NAME

fido_cbor_info_new, **fido_cbor_info_free**, **fido_dev_get_cbor_info**, **fido_cbor_info_aaguid_ptr**, **fido_cbor_info_extensions_ptr**, **fido_cbor_info_protocols_ptr**, **fido_cbor_info_transports_ptr**, **fido_cbor_info_versions_ptr**, **fido_cbor_info_options_name_ptr**, **fido_cbor_info_options_value_ptr**, **fido_cbor_info_algorithm_type**, **fido_cbor_info_algorithm_cose**, **fido_cbor_info_algorithm_count**, **fido_cbor_info_certs_name_ptr**, **fido_cbor_info_certs_value_ptr**, **fido_cbor_info_certs_len**, **fido_cbor_info_aaguid_len**, **fido_cbor_info_extensions_len**, **fido_cbor_info_protocols_len**, **fido_cbor_info_transports_len**, **fido_cbor_info_versions_len**, **fido_cbor_info_options_len**, **fido_cbor_info_maxmsgsiz**, **fido_cbor_info_maxcredbloblen**, **fido_cbor_info_maxcredcntlst**, **fido_cbor_info_maxcredidlen**, **fido_cbor_info_maxlargeblob**, **fido_cbor_info_maxrpid_minpinlen**, **fido_cbor_info_minpinlen**, **fido_cbor_info_fwversion**, **fido_cbor_info_uv_attempts**, **fido_cbor_info_uv_modality**, **fido_cbor_info_rk_remaining**, **fido_cbor_info_new_pin_required** - FIDO2 CBOR Info API

SYNOPSIS

#include <fido.h>

*fido_cbor_info_t **
fido_cbor_info_new(*void*);

void
fido_cbor_info_free(*fido_cbor_info_t **ci_p*);

int
fido_dev_get_cbor_info(*fido_dev_t *dev*, *fido_cbor_info_t *ci*);

*const unsigned char **
fido_cbor_info_aaguid_ptr(*const fido_cbor_info_t *ci*);

*char ***
fido_cbor_info_extensions_ptr(*const fido_cbor_info_t *ci*);

*const uint8_t **
fido_cbor_info_protocols_ptr(*const fido_cbor_info_t *ci*);

*char ***
fido_cbor_info_transports_ptr(*const fido_cbor_info_t *ci*);

*char ***
fido_cbor_info_versions_ptr(*const fido_cbor_info_t *ci*);

*char **
fido_cbor_info_options_name_ptr(*const fido_cbor_info_t *ci*);

*const bool **
fido_cbor_info_options_value_ptr(*const fido_cbor_info_t *ci*);

*const char **
fido_cbor_info_algorithm_type(*const fido_cbor_info_t *ci*, *size_t idx*);

int
fido_cbor_info_algorithm_cose(*const fido_cbor_info_t *ci*, *size_t idx*);

size_t
fido_cbor_info_algorithm_count(*const fido_cbor_info_t *ci*);

*char **
fido_cbor_info_certs_name_ptr(*const fido_cbor_info_t *ci*);

*const uint64_t **
fido_cbor_info_certs_value_ptr(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_certs_len(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_aaguid_len(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_extensions_len(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_protocols_len(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_transports_len(*const fido_cbor_info_t *ci*);

size_t
fido_cbor_info_versions_len(*const fido_cbor_info_t *ci*);

size_t

fido_cbor_info_options_len(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxmsgsiz(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxcredbloblen(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxcredcntlst(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxcredidlen(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxlargeblob(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_maxrpid_minpinlen(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_minpinlen(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_fwversion(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_uv_attempts(*const fido_cbor_info_t *ci*);

uint64_t
fido_cbor_info_uv_modality(*const fido_cbor_info_t *ci*);

int64_t
fido_cbor_info_rk_remaining(*const fido_cbor_info_t *ci*);

bool
fido_cbor_info_new_pin_required(*const fido_cbor_info_t *ci*);

DESCRIPTION

The **fido_cbor_info_new**() function returns a pointer to a newly allocated, empty *fido_cbor_info_t* type.

If memory cannot be allocated, NULL is returned.

The **fido_cbor_info_free**() function releases the memory backing **ci_p*, where **ci_p* must have been previously allocated by **fido_cbor_info_new**(). On return, **ci_p* is set to NULL. Either *ci_p* or **ci_p* may be NULL, in which case **fido_cbor_info_free**() is a NOP.

The **fido_dev_get_cbor_info**() function transmits a CTAP_CBOR_GETINFO command to *dev* and fills *ci* with attributes retrieved from the command's response. The **fido_dev_get_cbor_info**() function may block.

The **fido_cbor_info_aaguid_ptr**(), **fido_cbor_info_extensions_ptr**(), **fido_cbor_info_protocols_ptr**(), **fido_cbor_info_transports_ptr**(), and **fido_cbor_info_versions_ptr**() functions return pointers to the authenticator attestation GUID, supported extensions, PIN protocol, transports, and CTAP version strings of *ci*. The corresponding length of a given attribute can be obtained by **fido_cbor_info_aaguid_len**(), **fido_cbor_info_extensions_len**(), **fido_cbor_info_protocols_len**(), **fido_cbor_info_transports_len**(), or **fido_cbor_info_versions_len**().

The **fido_cbor_info_options_name_ptr**() and **fido_cbor_info_options_value_ptr**() functions return pointers to the array of option names and their respective values in *ci*. The length of the options array is returned by **fido_cbor_info_options_len**().

The **fido_cbor_info_algorithm_count**() function returns the number of supported algorithms in *ci*. The **fido_cbor_info_algorithm_cose**() function returns the COSE identifier of algorithm *idx* in *ci*, or 0 if the COSE identifier is unknown or unset. The **fido_cbor_info_algorithm_type**() function returns the type of algorithm *idx* in *ci*, or NULL if the type is unset. Please note that the first algorithm in *ci* has an *idx* (index) value of 0.

The **fido_cbor_info_certs_name_ptr**() and **fido_cbor_info_certs_value_ptr**() functions return pointers to the array of certification names and their respective values in *ci*. The length of the certifications array is returned by **fido_cbor_info_certs_len**().

The **fido_cbor_info_maxmsgsiz**() function returns the maximum message size attribute of *ci*.

The **fido_cbor_info_maxcredbloblen**() function returns the maximum "credBlob" length in bytes supported by the authenticator as reported in *ci*.

The **fido_cbor_info_maxcredcntlst**() function returns the maximum supported number of credentials in a single credential ID list as reported in *ci*.

The **fido_cbor_info_maxcredidlen**() function returns the maximum supported length of a credential ID

as reported in *ci*.

The **fido_cbor_info_maxrpid_minpinlen**() function returns the maximum number of RP IDs that may be passed to fido_dev_set_pin_minlen_rpid(3), as reported in *ci*. The minimum PIN length attribute is a CTAP 2.1 addition. If the attribute is not advertised by the authenticator, the **fido_cbor_info_maxrpid_minpinlen**() function returns zero.

The **fido_cbor_info_maxlargeblob**() function returns the maximum length in bytes of an authenticator's serialized largeBlob array as reported in *ci*.

The **fido_cbor_info_minpinlen**() function returns the minimum PIN length enforced by the authenticator as reported in *ci*. The minimum PIN length attribute is a CTAP 2.1 addition. If the attribute is not advertised by the authenticator, the **fido_cbor_info_minpinlen**() function returns zero.

The **fido_cbor_info_fwversion**() function returns the firmware version attribute of *ci*.

The **fido_cbor_info_uv_attempts**() function returns the number of UV attempts that the platform may attempt before falling back to PIN authentication. If 1, then all fido_dev_get_uv_retry_count(3) retries are handled internally by the authenticator and the platform may only attempt non-PIN UV once. The UV attempts attribute is a CTAP 2.1 addition. If the attribute is not advertised by the authenticator, the **fido_cbor_info_uv_attempts**() function returns zero.

The **fido_cbor_info_uv_modality**() function returns a bitmask representing different UV modes supported by the authenticator, as defined in the FIDO Registry of Predefined Values and reported in *ci*. See the *FIDO_UV_MODE_** definitions in <*fido/param.h*> for the set of values defined by libfido2 and a brief description of each. The UV modality attribute is a CTAP 2.1 addition. If the attribute is not advertised by the authenticator, the **fido_cbor_info_uv_modality**() function returns zero.

The **fido_cbor_info_rk_remaining**() function returns the estimated number of additional resident/discoverable credentials that can be stored on the authenticator as reported in *ci*. The estimated number of remaining resident credentials is a CTAP 2.1 addition. If the attribute is not advertised by the authenticator, the **fido_cbor_info_rk_remaining**() function returns -1.

The **fido_cbor_info_new_pin_required**() function returns whether a new PIN is required by the authenticator as reported in *ci*. If **fido_cbor_info_new_pin_required**() returns true, operations requiring PIN authentication will fail until a new PIN is set on the authenticator. The fido_dev_set_pin(3) function can be used to set a new PIN.

A complete example of how to use these functions can be found in the *example/info.c* file shipped with *libfido2*.

RETURN VALUES

The **fido_cbor_info_aaguid_ptr**(), **fido_cbor_info_extensions_ptr**(), **fido_cbor_info_protocols_ptr**(), **fido_cbor_info_transports_ptr**(), **fido_cbor_info_versions_ptr**(), **fido_cbor_info_options_name_ptr**(), and **fido_cbor_info_options_value_ptr**() functions return NULL if the respective field in *ci* is absent. If not NULL, returned pointers are guaranteed to exist until any API function that takes *ci* without the *const* qualifier is invoked.

SEE ALSO

fido_dev_get_uv_retry_count(3), fido_dev_open(3), fido_dev_set_pin(3), fido_dev_set_pin_minlen_rpid(3)

FIDO Registry of Predefined Values, https://fidoalliance.org/specs/common-specs/fido-registry-v2.2-rd-20210525.html, FIDO Alliance, 2021-05-25, Review Draft, Version 2.2.