Suid és sgid bit

A suid egy rövidítés, a „Set User Identification" kifejezés rövidítése. Azt jelenti, hogy a „felhasználói azonosító megváltoztatása". Ennek megértéséhez tudnunk kell azt, hogy idõnként szükség van arra, hogy egy egyszerû felhasználó egy privilégizált felhasználó jogaival rendelkezzen. Talán a legegyszerûbb eset a jelszó megváltoztatása. Egy egyszerû felhasználó nem írhatja közvetlenül a rendszer jelszófájlját, hiszen akkor bármikor korlátlan jogokhoz juthatna, de a saját jelszavát meg kell tudnia változtatni. Ehhez viszont írnia kell a jelszófájlba. Ezt az ellentmondást oldják fel úgy, hogy a programot ruházzák fel privilégizált jogokkal, a suid bit beállításával.

A *passwd* parancs engedélyei a következõk:

```
-rwsr-xr-x   1 root     root        28896 Jul 17  1998 /usr/bin/passwd
```

Látható, hogy a suid bit be van kapcsolva, így futásának idejére az õt futtató felhasználó rendszergazdai jogkört kap, tehát a root jogosultságaival olvassa és írja a */etc/passwd* fájlt (mivel a root tulajdonában van a fájl).

A Unix típusú rendszerekben a programok azokkal a felhasználói jogokkal futnak, amivel az õket elindító felhasználó rendelkezik. Ezt lehet megváltoztatni a suid és az sgid bitekkel. Egy suid bittel rendelkezõ program elindításakor a program a tulajdonosának jogaival fog futni, fájlokat olvasni, írni és más programokat futtatni.

Az sgid bit is egy rövidítés a „Set Group Identification" kifejezés rövidítése. Azt jelenti, hogy a „csoportazonosító megváltoztatása". Beállítása esetén a program annak a csoportnak a jogaival fog futni, akinek a fájl a birtokában van.

Hogy hogyan lehet ezeket a programokat megtalálni? Érdemes tanulmányozni a *checksecurity* (*/usr/sbin/checksecurity*) scriptet, nagyon tanulságos!

Az sgid bitet könyvtárak esetén is be lehet kapcsolni. Ennek az eredménye a következõ: ha ebben a könyvtárban bárki létrehoz egy fájlt (ehhez az többi jognak rendben kell lennie), akkor a fájl csoporttulajdonosa nem az a csoport lesz, amelyikbe a felhasználó tartozik, hanem az, akinek a könyvtár a birtokában van.

/usr/sbin/checksecurity

```
#!/bin/sh
# Security checks script - run daily out of the system crontab

set -e

PATH=/sbin:/bin:/usr/sbin:/usr/bin

LOG=/var/log
TMP=/var/log/setuid.new.tmp

umask 077
cd /

. /etc/checksecurity.conf

if [ "$CHECKSECURITY_DISABLE" = "TRUE" ] ; then
```

```
    exit
fi

#
# Check for NFS/AFS mounts that are not nosuid/nodev
#
#
# Check for NFS/AFS mounts that are not nosuid/nodev
#
if [ ! "$CHECKSECURITY_NONFSAFS" = "TRUE" ] ; then
   # temporarily disable error exit, as grep may give errors if no nfs/afs
   # are mounted.
   set +e
   nfssys=`mount | grep -E 'nfs|afs' | grep -vE
'\(.*(nosuid|noexec).*nodev.*\)'`
   nfssyscnt=`echo $nfssys |grep "[a-z]"| wc -l`
   set -e
   if [ $nfssyscnt -gt 0 ] ; then
      echo "The following NFS or AFS filesystems are mounted insecurely:"
      echo ""
      echo $nfssys
      echo ""
      echo "If this is intentional and you have supreme confidence in the"
      echo "security of the server for these file systems, you may disable"
      echo "this message by editing the value of CHECKSECURITY_NONFSAFS in"
      echo "the file /etc/checksecurity.conf."
   fi
fi

if [ "$CHECKSECURITY_NOFINDERRORS" = "TRUE" ] ; then
        exec 9>&2
        exec 2>/dev/null
fi


find `mount | grep -vE "$CHECKSECURITY_FILTER" | cut -d ' ' -f 3` \
     -xdev \( -type f -perm +06000 -o -type b -o -type c \) \
     -printf "%8i %5m %3n %-10u %-10g %9s %t %h/%f\n" \
| sort >$TMP

if [ "$CHECKSECURITY_NOFINDERRORS" = "TRUE" ] ; then
        exec 2>&9
fi

cd $LOG

test -f setuid.today || touch setuid.today

if cmp -s setuid.today $TMP >/dev/null
then
    :
else
        echo "`hostname` changes to setuid programs and devices:"
        diff setuid.today $TMP || [ $? = 1 ]
        mv setuid.today setuid.yesterday
        mv $TMP setuid.today
fi
rm -f $TMP
```