NAME

capsh - capability shell wrapper

SYNOPSIS

capsh [*OPTION*]...

DESCRIPTION

Linux capability support and use can be explored and constrained with this tool. This tool provides a handy wrapper for certain types of capability testing and environment creation. It also provides some debugging features useful for summarizing capability state.

OPTIONS

The tool takes a number of optional arguments, acting on them in the order they are provided. They are as follows:

--print Display prevailing capability and related state.

-- *[args]* Execute **/bin/bash** with trailing arguments. Note, you can use **-c 'command to execute'** for specific commands.

== Execute **capsh** again with remaining arguments. Useful for testing **exec**() behavior.

--caps=*cap-set* Set the prevailing process capabilities to those specified by *cap-set*. Where *cap-set* is a text-representation of capability state as per **cap_from_text(3).**

--drop=*cap-list* Remove the listed capabilities from the prevailing bounding set. The capabilities are a comma separated list of capabilities as recognized by the **cap_from_name(3)** function. Use of this feature requires that the capsh program is operating with **CAP_SETPCAP** in its effective set.

--inh=*cap-list* Set the inheritable set of capabilities for the current process to equal those provided in the comma separated list. For this action to succeed, the prevailing process should already have each of these capabilities in the union of the current inheritable and permitted capability sets, or the capsh program is operating with **CAP_SETPCAP** in its effective set.

--user=*username* Assume the identity of the named user. That is, look up the user's *uid* and *gid* with getpwuid(3) and their group memberships with getgrouplist(3) and set them all.

--uid=*id* Force all **uid** values to equal *id* using the setuid(2) system call.

--gid=<*id***>** Force all **gid** values to equal *id* using the setgid(2) system call.

--groups=<*id-list***>** Set the supplementary groups to the numerical list provided. The groups are set with the setgroups(2) system call.

--keep=<*0|1***>** In a non-pure capability mode, the kernel provides liberal privilege to the super-user. However, it is normally the case that when the super-user changes *uid* to some lesser user, then capabilities are dropped. For these situations, the kernel can permit the process to retain its capabilities after a setuid(2) system call. This feature is known as *keep-caps* support. The way to activate it using this script is with this argument. Setting the value to 1 will cause *keep-caps* to be active. Setting it to 0 will cause keep-caps to deactivate for the current process. In all cases, *keep-caps* is deactivated when an **exec**() is performed. See **--secbits** for ways to disable this feature.

--secbits=*N* XXX - need to document this feature.

--chroot=*path* Execute the chroot(2) system call with the new root-directory (/) equal to *path*. This operation requires**CAP_SYS_CHR OOT** to be in effect.

--forkfor=*sec*

--killit=*sig*

--decode=*N* This is a convenience feature. If you look at **/proc/1/status** there are some capability related fields of the following form:

CapInh: 0000000000000000 CapPrm: ffffffffffffffff CapEff: fffffffffffffeff CapBnd: ffffffffffffffff

This option provides a quick way to decode a capability vector represented in this form. For example, the missing capability from this effective set is 0x0100. By running:

capsh --decode=0x0100

we observe that the missing capability is: **cap_setpcap**.

--supports=*xxx* As the kernel evolves, more capabilities are added. This option can be used to verify the existence of a capability on the system. For example, **--supports=***cap_syslog* will cause capsh to promptly exit with a status of 1 when run on kernel 2.6.27. However, when run on kernel 2.6.38 it will silently succeed.

EXIT STATUS

Following successful execution the tool exits with status 0. Following an error, the tool immediately exits with status 1.

AUTHOR

Written by Andrew G. Morgan <morgan@kernel.org>.

REPORTING BUGS

Please report bugs to the author.

SEE ALSO

libcap(3), getcap(8), setcap(8) and capabilities(7).