# UAF Registry of Predefined Values

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


**Editors:**

Rolf Lindeman, Nok Nok Labs

Davit Baghdasaryan, Nok Nok Labs

Brad Hill, PayPal


**Contributors:**


**Abstract:**

This document defines all the strings and constants reserved by UAF protocols. The values defined in this document are referenced by various UAF specifications.

## 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 Terminology

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

String literals are enclosed in “”, e.g. “UAFV1-TLV”.

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

UAF 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 Introduction

This document defines the registry of UAF-specific constants that are used and referenced in various UAF specifications. It is expected that, over time, new constants will be added to this registry. For example new authentication algorithms and new types of authenticator characteristics will require new constants to be defined for use within the specifications.

# 3 Authenticator Characteristics

## 3.1 Authentication Factors

The USER_VERIFY constants are flags in a bitfield represented as a 64 bit long integer. They describe the methods and capabilities of an UAF authenticator for locally verifying a user. In most cases, the operational details of these methods are opaque to the server, but in some cases verification may involve transmission of attested measurement, such as for USER_VERIFY_LOCATION. These constants are used in the authoritative metadata for an authenticator, reported and queried through the UAF Discovery APIs, and used to form Authenticator policies in UAF protocol messages.

USER_VERIFY_PRESENCE 0x01

This flag will be set if the authenticator is able to confirm user presence in any fashion. If this flag and no other is set for user verification, the guarantee is only that the authenticator cannot be operated without some human intervention, not necessarily that the presence verification provides any level of authentication of the human's identity. (e.g. a device that requires a touch to activate)

USER_VERIFY_FINGERPRINT 0x02

This flag will be set if the authenticator uses any type of measurement of a fingerprint for User-to-Authenticator authentication.

USER_VERIFY_PASSCODE 0x04

This flag will be set if the authenticator uses a local-only passcode for User-to-Authenticator authentication.

USER_VERIFY_VOICEPRINT 0x08

This flag will be set if the authenticator uses a voiceprint for User-to-Authenticator authentication.

USER_VERIFY_FACEPRINT 0x10

This flag will be set if the authenticator uses any manner of face recognition to locally authenticate the user.

USER_VERIFY_LOCATION 0x20

This flag will be set if the authenticator uses any form of location sensor or measurement for User-to-Authenticator authentication and/or returns a location measurement to the Relying Party as an additional user verification.

USER_VERIFY_EYEPRINT 0x40

This flag will be set if the authenticator uses any form of eye biometrics for User-to-Authenticator authentication.

USER_VERIFY_PATTERN 0x80

This flag will be set if the authenticator uses a drawn pattern for User-to-Authenticator authentication.

USER_VERIFY_HANDPRINT 0x100

This flag will be set if the authenticator uses any measurement of a full hand (including palm-print, hand geometry or vein geometry) for User-to-Authenticator authentication.

USER_VERIFY_NONE 0x200

This flag will be set if the authenticator will respond without any user interaction.

## 3.2 Key Protection Types

The KEY_PROTECTION constants are flags in a bit field represented as a 64 bit long integer. They describe the method an authenticator uses to protect the private key material for FIDO registrations. These constants are used in the authoritative metadata for an authenticator, reported and queried through the UAF Discovery APIs, and used to form Authenticator policies in UAF protocol messages. When used in metadata describing an authenticator, several of these flags are exclusive with others - the certified metadata may have at most one of the mutually exclusive bits set to 1. When used in authenticator policy, any bit may be set to 1, e.g. to indicate that a server is willing to accept authenticators using either KEY_PROTECTION_SOFTWARE and KEY_PROTECTION_HARDWARE.

KEY_PROTECTION_SOFTWARE 0x01

This flag will be set if the authenticator uses software-based key management.

*Exclusive in authenticator metadata with KEY_PROTECTION_HARDWARE, KEY_PROTECTION_TEE, KEY_PROTECTION_SECURE_ELEMENT*

KEY_PROTECTION_HARDWARE 0x02

This flag will be set if the authenticator uses hardware-based key management.

*Exclusive in authenticator metadata with KEY_PROTECTION_SOFTWARE*

KEY_PROTECTION_TEE 0x04

This flag will be set if the authenticator uses the Trusted Execution Environment [TEE] for key management. In authenticator metadata, this flag should be set in conjunction with KEY_PROTECTION_HARDWARE.

*Exclusive in authenticator metadata with KEY_PROTECTION_SOFTWARE, KEY_PROTECTION_SECURE_ELEMENT*

KEY_PROTECTION_SECURE_ELEMENT 0x08

This flag will be set if the authenticator uses a Secure Element [SecureElement] for key management. In authenticator metadata, this flag should be set in conjunction with KEY_PROTECTION_HARDWARE.

*Exclusive in authenticator metadata with KEY_PROTECTION_TEE, KEY_PROTECTION_SOFTWARE*

KEY_PROTECTION_REMOTE_HANDLE 0x10

This flag will be set if the authenticator does not store per-Origin keys at the client, but relies on a server-provided key handle.

This flag MUST be set in conjunction with one of the other KEY_PROTECTION flags to indicate how the local key handle wrapping key and operations are protected.

Servers can unset this flag in authenticator policy if they are unprepared to store and return key handles, for example, if they have a requirement to respond indistinguishably to authentication attempts against userIDs that do and do not exist. Refer to [UAFProtocol] for more details.

## 3.3 Authenticator Attachment Hints

The ATTACHMENT_HINT constants are flags in a bit field represented as a 64 bit long. They describe the method an authenticator uses to communicate with the system on which the FIDO client software is executing. These constants are reported and queried through the UAF Discovery APIs, and used to form Authenticator policies in UAF protocol messages.

Because the connection state and topology of an authenticator may be transient, these values are only hints that can be used by server-supplied policy to guide the user experience, e.g. to prefer a device that is connected and ready for authenticating or confirming a low-value transaction, rather than one that is more secure but requires more user effort. These values are not reflected in authenticator metadata and cannot be relied on by the relying party, although some models of authenticator may provide attested measurements of similar data as part of UAF response messages.

ATTACHMENT_HINT_INTERNAL 0x01

This flag indicates that the authenticator is permanently attached to the system on which the FIDO client software is running.

A device such as a smartphone may have authenticator functionality that is able to be used both locally and remotely. In such a case, the FIDO client MUST filter and exclusively report only the relevant bit during Discovery and when performing policy matching.

ATTACHMENT_HINT_EXTERNAL 0x02

This flag indicates, for a hardware-based authenticator, that it is removable or remote from the system on which the FIDO client software is running.

A device such as a smartphone may have authenticator functionality that is able to be used both locally and remotely. In such a case, the FIDO client MUST filter and exclusively report only the relevant bit during Discovery and when performing policy matching.

ATTACHMENT_HINT_WIRED 0x04

Indicates that an external authenticator currently has an exclusive wired connection, e.g. through USB, Firewire or similar, to the system on which the FIDO client software is executing.

ATTACHMENT_HINT_WIRELESS 0x08

Indicates that an external authenticator communicates with the system on which the FIDO client software is executing through a personal area or otherwise non-routed wireless protocol, such as Bluetooth or NFC.

ATTACHMENT_HINT_NFC 0x10

Indicates that an external authenticator is able to communicate by NFC to the FIDO client software. As part of authenticator metadata, or when reporting characteristics through Discovery, if this flag is set, the ATTACHMENT_HINT_WIRELESS flag SHOULD also be set.

ATTACHMENT_HINT_BLUETOOTH 0x20

Indicates that an external authenticator is able to communicate using Bluetooth to the FIDO client software. As part of authenticator metadata, or

when reporting characteristics through Discovery, if this flag is set, the ATTACHMENT_HINT_WIRELESS flag SHOULD also be set.

ATTACHMENT_HINT_NETWORK 0x40
Indicates that the authenticator is not on the same system as the FIDO client software but communicates with it over a non-exclusive network. (e.g. over a TCP/IP LAN or WAN, as opposed to a point-to-point Bluetooth connection)

ATTACHMENT_HINT_READY 0x80
Indicates that an external authenticator is in a ready state. e.g. a Bluetooth connected device that is currently paired and connected or a USB device that is plugged in.

## 3.4 Secure Display Types

The SECURE_DISPLAY constants are flags in a bit field represented as a 64 bit long integer. They describe the availability and implementation of a secure display capability required for the Transaction Confirmation operation. These constants are used in the authoritative metadata for an authenticator, reported and queried through the UAF Discovery APIs, and used to form Authenticator policies in UAF protocol messages.

SECURE_DISPLAY_ANY 0x01
This flag indicates, that some form of secure display is available on this authenticator.

SECURE_DISPLAY_PRIVILEGED_SOFTWARE 0x02
This flag indicates, that a software-based secure display operating in a privileged context is available on this authenticator.

Software based displays are typically provided by the FIDO client software rather than the authenticator itself. A FIDO client that is capable of providing this capability MAY set this bit for all authenticators of type ATTACHMENT_EMBEDDED, even if the authoritative metadata for the authenticator does not indicate this capability.

SECURE_DISPLAY_TEE 0x04

This flag indicates that the authenticator implements a secure display in the Trusted Execution Environment [TEE].

SECURE_DISPLAY_HARDWARE 0x08

This flag indicates, that a secure display based on hardware assisted capabilities is available on this authenticator.

SECURE_DISPLAY_REMOTE 0x10

This flag indicates, that the secure display is provided on a distinct device from the system the FIDO client software is executing on.

# 4 Predefined Tags

The internal structure of UAF authenticator commands is a “Tag-Length-Value” (TLV) sequence. The **Tag** is a 2-byte unique unsigned value describing the type of field the data represents, the **Length** is a 2-byte unsigned value indicating the size of the value in bytes and **Value** is the variable-sized series of bytes which contain data for this item in the sequence.

Note that Tags are not used as bitflags.

## 4.1 Tags used in the protocol

The following tags have been allocated for data types in UAF protocol messages:

TAG_UAFV1_REG_RESPONSE 0x01
: The content of this TAG is Authenticator Response for Register command.

TAG_UAFV1_SIGN_RESPONSE 0x02
: The content of this TAG is Authenticator Response for Sign command.

TAG_UAFV1_KRD 0x03
: Indicates Key Registration Data.

TAG_UAFV1_SIGNEDDATA 0x04
: Indicates data signed by authenticator using Uauth.priv key.

TAG_ATTESTATION_CERT 0x05
: Indicates DER encoded Attestation Batch Certificate.

TAG_SIGNATURE 0x06
: Indicates a cryptographic signature.

## 4.2 Tags for crypto algorithms and types

These TAGs indicate the specific authentication algorithms, public key formats and other crypto relevant data.

## 4.3 Authentication Algorithms

UAF_ALG_SIGN_ECDSA_SHA256_RAW 0x01

ECDSA signature MUST have raw R and S buffers, encoded in big-endian order.

For example for ECC-P256 curve the signature MUST have the following form

[R (32 bytes), S (32 bytes)]

UAF_ALG_SIGN_ECDSA_SHA256_DER 0x02

DER encoded ECDSA signature,

i.e. DER encoded SEQUENCE { r INTEGER, s INTEGER }

UAF_ALG_SIGN_RSASSA-PSS_SHA256_RAW 0x03

RSASSA-PSS signature MUST have raw S buffers, encoded in big-endian order. For example for RSA 2048 the signature have the following form [ S (256 bytes) ]

UAF_ALG_SIGN_RSASSA-PSS_SHA256_DER 0x04

DER encoded RSASSA-PSS signature

## 4.4 Public Key Representation Formats

| | |
|---|---|
| UAF_ALG_KEY_ECC_NISTP256R1_X962_RAW | 0x100 |
| Raw ANSI X.9.62 formatted public key | |
| UAF_ALG_KEY_ECC_NISTP256R1_X962_DER | 0x101 |
| DER encoded ANSI X.9.62 formatted public key | |
| UAF_ALG_KEY_RSA_2048_PSS_RAW | 0x102 |
| Raw RSASSA-PSS formatted public key | |
| UAF_ALG_KEY_RSA_2048_PSS_DER | 0x103 |
| ASN.1 DER encoded RSASSA-PSS formatted public key | |

# 5 Assertion Schemes

Names of Assertion Schemes are string with a length of 8 characters.

UAF TLV based scheme “UAFV1TLV”

This assertion scheme allows the Authenticator and the FIDO Server to exchange an asymmetric authentication key generated by the Authenticator.

The Authenticator MUST generate a key pair (UAuth.pub/UAuth.priv) to be used with algorithm suites listed in section Crypto Suites (with prefix “UAF_ALG”).

This scheme is using a compact Tag Length Value (TLV) encoding for the KRD and SignData messages generated by the Authenticators. This is the default scheme for the UAF protocol.

# 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

**[UAFProtocol]** Rolf Lindemann, Davit Baghdasaryan, Eric Tiffany. FIDO Universal Authentication Framework Protocol. Version v1.0-rd-20140209, FIDO Alliance, February 2014. See http://fidoalliance.org/specs/fido-uaf-protocol-v1.0-rd-20140209.pdf

*Other References:*

**[FIPS 186-3]** NIST DIGITAL SIGNATURE STANDARD (DSS) (FIPS 186-3), National Institute of Standards and Technology, June 2009

**[FIPS 186-4]** NIST DIGITAL SIGNATURE STANDARD (DSS) (FIPS 186-4), National Institute of Standards and Technology, July 2013

[**PKCS#**1] Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 (RFC 3447), J. Jonsson et al, February 2003

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

**[RFC5639]** Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation (RFC 5639), M Lochter et al, March 2010

**[SecureElement]** Global Platform Card Specifications (Specifications)

**[SP800-131A]** NIST Special Publication 800-131A, Transitions: Recommendations for Transitioning the Use of Cryptographic Algorithms and Key Lengths, E. Barker et al, National Institute of Standards and Technology, January 2011

**[SP800-63-1]** NIST Electronic Authentication Guideline SP 800-63-1 (NIST SP 800-63-1). W. Burr et al, National Institute of Standards and Technology, December 2011

**[TEE]** Global Platform Trusted Execution Environment Specifications (Specifications)

**[TEESecureDisplay]** Trusted User Interface API Specification (Specifications)

[SecureElement] Global Platform Secure Element Specification (http://www.globalplatform.org/specifications.asp)