NAME

ldap_modify_ext, ldap_modify_ext_s - Perform an LDAP modify operation

LIBRARY

OpenLDAP LDAP (libldap, -lldap)

SYNOPSIS

```
#include <ldap.h>

int ldap_modify_ext(
        LDAP *ld,
        char *dn,
        LDAPMod *mods[],
        LDAPControl **sctrls,
        LDAPControl **cctrls,
        int *msgidp );

int ldap_modify_ext_s(
        LDAP *ld,
        char *dn,
        LDAPMod *mods[],
        LDAPControl **sctrls,
        LDAPControl **cctrls );

void ldap_mods_free(
        LDAPMod **mods,
        int freemods );
```

DESCRIPTION

The routine **ldap_modify_ext_s()** is used to perform an LDAP modify operation. *dn* is the DN of the entry to modify, and *mods* is a null-terminated array of modifications to make to the entry. Each element of the *mods* array is a pointer to an LDAPMod structure, which is defined below.

```
typedef struct ldapmod {
    int mod_op;
    char *mod_type;
    union {
        char **modv_strvals;
        struct berval **modv_bvals;
    } mod_vals;
```

```
} LDAPMod;
#define mod_values mod_vals.modv_strvals
#define mod_bvalues mod_vals.modv_bvals
```

The *mod_op* field is used to specify the type of modification to perform and should be one of LDAP_MOD_ADD, LDAP_MOD_DELETE, or LDAP_MOD_REPLACE. The *mod_type* and *mod_values* fields specify the attribute type to modify and a null-terminated array of values to add, delete, or replace respectively.

If you need to specify a non-string value (e.g., to add a photo or audio attribute value), you should set *mod_op* to the logical OR of the operation as above (e.g., LDAP_MOD_REPLACE) and the constant LDAP_MOD_BVALUES. In this case, *mod_bvalues* should be used instead of *mod_values*, and it should point to a null-terminated array of struct bervals, as defined in <lber.h>.

For LDAP_MOD_ADD modifications, the given values are added to the entry, creating the attribute if necessary. For LDAP_MOD_DELETE modifications, the given values are deleted from the entry, removing the attribute if no values remain. If the entire attribute is to be deleted, the *mod_values* field should be set to NULL. For LDAP_MOD_REPLACE modifications, the attribute will have the listed values after the modification, having been created if necessary. All modifications are performed in the order in which they are listed.

ldap_mods_free() can be used to free each element of a NULL-terminated array of mod structures. If *freemods* is non-zero, the *mods* pointer itself is freed as well.

ldap_modify_ext_s() returns a code indicating success or, in the case of failure, indicating the nature of the failure. See **ldap_error**(3) for details

The **ldap_modify_ext()** operation works the same way as **ldap_modify_ext_s()**, except that it is asynchronous. The integer that *msgidp* points to is set to the message id of the modify request. The result of the operation can be obtained by calling **ldap_result**(3).

Both **ldap_modify_ext()** and **ldap_modify_ext_s()** allows server and client controls to be passed in via the sctrls and cctrls parameters, respectively.

DEPRECATED INTERFACES

The **ldap_modify()** and **ldap_modify_s()** routines are deprecated in favor of the **ldap_modify_ext()** and **ldap_modify_ext_s()** routines, respectively.

Deprecated interfaces generally remain in the library. The macro LDAP_DEPRECATED can be defined to a non-zero value (e.g., -DLDAP_DEPRECATED=1) when compiling program designed to

use deprecated interfaces. It is recommended that developers writing new programs, or updating old programs, avoid use of deprecated interfaces. Over time, it is expected that documentation (and, eventually, support) for deprecated interfaces to be eliminated.

SEE ALSO

ldap(3), **ldap_error**(3),

ACKNOWLEDGEMENTS

OpenLDAP Software is developed and maintained by The OpenLDAP Project <http://www.openldap.org/>. **OpenLDAP Software** is derived from the University of Michigan LDAP 3.3 Release.