A Linux fájlrendszer biztonsága (1. rész)

A jogosultságok beállításának hibája számos gyakori Linux probléma – nem egyszer idegesítő – forrása, ezért kell elsajátítani a jogosultságok alapjait, és megtenni az első lépéseket a Linux biztonságának megértése felé.

A ***Linux Journal Paranoid pingvin*** rovatában az elmúlt négy évben elsősorban eszköz-központú cikkek jelentek meg. Megtudtuk, hogyan kell biztonságossá tenni a ***Sendmailt***, hogyan lehet a ***Stunnel*** segítségével ***SSL*** titkosítást alkalmazni, valamint egy egész sereg hatékony biztonsági program beállításának és futtatásának módját. A következő néhány oldalt ezzel szemben a a fájlrendszer jogosultságainak szenteljük, amely egyfelől a Linux biztonságának egyik legalapvetőbb, legfontosabb tényezője, másrészt – és ez az igazán meglepő – viszonylag elhanyagolt téma. Ha bölcsen használjuk, a felhasználók és a behatolók nehezebben tudnak visszaélni a jogosultságokkal. Ha azonban nemtörődöm módon állítjuk be, a kisebb gyenge pontok nagymértékben veszélyeztethetik a rendszert. Ezek a cikkek különösen hasznosak lehetnek a Linux újoncok számára, akik nem értik, mit jelenthet az a sok `drwxr-xr-x` halandzsa fájllistában. De ezek a cikkek, különösen a 2. rész, még a középhaladók számára is – esetleg akik még nem értik a ***setuid*** és a ***setgid*** pontos következményei – tartogathat újdonságokat.

Bevezetés: Minden fájl

Nem sokan tudják, hogy a ***UNIX*** és a hozzá hasonló rendszerekben alapvetően mindent fájlok képviselnek. A dokumentumokat, a képeket és akár a futtatható programokat is könnyű a merevlemezen elhelyezkedő fájlokként felfogni. Ugyan a könyvtárakat fájltárolóként képzeljük, pedig valójában olyan fájl, amely – ahogy sejtettük – más fájlokat tárol. Hasonlóképpen, annak ellenére, hogy a rendszerhez kapcsolt CD-ROM elég kézzelfoghatónak tűnik, de a Linux rendszermag számár az is csak egy fájl: egy különleges eszközfájl, a */dev/cdrom*. A rendszermag tulajdonképpen ebbe a különleges fájlba ír és ebből olvas ahhoz, hogy adatot küldjön róla, vagy hogy a CD-ROM meghajtóra írja. Valójában a legtöbb rendszerben a */dev/cdrom* jelképes hivatkozás a */dev/hdb*-re, vagy valamelyik más különleges fájlra. Ki sem találnánk, hogy a jelképes hivatkozás viszont nem más, mint egy olyan fájl, amely egy másik fájl elérési útvonalát tartalmazza.
Más fájlok, mint például a nevesített csővezetékek, bemeneti/kimeneti csatornaként viselkednek, lehetővé téve, hogy a programok adatokat adjanak át egymásnak.

Nem az a lényeg, hogy leírjunk minden egyes fájltípust, ami a ***Linuxban*** vagy a ***UNIX***-ban létezik, hanem, hogy szemléltessük, hogy szinte mindent fájlok képviselnek. Amint ezt felfogjuk, sokkal könnyebb lesz megérteni, hogy miért olyan fontos a fájlrendszer biztonsága, és hogyan működik.

Felhasználók, csoportok, jogosultságok

A ***Linuxban*** tulajdonképpen két olyan dolog van, amit nem fájlok képviselnek: a felhasználói fiókok és a csoportfiókok. Többféle fájl is tartalmaz adatokat a rendszer felhasználóiról és csoportjairól, de valójában egyik sem képviseli azokat.
A felhasználói fiók olyan valakit vagy valamit jelent, aki/ami képes a fájlok használatára. Ez azt jelenti, hogy emberek is és rendszerfolyamatok is használhatnak felhasználói fiókokat. Ha egy felhasználói fiók neve például webmester, az általában egy olyan embert jelent, aki webhelyeket tart fent, de szabványos ***lp*** Linux felhasználói fiókot a soros nyomtató démon (***lpd***) használja – az ***lpd*** program ***lp*** felhasználónéven fut. Később elmagyarázzuk, hogy mit jelent a programok számára, ha egymás elleni felhasználóként futnak.
A csoportfiók egyszerűen felhasználói fiók egy listája. Minden egyes felhasználói fiók egy fő csoporttagsággal van meghatározva, de valójában annyi csoporthoz tartozhat, amennyihez szükséges. A ***maestro*** (mester) felhasználónak például lehet a ***conductors*** (karmesterek) csoportban a fő tagsága, és tartozhat a ***pianists***-hoz (zongoristák) is.
A felhasználók fő csoporttagsága az */etc/password* fájlban található fiókbejegyzésben van meghatározva. A felhasználót további csoportokba is sorolhatjuk, az */etc/group* szerkesztésével, a felhasználónevet minden olyan csoportbejegyzés végéhez írva, amelyben a felhasználónak szerepelnie kell. Másik lehetőségként használhatjuk a `usermod` parancsot – további információkért lásd a `usermod(8)` súgóoldalt.
Az alábbi lista első sora a */etc/password* fájl `maestro`-ra vonatkozó bejegyzését mutatja, a következő kettő pedig a */etc/group* fájl megfelelő tartalmát.

```
maestro:x:200:100:Maestro Edward
Hizzersands:/home/maestro:/bin/bash

conductors:x:100:
pianists:x:102:maestro,volodyia
```

A *passwd* fájlban az első mező a felhasználói fiók nevét, a *maestro* nevet tartalmazza. A második (x) *maestro* jelszavának mutatója, ami arra utal, hogy a jelszó valójában az */etc/shadow* fájlban tárolódik. A harmadik mező a `maestro` számazonosítója (*uid*-ja), amely ebben az esetben 200. A negyedikben a felhasználó elsődleges csoportjának számazonosítója (*gid*) látható, amely ezúttal 100. A többi mező egy megjegyzést, `maestro` saját könyvtárát és alapértelmezett bejelentkezési héját tartalmazza.
A */etc/group* fájlban az egyes sorok csupán a csoportnevet, a csoportjelszót (általában nem használatos – az `x` a mutató), a csoport számazonosítóját (*gid*) és vesszővel elválasztva, a csoportba másodrendű tagként felvett felhasználók listáját tartalmazzák. Így láthatjuk, hogy a `conductors` csoport *gid*-je 100, ami megfelel `maestro` főcsoportként meghatározott *gid*-jének. Azt is látjuk, hogy a `pianist` csoportnak tagja a maestro felhasználó, és másodrendű tagként egy `volodyia` nevű felhasználó.
Az */etc/password* és az */etc/group* fájl felhasználói fiókok létrehozása, módosítása és törlése céljából való megváltoztatásának legegyszerűbb módja a `useradd`, `usermod` illetve `userdel` parancsok használata. A parancs írásmód helyett most inkább az elvre összpontosítunk, tehát elég annyit mondani, hogy mind a három parancs a csoporttagságok beállítására és módosítására szolgál, és a megfelelő súgóoldalakon jól dokumentáltak. Ahhoz, hogy egy rövid összefoglalót lássunk a használatról, gépeljük be a parancs után a `--help` paramétert, például `useradd --help`.

Egyszerű fájl-jogosultságok

Minden fájlnak két tulajdonosa van: egy felhasználó és egy csoport, mindegyik saját jogosultság-beállításokkal rendelkezik, amelyek meghatározzák, hogy a felhasználó mit csinálhat az adott fájllal – olvashatja, írhat bele és végrehajthatja. Egy harmadik beállításkészlet arra vonatkozik, hogy mások – olyan felhasználói fiókok, amelyek nem a fájl tulajdonosai, illetve nem tartoznak olyan csoporthoz, amely a fájl tulajdonosa – mit csinálhatnak a fájllal. Az *1. listán* a */home/maestro/baton_dealers.txt* hosszú fájllistája látható.
A jogosultságok a felhasználói engedélyek, a csoportengedélyek és a többi engedély sorrendje alapján vannak felsorolva. Az 1. listában bemutatott fájl esetében a tulajdonosa (maestro) olvashatja és írhatja a fájlt (`rw-`), a tulajdonos csoport (`conductors`) szintén olvashatja és írhatja a fájlt (`rw-`), de a többi felhasználó csak olvashatja. A jogosultságok azért kicsit bonyolultabbak. Az egyéb kategóriába sorolt felhasználók egy adott fájlra vonatkozó jogosultságok alapján bármilyen fájlt törölhetnek egy olyan könyvtárban, amelyre van írási jogosultságuk. Más szóval, ha egy felhasználónak csak olvasási jogosultsága van egy fájlra, akkor nem szerkesztheti, de ha a fájl könyvtárára írási jogosultsága van, akkor törölheti a fájlt.
Az olvasáson és az íráson kívül van egy harmadik jogosultság: a végrehajtás, amelyet `x` jelöl, amikor be van állítva. Ha a `maestro` ír egy héjprogramot, *punish_bassonists.sh* néven, és a jogosultságait a `-rwxrw-r--` értékre állítja, akkor úgy hajthatja végre ezt a héjprogramot, hogy beírja a nevét a parancssorba. Ha viszont elfelejti beállítani a végrehajtási engedélyt, nem tudja majd futtatni a programot, annak ellenére, hogy ő a tulajdonosa.

Parancsok és súgóoldalak

Ebben a cikkben a pontos írásmód és az adott parancsok használata helyett a fájlrendszer alapelveire összpontosítunk. De a kezdők biztosan szeretnék tudni, hogy egyáltalán hogyan kell futtatni ezeket a parancsokat, és hol találhatunk segítséget az írásmódhoz és az alkalmazáshoz.
Először is, minden példában és példahelyzetben a terminálablakban dolgozunk. A Microsoft *Windows* felhasználók úgy képzelhetik el a terminálablakot, mint egy *DOS* készenléti jelet, vagy parancsablakot. A terminálablak biztosítja a Linuxszal való párbeszéd legközvetlenebb formáját, vagyis minden parancsot kézzel írhatunk be, ahelyett, hogy egérkattintásokkal indítanánk el.
Ahhoz, hogy saját héjmunkamenetet indítsunk a *GNOME*-ból, kattintsunk a *Main Menu* (*Főmenü*) gombra, és válasszuk a *System Tools®Ternóminal* lehetőséget. A *KDE*-ben a terminálparancsot *konsole*-nak hívják, és saját ikonja van a tálcán: egy kagylóhéj egy számítógép-monitor előtt. Másik lehetőségként elindíthatjuk a *Run Program* (*Program futtatása*) párbeszédablakot, és a készenléti jelnél begépelhetjük, hogy *konsole*. Ahhoz, hogy gyakorlatilag bármelyik *Linux* parancshoz gyors segítséget kapjunk egy terminál/héj munkameneten belül, begépelhetjük az adott parancsot, utána pedig a `--help` lehetőséget. Ha például nem emlékszünk az `ls` parancs összes parancssor lehetőségére, amely a fájlokat és a könyvtárakat listázza, csak beírjuk az `ls --help` parancsot.
A `--help` lehetőség gyors, de nem mindegyik parancsnál működik. Amikor működik is, az eredmény néha túlságosan tömör. A parancsokra vonatkozó segítségkérés legjobb módja a `man` parancs használata. A súgóoldalak (man pages) teljes útbaigazítást biztosítanak a legtöbb Linux parancs használatával kapcsolatban, és gyakorlatilag az összes UNIX-hoz hasonló rendszerben jelen vannak. Ahhoz, hogy megtekintsük például az ls parancs súgóoldalát, gépeljük be a `man ls` parancsot. A súgóoldal felsorolásban a szóköz billentyű segítségével léphetünk tovább egy oldalt, a B gombbal pedig egy oldalt visszamehetünk, ha pedig begépeljük, hogy `/valami`, akkor megkeresi súgóoldalon a „`valami`" karaktersort.
Mi van azonban olyankor, ha nem tudjuk annak a parancsnak a nevét, amelyre szükségünk van? Erre való az `apropos` parancs. Gépeljük be például, hogy `apropos list`, hogy olyan különféle parancsokat láthassunk, amelyek valamit listáznak, majd annak a parancsnak a súgóoldalára ugorjunk, amelyre valószínűleg szükségünk van.

A jogosultságokat általában a `chmod` paranccsal lehet beállítani, ami a *„change mode"* (üzemmód-változtatás) rövidítése. A példát folytatva, tegyük fel, hogy a `maestro` meggondolja magát, és nem akarja, hogy a `conductors` (karmesterek) csoportja láthassa a pálcakereskedők (`baton dealers`) listáját. A *2. listán* látható parancsok segítségével eltávolíthatja a csoport olvasási/írási jogát.
A 2. lista példa `chmod` parancsában (`chmod go-rw`) a `go` arra utasítja a `chmod`-ot, hogy változtassa meg a csoport jogosultságait, és az egyéb jogosultságokat, az `-rw` pedig megszabja,

1. lista Jogosultságokat bemutató fájllisták

```
-rw-rw-r--  1  maestro  conductors  35414
➥Mar 25 01:38 baton_dealers.txt
```

2. lista Egy fájl jogosultságainak megváltoztatása a chmod paranccsal

```
bash-$ ls -l baton_dealers.txt

-rw-rw-r--  1  maestro  conductors  35414
➥Mar 25 01:38 baton_dealers.txt

bash-$ chmod go-rw baton_dealers.txt
bash-$ ls -l baton_dealers.txt

-rw-------  1  maestro  conductors  35414
➥Mar 25 01:38 baton_dealers.txt
```

3. lista Egy csoport által olvasható könyvtár

```
bash-$ chmod g+rx extreme_casseroles
bash-$ ls -l extreme_casseroles

drwxr-x---  8  biff  drummers  288  Mar 25
➥01:38 extreme_casseroles
```

hogy el kell távolítani a szóban forgó két kategória (a csoport és az egyéb) jogosultságait. Tehát a `chmod` parancs három részből áll: egy jogosultságkategóriából, amely az `u`, a `g` és az `o` valamilyen kombinációja. Itt állhat egyszerűen az a betű is (all = valamennyi), illetve szerepelhetnek benne műveleti jelek: a `-` az eltávolításnak, a `+` a hozzáadásnak felel meg. Ezeket mindig a hozzáadandó vagy eltávolítandó jogosultságok listája (általában `r`, `w` vagy `x`) követi.

Könyvtárjogosultságok

Már tudjuk, hogyan kell megváltoztatni a hagyományos fájlok alapvető jogosultságait, de mi a helyzet a könyvtárakkal? A könyvtárjogosultságok kicsit másképp működnek, mint a hagyományos fájlok jogosultságai. Az olvasás és az írás hasonló – a könyvtáraknál ezek a jogok a könyvtártartalom listázásának illetve a fájlok könyvtáron belüli létrehozásának és törlésének felelnek meg.

A könyvtáraknál viszont a végrehajtás kicsit kevésbé logikus. Itt a végrehajtás annak felel meg, ha bármit megváltoztatunk a könyvtárban, vagy a munkakönyvtárat az adott könyvtárra változtatjuk. Azaz, ha a felhasználónak vagy egy csoportnak végrehajtási jogosultsága van egy bizonyos könyvtárra, akkor a felhasználó/csoport listázhatja az adott könyvtár tartalmát, olvashatja a könyvtár fájljait (feltéve, hogy az egyes fájlok saját jogosultságai beletartoznak), és a munkakönyvtárat az adott könyvtárra változtathatjuk a `cd` paranccsal. Ha egy felhasználónak vagy egy csoportnak nincs végrehajtási jogosultsága az adott könyvtárra, akkor semmit nem listázhat vagy olvashat a könyvtáron belül, az ott lévő elemeken beállított jogosultságoktól függetlenül. Ha egy könyvtárra nincs végrehajtási jogunk, de olvasási jogunk van, és megpróbálunk az `ls` paranccsal listázni, olyan hibaüzenetet kapunk, amely tulajdonképpen felsorolja a könyvtár tartalmát. De ez nem működik olyankor, amikor nem rendelkezünk sem olvasási, sem végrehajtási jogosultsággal a könyvtárra vonatkozóan.

Tegyük fel, hogy a példarendszernek van egy `biff` nevű felhasználója, aki a `drummers` (dobosok) csoportba tartozik. Feltételezzük továbbá, hogy az ő könyvtára tartalmaz egy `extreme_casseroles` nevű alkönyvtárat, amit szeretne megosztani az ütőstársaival. Az *3. listán* látható, hogy `biff` hogyan állítaná be a könyvtár jogosultságait.

Az 3. listánál csak ***biffnek*** van lehetősége fájlokat létrehozni, megváltoztatni és törölni az `extreme_cassaroles` könyvtárban. A `drummers` csoport többi tagja listázhatja a tartalmát, és a `cd` paranccsal beléphet. A rendszerben viszont mindenki más számára tiltott lesz a listázás, azs olvasás, a `cd` parancs használata vagy bármilyen más, a könyvtárral kapcsolatos tevékenység.

A jogosultságok és a rendszergazda

Gyakorlatias kifejezéssel élve, a jogosultságok egyszerűen nem vonatkoznak a ***root*** felhasználóra, a rendszergazda bármelyik fájllal bármit, bármikor csinálhat. Ezért olyan fontos, hogy soha ne rendszergazdaként jelentkezzünk be, és ne használjuk a `su` parancsot, hogy rendszergazdává váljunk, kivéve, ha feltétlenül szükséges. Rendszergazdaként a fájl-jogosultságok nem védenek meg a saját hibáinktól. Ez nem azt jelenti, hogy ha rendszergazdaként lépünk be, az összes program teljesen figyelmen kívül hagyja a fájl-jogosultságokat. Ha a rendszergazda például a `vim` szerkesztő segítségével megpróbál megváltoztatni egy írásvédett fájlt, akkor a `:w!` parancsot (írás kényszerítése) kell használnia. A hagyományos `ZZ` vagy `:w` parancs ilyenkor hibaüzenetet ad vissza. Nagyon sok más parancsnak viszont nincs olyan szolgáltatása, amely ellenőrizné az ésszerűséget.

Összefoglalás

Ezek a Linux fájlrendszer biztonságának legalapvetőbb elvei és gyakorlati felhasználási módjai. A 2. részben tovább megyünk, és megbeszéljük (többek között) a ***setuid***-et, a ***setgid***-et, és a numerikus jogosultsági üzemmódokat. Addig is, csak óvatosan!

Linux Journal 2004. október, 126. szám

Mick Bauer – Hitelesített informatikai rendszerek biztonsági szakértője, a Linux Journal biztonsági szerkesztője, és biztonsági szakember a Minneapolisban, Minnesotában. A Building Secure Servers With Linux (O'Reilly, 2002) című könyv szerzője.