Amendment to TR-03112-6
Remote IFD
Version 1.0
27. January 2020
# Contents

1  Introduction........................................................................................................................................... 4
1.1  Key Words........................................................................................................................................... 4
1.2  Encoding............................................................................................................................................... 4
2  Connecting UD and IFD.......................................................................................................................... 4
2.1  Discovery............................................................................................................................................... 4
2.2  Pairing................................................................................................................................................... 5
2.3  Communication Establishment............................................................................................................ 5
2.4  Un-Pairing.............................................................................................................................................. 6
3  IFD Interface Messages.............................................................................................................................. 6
3.1  Message Formats.................................................................................................................................... 6
3.1.1  Generic Message Format.................................................................................................................. 6
3.1.2  Response Message Format.............................................................................................................. 6
3.1.3  Error Message Format..................................................................................................................... 7
3.2  EstablishContext.................................................................................................................................... 7
3.2.1  Message............................................................................................................................................. 7
3.2.2  Response.......................................................................................................................................... 7
3.3  Connect.................................................................................................................................................. 8
3.3.1  Message............................................................................................................................................. 8
3.3.2  Response.......................................................................................................................................... 8
3.4  Disconnect............................................................................................................................................ 8
3.4.1  Message............................................................................................................................................. 8
3.4.2  Response.......................................................................................................................................... 9
3.5  Transmit............................................................................................................................................... 9
3.5.1  Message............................................................................................................................................. 9
3.5.2  Response.......................................................................................................................................... 9
3.6  EstablishPACEChannel.......................................................................................................................... 10
3.6.1  Message............................................................................................................................................ 10
3.6.2  Response.......................................................................................................................................... 10
3.7  ModifyPIN............................................................................................................................................ 11
3.7.1  Message............................................................................................................................................ 11
3.7.2  Response.......................................................................................................................................... 11
3.8  GetIFDStatus......................................................................................................................................... 11
3.8.1  Requesting Status information........................................................................................................ 11
3.8.2  IFDStatus.......................................................................................................................................... 12
1 Introduction

This specification deals with connecting and interfacing to a remote IFD. An example is an eID-Client on a local computer (called User Device or UD) in conjunction with a card terminal connected via a local network, e.g. a smart phone in the same local network as the User Device. The terminal is called IFD in the following. A card terminal can support several readers (e.g. contact / contactless interfaces, or several devices connected to the IFD, which acts as a server in that case), which are called Slots in the following.

1.1 Key Words

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC 2119]. The key word “CONDITIONAL” is to be interpreted as follows:

CONDITIONAL: The usage of an item is dependent on the usage of other items. It is therefore further qualified under which conditions the item is REQUIRED or RECOMMENDED.

1.2 Encoding

All messages between IFD and UD SHALL be encoded as JSON messages ([RFC 7159]).

The messages SHALL follow the following rules:

- Absent values or absent OPTIONAL/CONDITIONAL parameters SHALL be encoded as null.
- Binary values (e.g. file contents or APDUs) SHALL be encoded as hexadecimal strings in big-endian format.
- Several messages MAY be sent at the same time. In this case, the messages MUST be grouped in an array of messages.

2 Connecting UD and IFD

2.1 Discovery

An IFD with an active reader announces itself via UDP broadcast to port 24727 with the following message:
The IFDName SHALL be a name of the IFD suitable for display to the user. Since this name might be not unique (e.g. if the make and model of the IFD is used as IFDName), IFDID is used to further disambiguate IFDs. The IFDID SHALL contain the certificate used by the IFD for pairing (see section 2.2).

Support for the protocol/protocol version defined in this specification is indicated by including IFDInterface_WebSocket_v2 in the SupportedAPI field. Further versions of this protocol will be denoted by v3, v4 and so on.

### 2.2 Pairing

To initiate the pairing of a UD and an IFD, a short password (e.g. a 4 digit numeric password) is exchanged between them out of band. Examples are generation of the password by one Device and manually entering it on the other Device, or using a QR code for transfer.

The UD initiates a TLS connection to the IFD with a PSK-cipher suite. UD and IFD MUST support TLS_RSA_PSK_WITH_AES_256_CBC_SHA and MAY support additional PSK-cipher suites compliant to [TR-03116], Part 4. Both UD and IFD use self-signed certificates for this connection. The certificates SHOULD be as small as possible and MUST NOT contain any identifying information beyond the contained public key. The pre-exchanged short password is used as PSK.

If the connection is established successfully (i.e. the short password is correct), the UD and the IFD register the certificate of the communication partner as the identity of the other side used for communication establishment (see below). To exchange human readable name for the UD and the IFD, an EstablishContext Message/Response pair (see section 3.2) SHOULD be exchanged.

The established channel MUST NOT be used for further communication. For further communication, an authenticated channel as described in the next section MUST be used.

*Note: Since RSA_PSK does not protect the short password against passive brute force attack, the short password MUST have a limited validity time, e.g. a few minutes, and/or the number of attempts to guess the short password MUST be strictly limited.*

### 2.3 Communication Establishment

To set up communication, a mutually authenticated TLS connection using the certificates registered during pairing (see above) is established. For the TLS connection, including cipher suite support, conformance to [TR-03116], Part 4, is REQUIRED.

If the connection establishment fails due to an unknown certificate, this is signalled by the TLS stack by a certificate related error. In this case, also the other peer SHOULD delete the certificate.

After the TLS connection is established, bidirectional WebSocket ([RFC 6455]) communication is initiated by the UD.
The IFD SHOULD NOT perform UDP broadcasts while a connection is established and MUST reject further connection attempts after a connection is established until this connection is released.

2.4 Un-Pairing

The IFD and the UD SHOULD offer the user the capability to un-pair devices. Un-pairing is performed by de-registering the certificate registered in the Pairing step.

3 IFD Interface Messages

All messages are sent inside the WebSocket connection between UD and IFD.

3.1 Message Formats

All messages contain a ContextHandle for addressing. A ContextHandle SHOULD be chosen as a random pseudo-unique identifier by the IFD, in order to enable the UD to distinguish different IFDs. To distinguish several Slots of the same IFD, a SlotHandle is contained in slot specific messages.

3.1.1 Generic Message Format

The generic message format is given as

```
{  
    "msg" : <string, message type>,  
    "ContextHandle" : <string>,  
    "SlotHandle" : <string>,  
    ...  
}
```

In some message, when no context/connection is established, the ContextHandle / the SlotHandle are omitted.

3.1.2 Response Message Format

The generic format of response messages is an extension of the generic format by result elements:

```
{  
    "msg" : <string, response message type>,  
    "ContextHandle" : <string>,  
    "SlotHandle" : <string>,  
    "ResultMajor" : <string>,  
    "ResultMinor" : <string>,  
    ...  
}
```

For ResultMajor / ResultMinor see the corresponding descriptions in [TR-03112], part 6.
3.1.3 Error Message Format

In case an error is encountered which is not related to a valid message (e.g. an unknown message), the party encountering the error SHALL send an error message:

```json
{
    "msg" : "IFDERROR",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "ResultMajor" : <string>,
    "ResultMinor" : <string>
}
```

If the error is not related to a defined Slot / a defined Context, the corresponding Handles are omitted from the error message.

3.2 EstablishContext

See also [TR-03112], part 6, section 3.1.1.

3.2.1 Message

To negotiate the protocol version for subsequent communication, and to establish a Context, the following message is sent immediately after communication establishment (see section 2.3). This message MUST be the first message send by the UD after the WebSocket connection between UD and IFD is established.

```json
{
    "msg" : "IFDEstablishContext",
    "Protocol" : <string>,
    "UDName" : <string>
}
```

The element `Protocol` MUST contain the identifier of one of the supported protocols advertised by the IFD in `SupportedAPI` during discovery.

The element `UDName` SHALL contain the human readable name of the UD, suitable for display to the user.

3.2.2 Response

Upon receiving an `EstablishContext` message, the IFD SHALL send the following message as response containing the `ContextHandle` chosen by the IFD.

```json
{
    "msg" : "IFDEstablishContextResponse",
    "ContextHandle" : <string>,
    "ResultMajor" : <string>,
    "ResultMinor" : <string>,
    "IFDName" : <string>
}
```
The element **IFDName** SHALL contain the human readable name of the IFD, suitable for display to the user. The name **SHOULD** be identical to the **IFDName** advertised for discovery (see section 2.1).

### 3.3 Connect

See also [TR-03112], part 6, section 3.2.1.

#### 3.3.1 Message

This message is used by the UD to connect to a card available in a Slot of the IFD.

```json
{
   "msg" : "IFDConnect",
   "ContextHandle" : <string>,
   "SlotName" : <string>,
   "exclusive" : <boolean>
}
```

If `exclusive` is set to `true`, no other application can connect to the card until the connection is released by the UD, i.e. the IFD **MUST** refuse any communication attempt to the card by other applications.

#### 3.3.2 Response

The IFD **SHALL** send a **ConnectResponse** message after receiving a **Connect** message from the UD. If the connection to the card was established successfully, a unique **SlotHandle** **MUST** be returned. If the connection failed, no handle is returned.

```json
{
   "msg" : "IFDConnectResponse",
   "ContextHandle" : <string>,
   "SlotHandle" : <string>,
   "ResultMajor" : <string>,
   "ResultMinor" : <string>
}
```

### 3.4 Disconnect

See also [TR-03112], part 6, section 3.2.2.

#### 3.4.1 Message

This message can be send by either the UD or the IFD to disconnect a card. After this message the **SlotHandle** is invalid, using the handle in further messages **SHALL** yield an error.
3.4.2 Response

This message SHALL be sent as answer to a Disconnect message.

```json
{
    "msg" : "IFDDisconnectResponse",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "ResultMajor" : <string>,
    "ResultMinor" : <string>
}
```

3.5 Transmit

See also [TR-03112], part 6, section 3.2.5.

3.5.1 Message

The Transmit function sends one APDU to the IFD. The IFD in turn sends the APDU to the connected card.

```json
{
    "msg" : "IFDTransmit",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "InputAPDU" : <string>,
    "DisplayText" : <string>
}
```

- **InputAPDU** *(REQUIRED)*
  Contains the APDU which is to be sent to the eCard.

- **DisplayText** *(OPTIONAL)*
  Contains a free text to be displayed to the user, e.g. to inform the user about the progress of the overall process. If no text is passed, the previous text stays unchanged.

If a PACE Secure Messaging channel between card and IFD is active (see section 3.6), the IFD SHALL encrypt and MAC the APDU before transmitting them to the card.

3.5.2 Response

This message is used by the IFD to return the response APDU to the command APDU transmitted via Transmit to the UD.
If a PACE Secure Messaging channel between card and IFD is active (see section 3.6), the IFD SHALL return the decrypted and verified plain response APDU.

### 3.6 EstablishPACEChannel

See also [TR-03112], part 6, section 3.3.1.

If the IFD supports a PIN Pad, the IFD MUST support the command EstablishPACEChannel as defined in [TR-03119], section D.1, and [PC/SC], Part 10 AMD1, section 2.6.16. This command is mapped to messages for the IFD Interface using the data structures defined in [PC/SC], Part 10 AMD1, section 2.6.16 (without the FEATURE_EXECUTE_PACE header defined in section 2.5.12), and [TR-03119], section D.2 as defined below.

#### 3.6.1 Message

```json
{
    "msg" : "IFDEstablishPACEChannel",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "InputData" : <string, see below>
}
```

The InputData SHALL contain the input data for EstablishPACEChannel as defined in [PC/SC], Part 10 AMD1, section 2.6.16, including the extension defined in [TR-03119], section D.2.

#### 3.6.2 Response

```json
{
    "msg" : "IFDEstablishPACEChannelResponse",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "ResultMajor" : <string>,
    "ResultMinor" : <string>,
    "ResultCode" : <string, see below>,
    "OutputData" : <string, see below>
}
```

The ResultCode SHALL contain a result value as defined in the result value table of [PC/SC], Part 10 AMD1, section 2.5.12.

The OutputData SHALL contain the output data for EstablishPACEChannel as defined in [PC/SC], Part 10 AMD1, section 2.6.16.
3.7 ModifyPIN

See also [TR-03112], part 6, section 3.3.2.

If the IFD supports a PIN Pad, the IFD MUST support the command ModifyPIN as defined in [TR-03119], section D.1, and [PC/SC], Part 10 to support PIN Management. This command is mapped to messages for the IFD Interface using the data structures defined in [PC/SC], Part 10 as defined below.

3.7.1 Message

```
{
    "msg" : "IFDModifyPIN",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "InputData" : <string, see below>
}
```

The InputData SHALL contain the structure PIN_MODIFY as defined in [PC/SC], Part 10, section 2.5.3.

3.7.2 Response

```
{
    "msg" : "IFDModifyPINResponse",
    "ContextHandle" : <string>,
    "SlotHandle" : <string>,
    "ResultMajor" : <string>,
    "ResultMinor" : <string>,
    "OutputData" : <string, see below>
}
```

The OutputData SHALL contain the response as defined in [PC/SC], Part 10, section 2.6.3.

3.8 GetIFDStatus

3.8.1 Requesting Status information

This message can be send by the UD to trigger the IFD to send a Status message.

```
{
    "msg" : "IFDgetStatus",
    "ContextHandle" : <string>,
    "SlotName" : <string>
}
```

If SlotName is empty, an IFDStatus message for all Slots is requested, otherwise a message for the named Slot is requested.
### 3.8.2 IFDStatus

This IFD SHALL send this message

- immediately after establishing a context with a UD (see section 3.2.1/3.2.2) for each Slot,
- upon change of the state of the IFD or one of the Slots for the affected Slots, and
- upon request by the UD via the IFDGetStatus message (see above) for the requested Slots.

```json
{
    "msg" : "IFDStatus",
    "ContextHandle" : <string>,
    "SlotName" : <string>,
    "PINPad" : <boolean>,
    "MaxAPDULength" : <number>,
    "ConnectedReader" : <boolean>,
    "CardAvailable" : <boolean>,
    "EFATR" : <string>,
    "EFDIR" : <string>
}
```

The message encapsulates the capabilities and current status of an IFD / Slot, and comprises the following elements:

- **SlotName** *(REQUIRED)*
  - SHALL contain a human readable name of the Slot, suitable for display to the user. The SlotName MUST be unique for the IFD.

- **PINPad** *(REQUIRED)*
  - SHALL be set to true if a PACE-capable PIN Pad is available and activated. Note that the status of this flag may change if the user activates/deactivates a pin entry capability on the IFD.

- **MaxAPDULength** *(REQUIRED)*
  - SHALL contain the maximum length for APDUs supported by the IFD / Slot.

- **ConnectedReader** *(REQUIRED)*
  - SHALL be set to true if the reader is available.

- **CardAvailable** *(REQUIRED)*
  - SHALL be set to true if and only if a card is present and recognized, and is available for Connect (i.e. not exclusively connected, see section 3.3).

- **EFATR** *(CONDITIONAL)*
  - If a card is available, this element SHALL contain the contents of EF.ATR/INFO of the card to enable recognition by the UD.

- **EFDIR** *(CONDITIONAL)*
  - If a card is available and contains a transparent file EF.DIR, this element SHALL contain the content of EF.DIR.
References

[TR-03116] BSI: TR-03116: Kryptographische Vorgaben für Projekte der Bundesregierung
[TR-03119] BSI: TR-03119: Requirements for Smart Card Readers Supporting eID and eSign Based on Extended Access Control
[RFC 2119] IETF: RFC 2119: S. Bradner: Key words for use in RFCs to Indicate Requirement Levels
[PC/SC] PC/SC Workgroup: PC/SC Workgroup Specifications 1.0/2.0