

NAME	getprofstr, putprofstr, setprofstr, endprofstr, getprofstrbyname, free_profstr – get Trusted Solaris 2.x user profile description
SYNOPSIS	<pre>cc [<i>flag ...</i>] <i>file ...</i> -ltsol -ltsol -lnsl -lcmd [<i>library ...</i>] #include <tsol/prof.h> profstr_t *getprofstrbyname(char *name, int src); profstr_t *getprofstr(int src); int putprofstr(profstr_t *res, int src); int setprofstr(int stayopen, int src); int endprofstr(int src); void free_profstr(profstr_t *profstr);</pre>
MT-LEVEL	MT-Safe
AVAILABILITY	SUNWtsolu
DESCRIPTION	<p>These functions are used to obtain entries describing Trusted Solaris 2.x user profiles from the tsolprof NIS+ database or the /etc/security/tsol/tsoluser file.</p> <p>getprofbyname() searches for information for a profile with the specified profile name given by the parameter <i>name</i>.</p> <p>The functions setprofstr(), getprofstr(), and endprofstr() are used to enumerate profile entries from the database. setprofstr() sets (or resets) the enumeration to the beginning of the set of Trusted Solaris profile entries. This function should be called before the first call to getprofstr(). A call to getprofbyname() leaves the enumeration position in an indeterminate state. If the <i>stayopen</i> flag is non-zero, the system may keep allocated resources such as open file descriptors until a subsequent call to endprofstr().</p> <p>Successive calls to getprofstr() return either successive entries or return NULL, indicating the end of the enumeration.</p> <p>endprofstr() may be called to indicate that the caller expects to do no further profile stry retrieval operations; the system may then deallocate resources it was using. It is still allowed, but possibly less efficient, for the process to call more profile stry retrieval functions after calling endprofstr().</p> <p>The functions getprofstrbyname() and getprofstr() are reentrant interfaces that allocate memory to store returned results, and are safe for use in both single-threaded and multithreaded applications. The function free_profstr() should be used to free the pointers returned by either getprofstrbyname() or getprofstr().</p>

The parameter *name* must be a pointer to the profile name in the form of a null-terminated character-string.

The parameter *src* may be set to any of `TSOL_SRC_FILES`, `TSOL_SRC_NISPLUS`, or `TSOL_SRC_SWITCH`, which are defined in `<tsol/tsol.h>`. For most applications the *src* parameter should be set to `TSOL_SRC_SWITCH`, indicating that the system should use the `/etc/nsswitch.conf` file to determine the ultimate source of the database. However, in certain administrative application, it may be prudent to use the `TSOL_SRC_FILES` option to for a read from the `/etc/security/tsol/tsoluser` file or the `TSOL_SRC_NISPLUS` option to force a read from the `tsoluser` NIS+ database.

The function `putprofstr` replaces an existing profile entry or adds a new entry if the profile name does not already exist. Currently the *src* parameter treated as `TSOL_SRC_NISPLUS`, forcing all information to be written to the NIS+ table. Use of files or of `nsswitch.conf(4)` may be supported in future releases.

For enumeration in multithreaded applications, the position within the enumeration is a process-wide property shared by all threads. `setprofstr()` may be used in a multithreaded application but resets the enumeration position for all threads. If multiple threads interleave calls to `getprofstr()`, the threads enumerate disjoint subsets of the `tsolprof(4TSOL)` database.

RETURN VALUES

User entries are represented by the `struct profstr_t` structure defined in `<tsol/prof.h>`:

```
typedef struct profstr_s {
    char *   name;      /* name of profile */
    char *   desc;      /* description */
    char *   auths;     /* comma separated list of authorization numbers */
    char *   actions;   /* semicolon separated action descriptions*/
    char *   cmds;      /* semicolon separated command descriptions*/
} profstr_t;
```

The function `getprofstrbyname()` returns a pointer to a `profstr_t` if it successfully locates the requested entry; otherwise it returns `NULL`.

The function `getprofstr()` returns a pointer to a `struct profstr_t` if it successfully enumerates an entry; otherwise it returns `NULL`, indicating the end of the enumeration.

The function `putprofstr()` return `0` on success.

ERRORS

The functions `getprofstrbyname()` and `getprofstr()` return `NULL` on on a failure.

NOTES

Programs that use the interfaces described here cannot be linked statically since the implementation of these functions employ dynamic loading and linking of shared objects at run time.

When compiling multithreaded applications, see **Intro (3)**, *Notes On Multithread Applications*, for information about the use of the `_REENTRANT` flag.