NAME

exec_attr – execution profiles database

SYNOPSIS

/etc/security/exec_attr

DESCRIPTION

/etc/security/exec_attr is a local database that specifies the execution attributes associated with profiles. The **exec_attr** file can be used with other sources for execution profiles, including the **exec_attr** NIS map and NIS+ table. Programs use the **getexecattr**(3) routines to access this information.

The search order for multiple execution profile sources is specified in the **/etc/nsswitch.conf** file, as described in the **nsswitch.conf**(4) man page.

A profile is a logical grouping of authorizations and commands that is interpreted by a profile shell to form a secure execution environment. The shells that interpret profiles are **pfcsh**, **pfksh**, and **pfsh**. See the **pfsh**(1) man page. Each user's account is assigned zero or more profiles in the **user_attr**(4) database file.

Each entry in the **exec_attr** database consists of one line of text containing seven fields separated by colons (:). Line continuations using the backslash (\) character are permitted. The basic format of each entry is:

```
name:policy:type:res1:res2:id:attr
```

name The name of the profile. Profile names are case-sensitive.

policy The policy that is associated with the profile entry. The only valid policy is **suser**.

type The type of object defined in the profile. The only valid type is: **cmd**.

res1 Reserved for future use.

res2 Reserved for future use.

id A string that uniquely identifies the object described by the profile. For a profile of type **cmd,** the id is either the full path to the command or the asterisk (∗) symbol, which is used to allow all commands. An asterisk that replaces the filename component in a pathname indicates all files in a particular directory. To specify arguments, the pathname should point to a shell script written to execute the command with the desired arguments.

attr An optional list of semicolon (;) separated key-value pairs that describe the security attributes to apply to the object upon execution. Zero or more keys may be specified. The list of valid key words depends on the policy enforced. The following key words are valid: **euid**, **uid, egid,** and **gid**.

euid and **uid** contain a single user name or a numeric user ID. Commands designated with **euid** run with the effective UID indicated, which is similar to setting the setuid bit on an executable file. Commands designated with **uid** run with both the real and effective UIDs. Setting **uid** may be more appropriate than setting the **euid** on privileged shell scripts.

egid and **gid** contain a single group name or a numeric group ID. Commands designated with **egid** run with the effective GID indicated, which is similar to setting the setgid bit on a file. Commands designated with **gid** run with both the real and effective GIDs. Setting **gid** may be more appropriate than setting **guid** on privileged shell scripts.

EXAMPLE

The following example shows the **audit** command specified in the Audit Control profile to execute with an effective user ID of root and effective group ID of bin.

Audit Control:suser:cmd:::/etc/init.d/audit:euid=0;egid=3

With the **nsswitch.conf** entry shown below, the local **exec_attr** file is consulted before the **exec_attr** NIS+ table.

exec_attr: files nisplus

FILES

/etc/nsswitch.conf

/etc/user_attr

/etc/security/exec_attr

CAVEATS

When deciding which authorization source to use (see **DESCRIPTION),** keep in mind that NIS+ provides stronger authentication than NIS.

Because the list of legal keys is likely to expand, any code that parses this database must be written to ignore unknown key-value pairs without error. When any new keywords are created, the names should be prefixed with a unique string, such as the company's stock symbol, to avoid potential naming conflicts.

The following characters are used in describing the database format and must be escaped with a backslash if used as data: colon (:), semicolon (;), equals (=), and backslash (\).

SEE ALSO

auths(1), **profiles**(1), **roles**(1), **makedbm**(1M), **getauthattr**(3), **getauusernam**(3), **getexecattr**(3), **getprofattr**(3), **getuserattr**(3), **kva_match**(3), **auth_attr**(4), **prof_attr**(4), **user_attr**(4)