© Kiskapu Kft. Minden jog fenntartva

A Sudo-ról dióhéjban

...avagy hogyan indíthatjuk programjainkat álnéven?

Bizonyára minden rendszergazdában felvetődött már a probléma, hogy hogyan lehetne azt megoldani, hogy bizonyos kitüntetett felhasználók elindíthassanak bizonyos védett programokat, amelyekhez különleges jogosultságok szükségesek, s mindezt anélkül, hogy az adott felhasználónak meglenne ez a különleges joga. Ez a leggyakrabban akkor fordul elő, amikor bizonyos folyamatokhoz – például adatbázis mentéséhez – root jogosultságra van szükség, tehát csak a rendszergazda végezheti. A rendszergazdának azonban a legritkább esetben feladata az ilyen folyamatok indítása, tehát el kell látni a megfelelő felhasználót az adott jogosultsággal. Ez azonban veszélyes, mert ezzel a felhasználó akár szándékosan, akár figyelmetlenségből kárt okozhat. Ősidők óta létezik a problémára egy megoldás: a `setuid/setgid` használata, amelyet az egyes fájlokra határozhatunk meg Posix környezetben. Ha pl. a `setuid` bit be van állítva egy állományra, akkor azt bárki futtatja, a program azon felhasználó jogaival fog futni, akié az állomány. Tipikus példája ennek a `passwd` parancs, amely azáltal tudja írni a jelszófájlt, hogy a parancs tulajdonosa a root felhasználó, s be van billenve a setuid bit. Nem nehéz azonban rájönni, hogy ezt alkalmazva mondjuk az adatbázis-mentés példára azt kapjuk, hogy minden (legalábbis nagyon sok) felhasználó tudja futtatni az adott parancsot, nem csak az, akit a rendszergazda szeretne.
A probléma tehát az, hogy nem lehet elég finom felbontásban engedélyezni bizonyos parancsok futtatását ezzel a módszerrel.

A megoldás: Sudo

Ennek a kis programocskának a segítségével lehetőségünk nyílik arra, hogy az előre meghatározott szabályok szerint bizonyos felhasználók bizonyos programokat root vagy más felhasználó nevében futtathassanak. Az, hogy ki, mit és hogyan futtathat, a program beállítási fájljában kell rögzítenie a gép rendszergazdájának, és ezzel máris megoldódott a problémánk: akár felhasználónként és parancsonként egyesével is meghatározható, hogy ki és mit indíthat. A dolog a hétköznapi használatban úgy működik, hogy a parancs helyett a `sudo <parancs>` utasítást adjuk ki az értelmezőnek. A Sudo ekkor ellenőrzi a felhasználó és a parancs összerendelését, s a megfelelő feltételek mellett elindítja a programot úgy, mintha azt valóban az adott parancs futtatására jogosult személy tette volna, s a futás végén visszatér az elindított parancs visszatérési értékével.

A programról

A Sudo alapértelmezetten része szinte minden ma használatos terjesztésnek, így tehát csak annyi a dolgunk, hogy a rendszer csomagkezelőjével megkeressük, és hozzáadjuk a telepített összetevők sorához. Ha valamilyen oknál fogva ez nem sikerülne, még mindig letölthetjük a legfrisseb változatot az alkalmazás honlapjáról (➲ http://www.courtesan.com/sudo/) – igaz, csak forrás formájában. A binárisok csak nagy kereskedelmi Unix változatokhoz érhetők el.
A telepítésről tehát ennyit. A Sudo beállításai az általam használt Debian, Red Hat ill. SuSE terjesztésekben kivétel nélkül a */etc/sudoers* fájlban találhatók. Nekünk szinte mindig ezzel a fájllal kell majd dolgozni ahhoz, hogy testreszabjuk a programunkat.

A Sudo használata

A már fentebb említett `sudo <parancs>` stílusú használat esetén alapértelmezetten meg kell adnunk a saját felhasználói jelszavunkat. A helyes jelszó megadása esetén egy 5 perces érvényességű jegyet kapunk, ami azt jelenti, hogy a parancs következő 5 percben történő ismételt futtatása esetén nem kell a jelszót újra beírni. Ezzel a megoldással szerették volna a program írói megakadályozni azt, hogy egy otthagyott parancsértelmezőbe illetéktelenek bármit beirkálhassanak. A jegy érvényességének ideje természetesen átállítható, mint ahogyan a jelszó megadásának kötelező volta is, akár minden parancsra egyesével – mindezt a sudoers fájlból. A program nevéből is adódóan (SUperuser DO!) alapértelmezetten root-ként próbálja meg futtatni a kívánt parancsokat, de ettől a megfelelő kapcsolók használatával eltérhetünk. A lényeg az, hogy ha nem rendszergazdaként használjuk a Sudo-t, akkor minden kiadott parancs mögött kell, hogy legyen valami a */etc/sudoers* fájlban, különben a program nem fog lefutni.
A Sudo egyébként igen aktívan naplóz. Naplózza a próbálkozásokat és a sikertelen futtatási kísérleteket, de igény szerint rögzíti a sikeres próbálkozásokat is. Ha valaki olyan próbálkozik, akinek nincs joga a sudoers szerint az adott parancshoz, még levelet is küld – alapértelmezetten a root felhasználónak. A programnak egyébként rengeteg átállítható alapértelmezett beállítása van.
Hogy megtudjuk mennyi, adjuk ki a `sudo -L` parancsot.
S ha már a parancsoknál tartunk, nézzünk meg egy-két gyakrabban használt darabot.

`sudo -l`: Ezzel a paranccsal kilistázhatjuk, hogy nekünk, mint felhasználóknak milyen parancsok futtatásához van jogunk a sudo segítségével.
`sudo -v`: Felülírja a Sudo-tól legutóbb kapott jegyet, amely ugye a futtatás során jön létre, majd avul el, stb. Ha szükséges, bekéri a jelszót. Ezzel gyakorlatilag azt jelezhetjük a sudo számára, hogy még a gép előtt vagyunk, ezáltal a jegyünk újabb n percig érvényes – ugyanez történik egyébként akkor is, ha futtatjuk az adott parancsot.
`sudo -k`: Érvényteleníti az jegyet, amit a legutóbbi futtatás során kaptunk, tehát más még véletlenül sem indíthatja el az adott parancsot, ha esetleg kijelentkezünk, vagy otthagyjuk a terminált.
`sudo -b <parancs>`: Ennek hatására a kért parancs a háttérben fog futni. Ez akkor lehet hasznos, ha valami hosszan futó folyamatot indítunk, de közben szeretnénk visszakapni a promptot.
`sudo -u <felhasználó> <parancs>`: Ez esetben nem rootként próbálja meg futtatni a parancsot, hanem a megadott felhasználó nevén.
Ezeken túlmenően még számos kapcsolót használhatunk, héjprogramot jelölhetünk ki, másik sajátkönyvtárat adhatunk meg, és még sorolhatnám. A kedves olvasó a program telepítése után a sudo kézikönyvben (`man sudo`) olvashat róla.

A Sudo testreszabása

Mint már említettem, a Sudo testreszabása a */etc/**sudoers*** fájlon keresztül történhet. A programnak része egy olyan parancs is, amellyel ezt a fájlt lehet szerkeszteni, ez pedig a `visudo`. Gyakorlatilag a rendszer alapértelmezett szövegszerkesztőjét hívja meg, ám láthatatlanul azért csinál mást is a háttérben: Lezárja a ***sudoers*** fájlt a többi folyamatok elöl, így azok nem férhetnek hozzá, tehát nem fordulhat elő az, hogy egyszerre ketten egymásnak ellentmondásos értékekkel töltik fel az állományt. Ezen túl a szerkesztés után a parancs ellenőrzi a fájl tartalmát, és csak akkor menti el, ha nincs benne szintaktikai hiba. Ha van, akkor újra szerkeszthetjük a fájlt, vagy eldobhatjuk a változtatásokat, kivételes esetben megkérhetjük rá, hogy a szintaktiai hiba ellenére is mentse el a változtatásokat.
Maga a ***sudoers*** egyébként az EBNF (Extended Backus-Naur Form) nyelvet használja, ezen a nyelven kell tehát mondatokat a ***sudoers*** fájlba beírni, s ezt tudja értelmezni a program. Nekünk szerencsére nem kell megijednünk ettől a leírónyelvtől, ugyanis egyrészt nagyon egyszerű, másrészt a mindennapi használat során a példamondatok átalakítása bőven elég ahhoz, hogy elkészítsük a számunkra szükséges kifejezéseket. Ha szükségesnek érezzük, a nyelv gyorstalpalója megtalálható a ***sudoers*** kézikönyvében (`man sudoers`), sok más egyéb hasznos tudnivaló mellett.
A fájl kétféle adatot tartalmaz. Egyiküket alias-nak nevezi a leírás – ezek valójában globális változók. A globális változók egy részével felhasználói ill. parancscsoportokat hozhatunk létre, másik részével beállíthatjuk a már fentebb említett alapértelmezett értékeket: mi legyen az alapértelmezett héjprogram, mi legyen az alapértelmezett sajátkönyvtár, és még sorolhatnám...
A másik csoportja az adatoknak a felhasználókra vonatkozó előírások. Ezek mondják meg, hogy egy adott parancsot mely felhasználó indíthat el és milyen körülmények között.
A ***sudoers*** fájlban használt leírónyelv lehetővé teszi, hogy a kialakított felhasználói csoportokkal, munkaállomás-csoportokkal, parancs-csoportokkal nagy számú felhasználótábort is könnyedén kezelhessünk, mi több a munkaállomás-csoportok alkalmazásával az is megoldható, hogy ugyanazt a ***sudoers*** fájlt használjuk egy hálózat több gépén is. Ez a gyakorlatban nagyjából a következőképpen működik: megadjuk, hogy mely munkaállomáson futtatható csak az adott parancs, ezáltal ha másik gépre másoljuk, ott az azonos nevű felhasználó nem fogja tudni indítani az adott parancsot. Ezt általánosítsuk most gondolatban mind a parancsok terében, mind a fizikai elhelyezkedés terében, s máris megkapjuk egy központosított rendszer körvonalait, amelyben egy helyről szabályozható igen finom felbontásban az egész géppark felhasználótábora.
Végezetül nézzünk néhány egyszerű példát, amire nekünk, egyszerű felhasználóknak is szüksége lehet. Mindenekelőtt vizsgáljuk meg, hogyan is néz ki egy felhasználóra vonatkozó előírás:

```
<felhasználó> <mely gépeken futtatható> =
➥[(<milyen néven futtatható>)]
<parancs1>,<parancs2>,....
```

Mint látjuk, tényleg nem valami bonyolult dolog. Amit érdemes még tudni: létezik egy `ALL` kulcsszó, amely annyit tesz a mondatban is, hogy 'minden esetben'. Említettem még, hogy kikapcsolható a jelszó bekérése a Sudo használata esetén. Ez az első parancs elé biggyesztett `NOPASSWD`: kulcsszóval érhetjük el. Ha azt szeretnénk, hogy csak bizonyos parancsok esetében ne kelljen megadni jelszót, akkor a 'bizonyos parancsok' felsorolása után a többi parancs elé tegyük a `PASSWD`: kulcsszót. Ezzel körülbelül el is mondtam mindent, amire otthoni, vagy egyéb kisebb környezetben szükségünk lehet, lássuk a példákat:
`bela ALL = (ALL) ALL`: Ez a sor a sudoers-ben bela-nak gyakorlatilag root jogot ad anélkül, hogy Béla ismerné a root-jelszót. Minden parancsot futtathat mindenki nevében, minden gépen.
`bela ALL = (ALL) NOPASSWD: ALL`: Ez a sor akkor jó, ha otthon a munkaállomást felhasználóként használjuk, de gyakran szükséges root jog az egyes parancsokhoz. Ilyenkor kényelmetlen volna mindig su parancs után begépelni a jelszót, stb, ezért beírhatjuk a fenti sort a */etc/**sudoers***-be, s kényelmesen gépezhetünk egész nap.
`bela konyveles = /usr/bin/db_backup`: Béla a 'konyveles' nevű gépen futtathatja az adatbázis-mentést elvégző scriptet root-ként, de ezen kívül semmi mást, és meg kell adnia a parancs kiadása után a jelszavát. Ezek alapján már bárki könnyedén behelyettesítheti a kívánt felhasználó-parancs összerendeléseket, s hozzáolvasva a felhasználói leírásokból (man) már egész kis rendszer kialakítására képes.

Komáromi Zoltán
(komi@kiskapu.hu)
23 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.

© Kiskapu Kft. Minden jog fenntartva