

**NAME**

**fido2-cred** - make/verify a FIDO2 credential

**SYNOPSIS**

**fido2-cred -M** [-bdhqr<sup>uv</sup>] [-c *cred\_protect*] [-i *input\_file*] [-o *output\_file*] *device* [*type*]

**fido2-cred -V** [-dhv] [-c *cred\_protect*] [-i *input\_file*] [-o *output\_file*] [*type*]

**DESCRIPTION**

**fido2-cred** makes or verifies a FIDO2 credential.

A credential *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.

When making a credential, the authenticator may require the user to authenticate with a PIN. If the **-q** option is not specified, **fido2-cred** will prompt the user for the PIN. If a *tty* is available, **fido2-cred** will use it to obtain the PIN. Otherwise, *stdin* is used.

The input of **fido2-cred** is defined by the parameters of the credential to be made/verified. See the *INPUT FORMAT* section for details.

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

If a credential is successfully created or verified, **fido2-cred** exits 0. Otherwise, **fido2-cred** exits 1.

The options are as follows:

**-M** Tells **fido2-cred** to make a new credential on *device*.

**-V** Tells **fido2-cred** to verify a credential.

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

**-c** *cred\_protect*

If making a credential, set the credential's protection level to *cred\_protect*, where *cred\_protect* is the credential's protection level in decimal notation. Please refer to *<fido/param.h>* for the set of possible values. If verifying a credential, check whether the credential's protection level was signed by the authenticator as *cred\_protect*.

- d** Causes **fido2-cred** to emit debugging output on *stderr*.
- h** If making a credential, enable the FIDO2 hmac-secret extension. If verifying a credential, check whether the extension data bit was signed by the authenticator.
- i** *input\_file*  
Tells **fido2-cred** to read the parameters of the credential from *input\_file* instead of *stdin*.
- o** *output\_file*  
Tells **fido2-cred** to write output on *output\_file* instead of *stdout*.
- q** Tells **fido2-cred** to be quiet. If a PIN is required and **-q** is specified, **fido2-cred** will fail.
- r** Create a resident credential. Resident credentials are called "discoverable credentials" in CTAP 2.1.
- u** Create a U2F credential. By default, **fido2-cred** will use FIDO2 if supported by the authenticator, and fallback to U2F otherwise.
- v** If making a credential, request user verification. If verifying a credential, check whether the user verification bit was signed by the authenticator.

## INPUT FORMAT

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

When making a credential, **fido2-cred** expects its input to consist of:

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

When verifying a credential, **fido2-cred** expects its input to consist of:

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

6. attestation signature (base64 blob);
7. attestation certificate (optional, base64 blob).

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

## OUTPUT FORMAT

The output of **fido2-cred** consists of base64 blobs, UTF-8 strings, and PEM-encoded public keys separated by newline characters (`'\n'`).

Upon the successful generation of a credential, **fido2-cred** outputs:

1. client data hash (base64 blob);
2. relying party id (UTF-8 string);
3. credential format (UTF-8 string);
4. authenticator data (base64 blob);
5. credential id (base64 blob);
6. attestation signature (base64 blob);
7. attestation certificate, if present (base64 blob).
8. the credential's associated 32-byte symmetric key ("`largeBlobKey`"), if present (base64 blob).

Upon the successful verification of a credential, **fido2-cred** outputs:

1. credential id (base64 blob);
2. PEM-encoded credential key.

## EXAMPLES

Create a new *es256* credential on `/dev/hidraw5`, verify it, and save the id and the public key of the credential in *cred*:

```
$ echo credential challenge | openssl sha256 -binary | base64 > cred_param
$ echo relying party >> cred_param
$ echo user name >> cred_param
$ dd if=/dev/urandom bs=1 count=32 | base64 >> cred_param
$ fido2-cred -M -i cred_param /dev/hidraw5 | fido2-cred -V -o cred
```

## SEE ALSO

`fido2-assert(1)`, `fido2-token(1)`

## CAVEATS

Please note that **fido2-cred** handles Basic Attestation and Self Attestation transparently. In the case of

Basic Attestation, the validity of the authenticator's attestation certificate is *not* verified.