NAME

mifare_desfire_connect, **mifare_desfire_disconnect**.
mifare_desfire_authenticate, **mifare_desfire_authenticate_aes**,
mifare_desfire_authenticate_iso, **mifare_desfire_change_key_settings**,
mifare_desfire_get_key_settings, **mifare_desfire_change_key**,
mifare_desfire_get_key_version, **mifare_desfire_create_application**,
mifare_desfire_create_application_3k3des,
mifare_desfire_create_application_aes,
mifare_desfire_create_application_iso,
mifare_desfire_create_application_3k3des_iso,
mifare_desfire_create_application_aes_iso,
mifare_desfire_delete_application, **mifare_desfire_get_application_ids**,
mifare_desfire_free_application_ids, **mifare_desfire_get_df_names**,
mifare_desfire_select_application, **mifare_desfire_format_picc**,
mifare_desfire_get_version, **mifare_desfire_free_mem**,
mifare_desfire_set_configuration, **mifare_desfire_set_default_key**,
mifare_desfire_set_ats, **mifare_desfire_get_card_uid**,
mifare_desfire_get_file_ids, **mifare_desfire_get_iso_file_ids**,
mifare_desfire_get_file_settings, **mifare_desfire_change_file_settings**,
mifare_desfire_create_std_data_file,
mifare_desfire_create_std_data_file_iso,
mifare_desfire_create_backup_data_file,
mifare_desfire_create_backup_data_file_iso,
mifare_desfire_create_value_file,
mifare_desfire_create_linear_record_file,
mifare_desfire_create_linear_record_file_iso,
mifare_desfire_create_cyclic_record_file,
mifare_desfire_create_cyclic_record_file_iso,
mifare_desfire_delete_file, **mifare_desfire_read_data**,
mifare_desfire_read_data_ex, **mifare_desfire_write_data**,
mifare_desfire_write_data_ex, **mifare_desfire_get_value**,
mifare_desfire_get_value_ex, **mifare_desfire_credit**,
mifare_desfire_credit_ex, **mifare_desfire_debit**, **mifare_desfire_debit_ex**,
mifare_desfire_limited_credit, **mifare_desfire_limited_credit_ex**,
mifare_desfire_write_record, **mifare_desfire_write_record_ex**,
mifare_desfire_read_records, **mifare_desfire_read_records_ex**,
mifare_desfire_clear_record_file, **mifare_desfire_commit_transaction**,
mifare_desfire_abort_transaction, — Mifare DESFire Manipulation Functions

LIBRARY

Mifare card manipulation library (libfreefare, −lfreefare)

SYNOPSIS

```
#include <freefare.h>

int
mifare_desfire_connect(MifareTag tag);

int
mifare_desfire_disconnect(MifareTag tag);

int
mifare_desfire_authenticate(MifareTag tag, uint8_t key_no,
    MifareDESFireKey key);

int
mifare_desfire_authenticate_aes(MifareTag tag, uint8_t key_no,
    MifareDESFireKey key);
```

int
mifare_desfire_authenticate_iso(*MifareTag tag*, *uint8_t key_no*, *MifareDESFireKey key*);

int
mifare_desfire_change_key_settings(*MifareTag tag*, *uint8_t settings*);

int
mifare_desfire_get_key_settings(*MifareTag tag*, *uint8_t *settings*, *uint8_t *max_keys*);

int
mifare_desfire_change_key(*MifareTag tag*, *uint8_t key_no*, *MifareDESFireKey new_key*, *MifareDESFireKey old_key*);

int
mifare_desfire_get_key_version(*MifareTag tag*, *uint8_t key_no*, *uint8_t *version*);

int
mifare_desfire_create_application(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*);

int
mifare_desfire_create_application_aes(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*);

int
mifare_desfire_create_application_3k3des(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*);

int
mifare_desfire_create_application_iso(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*, *int want_iso_file_identifiers*, *uint16_t iso_file_id*, *uint8_t *iso_file_name*, *size_t iso_file_name_len*);

int
mifare_desfire_create_application_3k3des_iso(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*, *int want_iso_file_identifiers*, *uint16_t iso_file_id*, *uint8_t *iso_file_name*, *size_t iso_file_name_len*);

int
mifare_desfire_create_application_aes_iso(*MifareTag tag*, *MifareDESFireAID aid*, *uint8_t settings*, *uint8_t key_no*, *int want_iso_file_identifiers*, *uint16_t iso_file_id*, *uint8_t *iso_file_name*, *size_t iso_file_name_len*);

int
mifare_desfire_delete_application(*MifareTag tag*, *MifareDESFireAID aid*);

int
mifare_desfire_get_application_ids(*MifareTag tag*, *MifareDESFireAID *aids[]*, *size_t *count*);

void
mifare_desfire_free_application_ids(*MifareDESFireAID aids[]*);

int
mifare_desfire_get_df_names(*MifareTag tag*, *MifareDESFireDF *dfs[]*, *size_t *count*);

int
mifare_desfire_select_application(*MifareTag tag*, *MifareDESFireAID aid*);

int
mifare_desfire_format_picc(*MifareTag tag*);

int
mifare_desfire_get_version(*MifareTag tag*,
*struct mifare_desfire_version_info *version_info*);

int
mifare_desfire_free_mem(*MifareTag tag*, *uint32_t *size*);

int
mifare_desfire_set_configuration(*MifareTag tag*, *bool disable_format*,
bool enable_random_uid);

int
mifare_desfire_set_default_key(*MifareTag tag*, *MifareDESFireKey key*);

int
mifare_desfire_set_ats(*MifareTag tag*, *uint8_t *ats*);

int
mifare_desfire_get_card_uid(*MifareTag tag*, *char **uid*);

int
mifare_desfire_get_file_ids(*MifareTag tag*, *uint8_t *files[]*,
*size_t *count*);

int
mifare_desfire_get_iso_file_ids(*MifareTag tag*, *uint16_t *files[]*,
*size_t *count*);

int
mifare_desfire_get_file_settings(*MifareTag tag*, *uint8_t file_no*,
*struct mifare_desfire_file_settings *settings*);

int
mifare_desfire_change_file_settings(*MifareTag tag*, *uint8_t file_no*,
uint8_t communication_settings, *uint16_t access_rights*);

int
mifare_desfire_create_std_data_file(*MifareTag tag*, *uint8_t file_no*,
uint8_t communication_settings, *uint16_t access_rights*,
uint32_t file_size);

int
mifare_desfire_create_std_data_file_iso(*MifareTag tag*, *uint8_t file_no*,
uint8_t communication_settings, *uint16_t access_rights*,
uint32_t file_size, *uint16_t iso_file_id*);

int
mifare_desfire_create_backup_data_file(*MifareTag tag*, *uint8_t file_no*,
uint8_t communication_settings, *uint16_t access_rights*,
uint32_t file_size);

int
mifare_desfire_create_backup_data_file_iso(*MifareTag tag*,
uint8_t file_no, *uint8_t communication_settings*,
uint16_t access_rights, *uint32_t file_size*, *uint16_t iso_file_id*);

int
mifare_desfire_create_value_file(*MifareTag tag*, *uint8_t file_no*,
    uint8_t communication_settings, *uint16_t access_rights*,
    int32_t lower_limit, *int32_t upper_limit*, *int32_t value*,
    uint8_t limited_credit_enable);

int
mifare_desfire_create_linear_record_file(*MifareTag tag*,
    uint8_t file_no, *uint8_t communication_settings*,
    uint16_t access_rights, *uint32_t record_size*,
    uint32_t max_number_of_records);

int
mifare_desfire_create_linear_record_file_iso(*MifareTag tag*,
    uint8_t file_no, *uint8_t communication_settings*,
    uint16_t access_rights, *uint32_t record_size*,
    uint32_t max_number_of_records);

int
mifare_desfire_create_cyclic_record_file(*MifareTag tag*,
    uint8_t file_no, *uint8_t communication_settings*,
    uint16_t access_rights, *uint32_t record_size*,
    uint32_t max_number_of_records, *uint16_t iso_file_id*);

int
mifare_desfire_create_cyclic_record_file_iso(*MifareTag tag*,
    uint8_t file_no, *uint8_t communication_settings*,
    uint16_t access_rights, *uint32_t record_size*,
    uint32_t max_number_of_records, *uint16_t iso_file_id*);

int
mifare_desfire_delete_file(*MifareTag tag*, *uint8_t file_no*);

ssize_t
mifare_desfire_read_data(*MifareTag tag*, *uint8_t file_no*, *off_t offset*,
    size_t length, *void *data*);

ssize_t
mifare_desfire_read_data_ex(*MifareTag tag*, *uint8_t file_no*,
    off_t offset, *size_t length*, *void *data*, *int cs*);

ssize_t
mifare_desfire_write_data(*MifareTag tag*, *uint8_t file_no*, *off_t offset*,
    size_t length, *void *data*);

ssize_t
mifare_desfire_write_data_ex(*MifareTag tag*, *uint8_t file_no*,
    off_t offset, *size_t length*, *void *data*, *int cs*);

int
mifare_desfire_get_value(*MifareTag tag*, *uint8_t file_no*, *int32_t *value*);

int
mifare_desfire_get_value_ex(*MifareTag tag*, *uint8_t file_no*,
    *int32_t *value*, *int cs*);

int
mifare_desfire_credit(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*);

int
mifare_desfire_credit_ex(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*,
    int cs);



int
mifare_desfire_debit(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*);

int
mifare_desfire_debit_ex(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*, *int cs*);

int
mifare_desfire_limited_credit(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*);

int
mifare_desfire_limited_credit_ex(*MifareTag tag*, *uint8_t file_no*, *int32_t amount*, *int cs*);

ssize_t
mifare_desfire_write_record(*MifareTag tag*, *uint8_t file_no*, *off_t offset*, *size_t length*, *void *data*);

ssize_t
mifare_desfire_write_record_ex(*MifareTag tag*, *uint8_t file_no*, *off_t offset*, *size_t length*, *void *data*, *int cs*);

ssize_t
mifare_desfire_read_records(*MifareTag tag*, *uint8_t file_no*, *off_t offset*, *size_t length*, *void *data*);

ssize_t
mifare_desfire_read_records_ex(*MifareTag tag*, *uint7_t file_no*, *off_t offset*, *size_t length*, *void *data*, *int cs*);

int
mifare_desfire_clear_record_file(*MifareTag tag*, *uint8_t file_no*);

int
mifare_desfire_commit_transaction(*MifareTag tag*);

int
mifare_desfire_abort_transaction(*MifareTag tag*);

DESCRIPTION

The **mifare_desfire_***() functions allows management of Mifare DESFire tags.

Card-level operations

The **mifare_desfire_connect**() and **mifare_desfire_disconnect**() functions activates and deactivates the provided *tag*. All **mifare_desfire_***() functions that operates on a *tag* require it to be on activated.

After activation, the selected application is the master application. It is possible to select another application using the **mifare_desfire_select_application**() function (see bellow).

The **mifare_desfire_get_version**() function retrieve various information about the provided *tag*, including UID, batch number, production date, and hardware and software information. Refer to the freefare.h header file for details about the *version_info* field.

The **mifare_desfire_free_mem**() functions returns the *size* of the free memory on the PICC (in bytes).

The **mifare_desfire_set_configuration**() function can be used to deactivate the format function when setting *disable_format* to a value different from 0, or swicth the card to use random UDI setting *enable_random_uid* to a non-zero value.



The *mifare_desfire_set_default_key* function sets the *key* argument as the default key for new applications.

The **mifare_desfire_set_ats** function replace the ATS bytes returned by PICC when it is selected.

The **mifare_desfire_get_card_uid** function can be used with a PICC configured for using random UID to retrieve the original UID of the target.

The **mifare_desfire_format_picc**() function resets *tag* to factory defaults. For this function to work, a previous authentication with the card master key is required.

Application-level operations

The **mifare_desfire_select_application**() function makes the application identified by *aid* the active one. Further file operations will be performed in the context of this application. After a call to *mifare_desfire_connect*, the default application is the card master application. It can be selected again calling the **mifare_desfire_select_application**() function either with an *aid* with all its fields set to 0, or by providing the NULL *aid*.

The **mifare_desfire_authenticate**() function performs an authentication using the key number *key_no* on the card and the *key* (3)DES key on *tag*.

The **mifare_desfire_authenticate_aes**() function performs an authentication using an AES *key*.

The **mifare_desfire_authenticate_iso**() function performs an ISO authentication using either a 3DES or a 3K3DES *key*.

The **mifare_desfire_get_key_settings**() function, returns the *settings* and the number of keys *max_keys* of the selected application.

The **mifare_desfire_change_key_settings**() function changes the selected application settings to *settings*. The application number of keys cannot be changed after the application has been created.

The **mifare_desfire_change_key**() changes the key *key_no* from *old_key* to *new_key* on *tag*. Depending on the application settings, a previous authentication with the same key or another key may be required.

The **mifare_desfire_get_key_version**() function retrieves the *version* of the key with number *key_no* of the selected application.

The **mifare_desfire_create_application**() function, creates an application with AID *aid*, the *settings* key settings and *key_no* authentication keys. Authentication keys are set to 0 after creation.

The **mifare_desfire_create_application_3k3des**() and **mifare_desfire_create_application_aes**() functions acts as the **mifare_desfire_create_application**() function except that the whole application is configured to use 3K3DES or AES keys. It is possible to achive the same result using the **mifare_desfire_create_application**() function and ORing the *key_no* argument with *APPLICATION_CRYPTO_3K3DES* or *APPLICATION_CRYPTO_AES* respectively.

The **mifare_desfire_create_application_iso**() acts as the **mifare_desfire_create_application**() function but allows one to specify if the created files within the application shall have an ISO file identifier setting *want_iso_file_identifiers* to a non-NULL value, a DF can be provided using *iso_file_id*, as long as an optional file name *iso_file_name* of length *iso_file_name_len* (in bytes).

The **mifare_desfire_create_application_3k3des_iso**() and **mifare_desfire_create_application_aes_iso**() function acts as the regular **mifare_desfire_create_application_3k3des**() and **mifare_desfire_create_application_aes**() functions, providing the same extensions ISO parameters of **mifare_desfire_create_application_iso**().



The **mifare_desfire_delete_application**() deletes the application identified by AID *aid*.

The **mifare_desfire_get_application_ids**() function returns a list of all applications of the card. The *aids* array has to be freed after usage calling **mifare_desfire_free_application_ids**().

The **mifare_desfire_get_df_names**() retrieves the list of DF *dfs* from *tag* and set *count* to the number of idems in the allocated array. Memory has to be freed by the user using free(3).

File-level operations

The **mifare_desfire_get_file_ids**() function returns the list of *count* files in the selected application as *files*. The memory allocated for *files* has to be reclaimed using free(3).

The **mifare_desfire_get_iso_file_ids**() function returns the list of *count* file ISO identifiers as *files*. The memory allocated for *files* has to be reclaimed using free(3).

The **mifare_desfire_get_file_settings**() function retrieves the *settings* of the file *file_no* of the selected application of *tag*.

The **mifare_desfire_change_file_settings**() function change the *communication_settings* and *access_rights* of the file *file_no* of the selected application of *tag*.

The **mifare_desfire_create_**∗() family of functions create a new file *file_no* with the provided *communication_settings* and *access_rights* on *tag.*

mifare_desfire_create_std_data_file()
creates a standard data file of size *file_size*.

mifare_desfire_create_backup_data_file()
creates a backup data file of size *file_size*.

mifare_desfire_create_value_file()
creates a value file of value *value* constrained in the range *lower_limit upper_limit*, and with the *limited_credit_enable* settings.

mifare_desfire_create_linear_record_file()
creates a linear record file that can hold *max_number_of_records* records of size *record_size*.

mifare_desfire_create_cyclic_record_file()
creates a cyclic record file that can hold *max_number_of_records* records of size *record_size*.

The **mifare_desfire_create_**∗**_iso**() family of functions acts as the functions without the *_iso* suffix but provide an additionnal argument *iso_file_id*.

The **mifare_desfire_delete_file**() removes the file *file_no* from the selected application of *tag*.

Data-level operations

The **mifare_desfire_read_data**() function reads *length* bytes of data from offset *offset* of the file *file_no* and copies it to *data*. If *length* is set to 0, the file is read to end. The function returns the number of bytes read.

The **mifare_desfire_write_data**() function writes *length* bytes of data from offset *offset* of the file *file_no* and copies it to *data*. The function returns the number of bytes written.

The **mifare_desfire_get_value**() reads the *value* of the file *file_no* of the selected application.

The **mifare_desfire_credit**() function adds *amount* to the value of the file *file_no* of the selected application.



The **mifare_desfire_debit**() function substracts *amount* to the value of the file *file_no* of the selected application.

to the value of the file *file_no* of the selected application.

The **mifare_desfire_limited_credit**() function adds *amount* to the value of the file *file_no* of the selected application.

The **mifare_desfire_write_record**() function writes *length* records starting at record *offset* of *data* in the file *file_no* and returns the number of bytes written.

The **mifare_desfire_read_records**() function reads *length* records starting at record *offset* from the file *file_no* and copy them to *data*, returning the number of bytes read.

The **mifare_desfire_clear_record_file**() function erase all records from the file *file_no* of the selected application.

The **mifare_desfire_commit_transaction**() validates the set of pending changes on the *tag*, while the **mifare_desfire_abort_transaction**() rollbacks the changes.

All data-manipulation functions that read data from and write data to files come with an ∗**_ex**() variant (e.g. **mifare_desfire_read_data_ex**()) which accepts an extra parameter *cs* that defines the communication settings to use. If not provided, the library will try to read-out this value from the file's configuration. Because reading this information may be denied, the ∗**_ex**() variant of functions still allows using the library for advanced usage.

RETURN VALUES

Unless stated otherwise, all other functions return a value greater than or equal to *0* on success or *-1* on failure.

SEE ALSO

freefare(3)

AUTHORS

Romain Tartiere ⟨romain AT il4p DOT org⟩