FIDO U2F USB Framing of APDUs

Specification Set: fido-u2f-v1.0-rd-20140209 REVIEW DRAFT

Editors:

Dirk Balfanz (balfanz@google.com)

Contributors:

Abstract:

Status:

This Specification has been prepared by FIDO Alliance, Inc. **This is a Review Draft Specification and is not intended to be a basis for any implementations as the Specification may change**. Permission is hereby granted to use the Specification solely for the purpose of reviewing the Specification. No rights are granted to prepare derivative works of this Specification. Entities seeking permission to reproduce portions of this Specification for other uses must contact the FIDO Alliance to determine whether an appropriate license for such use is available.

Implementation of certain elements of this Specification may require licenses under third party intellectual property rights, including without limitation, patent rights. The FIDO Alliance, Inc. and its Members and any other contributors to the Specification are not, and shall not be held, responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

THIS FIDO ALLIANCE SPECIFICATION IS PROVIDED "AS IS" AND WITHOUT ANY WARRANTY OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY EXPRESS OR IMPLIED WARRANTY OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Table of Contents

1 Notation

Type names, attribute names and element names are written in *italics*.

String literals are enclosed in “”, e.g. “UAF-TLV”.

In formulas we use “|” to denote byte wise concatenation operations.

U2F specific terminology used in this document is defined in [FIDOGlossary]

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 [RFC2119].

2 USB Token Descriptors

Note: Reading the 'FIDO U2F Overview' [U2FOverview] is recommended as a background for this document.

The device class is set to WinUSB.

You may use a vendor-id and product id (vid, pid) pair of your choosing. At this stage you will have to list the vid and pid in the Chrome extension contributed to the U2F working group and recompile the extension. Eventually, we will use the 'Product String Descriptor' to recognize U2F tokens. They must be prefixed by “FIDO U2F vx.y:” where x.y is the version number, starting at 0.9 as the writing of this document.

When the browser communicates with the USB token, in either direction (in or out), it uses bulk_transfer on respectively the in and out endpoint listed in the first interface advertising the 'Vendor Specific Class'.

3 Overall Frame Format

3.1 Request from browser to token

```
4 bytes transaction id | 1 byte cmd | 2 bytes length | data
```

See command set a token needs to support.

3.2 Response from token to browser

```
4 bytes transaction id | 1 byte status | 2 bytes length | data
```

Response always echoes its request's transaction id.

See response status definition.

4 Minimal Command Set

1. 0x81: echo.
   Data is to be echoed back.
2. 0x83: apdu.
   Data is apdu payload to be passed to smartcard. See below.
3. 0xbc: sync.
   Data is single byte to be echoed back. It drains and resets the communication queue between app and device. Device should answer to sync even when busy (for instance waiting on secure element).

5 Response status returned by token

1. Same value as cmd: all is well
2. 0xbf: error.
   data is 1 byte payload with specific error codes:
   - 0x05: timeout. Secure element timed out. Remediate with sync.
   - 0x06: busy. Device is servicing a request with a different transaction id.
   - (all other values RFU)

6 APDU payloads

We provide APDU payloads for all messages described in the U2F Raw Message Formats document [U2FRawMsgs].

We use the ISO 7816-4 specification for APDU payloads. See Fido u2f sample applet code for reference, but the general APDU format for the registration and authentication instructions are included below:

6.1 Registration Request Message

The raw registration request message (see the U2F Raw Message Formats document [U2FRawMsgs]) becomes the payload. The P1 byte may not be zero (non-zero values are reserved for future use). Example:

```
CL IN P1 P2 L0 L1 L2 --payload------------ Le
00 01 ?? 00 00 00 40 [64 bytes raw message] 00 00
```

6.2 Enrollment Response Message: Success

The payload of the response message is the raw enrollment response message as described in the U2F Raw Message Formats document [U2FRawMsgs]. Example:

```
 --payload------------------ ISO7816.SW_NO_ERROR
 [e.g. 457 bytes raw message] 90 00
```

6.3 Authentication Request Message

The control byte becomes P1. The raw enrollment request message (without the control byte) becomes the payload. Example:

```
CL IN P1 P2 L0 L1 L2                                          Le
00 02 03 00 00 00 81 [129 bytes raw message w/o control byte] 00 00
```

6.4 Authentication Response Message: Success

The payload of the response message is the raw authentication response message as described in the U2F Raw Message Formats document [U2FRawMsgs].

Including SW1-2 0x90 0x00. Example:

```
--payload----------------- ISO7816.SW_NO_ERROR
[e.g. 82 bytes raw message] 90 00
```


6.5 Response Messages: Error: Test Of User Presence Required

Both the authentication and registration response error messages indicating that a test-of-user-presence is required feature an empty payload followed by the 0x69 0x85 error code:

```
[] 69 85 (ISO7816.SW_CONDITIONS_NOT_SATISFIED)
```

6.6 Authentication Response Messages: Error: Bad Key Handle

The authentication response message:error:bad key handle features an empty payload followed by the 0x6A 0x80 error code:

```
[] 6A 80 (ISO7816.SW_WRONG_DATA)
```

6.7 GetVersion request message

The GetVersion request message's USB framing as an APDU looks like this:

```
CL IN P1 P2 L0 L1 L2 Le
00 03 00 00 00 00 00 00 00
```

6.8 GetVersion response message

The payload of the GetVersion response message is the string 'U2F_V2'.

```
USB FRAMING     | APDU PAYLOAD
CID      CM LEN | U 2 F _ V 2 SW12
01000003 83 0008 5532465F5632 9000
                 (e.g. 'U2F_V2' 90 00 (ISO7816.SW_NO_ERROR))
```

These APDU frames are to be inserted inside the data (aka payload) portion of the USB frame.

The USB device should consistency check that the L0 L1 L2 length is equal to USB request frame length minus 9 (APDU + data + Le).

Note that in the case of the enroll response, the reply size is dictated by enroll attestation cert, which is large and of unknown length.

6.9 Future Considerations

Future versions may choose to present U2F devices as HID class devices in addition to or in place of winUSB to widen the driverless compatibility across more client OS platforms.

Bibliography

FIDO Alliance Documents:

[FIDOGlossary] Rolf Lindemann, Davit Baghdasaryan, Brad Hill, John Kemp. FIDO Technical Glossary. Version v1.0-rd-20140209, FIDO Alliance, February 2014. See http://fidoalliance.org/specs/fido-glossary-v1.0-rd-20140209.pdf

[**U2FOverview**] Sampath Srinivas, Dirk Balfanz, Eric Tiffany. FIDO Universal 2nd Factor (U2F) Overview. Version v1.0-rd-20140209, FIDO Alliance, February 2014. See http://fidoalliance.org/specs/fido-u2f-overview-v1.0-rd-20140209.pdf

[**U2FRawMsgs**] Dirk Balfanz. FIDO U2F Raw Message Formats. Version v1.0-rd-20140209, FIDO Alliance, February 2014. See http://fidoalliance.org/specs/fido-u2f-raw-message-formats-v1.0-rd-20140209.pdf

Other References:

[RFC2119] Key words for use in RFCs to Indicate Requirement Levels (RFC2119), S. Bradner, March 1997