllegalParams   - The LEN is different from the DATA size
5.2.3 Authenticate

Message where the AP specifies the authentication to the IC card.

The AP requires that the challengedata be obtained by the RequestChallenge message before sending this message.

When this message is generated,
- In the case of the owner mode, the AP specifies the authentication information, including the authentication mode, mode (=0002h); the challengedata that holds it and the secret information (PIN for owner authentication set in the IC card).
- In the case of the non-authentication mode, it specifies the authentication mode, mode (=0000h).

The IC card that accepts the message only authenticates when the transmission source domain is equivalent to the domain of the IC card eTRON ID (see section 3.4.2 Authentication methods). When authentication is successful, the TENEIC card shifts to the specified authentication mode. When the authentication fails, the specified authentication mode does not shift.

When the above process ends normally, the IC card sends the Authmode message, including the authentication mode, mode, after shifting to the transmission source AP.

When the above processing ends abnormally, the IC card sends the below messages to the transmission source AP.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the transmission source domain and the domain of the IC card eTRON ID are different, it sends an AccessViolation message.
- When the mode value is neither 0000h nor 0002h, it sends an IllegalParameters message.

Parameters:

DATA details
- In case of the owner mode
  - mode Authentication mode 2B
  - Authenticator Information for authentication 20B
    - Hash value h (ch | pw) of the challengedata (ch) and secret information (pw) obtained from the IC card.
- In case of the non-authentication mode
  - mode Authentication mode 2B
    - Specifies the owner mode: mode=0002h
    - Specifies the non-authentication mode: mode=0000h

Output:

AuthMode
- DATA details
  - mode Authentication mode of the shifted results 2B
    - Shift to the owner mode: mode=0002h
    - Shift to the non-authentication mode: mode=0000h

Exceptional Output:

- (Error message) (Cause)
  - AccessViolation - The IC card is in the LOCKED state /
    the message transmission source is not local access
  - IllegalParams - The LEN is different from the DATA size /
    The mode is not 0000h or 0002h
5.2.4 CreateFile

Message in which the AP specifies the creation of a file to the IC card.

When this message is generated, the AP specifies the file content, fileDATA, which is the file information; the number of files, fileCnt; the file access level, fileACL and the folder identifier for the file storage destination, folderID. If the authentication state paired with the transmission source identifier is the owner mode, the IC card that accepted the message creates a file in the specified folder and allocates the CreatedID that is the file identifier.

The generated file is allocated by the eTRON ID within the owner certificate of section 3.2.2. The file is managed in combination with this eTRON ID.

When the above processing has ended normally, the IC card sends a SuccessfulFileOperation message including createdCnt, the number of files generated, and the above createdID to the transmission source AP.

When the above processing has ended abnormally, the IC card sends the below messages to the transmission source AP.

- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the transmission source has not been authenticated as the owner, it sends an AccessViolation message.
- When the folder indicated by the folderID does not exist, it sends an ObjectNotFound message
- When the fileCnt is 0, it sends an IllegalParameters message.
- When the LEN value exceeds the maximum storage file size, or when there is no available capacity for creating new files it sends a MemoryOverflow message
- When the number of files following addition exceeds FFFFFFFFh it sends a MaximumNumberExceeded

**Parameters:**

DATA details

- folderID: Folder identifier of the file storage destination 2B
- fileCnt: Number of files 4B
- fileACL: File access level 1B
- fileLEN: File length 2B
- fileDATA: File contents (filelen) B

**Output:**

SuccessfulFileOperation

DATA details

- MessageType: CreateFile MessageType 2B
- createdID: Identifier of created file 2B
- createdCnt: Number of files created 4B

**Exceptional Output:**

<table>
<thead>
<tr>
<th>Error message</th>
<th>(Cause)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>The IC card is in the LOCKED state/ the authentication state corresponding to the transmission source is not the owner mode</td>
</tr>
<tr>
<td>ObjectNotFound</td>
<td>The folder indicated by the folderID does not exist</td>
</tr>
<tr>
<td>IllegalParameters</td>
<td>It inputs 0 for the fileCnt</td>
</tr>
<tr>
<td>MemoryOverflow</td>
<td>The LEN exceeds the maximum storage file size/ there is no available card capacity</td>
</tr>
<tr>
<td>MaximumNumberExceed</td>
<td>The fileCnt of the files following addition exceeds FFFFFFFFh</td>
</tr>
</tbody>
</table>
5.2.5 DeleteFile

Message in which the AP specifies the deletion of files to the IC card.

When this message is input, the AP specifies the storage destination folder identifier of the files to be deleted, folderID; the identifier of the file to be deleted, fileID and the number of files to be deleted, fileCnt.

If the authentication state that is paired with the transmission source identifier is the owner mode, the IC card that accepts the message deletes the specified files.

When the above process ends normally, the IC card outputs a SuccessfulFileOperation message that includes the identifier of the deleted files, deletedID, and the number that were deleted, deletedCnt, to the transmission source.

The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the transmission state paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When there is no folder indicated with the folderID, it sends an ObjectNotFound message.
- When delfileCnt is 0, it sends a IllegalParameters message.
- When the number of files when a message is accepted is less than the delfileCnt, it sends a MaximumNumberExceed message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>2B</th>
<th>4B</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderID</td>
<td>Folder identifier of the storage destination of the file to be deleted</td>
<td></td>
</tr>
<tr>
<td>fileID</td>
<td>Identifier of the file to be deleted</td>
<td></td>
</tr>
<tr>
<td>fileCnt</td>
<td>Number of files to be deleted</td>
<td></td>
</tr>
</tbody>
</table>

Output:

SuccessfulFileOperation:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>2B</th>
<th>4B</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageType</td>
<td>DeleteFile MessageType</td>
<td></td>
</tr>
<tr>
<td>deletedID</td>
<td>Identifier of deleted file</td>
<td></td>
</tr>
<tr>
<td>deletedCnt</td>
<td>Number of files deleted</td>
<td></td>
</tr>
</tbody>
</table>

Exceptional Output:

<table>
<thead>
<tr>
<th>Error message</th>
<th>Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>The IC card is in the LOCKED state/ the authentication state corresponding to the transmission source is not the owner mode</td>
</tr>
<tr>
<td>ObjectNotFound</td>
<td>The folder indicated by the folderID does not exist</td>
</tr>
<tr>
<td>IllegalParameters</td>
<td>The LEN is different from the DATA size/ it inputs 0 for the fileCnt</td>
</tr>
<tr>
<td>MaximumNumberExceed</td>
<td>The current number of files is less than the delfileCnt</td>
</tr>
</tbody>
</table>
5.2.6 MoveFile

Message in which the AP specifies the movement of files between folders to the IC card.

When this message is generated, the AP specifies the identifier of the file(s) to be moved, fileID; the number of files to be moved, fileCnt; the folder identifier of the storage source, folderID; the folder identifier of the storage destination, dstfolderID and the copyflag that differentiates movements and copies.

If the authentication state that is paired with the transmission source identifier is the owner mode, the IC card that accepts the message moves the file(s).

When there are files that match the issuerID, which is the eTRON ID of the creator of the files for the storage destination folder; the access level, fileACL, and the file content, fieDATA, the IC card adds the concerned file to the fileCnt portion.

When the fileCnt is not the same as the number of files, the IC card completely deletes the files from the storage source folder.

When the copyflag is not 0, it changes to file copying and the number of files of the movement source is not subtracted.

When the above process ends normally, the IC card sends a SuccessfulFileOperation message to the AP of the transmission source.

The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the transmission source has not been authenticated as the owner, and when a copy of the file has not been approved when copyflag ≠ 00h, it sends an AccessViolation message.
- When there is no folder indicated with the folderID, and no file specified with the fileID, and when the number to be moved exceeds the number of files, it sends an ObjectNotFound message.
- When 0 has been input for the fileCnt and when the folderID and the dstfolderID of the movement destination are the same, it sends an IllegalParameters message.
- When the file length specified by the fileID has exceeded the maximum storable file length, and when there is no available capacity when copying, it sends a MemoryOverflow message.
- When the number of movement destination files exceeds FFFFFFFFh, it sends a MaximumNumberExceed message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderID</td>
<td>Folder identifier of the storage destination of the movement destination file</td>
</tr>
<tr>
<td>copyFlag</td>
<td>Specification of the movement/copy (=00h: movement ≠ 00h: copy)</td>
</tr>
<tr>
<td>fileID</td>
<td>Identifier of the movement destination file</td>
</tr>
<tr>
<td>fileCnt</td>
<td>Number of files to be moved</td>
</tr>
<tr>
<td>dstfolderID</td>
<td>Folder identifier of the movement destination</td>
</tr>
</tbody>
</table>

Output:

SuccessfulFileOperation

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageType</td>
<td>MoveFile MessageType</td>
</tr>
<tr>
<td>fileID</td>
<td>Identifier ID of the movement destination file</td>
</tr>
<tr>
<td>fileCnt</td>
<td>Current number of files of the movement destination</td>
</tr>
</tbody>
</table>
Exceptional Output:
(Error message) (Cause)
AccessViolation - The IC card is in the LOCKED state/
the authentication state corresponding to the transmission source is not
the owner mode /
the file copy authority has not been approved for copyflag ≠ 00h
ObjectNotFound - The folder indicated by the folderID or the file indicated by the fileID
does not exist
IllegalParamets - The LEN is different from the DATA size
the dstfolderID is the same as the folderID/ it inputs 0 for the fileCnt
MemoryOverflow - The file length specified by the fileID exceeds the maximum storable file size/
there is no available card capacity
MaximumNumberExceeds - The number of movement destination files exceeds FFFFFFFFh
5.2.7 RequestFileInfo

Message in which the AP specifies the reading of files to the IC card.

When this message is input, the AP specifies the identifier of the file to be read, fileID; the identifier of the folder where the file is stored, folderID; the read start address, start, and the length to be read, len.

When the authentication state that is paired with the transmission source identifier is the owner mode, or when the authentication state is the non-authentication mode and read privileges are permitted, the IC card that accepts the message sends file information.

When the above process ends normally, the IC card sends a FileInfo message including the file information to the transmission source AP.

The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.

- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, and when folder read authority is not permitted, it sends an AccessViolation message.
- When there is no folder or file indicated at the time of the input, it sends an ObjectNotFound message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>2B</th>
<th>4B</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderID</td>
<td>Folder identifier of the storage destination of the file to be read</td>
<td>2B</td>
</tr>
<tr>
<td>fileID</td>
<td>Identifier of the file to be read</td>
<td>2B</td>
</tr>
<tr>
<td>start</td>
<td>Read start address</td>
<td>2B</td>
</tr>
<tr>
<td>len</td>
<td>(Offset value from the front of the file data)</td>
<td>2B</td>
</tr>
</tbody>
</table>

Output:

FileInfo

<table>
<thead>
<tr>
<th>DATA details</th>
<th>2B</th>
</tr>
</thead>
<tbody>
<tr>
<td>filelen</td>
<td>File data length</td>
</tr>
<tr>
<td>(data length of the actual file)</td>
<td></td>
</tr>
<tr>
<td>fileCnt</td>
<td>The number of files</td>
</tr>
<tr>
<td>fileACL</td>
<td>File access level</td>
</tr>
<tr>
<td>issuerID</td>
<td>eTRON ID of the file creator</td>
</tr>
<tr>
<td>readLen</td>
<td>Size of the fileDATA</td>
</tr>
<tr>
<td>(data length of the read file)</td>
<td></td>
</tr>
<tr>
<td>fileDATA</td>
<td>File content</td>
</tr>
</tbody>
</table>

Exceptional Output:

(Error message) (Cause)
AccessViolation  - The IC card is in the LOCKED state/
the authentication state corresponding to the transmission source is not the owner mode /
the folder read authority has not been acknowledged

ObjectNotFound   - The folder indicated by the folderID or the file indicated by the fileID does not exist

IllegalParaments - The LEN is different from the DATA size
5.2.8 RequestFileList

Message in which the AP specifies the outputting of a file list to the IC card.
When this message is generated, the AP specifies the folder identifier that obtains the file list, folderID; the read start address, start, and the length to be read, len.

If the authentication state that is paired with the transmission source identifier is the owner mode, or when the authentication state is the non-authentication mode and read privileges are not permitted, the IC card that accepts the message outputs a file list.

When the above process ends normally, the IC card sends a FileList message including the file list to the transmission source.

The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, and when folder read authority is not permitted, it sends an AccessViolation message.
- When there is no folder, it sends an ObjectNotFound message.
- When the size of the file list to be read exceeds the maximum transmittable size, it sends a MessageSizeOverflow message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>folderID</td>
<td>Folder identifier that acquires the file list</td>
</tr>
<tr>
<td>start</td>
<td>Read start address</td>
</tr>
<tr>
<td>(Offset value from the front of the file data)</td>
<td></td>
</tr>
<tr>
<td>len</td>
<td>Read length</td>
</tr>
</tbody>
</table>

Output:

<table>
<thead>
<tr>
<th>FileList</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>DATA details</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>filenum</td>
<td>Number of files managed by the file list</td>
</tr>
<tr>
<td>fileinfo</td>
<td>List of files</td>
</tr>
</tbody>
</table>

fileinfo details
- fileID: File ID | 2B |
- filelen: Actual file size | 2B |
- filecnt: The number of files | 4B |
- fileACL: File access level | 1B |
- issuerID: File issuer ID | 16B |
- readLen: Length of the read file data | 2B |
- fileDATA: Read file data | (readLen) B |

Exceptional Output:

<table>
<thead>
<tr>
<th>(Error message)</th>
<th>(Cause)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>- The IC card is in the LOCKED state/ the authentication state corresponding to the transmission source is not the owner mode / the folder read authority has not been acknowledged</td>
</tr>
<tr>
<td>ObjectNotFound</td>
<td>- The folder indicated by the folderID does not exist</td>
</tr>
<tr>
<td>IllegalParams</td>
<td>- The LEN is different from the DATA size</td>
</tr>
<tr>
<td>MemoryOverflow</td>
<td>- The size of the file list exceeds the maximum transmittable size</td>
</tr>
</tbody>
</table>
5.2.9 CreateFolder

Message in which the AP specifies the generation of a folder to the IC card.
When this message is generated, the AP specifies folder name, the folder name to be created and folderACL, the folder access level.
If the authentication state that is paired with the transmission source identifier is the owner mode, the IC card that accepts the message generates a folder and allocates an identifier, the CreatedfolderID.

When the above process ends normally, the IC card sends a SuccessfulFolderOperation message including the CreatedfolderID.
The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When there is already a folder the same as the folder name to be generated, it sends an IllegalParameters message.
- When there is no available capacity for generating folders in the card, it sends a MemoryOverflow message.

Parameters:

- DATA details
  - foldername: The folder name to be created 16B
  - folderACL: The folder access level 1B

Output:

SuccessfulFileOperation

- DATA details
  - MessageType: CreateFile MessageType 2B
  - CreatedfolderID: Identifier of created folder 2B

Exceptional Output:

- (Error message) (Cause)
  - AccessViolation: The IC card is in the LOCKED state/
    the authentication state corresponding to the transmission source is not
    the owner mode
  - IllegalParams: The LEN is different from the DATA size/
    there is a file with the same name
  - MemoryOverflow: There is no available card capacity
5.2.10 DeleteFolder

Message in which the AP specifies the deletion of a folder to the IC card.

When this message is generated, the AP specifies the identifier of the folder to be deleted, folderID, and the file deletion mode, mode. When mode=00h, it cannot delete the folder, when there are files located within the folder (abnormal end). When mode=01h, it deletes the respective files within the folder.

If the authentication state that is paired with the transmission source identifier is the owner mode, it deletes the file.

When the above process ends normally, the IC card sends to the transmission source AP a SuccessfulFolderOperation message, including the identifier of the deleted file, the deletedfolderID.

The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.

- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the transmission source is not authenticated as the owner, and when there are no files within the folder when mode=00h has been specified, and when files are being exchanged, it sends an AccessViolation message.
- When there is no folder indicated by Folderid, it sends an ObjectNotFound message.
- If the mode is neither 00h or 01h, it sends an IllegalParameters message.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>folderID</td>
<td>Folder identifier of the folders to be deleted</td>
<td>2B</td>
</tr>
<tr>
<td>mode</td>
<td>Folder deletion mode</td>
<td>1B</td>
</tr>
<tr>
<td></td>
<td>· mode=00h: When there are files in the folder, it ends abnormally</td>
<td></td>
</tr>
<tr>
<td></td>
<td>· mode=01h: When there are files, it deletes each file</td>
<td></td>
</tr>
</tbody>
</table>

**Output:**

SuccessfulFileOperation

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageType</td>
<td>DeleteFolder message MessageType</td>
<td>2B</td>
</tr>
<tr>
<td>deletedfolderID</td>
<td>Deleted folder identifier</td>
<td>2B</td>
</tr>
</tbody>
</table>

**Exceptional Output:**

AccessViolation

- The IC card is in the LOCKED state/
- the authentication state corresponding to the transmission source is not the owner mode/
- when mode=00h, there are files within the folder/
- files are being exchanged

ObjectNotFound

- The folder indicated by the folderID does not exist

IllegalParamets

- The LEN is different from the DATA size/
- it is neither 00h or 01h
5.2.11 RequestFolderList

Message in which the AP specifies the outputting of a folder list to the IC card.
The DATA, to be specified by the AP when this message is generated, is empty.
The IC card that accepts the message obtains the folder list within the card, without relying on the authentication state that is paired with the transmission source identifier.

When the above process ends normally, the IC card sends the FolderList message including the folder list to the transmission source.
The IC card sends the below messages to the transmission source, when the above processing has ended abnormally.
- When the sizes of the LEN and actual DATA are different, it sends an IllegalParameters message.

Parameters:

DATA details
None

Output:
FolderList
DATA details
foldernum Number of folders 2B
folderinfo Folder list (foldernum*19) B

folderinfo details
folderID Folder identifier 2B
foldername Folder name 16B
folderACL Folder ACL 1B

Exceptional Output:
(Errow message) (Cause)
AccessViolation - The IC card is in the LOCKED state
IllegalParameters - The LEN is different from the DATA size
5.2.12 RequestCardInfo

Message in which the AP specifies the outputting of a IC card information to the IC card.
The DATA, to be specified by the AP when this message is generated, is empty.
The IC card that accepts the message obtains the card information, without relying on the authentication state that is paired with the transmission source identifier.
When the above process normally ends, the IC card sends the CardInfo message including the card information to the transmission source.
The IC card sends the below message, when the above processing has ended abnormally.
- When the sizes of the LEN and actual DATA are different, it sends an IllegalParameters message.

Parameters:
- DATA details
  - None

Output:
CardInfo
- DATA details
  - ICCState: State of the ICC 1B
  - SignAlgorithm: Algorithm of signatures useable by the ICC 1B
  - KeyAlgorithm: Algorithm of keys useable by the ICC 1B
  - Certlen: Public key certificate length of the ICC 1B
  - Cert: Public key certificate of the ICC (Certlen) B
  - MaxFolderNum: Maximum number of folders that can be created 2B
  - MaxFileNum: Maximum number of files that can be created 2B
  - MaxFileSize: Maximum file size that can be created 2B
  - AuthMode: Current authentication mode of RequestCardInfo Src 2B

Exceptional Output:
- (Error message) (Cause)
  - IllegalParameters - The LEN is different from the DATA size
5.2.13 BackUpCard

Message in which the AP specifies the outputting of IC card information to the IC card.
The DATA, to be specified by the AP when this message is generated, is empty.

If the authentication state that is paired with the transmission source identifier is the owner mode, the IC card that accepts the message obtains the savefile information; generates a random number, random; generates backupinfo that is the value linking the random number with the file information to be saved and allocates a signature, sign. The IC card stores the random until the backup mentioned in section 3.5 is completed.

When the above process ends normally, the IC card sends BackUpInfo message including the backupinfo and sign to the transmission source.

The IC card sends the below message, when the above processing has ended abnormally.

- When the sizes of the LEN and actual DATA are different, it sends an IllegalParameters message.
- When the transmission source is not the owner mode, and when there are files being exchanged, it sends an AccessViolation message.
- When the size of the save file information exceeds the maximum transmittable size, it sends a MaximumSizeOverflow message.

Parameters:

DATA details
None

Output:
BackUpInfo
DATA details
backupinfo[n]  Length of the savefile information + Length of the random number 2B
signlen[n]    Signature length 2B
backupinfo    Save file information | random (backupinfo[n] B)
sign[n]       Signature for backupinfo by IC card (signlen[n] B)

Exceptional Output:
AccessViolation  - The IC card is in the LOCKED state/
the authentication state corresponding to the transmission source is not
the owner mode /
files are being exchanged

IllegalParams  - The LEN is different from the DATA size

MemoryOverflow  - The size of the save file exceeds the maximum transmittable size
5.2.14 ReformatCard

Message in which the AP specifies the resizing of the file domain to the IC card.
When this message is generated, the AP specifies the fileSize that is the size to be resized.
If the authentication state that is paired with the transmission source identifier is the owner mode, the IC card that accepts the message resizes the file domain.

When the above process ends normally, the IC card sends a SuccessfulOperation message to the transmission source AP.
The IC card sends the below messages, when the above processing has ended abnormally.
- When the sizes of the LEN and actual DATA are different, it sends an IllegalParameters message.
- When the authentication state that is paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When 0 has been input for the fileSize and the fileSize exceeds the buffer domain size, it sends a MemoryOverflow message.

Parameters:
DATA details
fileSize The size to be resized 2B

Output:
SuccessfulOperation
DATA details
MessageType MessageType of the ReformatCard message 2B

Exceptional Output:
(Error message) (Cause)
AccessViolation - The IC card is in the LOCKED state/
the authentication state corresponding to the transmission source is not
the owner mode
IllegalParameters - The LEN is different from the DATA size
MemoryOverflow - 0 is entered for the fileSize/
The fileSize exceeds the buffer domain size
5.2.15 RestoreCard

Message in which the AP specifies the restoring of written file information to the IC card.

When this message is generated, the AP specifies backupinfo, the save file information obtained by the BackUpInfo; backupinfolen, the file information length; sign, the signature, and signlen, the signature length.

If the authentication state that is paired with the transmission source identifier is the owner mode, verifies the signature, sign, and if it is verified as correct, crosschecks whether the random held by it is equivalent to the random number within the backupinfo, and if it is equivalent, it writes of backupinfo back into the memory domain of the IC card.

When the above process ends normally, the IC card sends a SuccessfulOperation message to the transmission source.

The IC card sends the below messages, when the above processing has ended abnormally.

- When the sizes of the LEN and actual DATA are different, it sends an IllegalParameters message.
- When the authentication state that is paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- If the random number held by the IC card upon when the BackUp message is executed is different from the random number within the file information, and if the verification of the sign has failed, it sends an IllegalParameters message.
- When the filelen within the file information exceeds the maximum storable file of the IC card, and when there is no available card capacity, it sends a MemoryOverflow message.

Parameters:

- Backupinfolen: File information length, 2B
- Signlen: Signature length, 2B
- backupinfo: File information, (backupinfolen) B
- sign: Signature by IC card, (signlen) B

Output:

SuccessfulOperation

- MessageType: Message Type of the RestoreCard message, 2B

Exceptional Output:

- AccessViolation
  - The IC card is in the LOCKED state/
  - the authentication state corresponding to the transmission source is not the owner mode

- IllegalParameters
  - The LEN is different from the DATA size/
  - the value of random number held by the IC card at the time of the BackUpCard is different from the random number within the file information/
  - sign verification has failed

- MemoryOverflow
  - The filelen, which is the file length within the file information, exceeds the maximum storable files of the IC card/
  - there is no vacant capacity

Copyright © 2005 by T-Engine Forum. All rights reserved.
5.3 Output messages from the IC card to the application

5.3.1 DelegatedID

Message in which the IC card issues an ID to the AP.
This message is generated as an output message to the RequestID message.
The IC card generates a message with the identifier AP_ID to be issued to the AP as the DATA.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Identifier to be issued to the AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP_ID</td>
<td>16B</td>
</tr>
</tbody>
</table>
5.3.2 Challenge

Message in which the IC card outputs challenge data to the AP.
This message is generated as an output message to the RequestChallenge message.
The IC card generates a message with the challengedata that is the challenge as the DATA.

Parameters:
- DATA details
  - challengedata  Challenge data  20B
5.3.3 AuthMode

Message in which the IC card outputs the authentication mode after the shift to the AP.
This message is generated as an output message to the Authenticate message.
The IC card generates a message with the mode that is the verification mode following the shift as the DATA.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Verification mode following the shift</th>
<th>2B</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>Shift to the owner mode: mode=0002h</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shift to the non-authentication mode: mode=0000h</td>
<td></td>
</tr>
</tbody>
</table>
5.3.4 SuccessfulFileOperation

Message in which the IC card notifies the file operation results for creating/deleting/moving files to the AP. This message is created as the output message for the CreateFile message, DeleteFile message and MoveFile message.

The IC card creates the message using as DATA the MessageType for the message that performed the processing; the fileID, the file identifier and the fileCnt, the number of files. The meaning of the fileCnt varies depending on the operation.

- It outputs the number of files created for the CreateFile message
- It outputs the number of files deleted for the DeleteFile message
- It outputs the current number of files in the movement destination folder for the moveFile message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageType</td>
<td>MessageType of the message that was processed 2B</td>
</tr>
<tr>
<td>fileID</td>
<td>Identifier of created/deleted/move destination files 2B</td>
</tr>
<tr>
<td>fileCnt</td>
<td>Number of created/deleted/move destination files 4B</td>
</tr>
</tbody>
</table>
5.3.5 SuccessfulFolderOperation

Message in which the IC card notifies the file operation results for folder creation/deletion to the AP. This message is created as the output message for the CreateFolder message, and DeleteFolder message. The IC card creates the message using as DATA the MessageType for message that performed the processing; the identifier of the folders that were created/deleted, the folderID.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Message details</th>
</tr>
</thead>
<tbody>
<tr>
<td>MessageType</td>
<td>MessageType of the message that was processed</td>
</tr>
<tr>
<td>folderID</td>
<td>Identifier of created/deleted files</td>
</tr>
</tbody>
</table>
5.3.6 SuccessfulOperation

Message in which the IC card notifies the normal end of the processing of input messages to the AP.
This message is created as the output message for the ReformatCard message, RestoreCard message and
CancelExchange message.
The IC card creates the message using as DATA the message type for the message that performed the
processing.

Parameters:
DATA details
MessageType MessageType of the message that was processed 2B
5.3.7 FileInfo

Message in which the IC card outputs file information to the AP. This message is created as the output message for the RequestFileInfo message.

The IC card creates the message using as DATA fileDATA, which is the file content; fileCnt, which is the number of files; fileACL, which is the file access level, issuerID which is the identifier of the file creator; readLen, which is the service of fileDATA, the file content that was read; and filelen, which is the actual file length.

**Parameters:**

DATA details
- `filelen` The data length of the file (the data length of the actual file) 2B
- `fileCnt` The number of files 4B
- `fileACL` The file access level 1B
- `issuerID` The eTRON ID of the file creator 16B
- `readLen` The size of the fileDATA (data length of the read files) 2B
- `fileDATA` The file content (readLen) B
5.3.8 FileList

Message in which the IC card outputs the file list to the AP.
This message is created as the output message for the RequestFileList message.
The IC card creates the message using as DATA filenum, which is the number of files within the folder list.
and the file information equivalent to the filenum.
The file information is the file identifier, fileID; the file content, fileDATA; number of files, fileCnt; the file
access level, fileACL; the issuer ID, which is the identifier of the file creator; the size of the fileDATA that
was read, readLen, and the filelen, which is the actual file length.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>filenum</td>
<td>The number of files managed by the file list</td>
<td>2B</td>
</tr>
<tr>
<td>fileinfo</td>
<td>List of files</td>
<td>filenum*(27+readlen) B</td>
</tr>
<tr>
<td>fileID</td>
<td>File identifier</td>
<td>2B</td>
</tr>
<tr>
<td>filelen</td>
<td>Actual file size</td>
<td>2B</td>
</tr>
<tr>
<td>filecnt</td>
<td>Number of files</td>
<td>2B</td>
</tr>
<tr>
<td>fileACL</td>
<td>File ACL</td>
<td>1B</td>
</tr>
<tr>
<td>issuerID</td>
<td>eTRON ID of the file creator</td>
<td>16B</td>
</tr>
<tr>
<td>readlen</td>
<td>Length of the read file data</td>
<td>2B</td>
</tr>
<tr>
<td>fileDATA</td>
<td>Read file data</td>
<td>(readlen) B</td>
</tr>
</tbody>
</table>
5.3.9 FolderList

Message in which the IC card outputs the folder list to the AP.
This message is created as the output message for the RequestFolderList message.
The IC card creates the message using as DATA foldernum, which is the number of folders and folder information equivalent to the foldernum.
The folder information is the folder identifier, folderID; the folder name, foldername and the folder access level, folderACL.

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Details</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>foldernum</td>
<td>The number of folders</td>
<td>2B</td>
</tr>
<tr>
<td>folderinfo</td>
<td>List of folders</td>
<td>(foldernum*19)B</td>
</tr>
<tr>
<td>folderID</td>
<td>Folder identifier</td>
<td>2B</td>
</tr>
<tr>
<td>foldername</td>
<td>Folder name</td>
<td>16B</td>
</tr>
<tr>
<td>folderACL</td>
<td>Folder ACL</td>
<td>1B</td>
</tr>
</tbody>
</table>
5.3.10 CardInfo

Message in which the IC card outputs the IC card information to the AP. This message is created as the output message for the RequestFolderList message.

The IC card creates the message using as DATA ICCState, which indicates permission/prohibition on writing to the card; the SignalAlgorithm, which is the signature algorithm enabling execution by the card; the KeyAlgorithm, which is the key algorithm enabling execution by the card; Cert, which is the public key certificate of the card; MaxFolderNum, the maximum number of creatable folders; MaxFileNum, the maximum number of creatable files; MaxFileSize, the maximum creatable file size, and authmode, which is the AP authentication mode of the transmission source of the RequestCardInfo message.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICCState</td>
<td>State of the ICC</td>
<td>1B</td>
</tr>
<tr>
<td>SignalAlgorithm</td>
<td>Signature algorithm usable by the ICC</td>
<td>1B</td>
</tr>
<tr>
<td>KeyAlgorithm</td>
<td>Key algorithm usable by the ICC</td>
<td>1B</td>
</tr>
<tr>
<td>Certlen</td>
<td>Public key certificate length of ICC</td>
<td>2B</td>
</tr>
<tr>
<td>Cert</td>
<td>Public key certificate of the ICC</td>
<td>(Certlen) B</td>
</tr>
<tr>
<td>MaxFolderNum</td>
<td>Maximum number of creatable folders</td>
<td>2B</td>
</tr>
<tr>
<td>MaxFileNum</td>
<td>Maximum number of creatable files</td>
<td>2B</td>
</tr>
<tr>
<td>MaxFileSize</td>
<td>Maximum creatable file size</td>
<td>2B</td>
</tr>
<tr>
<td>AuthMode</td>
<td>Current authentication mode of the Src for RequestCardInfo</td>
<td>2B</td>
</tr>
</tbody>
</table>
5.3.11 BackUpInfo

Message in which the IC card outputs the file information to be backed up to the AP. This message is created as the output message for the BackUpCard message.

The IC card creates the message using as DATA backupinfo, which is the linking data between the file information and the random number; the sign, the signature of the IC card for the backupinfo; the signlen, the signature length and the backupinfolen, the size of the backupinfo.

Parameters:

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>backupinfolen</td>
<td>The file information length + random number length</td>
<td>2B</td>
</tr>
<tr>
<td>signlen</td>
<td>Signature length</td>
<td>2B</td>
</tr>
<tr>
<td>backupinfo</td>
<td>Information length + random number length</td>
<td>(backupinfolen) B</td>
</tr>
<tr>
<td>sign</td>
<td>Signature for backupinfo by IC card</td>
<td>(signlen) B</td>
</tr>
</tbody>
</table>
5.3.12 UnsupportedMessage

Message in which the IC card notifies the AP that the message type of the input message is other than the specified value.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

**Parameters:**

- **DATA details**
  - **errorCode** Message that indicates the nature of error (information) 2B
  - **MessageType** Message Type of the message that was processed 2B
5.3.13 AccessViolation

Message in which the IC card notifies the AP that it will not permit access to the folders and files specified in the input message.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

Parameters:

- **DATA details**
  - errorCode: Message that indicates the nature of error (information) 2B
  - MessageType: MessageType of the message that was processed 2B
5.3.14 ObjectNotFound

Message in which the IC card notifies the AP that none of folders and files specified in the input message are present.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

**Parameters:**
- errorCode: Message that indicates the nature of error (information) 2B
- MessageType: MessageType of the message that was processed 2B
5.3.15 IllegalParameters

Message in which the IC card notifies the AP that the input message parameters are illegal. The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorCode</td>
<td>Message that indicates the cause of error (information)</td>
<td>2B</td>
</tr>
<tr>
<td>MessageType</td>
<td>Message Type of the message that was processed</td>
<td>2B</td>
</tr>
</tbody>
</table>
5.3.16 MemoryOverflow

Message in which the IC card notifies the AP that the input message exceeds the upper limit/ lower limit for the domain.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request

**Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorCode</td>
<td>Message that indicates the cause of error (information)</td>
<td>2B</td>
</tr>
<tr>
<td>MessageType</td>
<td>Message Type of the message that was processed</td>
<td>2B</td>
</tr>
</tbody>
</table>
5.3.17 MaximumNumberOfExceeded

Message in which the IC card notifies the AP that the number of files of the input message exceeds the upper limit/ lower limit.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

**Parameters:**

- **DATA details**
  - `errorCode` Message that indicates the cause of error (information) 2B
  - `MessageType` Message Type of the message that was processed 2B
5.3.18 MemorySizeOverflow

Message in which the IC card notifies the AP that the value of the message originally to be output exceeds the message size that can be sent and received.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request

Parameters:
- errorCode: Message that indicates the cause of error (information) 2B
- MessageType: MessageType of the message that was processed 2B
5.3.19 ExchangeSuspended

Message in which the IC card notifies the AP that the exchange has been interrupted in mid-process. The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request.

**Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorCode</td>
<td>Message that indicates the cause of error (information)</td>
<td>2B</td>
</tr>
<tr>
<td>MessageType</td>
<td>MessageType of the message that was processed</td>
<td>2B</td>
</tr>
</tbody>
</table>
5.3.20 IncompatibleStatus

Message in which the IC card notifies the AP that the exchange process specified by the value of the input message cannot be located and that the relationship between the input message and the current exchange state is not consistent.

The errorCode is the message (information) indicating the cause of the error and the content is the incorporated request

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
</tr>
</thead>
<tbody>
<tr>
<td>errorCode</td>
</tr>
<tr>
<td>Message that indicates the cause of error (information)</td>
</tr>
<tr>
<td>2B</td>
</tr>
<tr>
<td>MessageType</td>
</tr>
<tr>
<td>MessageType of the message that was processed</td>
</tr>
<tr>
<td>2B</td>
</tr>
</tbody>
</table>
6 Exchange Control Messages

This chapter stipulates the messages and the data formats for processing exchanges. Section 6.1 explains the details of the protocol enabling the exchange. The respective messages for exchange control are stipulated from section 6.2.

6.1 Exchange protocol details

This section explains the detailed operation of the ‘optimistic exchange protocol for authority values,’ which is the method, adopted for the safe and impartial exchange functions realized by the TENeT IC card, which incorporates a TENeT message within these specifications.

6.1.1 Cast of characters

The characters that execute the ‘optimistic exchange protocol for authority values’ are the owner of the authority value, the TENeT IC card in which the authority values is stored and the TTP (Trusted Third Party) that performs the role of a mediator server to assure impartiality.

Because exchange protocols are executed between two parties, there is a side that proposes the exchanging of the authority values and a proposal receiving side. The explanation in this chapter refers to the proposing side as A and the receiving side as B. There is a TENeT IC card and an application program on a portable terminal operated by the owner on both the proposing and the receiving side.

This chapter designates the TENeT IC card as ICC\textsubscript{X} and the application program as AP\textsubscript{X}. X indicates either A or B. The ICC\textsubscript{X}ID is the identifier for ICC\textsubscript{X}, AP\textsubscript{X}ID is the identifier for the application and ttpID is the identifier for the TTP.

Figure 1 shows the relationship among the characters.

6.1.2 Protocol procedure

These protocols are composed of a main protocol, an [M5] abort protocol, and a resolve[M6] protocol. This
section explains the protocol procedures. ICCA and ICCB respectively hold the session $S_A$ and $S_B$ sets that manage the status transitions during exchange, TTP holds the $S_{resolve}$ set that saves the resolve enabled exchange information and the $S_{abort}$ set that holds the abort enabled exchange information.

6.1.2.1 Main protocol

The exchange protocol normally completes the exchange processing by executing the main protocol. Here following is indicated the procedure for the main protocol.

1. APA generates ConditionData including the conditions of the authority value $v_1$ and conditions of the authority value to be exchanges.
2. APA generates the $\text{StartExchange} \{\text{APBID, ConditionData and ttpID}\}$ that is the exchange instruction message including the ConditionData and sends it to ICCA.
3. When accepting a StartExchange, ICCA generates the random number $n_1$, then generates the exchange proposal message, the $\text{Offer} \{\text{APBID, ConditionData and ttpID}\}$, which includes the $n_1$, ConditionData and ttpid and sends it to the APB.

The ICCA stores the $(\text{ConditionData, ttpID, n1 and tid})$ after sending the offer. The tid is the thread ID (see “e2TP message specifications”) included in the StartExchange header.
4. When the Offer is accepted, if the ConditionData included in the Offer has been agreed to, the APB defines the exchange content, in other words $v_1$ and $v_2$, based on the ConditionData conditions, and sends the $\text{AgreeExchange} \{\text{APAID, ttpID, v1, v2 and n1}\}$ to the ICCB.
5. The ICCB generates the random number, $n_2$; the $s_1:=h<\text{ttpid}|v_1|v_2|n_1>$, which is the identification information within the proposal and the $s_2:=h<n_2>$, which is the session information.

The ICCB signs it using the its own secret key for the linkage value $(s_1|s_2)$ of $s_1$ and $s_2$ and deletes (withdraws) $s_2$, and then generates the exchange consent message $\text{Agreement}: \{\text{ICCBID, APBID, v1, v2 (s1|s2)PkB, and CertB}\}$ and sends it to APA.

The APB stores the $(v_1, v_2, n_1, s_1, s_2$ and tid) in $S_B$ after sending the Agreement. Tid...
6. When the Agreement is received, if the APA confirms the ConditionData conditions were met for the exchange content $v_1$ and $v_2$ defined by APB, it sends the $\text{ConfirmExchange}: \{\text{ICCBID, APBID, v1, v2, (s1|s2)PkB’ CertB}\}$ to ICCA
7. When ConfirmExchange is received, if the signature of $(s1|s2)PKB$ is verified as correct, the ICCA generates the ttpid and n1 stored in $S_A$ and the hash value $h<\text{ttpid}|v_1|v_2|n_1>$ using the received $v_1$ and $v_2$, and then crosschecks it with the generated hash value.

If the hash value generated is equivalent to $s_1$, ICCA deletes $v_1$ and implements the signature for $s_2$ using its own secret key.

It sends the $\text{Confirmation}: \{\text{ICCAID, APBID, (s2)PkB’ CertA}\}$ to ICCB.

Once the Confirmation is sent, ICCA stores $(s_1, s_2)$ in $S_A$.
8. ICCB accepts the Confirmation.

If the signature verification of the accepted (s2)PKA is done correctly, the ICCB crosschecks it with the session information stored by s2 in SB.

If there is session information within SB equivalent to s2, ICCB obtains the v1 to be exchanged.

It generates a $\text{Commitment}: \{\text{APAID, n2}\}$, which concludes the exchange.

The ICCB deletes the stored value.

The ICCB sends the Commitment to the ICCA, and then completes the exchange processing.
9. The ICCB sends $\text{ExchangeCommitted}: \{ \}$, which notifies resolve of the exchange, to APB.
10. ICCA accepts the Commitment.
ICCA generates the n2 hash value for the accepted n2, and then crosschecks it with the session value stored in SA.

If there is session information within the SA equivalent to the hash value of n2, the ICCA stores the v2 that is to be exchanged, and then completes the exchange.

11. The ICCA sends ExchangeCommitted: { }, which notifies resolve of the exchange to the APA.

The ICCA deletes the value stored in S_A.

---

**Figure 3 Main protocol procedure**

Figure 2 shows the main protocol procedure. The Resolvable/Abortable interval of the red arrow section shown in Figure 2 is the biased state. Only one of the two implementing the exchange is in the state where it is receiving the authority value from the opposite party or in the state where it is deleting its own authority value. When the processing is interrupted in this interval, the respective APs implement recovery processing with the TTP, without restarting the communication with the opposite party, inducing the IC card to recover from the above state to the unbiased state where it is consistent with the state of the other party. In the recovery processing, the AP_X instructs the execution of the abort/resolve request protocol, and then performs the processing between the ICC_X and the TTP.

Specifically, when the Abortable interval (interval from the sending of the Agreement to the acceptance of the Confirmation) is optional, the ICC_B can execute the interruption request protocol with the TTP, and then interrupt the processing (restoration of the electronic value). Likewise, at the optional time of the Resolvable interval (interval between the generation of the Confirmation and the acceptance of the Commitment), the ICC_A can end the processing by executing a resolve request protocol with the TTP. The interruption request protocol and the resolve request protocol are explained from the following section.

### 6.1.2.2 Interruption request protocol

When it has ended abnormally, execution of the interruption request protocol causes the exchange protocol to end the exchange processing. The interruption request protocol procedure is shown in Figure 3.

1. AP_B uses the ExgThreadID, the identifier of the exchange whose processing is to be interrupted, to generate a RecoverExchange: {ExgThreadID}, which it sends to the ICC_B.
2. The ICC_B accepts the RecoverExchange and if there is a tid equivalent to the ExgThreadID in SB, it signs using its own secret key for the s2 corresponding to the exgthreadID within the SB and the linkage value (flag|s2) of the flag, which is the flag indicating the exchange interruption request, and generates an ArbitrationRequest: \{AP_BID, (flag|s2)PK_B, Cert_B\}, which is the exchange interruption request, and then sends it to the TTP, for which the ttpid corresponding to the exgthreadid within the SB is the identifier.

3. The TTP accepts the ArbitrationRequest and, if the signature is verified correctly, 
   - When there is no s2 in S_resolve,
     it signs using its own secret key for the flag that is the flag indicating the exchange interruption request and for the linkage value of the accepted s2 (flag|s2),
     and sends the Arbitration \{AP_BID, (flag|s2)PK_T, Cert_T\} that is the interruption permission to the ICC_B.
     TTP stores s2 in S_abort, after the sending of the Arbitration.
   - When there is an s2 in S_resolve,
     it signs using its own secret key for the flag that is the flag indicating the exchange resolve request and for the linkage value of the accepted s2 (flag|s2),
     and sends the Arbitration \{AP_BID, (flag|s2)PK_T, Cert_T\} that is the resolve permission to the ICC_B.

4. The ICC_B accepts the Arbitration \{(flag|s2)PK_T, Cert_T\}, and then if, after signature verification, there is session information within the SB equivalent to the S2,
   - If the flag indicates interruption permission,
     it sends to the AP_B ExchangeAborted: \{ \}, which notifies interruption of the exchange following restoration of the V2.
   - If the flag indicates resolve permission,
     it sends to the AP_A ExchangeCommitted: \{ \}, which notifies resolve of the exchange following restoration of the V1

The ICC_B deletes the value stored in the SB, after sending notification of the interruption/resolve of the exchange.

---

![Abort Request Protocol Procedure](image-url)

**Figure 3** Abort request protocol procedure

Copyright © 2005 by T-Engine Forum. All rights reserved.
6.1.2.3 Resolve request protocol

When it has ended abnormally, the exchange protocol ends the exchange processing, by executing a resolve request protocol. Figure 4 shows the resolve request protocol procedure.

1. AP\textsubscript{A} uses the ExgThreadID, the identifier of the exchange whose processing is to be interrupted, to generate a RecoverExchange: \{ExgThreadID\}, which it sends to the ICC\textsubscript{A}.

2. The ICC\textsubscript{A} accepts the RecoverExchange and if there is a tid equivalent to the ExgThreadID in SA, it signs using its own secret key for the s2 corresponding to the exgthreadID within the SB and the linkage value (flag|s2) of the flag, which is the flag indicating the exchange interruption request, and generates an ArbitrationRequest: \{AP\textsubscript{B}ID, (flag|s2)\textsubscript{PKA}, Cert\textsubscript{A}\}, which is the exchange interruption request, and then sends the ttpid corresponding to the exgthreadid within the SB to the TTP.

3. The TTP accepts the ArbitrationRequest and, if the signature is verified correctly,

- When there is no s2 in S\textsubscript{abort},
  
  it signs using its own secret key for the flag that is the flag indicating the exchange resolve request and for the linkage value of the accepted s2 (flag|s2),
  
  and sends the Arbitration \{AP\textsubscript{A}ID, (flag|s2)\textsubscript{PKT}, Cert\textsubscript{T}\} that is the resolve permission to the ICC\textsubscript{A}.

- When there is an s2 in S\textsubscript{abort},
  
  it signs using its own secret key for the flag that is the flag indicating the exchange interruption request and for the linkage value of the accepted s2 (flag|s2),
  
  and sends the Arbitration \{AP\textsubscript{A}ID, (flag|s2)\textsubscript{PKT}, Cert\textsubscript{T}\} that is the interruption permission to the ICC\textsubscript{A}.

4. The ICC\textsubscript{A} Arbitration \{(flag|s2)\textsubscript{PKT}, Cert\textsubscript{T}\}, and then if, after signature verification, there is session information within the S\textsubscript{A} equivalent to the S2,

- If the flag indicates interruption permission,
  
  it sends to the AP\textsubscript{A} ExchangeAborted: \{ \}, which notifies interruption of the exchange following restoration of the V1.

- If the flag indicates resolve permission,
  
  it sends to the AP\textsubscript{A} ExchangeCommitted: \{ \}, which notifies resolve of the exchange following restoration of the V2

The ICC\textsubscript{A} deletes the value stored in the S\textsubscript{A}, after sending notification of the interruption/resolve of the exchange.
6.2 State record/ state table

The IC cards of the exchange proposing side and receiving side manage the information to be used in the exchange processing as state records. The IC card manages the transition state and values by the state records shown in Table 11 for the exchange processing during execution. The IC card continuously arranges the state records on the memory, and then manages them as a state table.

The IC card releases the state records in accordance with the resolve of the exchange processing.

Table 12 shows the state values of the state records, indicating the transition of the state of the exchange control messages mentioned in section 5.1.2.

<table>
<thead>
<tr>
<th>#</th>
<th>Data</th>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>state</td>
<td>State of the exchange processing</td>
</tr>
<tr>
<td>2</td>
<td>extId</td>
<td>Thread ID during normal exchange processing</td>
</tr>
<tr>
<td>3</td>
<td>ttpID</td>
<td>eTRON ID of the TTP</td>
</tr>
<tr>
<td>4</td>
<td>s1</td>
<td>Hash value of the exchange content</td>
</tr>
<tr>
<td>5</td>
<td>s2</td>
<td>Hash value of the random number n2</td>
</tr>
<tr>
<td>6</td>
<td>n1/n2</td>
<td>Random number</td>
</tr>
<tr>
<td>7</td>
<td>folderID</td>
<td>Folder ID of the folder to be exchanged</td>
</tr>
<tr>
<td>8</td>
<td>fileCnt1</td>
<td>Number of exchange source files</td>
</tr>
<tr>
<td>9</td>
<td>fileACL1</td>
<td>File ACL of the exchange source</td>
</tr>
<tr>
<td>10</td>
<td>issuerID1</td>
<td>eTRON ID of the issuer of the exchange source file</td>
</tr>
<tr>
<td>11</td>
<td>fileLen1</td>
<td>Exchange source file length</td>
</tr>
<tr>
<td>12</td>
<td>fileData1</td>
<td>Exchange source file data</td>
</tr>
<tr>
<td>13</td>
<td>fileCnt2</td>
<td>Number of exchange destination files</td>
</tr>
<tr>
<td>14</td>
<td>fileACL2</td>
<td>Exchange destination file ACL</td>
</tr>
<tr>
<td>15</td>
<td>issuerID2</td>
<td>eTRON ID of the issuer of the exchange destination file</td>
</tr>
<tr>
<td>16</td>
<td>fileLen2</td>
<td>Exchange destination file length</td>
</tr>
<tr>
<td>17</td>
<td>fileData2</td>
<td>Exchange destination file data</td>
</tr>
<tr>
<td>#</td>
<td>Value of the state</td>
<td>Content of the state</td>
</tr>
<tr>
<td>----</td>
<td>--------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>None</td>
<td>Initial state (exchange not yet started)</td>
</tr>
<tr>
<td>2</td>
<td>Cancelable</td>
<td>Awaiting agreement state</td>
</tr>
<tr>
<td>3</td>
<td>Abortable</td>
<td>Awaiting confirmation state</td>
</tr>
<tr>
<td>4</td>
<td>Resolvable</td>
<td>Awaiting commitment state</td>
</tr>
<tr>
<td>5</td>
<td>Wait_abort</td>
<td>Awaiting interruption permission state</td>
</tr>
<tr>
<td>6</td>
<td>Wait_commit</td>
<td>Awaiting resolve permission state</td>
</tr>
</tbody>
</table>
6.3 Inputting messages from the application/IC card to the IC card

6.3.1 Main protocol messages

6.3.1.1 StartExchange

Message in which the AP_A indicates the start of the exchange to the ICC_A card.

When this message is input, the AP_A specifies AP_AID, the identifier of the AP_A of the opposite end of the exchange; ttpID, the identifier of the TTP and the ConditionData, the exchange conditions. (See section 6.1.2)

When the authentication state corresponding to the identifier of the transmission source of this message is the owner mode, and when there is no state record with the same state ID as the tid, which is the thread ID included in the e^2TP message, the ICC_A generates the random number n1, and then sends to the AP_B whose identifier is AP_BID, an Offer message including the AP_AID, the identifier of the transmission source; the ttpID, the identifier of the and ConditionData, the exchange conditions.

The ICC_A generates a new state record in which the state is Cancelable, and that has {ConditionData, ttpID, n1 and tid}.

The ICC_A generates a state record in which the state is Cancelable, after the outputting of the Offer message, and that has n1, tid, tppid and ConditionData.

When the above process ends abnormally, the ICC_A sends the below messages to the AP_A.

- When the sizes of the LEN and DATA included in the e^2TP message are different, it sends an IllegalParameters message.
- When the authentication state corresponding to the transmission source identifier is not the owner mode, it outputs an AccessViolation message.
- When there is no available domain for generating state records, it outputs a MemoryOverflow message.
- When there is already a state record with the same thread ID, it outputs an IncompatibleStatus message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>APAID</td>
<td>The identifier of the AP_B</td>
<td>16B</td>
</tr>
<tr>
<td>ttpID</td>
<td>The eTRON ID of the TTP</td>
<td>16B</td>
</tr>
<tr>
<td>ConditionDataSize</td>
<td>Condition data size</td>
<td>2B</td>
</tr>
<tr>
<td>ConditionData</td>
<td>Condition data</td>
<td>(ConditionDataSize) B</td>
</tr>
</tbody>
</table>

Output:

Offer

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>APAID</td>
<td>The identifier of the AP_A</td>
<td>16B</td>
</tr>
<tr>
<td>ttpID</td>
<td>The eTRON ID of the TTP</td>
<td>16B</td>
</tr>
<tr>
<td>ConditionDataSize</td>
<td>Condition data size</td>
<td>2B</td>
</tr>
<tr>
<td>ConditionData</td>
<td>Condition data</td>
<td>(ConditionDataSize) B</td>
</tr>
<tr>
<td>n1</td>
<td>Random number</td>
<td>20B</td>
</tr>
</tbody>
</table>

Exceptional Output:

<table>
<thead>
<tr>
<th>(Error message)</th>
<th>(Cause)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>- The IC card is in the LOCKED state/ the authentication state corresponding to the transmission source is not the owner mode /</td>
</tr>
<tr>
<td>IllegalParameters</td>
<td>- The LEN is different from the DATA size</td>
</tr>
<tr>
<td>MemoryOverflow</td>
<td>- There is no available card capacity</td>
</tr>
<tr>
<td>IncompatibleStatus</td>
<td>- A state record for the same thread ID already exists/</td>
</tr>
</tbody>
</table>
6.3.1.2 Offer

Message in which the ICC_A proposes exchange to the AP_B.

The ICC_A generates a message using as DATA the AP_A ID, the identifier StartExchange transmission source; the ttpID, the TTP identifier; ConditionData, the proposed content and n1, a random number.

After accepting the Offer message, when the AP_B consents to the proposed content that is the ConditionDATA, it outputs to the ICC_B an AgreeExchange message that inputs the Offer DATA value as is.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP_A ID</td>
<td>The identifier of the AP_A</td>
<td>16B</td>
</tr>
<tr>
<td>ttpID</td>
<td>The eTRON ID of the TTP</td>
<td>16B</td>
</tr>
<tr>
<td>ConditionDataSize</td>
<td>Condition data size</td>
<td>2B</td>
</tr>
<tr>
<td>ConditionData</td>
<td>Condition data</td>
<td>(ConditionDataSize) B</td>
</tr>
<tr>
<td>n1</td>
<td>Random number</td>
<td>20B</td>
</tr>
</tbody>
</table>
6.3.1.3 AgreeExchange

Message in which the APB specifies the start of the exchange to the ICCA.

When this message is input, the APB specifies APBID, the identifier of the StartExchange; ttpID, the identifier of the TTP; the exchange content specified by the APB (v1, the file content of exchange proposal source; num1, the number; size1, the size; acl1, the access level; issuerID1, the creator identifier; folderID1, the identifier of the v1 storage source folder; Size2, the size; acl2, the access level2; issuerID2, the creator identifier; and folderID2, the identifier of v2 storage source folder) and n1, a random number.

When the authentication state corresponding to the identifier of the transmission source of this message is the owner mode, and when there is no state record with the same state ID as the tid, which is the thread ID included in the eTP message, and when the transfer privilege of v2 has been permitted, the ICCA generates the random number, n2, and then generates the proposal content identification information s1:=h<v1|v2|n1 and session information s2:=h<n2>.

The ICCB deletes (withdraws) the v2 information from the folder, and then outputs an Agreement message that includes msg:= (s1|s2) which links s1 and s2; sign:=(s1|s2)PKB, the signature for the msg; CertB, the public key certificate; the information for v1 and v2; and ICCBID, and APBID, the identifiers for the exchange receiving side

The ICCB generates a state record in which the state is Abortable, after outputting Agreement, and that has {n2, s1, s2, tid, ttpID and the information for v1 and v2}.

When the above process ends abnormally, the ICCB sends the below messages to the APB.
- When the sizes of the LEN and DATA are different, and when both 0 has been input for both num1 and num2, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When there is no available domain for generating state records, it sends a MemoryOverflow message.
- When there is already a state record with the same thread ID, it outputs an IncompatibleStatus message.
- When the indicated folder and file do not exist, it sends an ObjectNotFound message.
- When the number of v2 is less than num2, it sends a MaximumNumberOfExceeded message

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
</tr>
</thead>
<tbody>
<tr>
<td>APAID</td>
</tr>
<tr>
<td>ttpID</td>
</tr>
<tr>
<td>folderID1</td>
</tr>
<tr>
<td>folderID2</td>
</tr>
<tr>
<td>num1</td>
</tr>
<tr>
<td>acl1</td>
</tr>
<tr>
<td>issuerID1</td>
</tr>
<tr>
<td>size1</td>
</tr>
<tr>
<td>v1</td>
</tr>
<tr>
<td>num2</td>
</tr>
<tr>
<td>acl2</td>
</tr>
<tr>
<td>issuerID2</td>
</tr>
<tr>
<td>size2</td>
</tr>
<tr>
<td>v2</td>
</tr>
<tr>
<td>n1</td>
</tr>
</tbody>
</table>

Output:
Agreement
### DATA details

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICC ID</td>
<td>eTRON ID of the recipient side ICC</td>
<td>16B</td>
</tr>
<tr>
<td>AP ID</td>
<td>The identifier of the recipient side AP</td>
<td>16B</td>
</tr>
<tr>
<td>msglen</td>
<td>msg length</td>
<td>2B</td>
</tr>
<tr>
<td>signlen</td>
<td>sign length</td>
<td>2B</td>
</tr>
<tr>
<td>Certlen</td>
<td>Recipient side public key certificate size</td>
<td>2B</td>
</tr>
<tr>
<td>msg</td>
<td>(s1</td>
<td>s2) intended for the signature</td>
</tr>
<tr>
<td>sign</td>
<td>signature</td>
<td>(signlen) B</td>
</tr>
<tr>
<td>Cert B</td>
<td>Recipient side public key certificate</td>
<td>(Certlen) B</td>
</tr>
<tr>
<td>num1</td>
<td>Number of v1 files</td>
<td>4B</td>
</tr>
<tr>
<td>acl1</td>
<td>ACL of v1 files</td>
<td>1B</td>
</tr>
<tr>
<td>issuerid1</td>
<td>Issuer ID of v1 file</td>
<td>16B</td>
</tr>
<tr>
<td>size1</td>
<td>File size of v1 file</td>
<td>2B</td>
</tr>
<tr>
<td>v1</td>
<td>Body of the v1 file</td>
<td>(size1) B</td>
</tr>
<tr>
<td>num2</td>
<td>Number of v2 files</td>
<td>4B</td>
</tr>
<tr>
<td>acl2</td>
<td>ACL of v2 files</td>
<td>1B</td>
</tr>
<tr>
<td>issuerid2</td>
<td>Issuer ID of v2 file</td>
<td>16B</td>
</tr>
<tr>
<td>size2</td>
<td>File size of v2 file</td>
<td>2B</td>
</tr>
<tr>
<td>v2</td>
<td>Body of the v2 file</td>
<td>(size2) B</td>
</tr>
</tbody>
</table>

### Exceptional Output:

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>The IC card is in the LOCKED state/</td>
</tr>
<tr>
<td></td>
<td>the authentication state corresponding to the transmission source</td>
</tr>
<tr>
<td></td>
<td>is not the owner mode /</td>
</tr>
<tr>
<td></td>
<td>The transfer authority of the file indicated in v2 has not been</td>
</tr>
<tr>
<td></td>
<td>approved</td>
</tr>
<tr>
<td>ObjectNotFound</td>
<td>The folder indicated by the folderid is not present/</td>
</tr>
<tr>
<td></td>
<td>The file indicated by {acl2, issuerid2, size2, v2} is not present</td>
</tr>
<tr>
<td>IlleagalParams</td>
<td>The LEN is different from the DATA size/</td>
</tr>
<tr>
<td></td>
<td>0 is input for both num1 and num2 **</td>
</tr>
<tr>
<td>MemoryOverflow</td>
<td>size1, which is the v1 file size, exceeds the maximum storable file</td>
</tr>
<tr>
<td></td>
<td>length/</td>
</tr>
<tr>
<td></td>
<td>There is no available card capacity</td>
</tr>
<tr>
<td>MaximumNumberExceeded</td>
<td>The Cnt of the specified v2 is less than num2</td>
</tr>
<tr>
<td>IncompatibleStatus</td>
<td>The same thread ID already exists</td>
</tr>
</tbody>
</table>
6.3.1.4 Agreement

Message in which the ICCB indicates agreement with the proposed content of the exchange to the AP A.

The ICCB generates a message using as DATA APBID, the identifier transmission source of AgreeExchange; ICCBID, the identifier of the transmission source of this message; msg, the value linking the s1 and s2, the identifiers of the exchange; sign, the signature allocated to the msg by ICCB; CertB, the public key certificate of ICCB and the information of files v1 and v2 to be exchanged.

When the AP A confirms that the content of the defined v1 and v2 meet the ConditionData conditions, it sends to the ICCA a ConfirmExchange message that inputs the Agreement DATA as is.

Parameters:

- **DATA details**
  - ICCBID: The eTRON ID of the ICCB (16B)
  - APBID: The identifier of the AP B (16B)
  - msglen: msg length (2B)
  - signlen: sign length (2B)
  - Certlen: Recipient side public key certificate size (2B)
  - Msg: (s1|s2) intended for the signature (msglen) B
  - sign: signature (signlen) B
  - CertB: Recipient side public key certificate (Certlen) B
  - num1: Number of v1 files (4B)
  - acl1: ACL of v1 files (1B)
  - issuerid1: Issuer ID of v1 file (16B)
  - size1: File size of v1 file (2B)
  - v1: Body of the v1 file (size1) B
  - num2: Number of v2 files (4B)
  - acl2: ACL of v2 files (1B)
  - issuerid2: Issuer ID of v2 file (16B)
  - size2: File size of v2 file (2B)
  - v2: Body of the v2 file (size2) B
6.3.1.5 ConfirmExchange

Message in which the AP_A specifies the continuation of the exchange to the ICC_A. When this message is input, the AP_A specifies AP_BID, the identifier transmission source of AgreeExchange; ICC_BID, the identifier of the transmission source of the Agreement message; msg, the s1 and s2 linkage data; sign, the signature allocated to the msg; and the v1 and v2 file information.

The ICC_A accepts the message only when the authentication state corresponding to the identifier of the transmission source of this message is the owner mode, when there is no state record with the same state ID as the tid, which is the thread ID included in the e²TP message, and if the v1 file transfer privilege is permitted.

The ICC_A verifies the sign, the signature allocated to the msg. If the signature is verified to be correct, it picks out s1 from the msg and generates the ttpid and n1 stored in the state record and h<ttpid|v1|v2|n1>, the hash value using the received v1 and v2, and then crosschecks the generated hash values with s1.

If the crosscheck shows it to be correct, it deletes (withdraws) v1 from the folder, and sends a Confirmation to ICC_B including the sign, the signature allocated to S1 and S2, Cert_A, the public key certificate of the ICC_A; the AP_BID, the identifier of the transmission source for this message and the AP_AID, the identifier of the AgreeExchange transmission source.

Following the outputting of the Confirmation, the ICC_A updates the state value of the state record with the thread ID equivalent to tid from Cancelable to Resolvable.

The ICC_A updates the state of the state record including the tid from Cancelable to Resolvable, and stores the v1 and v2 file information and the s1 and s2 exchange identifiers.

When the above process ends abnormally, the ICC_A sends the below messages to the AP_A.
- When the sizes of the LEN and DATA are different,
  when 0 has been input for both num1 and num2,
  when the authentication state paired with the transmission source identifier is not the owner mode,
  when there is no specified folder and file, and
  when the signature allocation or signature verification has failed,
  the ICC_A sends an ExchangeSuspended message.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICC_BID</td>
<td>eTRON ID ICC_B</td>
<td>16B</td>
</tr>
<tr>
<td>AP_BID</td>
<td>The AP_B of the eTRON ID</td>
<td>2B</td>
</tr>
<tr>
<td>msglen</td>
<td>msg length</td>
<td>2B</td>
</tr>
<tr>
<td>signlen</td>
<td>signature length</td>
<td>2B</td>
</tr>
<tr>
<td>certlen</td>
<td>Recipient side public key certificate size</td>
<td>2B</td>
</tr>
<tr>
<td>msg</td>
<td>(s1</td>
<td>s2) intended for the signature</td>
</tr>
<tr>
<td>sign</td>
<td>signature</td>
<td>(signlen) B</td>
</tr>
<tr>
<td>Cert_B</td>
<td>Recipient side public key certificate</td>
<td>(Certlen) B</td>
</tr>
<tr>
<td>folderID1</td>
<td>Folder ID of the v1 storage source</td>
<td>2B</td>
</tr>
<tr>
<td>folderID2</td>
<td>Folder ID of the v2 storage destination</td>
<td>2B</td>
</tr>
<tr>
<td>num1</td>
<td>Number of v1 files</td>
<td>4B</td>
</tr>
<tr>
<td>acl1</td>
<td>ACL of v1 files</td>
<td>1B</td>
</tr>
<tr>
<td>issuerID1</td>
<td>Issuer ID of v1 file</td>
<td>16B</td>
</tr>
<tr>
<td>size1</td>
<td>File size of v1 file</td>
<td>2B</td>
</tr>
<tr>
<td>v1</td>
<td>Body of the v1 file</td>
<td>(size1) B</td>
</tr>
<tr>
<td>num2</td>
<td>Number of v2 files</td>
<td>4B</td>
</tr>
<tr>
<td>acl2</td>
<td>ACL of v2 files</td>
<td>1B</td>
</tr>
</tbody>
</table>

Copyright © 2005 by T-Engine Forum. All rights reserved.
issuerid2  Issuer ID of v2 file   16B
size2    File size of v2 file   2B
v2       Body of the v2 file   (size2) B

Output:
Agreement
DATA details
OriginAPID   eTRON ID of the Originator side ICC   16B
RecipientAPID eTRON ID of the Recipient side ICC   16B
msglen     msg length     2B
signlen     sign length     2B
certlen    Originator side public key certificate size  2B
msg         s2 intended for the signature     (msglen) B
sign        signature     (signlen) B
CertA       Originator side public key certificate     (certlen) B

Exceptional Output:
(Error message)  (Cause)
ExchangeSuspended -The IC card is in the LOCKED state/
The LEN is different from the DATA size•
The authentication state corresponding to the transmission source is not
the owner mode /
The transfer authority of the v1 file has not been approved/
The folder indicated in folderid1 does not exist/
The file that is {acl1, issuerid1, size1, v1} does not exist/
0 is input for both num1 and num2/
The Cert format is different/
The Cert verification has failed/
The signature verification has failed/
6.3.1.6 Confirmation

Message in which the ICC\(_A\) indicates its agreement to the defined exchange content to the ICC\(_A\).

The ICC\(_A\) generates this message using as DATA the AP\(_A\)ID, the identifier of the ConfirmExchange transmission source; the AP\(_B\)ID, the identifier of the AgreeExchange transmission source; the msg, which includes the s\(_2\), the exchange identifier; and the sign, the signature for the msg.

If the state value of the state record that has the same thread ID as the tid that is the thread ID of the e\(^2\)TP message is Abortable, the ICC\(_B\) verifies the sign, the signature allocated to the msg. If the signature is verified as correct, the ICC\(_B\) acquires vi, and then sends a Commitment to ICC\(_A\). It also sends ExchangedCommitted, which notifies resolve of the exchange from ICC\(_B\) to AP\(_B\).

The ICC\(_B\) deletes the state record including the tid (It updates the state value to None.)

After sending Commitment/ExchangeCommitted, the ICC\(_B\) releases a state record that has the same thread ID as the tid.

When the above process ends abnormally, the ICC\(_B\) sends the below messages to the AP\(_B\).

- When the sizes of the LEN and DATA are different,
- when the Parameter value is different from the corresponding data size,
- when the signature verification has failed,
- when there is already a v\(_1\) in the ICC\(_B\) and the addition result of num\(_1\) exceeds the upper limit,
- when there is no v\(_1\) in ICC\(_B\), and there is no available capacity in the file,

the ICC\(_B\) sends an ExchangeSuspended message.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Description</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP(_A)ID</td>
<td>The identifier of the AP(_A)</td>
<td>16B</td>
</tr>
<tr>
<td>AP(_B)ID</td>
<td>The identifier of the AP(_B)</td>
<td>16B</td>
</tr>
<tr>
<td>msglen</td>
<td>msg length</td>
<td>2B</td>
</tr>
<tr>
<td>signlen</td>
<td>sign length</td>
<td>2B</td>
</tr>
<tr>
<td>certlen</td>
<td>Originator side public key certificate size</td>
<td>2B</td>
</tr>
<tr>
<td>msg</td>
<td>s(_2) intended for the signature</td>
<td>(msglen) B</td>
</tr>
<tr>
<td>sign</td>
<td>signature</td>
<td>(signlen) B</td>
</tr>
<tr>
<td>CertA</td>
<td>Originator side public key certificate</td>
<td>(certlen) B</td>
</tr>
</tbody>
</table>

**Output:**

- ExchangeCommitted - Sends to ICC\(_A\)
- Commitment - Sends to AP\(_B\)

**DATA details**

- None

- AP\(_A\)ID: The identifier of the AP\(_A\) 16B
- N2: Random number 20B
Exceptional Output:

(Error message)  (Cause)
ExchangSuspended
- The IC card is in the LOCKED state/
  The authentication state corresponding to the transmission source is not
  the owner mode/
  The LEN is different from the DATA size/
  The msglen, siglen and certlen values are different from the data/
  The Cert format is different/
  The Cert verification has failed/
  The signature verification has failed/
  The addition result of file v1 exceeds FFFFFFFFh/
  There is no available card capacity/
  There is no state record with an equivalent thread ID
  The received s2 does not match the s2 within the state record with an
  equivalent thread ID/
6.3.1.7 Commitment

Message in which the ICC_B notifies the acquisition of v1 to the ICC_A. The ICC_B generates this message using as DATA the APAID, the identifier of the StartExchange transmission source and n2, a random number.

When the state value of the state record with the same ExgThreadId as the tid that is the thread ID of the e^{2}TP message is Resolvable, it crosschecks whether the s2 within the state record is equivalent to the h<n2>, the hash value of n2.
If it is equivalent, it acquires v2, and then sends ExchangeCommitted, which notifies resolve of the exchange, to the AP_A.
The ICC_A deletes the state record that includes the tid (It updates the state value).

When the above process ends abnormally, the ICC_B sends the below messages to the AP_B.
- When the sizes of the LEN and DATA are different,
  - when there is already a v2 in the ICC_B and the addition result of num2 exceeds the upper limit,
  - when there is no v2 in ICC_B, and there is no available capacity in the file,
    the ICC_B sends an ExchangeSuspended message.

**Parameters:**
DATA details
- APAID  The identifier of the APA 16B
- n2  Random number n2 20B

**Output:**
- ExchangeCommitted
  DATA details
    None

**Exceptional Output:**
- ExchangeSuspended
  (Error message)  (Cause)
    - The IC card is in the LOCKED state/
    - The LEN is different from the DATA size/
    - The addition result of number of v2 files exceeds FFFFFFFFh/
    - There is no available card capacity
6.3.2 Interruption request/resolve request protocol message

6.3.2.1 RecoverExchange

Message in which the AP\textsubscript{X} indicates an abort request/resolve request to the ICC\textsubscript{X}.

If the authentication state corresponding to the transmission source identifier is the owner mode, the ICC\textsubscript{X} processes it according to the state value of the state record that includes the ExgThreadID.

If the state is Cancelable, the ICC\textsubscript{X} sends an ExchangeAborted message to the transmission source AP.

Following the ExchangeAborted message, the ICC\textsubscript{X} releases the state record that includes the ExgThreadID.

If the state is either Abortable or Wait_Abort, the ICC\textsubscript{X} executes the abort request protocol. If the state is Resolvable or Wait_Commit, it executes the resolve request protocol. (See section 6.1.2)

The ICC\textsubscript{X} sends an ArbitrationRequest message that includes the msg linking the s2, session identifier, with the flag, the flag that identifies the abort request and resolve request (flag| s2); the sign, the signature for the msg; the identifier of the transmission source for this message and the CertX, public key certificate of the ICC\textsubscript{X}.

When the above process ends abnormally, the ICC\textsubscript{X} sends the below messages to the AP\textsubscript{X}.
- When the sizes of the LEN and DATA are different,
- when the SrcID has not been authenticated as the owner,
- when there is no state record with the same thread ID as the received ExgThreadID,
- when the state value is a value other than the specified value,
the ICC\textsubscript{X} sends an ExchangeSuspended message.

Parameters:

\begin{itemize}
  \item DATA details
    \begin{itemize}
      \item ExgThreadID: ExgThreadID for the exchange content to be recovered (16B)
    \end{itemize}
\end{itemize}

Output:

ArbitrationRequest

\begin{itemize}
  \item DATA details
    \begin{itemize}
      \item RecoverAPID: Identifier of the RecoverExchange (16B)
      \item msglen: msg length (2B)
      \item signlen: sign length (2B)
      \item certlen: Public key certificate size of X (2B)
      \item msg: (flag| s2) intended for the signature (msglen) B
      \item sign: signature (signlen) B
      \item CertX: Public key certificate of X (certlen) B
    \end{itemize}
\end{itemize}

\(^(*)\) Concerning the flag

flag = 0: Abort request
flag = 1: Resolve request

ExchangeAborted

\begin{itemize}
  \item DATA details
  \begin{itemize}
    \item None
  \end{itemize}
\end{itemize}
Exceptional Output:

(Error message) ExchagSuspended (Cause)
- The IC card is in the LOCKED state/ The LEN is different from the DATA size/
- The authentication state corresponding to the transmission source is not the owner mode/
- There is no state record with the same thread ID as the ExgThreadID
- The state value is a value other than the stipulated value/
6.3.2.2 ArbitrationRequest

Message in which the ICC_X indicates an abort request/resolve request to the ICC_X.

The ICC_X specifies the AP_XID, the identifier of the RecoverExchange transmission source; the s2, the exchange identifier; the identifier of the message transmission source; the msg, the value linking the session information, s2 with the flag, the flag that classifies the abort request/recovery request; sign, the signature for the msg and CertX, the public key certificate of the ICC_X.

Following receipt of this message, the TTP sends an Arbitration message to the ICC_XID, the identifier of the ArticleRequest transmission source.

**Parameters:**

- **DATA details**
  - RecoverAPID: SrcID of the RecoverExchange 16B
  - msglen: msg length 2B
  - signlen: sign length 2B
  - certlen: Public key certificate size of X 2B
  - msg: (flag(*)|s2) intended for the signature (msglen) B
  - sign: signature (signlen) B
  - CertX: Public key certificate of X (certlen) B

(*) Concerning the flag

- flag = 0: Abort request
- flag = 1: Resolve request
6.3.2.3 Arbitration

Message in which the TTP assigns abort permission or resolve permission to the ICC_x.

The TTP creates the message using as DATA the AP_x ID, the identifier of the RecoverExchange; the msg, the linkage value of the flag classifying the abort request/recovery request with the s2, the session identifier and the sign, the signature for the msg.

When the state value of the state record that includes s2 is Wait_Commit and Wait_Abort, it verifies the sign for the signature.

When the sign for the signature is verified as correct,
if the flag is for abort permission,
- if it is ICC_A, it restores the retired v2, and then sends ExchangeAborted to the AP_A.
  After sending the ExchangeAborted message, it releases the state record that includes the s2.
- if it is ICC_B, it restores the retired v2, and then sends ExchangeAborted to the AP_B.
  After sending the ExchangeAborted message, it releases the state record that includes the s2.

if the flag is for resolve permission,
- if it is ICC_A, it acquires v2, and then sends ExchangeCommitted to the AP_A.
  After sending the ExchangeCommitted message, it releases the state record that includes the s2.
- if it is ICC_B, it acquires v1, and then sends ExchangeCommitted to the AP_B.
  After sending the ExchangeCommitted message, it releases the state record that includes the s2.

After sending ExchangeCommitted/ExchangeAborted, it releases the state record with the same ThreadID as the ThreadID of the e2TP message.

When the above process ends abnormally, the ICC_B sends the below messages to the AP_B.
- When the sizes of the LEN and DATA are different, it sends an ExchangeSuspended message.
- When there is no state record that includes s2,
  if the state value of the state record that includes s2 is neither Wait_Commit or Wait_Abort, it sends an IncompatibleStatus message.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Identifier of the RecoverExchange transmission source</th>
<th>16B</th>
</tr>
</thead>
<tbody>
<tr>
<td>RecoverAPID</td>
<td>msg length</td>
<td>2B</td>
</tr>
<tr>
<td>msglen</td>
<td>sign length</td>
<td>2B</td>
</tr>
<tr>
<td>signlent</td>
<td>Public key certificate size of TTP</td>
<td>2B</td>
</tr>
<tr>
<td>certlent</td>
<td>(flag(*)</td>
<td>s2) intended for the signature</td>
</tr>
<tr>
<td>msg</td>
<td>signature</td>
<td>(signlent) B</td>
</tr>
<tr>
<td>sign</td>
<td>CertT</td>
<td>(certlent) B</td>
</tr>
<tr>
<td>CertT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(*) Concerning the flag
flag = 0: Abort request
flag = 1: Resolve request

**Output:**

ExchangeCommitted
- DATA details
  None

Exceptional Output:
- DATA details
  None
Exceptional Output:

(Error message) (Cause)
ExchangSuspended - The IC card is in the LOCKED state/
  The authentication state corresponding to the transmission source is not
  the owner mode/
  The LEN is different from the DATA size
IncompatibleStatus  - There is no exchange processing/ The mid-exchange state is illegal
6.3.2.4 CancelExchange

Message in which the APₐ indicates to the ICCₓ abort of the exchange processing where the state is the Cancelable interval.

When this message is input, the APₐ specifies ChkThreadID, which is thread ID for the exchange processing to be interrupted.

When the processing ends normally, the ICCₐ sends an ExchangeAborted. After sending the ExchangeAborted, the ICCₐ releases the state record that includes the ChkThreadID.

If the LEN and DATA sizes are different, the ICCᵦ sends an AccessViolation message.

If there is no state record that includes the same thread ID as the ChkThreadID, and
If the state of the state record that includes the same thread ID as the ChkThreadID is not Cancelable, it sends an IncompatibleStatus message.

Parameters:

- DATA details
  - ChkThreadID  ThreadID of the exchange processing to be interrupted  16B

Output:

- ExchangeAborted
- DATA details
  - MessageType  MessageType of the message that was processed  16B

Exceptional Output:

- (Error message) (Cause)
  - AccessViolation
    - The IC card is in the LOCKED state/
    - The authentication state corresponding to the transmission source is not the owner mode/
  - IncompatibleStatus
    - There is no state record that includes a ChkThreadID/
    - The state is not Cancelable
  - IllegalParameters
    - The LEN is different from the DATA size
6.3.3 Exchange state management messages

6.3.3.1 RequestExgStatusInfo

Message in which the AP$_X$ acquires the exchange processing information, which has the thread ID specified for the ICC$_X$ as the identifier.

When this message is input, it specifies the ChkThreadID, which is the thread ID that performs the search.

If the authentication state corresponding to the transmission identifier is the owner mode, ICC$_X$ sends the ExgStatusInfo that includes v1 and v2 files to be exchanged; the state, the exchange state, and the ExgthreadID that is the thread ID. The data included in the ExchangeStatusInfo varies when the state is Cancelable and when it is otherwise.

When the above process ends abnormally, the ICC$_X$ sends the below messages to the AP$_X$.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When the state record indicated by the ChkThreadID is not there, it sends an ObjectNotFound message.

Parameters:

<table>
<thead>
<tr>
<th>DATA details</th>
<th>ChkThreadID</th>
<th>ThreadID to be searched</th>
<th>20B</th>
</tr>
</thead>
</table>

Output:

ExgStatusInfo

<table>
<thead>
<tr>
<th>DATA details</th>
<th>status</th>
<th>state of the searched exchange content</th>
<th>1B</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExgThreadID</td>
<td>ttpID</td>
<td>ttpid of the searched exchange content</td>
<td>16B</td>
</tr>
<tr>
<td>folderID1</td>
<td>folderID1 of the searched exchange content</td>
<td>2B</td>
<td></td>
</tr>
<tr>
<td>folderID2</td>
<td>folderID2 of the searched exchange content</td>
<td>2B</td>
<td></td>
</tr>
<tr>
<td>ConditionDataSize</td>
<td>Condition data size</td>
<td>2B</td>
<td></td>
</tr>
<tr>
<td>ConditionData</td>
<td>Condition data</td>
<td>(ConditionDATASize) B</td>
<td></td>
</tr>
</tbody>
</table>

(When state = Cancelable)

status | state of the searched exchange content | 1B |
ExgThreadID | Thread ID of the searched exchange content | 20B |
ttpID | ttpid of the searched exchange content | 16B |
folderID1 | folderID1 of the searched exchange content | 2B |
folderID2 | folderID2 of the searched exchange content | 2B |
num1 | num of v1 of the searched exchange content | 4B |
acl1 | acl of v1 of the searched exchange content | 1B |
issuerid1 | issuerid of v1 of the searched exchange content | 16B |
size1 | size of v1 of the searched exchange content | 2B |
v1 | Content of v1 of the searched exchange content | (size1) B |
num2 | num of v2 of the searched exchange content | 4B |
acl2 | acl of v2 of the searched exchange content | 1B |
issuerid2 | issuerid of v2 of the searched exchange content | 16B |
size2 | size of v2 of the searched exchange content | 2B |
v2 | Content of v2 of the searched exchange content | (size2) B |
### Exceptional Output:

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessViolation</td>
<td>- The IC card is in the LOCKED state/ The authentication state corresponding to the transmission source is not the owner mode/</td>
</tr>
<tr>
<td>ObjectNotFound</td>
<td>- There is no state record that is indicated by the ChkThreadID</td>
</tr>
<tr>
<td>IllegalParameters</td>
<td>- The LEN is different from the DATA size</td>
</tr>
</tbody>
</table>
6.3.3.2 RequestExgStatusList

Message in which the APx requests the sending of the thread ID from the ICCx.
The value that the APx specifies when this message is created is empty.

If the authentication state corresponding to the transmission source identifier is the owner mode, the ICCx sends the ExgStatusList, a list pairing the state, which is the exchange state, with the ExgthreadID, which is the thread ID during exchange.

When the above process ends abnormally, the ICCx sends the below messages to the APx.
- When the sizes of the LEN and DATA are different, it sends an IllegalParameters message.
- When the authentication state paired with the transmission source identifier is not the owner mode, it sends an AccessViolation message.
- When the state record including the ChkThreadID is not there, it sends an ObjectNotFound message.

Parameters:

DATA details
None

Output:

ExgStatusList

DATA details
ThreadIDList Thread ID list
ThreadIDNum Number of exchange items 2B
ThreadIDInfo List of exchange items
state State control parameters 1B
ExgThreadID Thread ID during normal exchange processing 20B

Exceptional Output:

(Error message) (Cause)
AccessViolation -The IC card is in the LOCKED state/
The authentication state corresponding to the transmission source is not the owner mode
IllegalParameters -The LEN is different from the DATA size
6.4 Output messages from IC cards to applications

6.4.1 ExchangeCommitted

Message in which the ICCₓ notifies the APₓ that the exchange processing has been completed. The ICCₓ generates a message with data that is empty.

Parameters:

- DATA details
  - None
6.4.2 ExchangeAborted

Message in which the ICC$_X$ notifies the AP$_X$ that the exchange processing has been interrupted. The ICC$_X$ generates a message with data that is empty.

**Parameters:**
- DATA details
- None
6.4.3 **ExgStatusInfo**  
Message in which the ICC\textsubscript{X} notifies the AP\textsubscript{X} of the state of the exchange processing.  
This message is generated as an output message for the RequestStatusInfo message.  
The ICC\textsubscript{X} creates the message using as DATA the exchange processing information containing the searched ThreadID.

**Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>status</strong></td>
<td>state of the searched exchange content</td>
<td>1B</td>
</tr>
<tr>
<td><strong>ExgThreadID</strong></td>
<td>ExgThreadID of the searched exchange content</td>
<td>20B</td>
</tr>
<tr>
<td><strong>ttPID</strong></td>
<td>ttpid of the searched exchange content</td>
<td>16B</td>
</tr>
<tr>
<td><strong>folderID1</strong></td>
<td>folderID1 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>folderID2</strong></td>
<td>folderID1 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>ConditionDataSize</strong></td>
<td>Condition data size</td>
<td>2B</td>
</tr>
<tr>
<td><strong>ConditionData</strong></td>
<td>Condition data</td>
<td>(ConditionDataSize) B</td>
</tr>
</tbody>
</table>

(When state = other than Cancelable)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>status</strong></td>
<td>state of the searched exchange content</td>
<td>1B</td>
</tr>
<tr>
<td><strong>ExgThreadID</strong></td>
<td>Thread ID of the searched exchange content</td>
<td>20B</td>
</tr>
<tr>
<td><strong>ttPID</strong></td>
<td>ttpid of the searched exchange content</td>
<td>16B</td>
</tr>
<tr>
<td><strong>folderID1</strong></td>
<td>folderID1 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>folderID2</strong></td>
<td>folderID2 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>num1</strong></td>
<td>num of v1 of the searched exchange content</td>
<td>4B</td>
</tr>
<tr>
<td><strong>acl1</strong></td>
<td>acl of v1 of the searched exchange content</td>
<td>1B</td>
</tr>
<tr>
<td><strong>issuerid1</strong></td>
<td>issuerid of v1 of the searched exchange content</td>
<td>16B</td>
</tr>
<tr>
<td><strong>size1</strong></td>
<td>size of v1 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>v1</strong></td>
<td>Content of v1 of the searched exchange content</td>
<td>(size1) B</td>
</tr>
<tr>
<td><strong>num2</strong></td>
<td>num of v2 of the searched exchange content</td>
<td>4B</td>
</tr>
<tr>
<td><strong>acl2</strong></td>
<td>acl of v2 of the searched exchange content</td>
<td>1B</td>
</tr>
<tr>
<td><strong>issuerid2</strong></td>
<td>issuerid of v2 of the searched exchange content</td>
<td>16B</td>
</tr>
<tr>
<td><strong>size2</strong></td>
<td>size of v2 of the searched exchange content</td>
<td>2B</td>
</tr>
<tr>
<td><strong>v2</strong></td>
<td>Content of v2 of the searched exchange content</td>
<td>(size2) B</td>
</tr>
</tbody>
</table>
6.4.4 ExgStatusInfo

Message in which the ICC\_X notifies to the AP\_X the ThreadID list in mid-exchange.
This message is generated as an output message corresponding to the RequestExgStatusList message.
The ICC\_X creates the message using as DATA the ExgStatusList that is a list pairing the ExgthreadID,
which is the thread ID, and the state, the transition state.

**Parameters:**

<table>
<thead>
<tr>
<th>DATA details</th>
<th>Thread ID list</th>
</tr>
</thead>
<tbody>
<tr>
<td>ThreadIDList</td>
<td>Thread ID list</td>
</tr>
<tr>
<td>ThreadIDNum</td>
<td>Number of exchange items</td>
</tr>
<tr>
<td>ThreadIDInfo</td>
<td>List of exchange items</td>
</tr>
<tr>
<td>state</td>
<td>State control parameters</td>
</tr>
<tr>
<td>ExgThreadID</td>
<td>Thread ID during normal exchange processing</td>
</tr>
</tbody>
</table>
Appendix. APDU Command

This chapter stipulates as APDU commands, those commands for which it is unnecessary to specify a message transmission source and destination, or which cannot be specified, to the IC card. The IC card accepts the APDU commands stipulated in this chapter, processes them and outputs the results in an APDU response format. Only IC cards conforming to ISO7816 can accept APDU commands.

A.1 ReqIccID

IC card users use this command to read IC card eTRON IDs. IC cards output APDU responses that include eTRON IDs.

ReqIccID command format

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Data Size</th>
<th>Specified Value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
<td>1B</td>
<td>80h</td>
<td>Non SeM*, not conforming to the ISO/IEC7816-4</td>
</tr>
<tr>
<td>INS</td>
<td>1B</td>
<td>F4h</td>
<td>ReqIccID command</td>
</tr>
<tr>
<td>P1</td>
<td>1B</td>
<td>00h</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>1B</td>
<td>00h</td>
<td></td>
</tr>
<tr>
<td>Lc</td>
<td>3B</td>
<td>000000h</td>
<td>Fixed value</td>
</tr>
</tbody>
</table>

*SeM: Secure messaging

ReqIccID response format

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Data Size</th>
<th>Specified Value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>Response DATA block</td>
<td>16B</td>
<td></td>
<td>See DATA details</td>
</tr>
<tr>
<td>SW1</td>
<td>Command processing status</td>
<td>1B</td>
<td>**h</td>
<td>See Specified values: SW1/SW2</td>
</tr>
<tr>
<td>SW2</td>
<td>Command processing modifier</td>
<td>1B</td>
<td>**h</td>
<td>See Specified values: SW1/SW2</td>
</tr>
</tbody>
</table>

DATA details

<table>
<thead>
<tr>
<th>Data</th>
<th>Content</th>
<th>Date Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETRON ID</td>
<td>ICC eTRON ID (domain 12)</td>
<td></td>
</tr>
</tbody>
</table>

SW1-SW2 list

<table>
<thead>
<tr>
<th>Content</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal end</td>
<td>9000h</td>
</tr>
<tr>
<td>LEN error</td>
<td>6700h</td>
</tr>
<tr>
<td>Execution conditions unfulfilled</td>
<td>6985h</td>
</tr>
<tr>
<td>P1-P2 error</td>
<td>6A86h</td>
</tr>
<tr>
<td>INS error</td>
<td>6D00h</td>
</tr>
<tr>
<td>CLA error</td>
<td>6E00h</td>
</tr>
</tbody>
</table>
A.2 Unlock

The IC card user uses this command to release the LOCKED state of the information ICCState that controls the reading of the IC card and shifts it to the UNLOCKED state.

When the IC card authenticates the PIN for UNLOCKING, it shifts the card state from the LOCKED to the UNLOCKED state, and then outputs the APDU response. When shifted to the UNLOCKED state with this command, the file information that was output from the IC card by the BackupInfo message cannot be written back in by the RestoreCard message.

Unlock command format

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Data Size</th>
<th>Specified Value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLA</td>
<td>Command class</td>
<td>1B</td>
<td>80h</td>
<td>Non SeM*, not conforming to the ISO/IEC7816-4</td>
</tr>
<tr>
<td>INS</td>
<td>Command code</td>
<td>1B</td>
<td>F6h</td>
<td>Unlock command</td>
</tr>
<tr>
<td>P1</td>
<td>Command parameter 1</td>
<td>1B</td>
<td>00h</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>Command parameter 2</td>
<td>1B</td>
<td>00h</td>
<td></td>
</tr>
</tbody>
</table>

*SeM: Secure messaging

ReqIccID response format

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Data Size</th>
<th>Specified Value</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1</td>
<td>Command processing status</td>
<td>1B</td>
<td>**h</td>
<td>See Specified values: SW1/SW2</td>
</tr>
<tr>
<td>SW2</td>
<td>Command processing modifier</td>
<td>1B</td>
<td>**h</td>
<td>See Specified values: SW1/SW2</td>
</tr>
</tbody>
</table>

SW1-SW2 list

<table>
<thead>
<tr>
<th>Content</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal end</td>
<td>9000h</td>
</tr>
<tr>
<td>LEN error</td>
<td>6700h</td>
</tr>
<tr>
<td>Memory write failed</td>
<td>6581h</td>
</tr>
<tr>
<td>Execution conditions unfulfilled</td>
<td>6985h</td>
</tr>
<tr>
<td>P1-P2 error</td>
<td>6A86h</td>
</tr>
<tr>
<td>INS error</td>
<td>6D00h</td>
</tr>
<tr>
<td>CLA error</td>
<td>6E00h</td>
</tr>
</tbody>
</table>
Bibliography