NAME

subusers − an overview of the subuser toolkit

DESCRIPTION

The subuser toolkit gives users the opportunity to protect themselves from potentially malicious code that they want to run, by giving them the ability to create protection domains at any time. The protection domains provided are conventional Unix accounts, and the protection comes from the standard Unix user isolation semantics.

The toolkit keeps all the information it needs about the ancestry relationships in one file, **/etc/subusers**. See **/etc/subusers**(5) for information on its format. Most of the utilities are simply wrappers around the standard Unix utilities after which they are named, that just check this file to make sure the calling user is an ancestor of the named user.

QUALIFICATION

Users can be thought of as having either a "fully qualified" user name or a "local" user name. If alice has a subuser bob, and bob has a subuser charlie, then charlie's fully qualified user name is alice_bob_charlie. To alice, charlie's local user name is bob_charlie, and to bob, charlie's local user name is just charlie. The tools expect local user names as commandline arguments.

TYPICAL USAGE

A typical duty cycle begins when the user decides they want to create a subuser to insulate himself from some program. The user uses **subuseradd** to create a subuser. This creates a new Unix user with all the rimmings, including an entry in **/etc/passwd** and a home directory. **subuseradd** really merely calls **useradd**, so it should observe any oddities of your distribution.

In order to run something as the subuser, the user can invoke either **subusersu** or **subusersudo**. These programs function somewhat like the standard programs that they are named after, except they additionally perform a check against **/etc/subusers** to see if the calling user is an ancestor of the target user.

Once the subuser has performed some computation, it is common for the parent user to want access to some output files. This is, however, impossible because of the very semantics that protect users from each other. So, in order to get access to files owned by a subuser, the toolkit provides **subuserchown**, which allows users to change the ownership of files belonging to their descendants.

In the event a user wants to delete a subuser, the tool **subuserdel** deletes the subuser and all its files. See subuserdel(1) for a discussion of the current limitations of this tool.

FILES

/etc/subusers
: the file containing the ancestry relationships between users.

SEE ALSO

subuseradd(1) subuserdel(1) subuserchown(1) subusersu(1) subusersudo(1)