

**NAME**

**fido\_cred\_new**, **fido\_cred\_free**, **fido\_cred\_pin\_minlen**, **fido\_cred\_prot**, **fido\_cred\_fmt**, **fido\_cred\_rp\_id**, **fido\_cred\_rp\_name**, **fido\_cred\_user\_name**, **fido\_cred\_display\_name**, **fido\_cred\_authdata\_ptr**, **fido\_cred\_authdata\_raw\_ptr**, **fido\_cred\_clientdata\_hash\_ptr**, **fido\_cred\_id\_ptr**, **fido\_cred\_aaguid\_ptr**, **fido\_cred\_largeblob\_key\_ptr**, **fido\_cred\_pubkey\_ptr**, **fido\_cred\_sig\_ptr**, **fido\_cred\_user\_id\_ptr**, **fido\_cred\_x5c\_ptr**, **fido\_cred\_attstmt\_ptr**, **fido\_cred\_authdata\_len**, **fido\_cred\_authdata\_raw\_len**, **fido\_cred\_clientdata\_hash\_len**, **fido\_cred\_id\_len**, **fido\_cred\_aaguid\_len**, **fido\_cred\_largeblob\_key\_len**, **fido\_cred\_pubkey\_len**, **fido\_cred\_sig\_len**, **fido\_cred\_user\_id\_len**, **fido\_cred\_x5c\_len**, **fido\_cred\_attstmt\_len**, **fido\_cred\_type**, **fido\_cred\_flags**, **fido\_cred\_sigcount** - FIDO2 credential API

**SYNOPSIS**

```
#include <fido.h>
```

```
fido_cred_t *
```

```
fido_cred_new(void);
```

```
void
```

```
fido_cred_free(fido_cred_t **cred_p);
```

```
size_t
```

```
fido_cred_pin_minlen(const fido_cred_t *cred);
```

```
int
```

```
fido_cred_prot(const fido_cred_t *cred);
```

```
const char *
```

```
fido_cred_fmt(const fido_cred_t *cred);
```

```
const char *
```

```
fido_cred_rp_id(const fido_cred_t *cred);
```

```
const char *
```

```
fido_cred_rp_name(const fido_cred_t *cred);
```

```
const char *
```

```
fido_cred_user_name(const fido_cred_t *cred);
```

```
const char *
```

```
fido_cred_display_name(const fido_cred_t *cred);
```

*const unsigned char \**  
**fido\_cred\_authdata\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_authdata\_raw\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_clientdata\_hash\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_id\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_aaguid\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_largeblob\_key\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_pubkey\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_sig\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_user\_id\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_x5c\_ptr**(*const fido\_cred\_t \*cred*);

*const unsigned char \**  
**fido\_cred\_attstmt\_ptr**(*const fido\_cred\_t \*cred*);

*size\_t*  
**fido\_cred\_authdata\_len**(*const fido\_cred\_t \*cred*);

*size\_t*  
**fido\_cred\_authdata\_raw\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_clientdata\_hash\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_id\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_aaguid\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_largeblob\_key\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_pubkey\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_sig\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_user\_id\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_x5c\_len**(*const fido\_cred\_t \*cred*);

*size\_t*

**fido\_cred\_attstmt\_len**(*const fido\_cred\_t \*cred*);

*int*

**fido\_cred\_type**(*const fido\_cred\_t \*cred*);

*uint8\_t*

**fido\_cred\_flags**(*const fido\_cred\_t \*cred*);

*uint32\_t*

**fido\_cred\_sigcount**(*const fido\_cred\_t \*cred*);

## DESCRIPTION

FIDO2 credentials are abstracted in *libfido2* by the *fido\_cred\_t* type. The functions described in this page allow a *fido\_cred\_t* type to be allocated, deallocated, and inspected. For other operations on *fido\_cred\_t*, please refer to [fido\\_cred\\_set\\_authdata\(3\)](#), [fido\\_cred\\_exclude\(3\)](#), [fido\\_cred\\_verify\(3\)](#), and [fido\\_dev\\_make\\_cred\(3\)](#).

The **fidocred\_new()** function returns a pointer to a newly allocated, empty *fidocred\_t* type. If memory cannot be allocated, NULL is returned.

The **fidocred\_free()** function releases the memory backing *\*cred\_p*, where *\*cred\_p* must have been previously allocated by **fidocred\_new()**. On return, *\*cred\_p* is set to NULL. Either *cred\_p* or *\*cred\_p* may be NULL, in which case **fidocred\_free()** is a NOP.

If the CTAP 2.1 FIDO\_EXT\_MINPINLEN extension is enabled on *cred*, then the **fidocred\_pin\_minlen()** function returns the minimum PIN length of *cred*. Otherwise, **fidocred\_pin\_minlen()** returns zero. See **fidocred\_set\_pin\_minlen(3)** on how to enable this extension.

If the CTAP 2.1 FIDO\_EXT\_CRED\_PROTECT extension is enabled on *cred*, then the **fidocred\_prot()** function returns the protection of *cred*. Otherwise, **fidocred\_prot()** returns zero. See **fidocred\_set\_prot(3)** for the protection policies understood by *libfido2*.

The **fidocred\_fmt()** function returns a pointer to a NUL-terminated string containing the attestation statement format identifier of *cred*, or NULL if *cred* does not have a format set.

The **fidocred\_rp\_id()**, **fidocred\_rp\_name()**, **fidocred\_user\_name()**, and **fidocred\_display\_name()** functions return pointers to NUL-terminated strings holding the relying party ID, relying party name, user name, and user display name attributes of *cred*, or NULL if the respective entry is not set.

The **fidocred\_authdata\_ptr()**, **fidocred\_authdata\_raw\_ptr()**, **fidocred\_clientdata\_hash\_ptr()**, **fidocred\_id\_ptr()**, **fidocred\_aaguid\_ptr()**, **fidocred\_largeblob\_key\_ptr()**, **fidocred\_pubkey\_ptr()**, **fidocred\_sig\_ptr()**, **fidocred\_user\_id\_ptr()**, **fidocred\_x5c\_ptr()**, and **fidocred\_attstmt\_ptr()** functions return pointers to the CBOR-encoded and raw authenticator data, client data hash, ID, authenticator attestation GUID, "largeBlobKey", public key, signature, user ID, x509 certificate, and attestation statement parts of *cred*, or NULL if the respective entry is not set.

The corresponding length can be obtained by **fidocred\_authdata\_len()**, **fidocred\_authdata\_raw\_len()**, **fidocred\_clientdata\_hash\_len()**, **fidocred\_id\_len()**, **fidocred\_aaguid\_len()**, **fidocred\_largeblob\_key\_len()**, **fidocred\_pubkey\_len()**, **fidocred\_sig\_len()**, **fidocred\_user\_id\_len()**, **fidocred\_x5c\_len()**, and **fidocred\_attstmt\_len()**.

The authenticator data, x509 certificate, and signature parts of a credential are typically passed to a FIDO2 server for verification.

The **fidocred\_type()** function returns the COSE algorithm of *cred*.

The **fidocred\_flags()** function returns the authenticator data flags of *cred*.

The **fido\_cred\_sigcount()** function returns the authenticator data signature counter of *cred*.

## RETURN VALUES

The authenticator data returned by **fido\_cred\_authdata\_ptr()** is a CBOR-encoded byte string, as obtained from the authenticator. To obtain the decoded byte string, use **fido\_cred\_authdata\_raw\_ptr()**.

If not NULL, pointers returned by **fido\_cred\_fmt()**, **fido\_cred\_authdata\_ptr()**, **fido\_cred\_clientdata\_hash\_ptr()**, **fido\_cred\_id\_ptr()**, **fido\_cred\_aaguid\_ptr()**, **fido\_cred\_largeblob\_key\_ptr()**, **fido\_cred\_pubkey\_ptr()**, **fido\_cred\_sig\_ptr()**, and **fido\_cred\_x5c\_ptr()** are guaranteed to exist until any API function that takes *cred* without the *const* qualifier is invoked.

## SEE ALSO

**fido\_cred\_exclude(3)**, **fido\_cred\_set\_authdata(3)**, **fido\_cred\_set\_pin\_minlen(3)**, **fido\_cred\_set\_prot(3)**, **fido\_cred\_verify(3)**, **fido\_credman\_metadata\_new(3)**, **fido\_dev\_largeblob\_get(3)**, **fido\_dev\_make\_cred(3)**