## NAME

**fido2-assert** - get/verify a FIDO2 assertion

## SYNOPSIS

**fido2-assert -G** [**-bdhpruv**] [**-t** *option*] [**-i** *input_file*] [**-o** *output_file*] *device*
**fido2-assert -V** [**-dhpv**] [**-i** *input_file*] *key_file* [*type*]

## DESCRIPTION

**fido2-assert** gets or verifies a FIDO2 assertion.

The input of **fido2-assert** is defined by the parameters of the assertion to be obtained/verified. See the *INPUT FORMAT* section for details.

The output of **fido2-assert** is defined by the result of the selected operation. See the *OUTPUT FORMAT* section for details.

If an assertion is successfully obtained or verified, **fido2-assert** exits 0. Otherwise, **fido2-assert** exits 1.

The options are as follows:

**-G** Tells **fido2-assert** to obtain a new assertion from *device*.

**-V** Tells **fido2-assert** to verify an assertion using the PEM-encoded public key in *key_file* of type *type*, where *type* may be *es256* (denoting ECDSA over NIST P-256 with SHA-256), *rs256* (denoting 2048-bit RSA with PKCS#1.5 padding and SHA-256), or *eddsa* (denoting EDDSA over Curve25519 with SHA-512). If *type* is not specified, *es256* is assumed.

**-b** Request the credential’s "largeBlobKey", a 32-byte symmetric key associated with the asserted credential.

**-h** If obtaining an assertion, enable the FIDO2 hmac-secret extension. If verifying an assertion, check whether the extension data bit was signed by the authenticator.

**-d** Causes **fido2-assert** to emit debugging output on *stderr*.

**-i** *input_file*
Tells **fido2-assert** to read the parameters of the assertion from *input_file* instead of *stdin*.

**-o** *output_file*
Tells **fido2-assert** to write output on *output_file* instead of *stdout*.

**-p** If obtaining an assertion, request user presence. If verifying an assertion, check whether the user presence bit was signed by the authenticator.

**-r** Obtain an assertion using a resident credential. If **-r** is specified, **fido2-assert** will not expect a credential id in its input, and may output multiple assertions. Resident credentials are called "discoverable credentials" in CTAP 2.1.

**-t** *option*

Toggles a key/value *option*, where *option* is a string of the form "key=value". The options supported at present are:

**up**=*true*|*false*

Asks the authenticator for user presence to be enabled or disabled.

**uv**=*true*|*false*

Asks the authenticator for user verification to be enabled or disabled.

**pin**=*true*|*false*

Tells **fido2-assert** whether to prompt for a PIN and request user verification.

The **-t** option may be specified multiple times.

**-u** Obtain an assertion using U2F. By default, **fido2-assert** will use FIDO2 if supported by the authenticator, and fallback to U2F otherwise.

**-v** If obtaining an assertion, prompt the user for a PIN and request user verification from the authenticator. If verifying an assertion, check whether the user verification bit was signed by the authenticator.

If a *tty* is available, **fido2-assert** will use it to obtain the PIN. Otherwise, *stdin* is used.

## INPUT FORMAT

The input of **fido2-assert** consists of base64 blobs and UTF-8 strings separated by newline characters ('\n').

When obtaining an assertion, **fido2-assert** expects its input to consist of:

1. client data hash (base64 blob);
2. relying party id (UTF-8 string);
3. credential id, if credential not resident (base64 blob);

4. hmac salt, if the FIDO2 hmac-secret extension is enabled (base64 blob);

When verifying an assertion, **fido2-assert** expects its input to consist of:

1. client data hash (base64 blob);
2. relying party id (UTF-8 string);
3. authenticator data (base64 blob);
4. assertion signature (base64 blob);

UTF-8 strings passed to **fido2-assert** must not contain embedded newline or NUL characters.

## OUTPUT FORMAT

The output of **fido2-assert** consists of base64 blobs and UTF-8 strings separated by newline characters ('\n').

For each generated assertion, **fido2-assert** outputs:

1. client data hash (base64 blob);
2. relying party id (UTF-8 string);
3. authenticator data (base64 blob);
4. assertion signature (base64 blob);
5. user id, if credential resident (base64 blob);
6. hmac secret, if the FIDO2 hmac-secret extension is enabled (base64 blob);
7. the credential's associated 32-byte symmetric key ("largeBlobKey"), if requested (base64 blob).

When verifying an assertion, **fido2-assert** produces no output.

## EXAMPLES

Assuming *cred* contains a *es256* credential created according to the steps outlined in fido2-cred(1), obtain an assertion from an authenticator at */dev/hidraw5* and verify it:

```
$ echo assertion challenge | openssl sha256 -binary | base64 > assert_param
$ echo relying party >> assert_param
$ head -1 cred >> assert_param
$ tail -n +2 cred > pubkey
$ fido2-assert -G -i assert_param /dev/hidraw5 | fido2-assert -V pubkey es256
```

## SEE ALSO

fido2-cred(1), fido2-token(1)