NAME

getexecattr, free_execattr, setexecattr, endexecattr, getexecuser, getexecprof, match_execattr — get execution profile entry

SYNOPSIS

cc [*flag* ...] *file* ... **–lsecdb –lsocket –lnsl –lintl** [*library* ...]

#include <exec_attr.h>

#include <secdb.h>

execattr_t *getexecattr(void);

void free_execattr(execattr_t **ep*);

void setexecattr(void);

void endexecattr(void);

execattr_t *getexecuser(const char **username*, **const char** **type*,**const char** **id*, **int** *search_flag);*

execattr_t *getexecprof(const char **profname*, **const char** **type*,**const char** **id*, **int** *search_flag);*

execattr_t *match_execattr(execattr_t **ep*, **char** **profname*,**char** **type*, **char** **id);*

DESCRIPTION

The **getexecattr**( ) function returns a single **exec_attr** entry. Entries can come from any of the sources specified in the **nsswitch.conf**(4) file.

Successive calls to **getexecattr**( ) return either successive **exec_attr** entries or **NULL**. Because **getexecattr**( ) always returns a single entry, the **next** pointer in the **execattr_t** data structure points to **NULL**.

The internal representation of an **exec_attr** entry is an **execattr_t** structure defined in **<exec_attr.h>** with the following members:

```
char *name; /* name of the profile */
char *type; /* type of profile */
char *policy; /* policy under which the attributes are relevant*/
char *res1; /* reserved for future use */
char *res2; /* reserved for future use */
char *id; /* unique identifier */
kva_t*attr; /* attributes */
struct execattr_s *next; /* optional pointer to next profile */
```

The **free_execattr** function call releases memory. It follows the **next** pointers in the **execattr_t** structure so that the entire linked list is released.

The **setexecattr**( ) function “rewinds” to the beginning of the enumeration of **exec_attr** entries. Calls to **getexecuser**( ) can leave the enumeration in an indeterminate state. Therefore, **setexecattr**( ) should be called before the first call to **getexecattr**( ).

The **endexecattr**( ) function can be called to indicate that **exec_attr** processing is complete; the library can then close any open **exec_attr** file, deallocate any internal storage, and so forth.

The **getexecuser**( ) function is used to return a linked list of entries filtered by the function's arguments. Only entries assigned to the specified **username**, as described in the **passwd**(4) database, and containing the specified **type** and **id**, as described in the **exec_attr**(4) database, are placed in the list. **getexecuser**( ) is different from the other functions in its family because it spans two databases: **getexecuser**( ) looks up the list of profiles assigned to a user in the **user_attr** database then looks up each profile in the **exec_attr** database.

The **getexecprof**( ) function is used to return a linked list of entries that have components matching the functions arguments. Only entries in the database matching the argument **profname**, as described in **exec_attr**, and containing the **type** and **id**, also described in **exec_attr**, are placed in the list.

Using **getexecuser**( ) and **getexecprof**( ), programmers can search for any **type** argument, such as the manifest constant KV_COMMAND. The arguments are logically AND'd together so that only entries exactly matching all of the arguments are returned. Wildcard matching applies if there is no exact match for an id. Any argument can be assigned the **NULL** value to indicate that it is not used as part of the matching criteria. The **search_flag** controls whether the function returns the first match (GET_ONE), setting the **next** pointer to **NULL** or all matching entries (GET_ALL), using the **next** pointer to create a linked list of all entries that meet the search criteria. See **EXAMPLES.**

Once a list of entries is returned by **getexecuser**( ) or **getexecprof**( ), the convenience function **match_execattr**( ) can be used to identify an individual entry. **match_execattr**( ) returns a pointer to the individual element with the same profile name (**profname**), type name (**type**), and **id**. Function parameters set to **NULL** are not used as part of the matching criteria. In the event that multiple entries meet the matching criteria, only a pointer to the first entry is returned. **kv_match**( ) can be used to look up a key in a key-value array.

RETURN VALUES

Those functions returning data only return data related to the active policy. The **getexecattr**( ) function returns a pointer to a **execattr_t** if it successfully enumerates an entry; otherwise it returns **NULL**, indicating the end of the enumeration.

Individual attributes may be referenced in the **attr** structure by calling the **kva_match**(3) function. See **getauthattr**(3)**.**

FILES

/etc/nsswitch.conf	stores configuration file lookup information for the name server switch
/etc/user_attr	stores extended user attributes
/etc/security/exec_attr	stores execution profiles

EXAMPLES

The following finds all profiles that have the **ping** command.

```
if ((execprof=getexecprof(NULL, KV_COMMAND, "/usr/sbin/ping",
    GET_ONE)) == NULL) {
        /* do error */
}
```

The following finds the entry for the **ping** command in the Network Administration Profile.

```
if ((execprof=getexecprof("Network Administration", KV_COMMAND,
    "/usr/sbin/ping", GET_ALL))==NULL) {
        /* do error */
}
```

The following tells everything that can be done in the Filesystem Security profile.

```
if ((execprof=getexecprof("Filesystem Security", KV_NULL, NULL,
    GET_ALL))==NULL)) {
        /* do error */
}
```

The following tells if the **tar** command is in a profile assigned to user wetmore.

```
if ((execprof=getexecuser("wetmore", KV_COMMAND, "/usr/bin/tar",
    GET_ONE))==NULL) {
        /* do error */
}
```

NOTE: If there is no exact profile entry, the example above returns the wildcard (*), if there is one defined.

ATTRIBUTES

See **attributes**(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
MT-Level	MT-Safe.

SEE ALSO

getauthattr(3), **getuserattr**(3), **kva_match**(3), **exec_attr**(4), and **user_attr**(4)

NOTES

getexecattr( ), **getexecuser**( ), and **getexecprof**( ) all allocate memory for the pointers they return. This memory should be de-allocated with the **free_execattr**( ) call. The **match_execattr**( ) function does not allocate any memory. Therefore, pointers returned by this function should not be de-allocated.

Programs that use the interfaces described in this manual page cannot be linked statically, since the implementations of these functions employ dynamic loading and linking of shared objects at run time. Note that these interfaces are reentrant even though they do not use the _r suffix naming convention.