Konfiguracja i stosowanie list kontroli dostępu do plików

ACL – Listy kontroli dostępu

Tradycyjny system plików Linuksa zapewnia jedynie podstawowy poziom zabezpieczenia danych. Prawa dostępu przyznawane są na poziomie pojedynczych plików, a nie indywidualnych potrzeb użytkownika. Jeżeli potrzebny jest mechanizm selektywnego dostępu dla różnych użytkowników, najlepszym rozwiązaniem jest użycie list kontroli dostępu.

VOLKER SCHMITT

Komputery typu mainframe zapewniają tego typu narzędzia już od lat, pozwalając na tworzenie grupy użytkowników z różnymi prawami dostępu do plików. Jednym z przykładów może być implementacja RACF dla systemu operacyjnego z/OS dla maszyn mainframe firmy IBM.

Od pewnego czasu dostępne są poprawki dla jądra w wersji 2.4, umożliwiające osiągnięcie podobnej funkcjonalności w systemie Linux. Zanim jednak zajmiemy się przydzielaniem uprawnień do plików wielu użytkownikom, przyjrzyjmy się tradycyjnym rozwiązaniom.

Uprawnienia tradycyjne

Poza kontrolą dostępu do komputera lokalnego poprzez hasło, system operacyjny stosuje dodatkowe mechanizmy określające dostęp do poszczególnych obiektów (pliki, katalogi, urządzenia). Aby taki mechanizm mógł działać prawidłowo, każdy obiekt i każdy proces posiadają dwa atrybuty: właściciela i grupę.

Ponadto, każdy obiekt może ściśle określać uprawnienia dla właściciela, grupy i innych użytkowników systemu. Po wykonaniu polecenia pokazującego zawartość katalogu (*ls l*), po lewej stronie w kolumnie zostaną wypisane uprawnienia (Ramka 1) do poszczególnych katalogów i plików.

Podczas każdej próby uzyskania dostępu do pliku, system Linux sprawdza, czy program lub użytkownik posiada odpowiednie uprawnienia. Warto wiedzieć, że podczas przetwarzania brany jest pod uwagę użytkownik i identyfikator grupy, ale dodatkowo także sprawdzany jest efektywny identyfikator użytkownika (*effective user ID*) i grupy (który wynika np. z przynależności do grup – *effective group ID*).

Efektywne ID użytkownika pochodzi zwykle z identyfikatora użytkownika użytego podczas uruchamiania programu, podobnie jest z efektywnym identyfikatorem grupy. Jeżeli dany program uruchomi samodzielnie podprogram, ten odziedziczy identyfikatory użytkownika i grupy z wywołanego procesu. W tym przypadku efektywne ID użytkownika jest takie samo, jak prawdziwego użytkownika uruchamiającego właściwy program.

Efektywne ID aktualnego użytkownika zostanie zmienione, gdy dla programu ustawiono bit S (tzw. sticky bit). W takim przypadku

Ramka 1. Uprawnienia do obiektów w systemie Linux.

Znacznik	Opis	Znaczenie
r	odczyt	Odczyt lub kopiowanie pliku
w	zapis	Zapis lub modyfikacja pliku
x	wykonanie	Uruchamianie programu zwartego w pliku

Tabela 1: Minimalne uprawnienia do operacji na plikach.

Operacja	Katalog	Plik
Odczyt (cat, more, less, pg)	--x	r--
Zapis (cat, >, ed)	--x	-w-
Zmiana nazwy (mv)	-wx	-
Uruchomienie programu	--x	--x

kolejny proces nie dziedziczy ID i grupy użytkownika lub programu nadrzędnego. Zamiast tego, proces zostanie uruchomiony z przywilejami użytkownika, oznaczonego jako właściciel pliku binarnego w drzewie systemu plików. Zatem poza ID użytkownika uruchamiającego program, jest także identyfikator użytkownika-właściciela pliku. Można to zilustrować prostym przykładem:

```
root@maus:# cp /bin/sleep ➥
/bin/rootsleep
root@maus:# chmod u+s ➥
/bin/rootsleep
root@maus:# su volker -c ➥
„rootsleep 25" &
root@maus:# /bin/ps -aeo pid,➥
euser,ruser,command|grep fdisk
12887 root volker rootsleep 25
```

Narzędzie *passwd*

Narzędzie *passwd* jest dobrym przykładem wykorzystania bitu S. System Linux przechowuje zaszyfrowane hasła użytkowników w pliku o nazwie */etc/passwd*. Ze względów bezpieczeństwa plik ten jest chroniony, przyznając prawo do zapisu wyłącznie dla użytkownika *root*.

Zwykli użytkownicy muszą jednak czasem sami zmieniać swoje hasła do systemu. Aby to umożliwić, program *passwd* ustawia bit *S* dla danego użytkownika – pewne dystrybucje Linuksa robią to także dla grupy. W tym przypadku, na liście katalogów utworzonych przez *ls l* zamieniamy *x* na *s*:

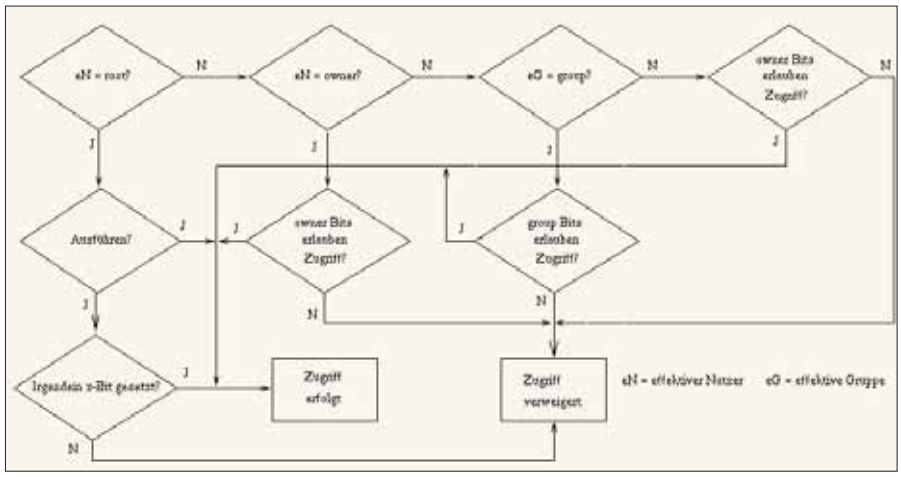
```
v@maus:~> ls -l /usr/bin/passwd
-rwsr-xr-x 1 root shadow 27604 ➥
Sep 20 2001 /usr/bin/passwd

v@maus:~> ls -l /etc/passwd
-rw-r--r-- 1 root root 2070 ➥
Feb 10 2002 /etc/passwd
```

Każdy użytkownik może uruchomić narzędzie *passwd*, a dodatkowo dla właściciela ustawiany jest bit *S*. Oznacza to, że proces odziedziczy uprawnienia należące do użytkownika efektywnego, czyli *root-a*. Ponieważ użytkownik efektywny ma prawo zapisu do pliku */etc/passwd*, użytkownicy mogą modyfikować plik, a w rezultacie zmieniać swoje hasła wedle własnych potrzeb. Oczywiście program passwd umożliwia zmianę wyłącznie swojego hasła.

Tabela 2: Minimalne uprawnienia do operacji na katalogach.

Operacja	Katalog ./..	Katalog ./.
Wyświetlenie zawartości katalogu (ls)	r-x	r-x
Usunięcie katalogu (rm -r)	rwx	rwx
Utworzenie pliku (cp, mv, ln)	--x	-wx



Rysunek 1. Procedura sprawdzania uprawnień dostępu do pliku wykonywana przez system.

Uprawnienia minimalne

Pojedyncze operacje na plikach wymagają określenia pewnych minimalnych uprawnień do odczytu, zapisu i dostępu do plików. W Tabeli 1 umieszczono kilka przykładów, wraz z wyjaśnieniami dotyczącymi praw dostępu. Tabela 2 pokazuje różne interpretacje wpisów *w* i *x* dla katalogów i plików. Uprawnienia zapisu można zinterpretować jako zgodę na modyfikację lub dodanie do listy plików. Jeżeli ustawiono parametr *x*, dany katalog stanie się katalogiem domyślnym lub częścią ścieżki dostępu. Rysunek 1 pokazuje przepływ informacji o prawach dostępu, w sytuacji gdy proces chce uzyskać dostęp do pliku.

Łatwy i wygodny

Wycieczka w świat tradycyjnych praw dostępu pokazuje podstawę mechanizmu sterowania dostępem do różnych plików w oparciu o dziewięć bitów plus dwa bity na każdy plik. Zaletą tego rozwiązania jest absolutna prostota.

Niestety, jeżeli administrator systemu będzie chciał ustawić wszystkie możliwe uprawnienia dostępu do pliku n użytkownikom, będzie potrzebował 2n różnych grup. Jest to oczywiście skrajnie niepraktyczne, zatem nowsze systemy operacyjne wykorzystują pojęcie grupy rozszerzonej - listy kontroli dostępu, w skrócie ACL (z ang. Access Control List). ACL umożliwia bardzo szczegółową kontrolę uprawnień w porównaniu z tradycyjnym modelem uprawnień.

Przy użyciu list kontroli dostępu system operacyjny dołącza do pliku dodatkową listę uprawnień dla określonych użytkowników i grup. Umożliwia to przypisanie uprawnienia odczytu lub zapisu dla pojedynczego pliku dwóm lub trzem użytkownikom, a nie tylko całej grupie. System przypisuje użytkowników indywidualnie, zgodnie z nadanymi przywilejami.

Najlepszym sposobem zrozumienia działania listy kontroli dostępu jest dokładniejsze przyjrzenie się poleceniom *getfacl* i *setfacl*, które wykorzystywane są do odczytu i zapisu tych uprawnień.

Następny proszę

Polecenie *getfacl* wyświetla listę kontroli dostępu dla pliku. Zwraca ono kilkanaście in-

Ramka 2. Przykład polecenia *getfacl*.

```
v@maus:> getfacl filename
```

zwraca następujące wyniki

```
01 # file: filename
02 # owner: userid-of-owner
03 # group: groupid
04 user::permissions
05 user:other-userid:permissions
06 group::permissions
07 group:other-groupid:permissions
08 mask:permissions
09 world:permissions
10 default:user::permissions
11 default:user:other-userid:
   permissions
12 default:group::permissions
13 default:group:other-groupid:
   permissions
14 default:mask:permissions
15 default:world:permissions
```

formacji dla każdego z plików, rozdzielając informacje przy pomocy pustych wierszy (patrz Ramka 2). Pierwszy wiersz każdego bloku zawiera nazwę pliku, drugi pokazuje (tradycyjnego) właściciela pliku, a trzeci (tradycyjną) grupę, do której jest on przypisany. Kolejne informacje, to szczegółowa lista kontroli dostępu dla określonego pliku.

Program najpierw wyświetla uprawnienia użytkownika, a następnie wszystkich innych użytkowników z listy ACL. Zamiast ponownego wyświetlania w pierwszym wierszu nazwy aktywnego użytkownika, wstawiane są dwa dwukropki. W opisywanym przez nas przykładzie użytkownik dodany do listy kontroli dostępu został zdefiniowany pomiędzy dwoma dwukropkami.

Na końcu każdego wiersza program wyświetla specjalne przywileje, przypisane każdemu z użytkowników według maski *rwx*. Wiersze zawierające wpisy użytkowników występują po opisach grup prezentowanych w ten sam sposób.

Kolejne dwa wiersze to maska z opisem uprawnień i tradycyjnymi uprawnieniami dla dowolnego użytkownika.

Ostatnie kilka wierszy w naszym przykładzie zawiera domyślne wpisy dla innych użytkowników i grup oraz maskę ACL, wskazującą uprawnienia domyślne.

Dla przypisania nowych list kontroli dostępu, tworzonych przez użytkowników w obecnym katalogu, system operacyjny wymaga wartości domyślnych. Następnie, w celu sprawdzenia uprawnień użytkowników i grup, system przekazuje wartości domyślne do katalogu głównego. Tak więc nowy plik odziedziczy wpisy domyślne, jak pokazano w Ramce 3.

Polecenie *getfacl* posiada następujące opcje: *-d*, dzięki której wyświetlone zostaną wyłącznie wpisy domyślne, oraz *a*, dzięki której poznamy listę kontroli dostępu bez wartości domyślnych.

Wydajna ochrona

Zamiast określania uprawnień dla kolejnych użytkowników i grup na liście kontroli dostępu, można użyć maski, która ograniczy uprawnienia dla całego wpisu, poza wpisem aktualnego właściciela.

Tabela 3: Opcje setfacl.

Atrybut	Opis	Znaczenie
--set	set	
-m	modyfikuj	
-x	usuń	usuwa wpis w ACL
-d	skasuj	usuwa całą listę ACL

Jeżeli użyjesz maski do usunięcia uprawnień dostępu, pozostawiając jedynie uprawnienia do odczytu (*Read*) i uruchamiania (*Execute*), operacja ta będzie dotyczyła wyłącznie użytkowników dodatkowych, bez względu na wpisy na liście ACL. Aby umożliwić z kolei dostęp, wpis musi być wykonany dla określonego użytkownika lub grupy i w tym samym czasie zatwierdzony przez maskę. Linux sprawdza pary wpisów na liście ACL, potwierdzając nadanie odpowiednich uprawnień.

Mogłoby to doprowadzić do nieumyślnego przyznawania uprawnień dla wpisów na liście ACL, łagodniejszych niż te, które w rzeczywistości mają mieć zastosowanie. Na szczęście można to szybko sprawdzić, ponieważ polecenie *getfacl* wylicza automatycznie uprawnienia efektywne i wyświetla je w komentarzu *#effective:* w wierszu pokazującym uprawnienia konkretnego użytkownika (patrz Ramka 4).

Jednakże, musimy być świadomi faktu, że aktywne uprawnienia na liście ACL nie mają nic wspólnego z bieżącym użytkownikiem lub grupą dla danego procesu. Ale jak modyfikować lub rozszerzać listy ACL?

Nowi użytkownicy

W przypadku nowych użytkowników (a zatem modyfikacji ACL) musimy skorzystać z drugiego bliźniaczego polecenia - setfacl. Poszczególne opcje tego polecenia, modyfikujące listę ACL, opisano w Tabeli 3. Najprostszym sposobem użycia setfacl jest wykorzystanie parametru set przy definiowaniu wpisu:

```
v@maus:~> setfacl --set ➥
acl-entry file
```

W Ramce 5 umieszczono wykaz wartości dopuszczalnych dla wpisu *acl-.entry*.

Parametr *m* ma w zasadzie takie samo działanie jak parametr *--set*, pod warunkiem, że nie ma obecnie żadnego wpisu dla tego użytkownika lub grupy. W innym przypadku uprawnienia zostaną nadpisane nowymi wartościami.

Parametr *x* usuwa wpis z listy ACL bez zmiany właściciela oraz grupy, nie zmieniając przy tym innych wpisów. Przy usuwaniu wpisu cytowanie uprawnień nie jest konieczne.

Jeżeli ustawimy zbyt restrykcyjną maskę na liście ACL, pozbawimy dostępu innych użytkowników, a przecież wyraźnym celem tego rozwiązania miało być zapewnienie dostępu użytkownikom. Generalnie powinniśmy unikać wpisów masek ograniczających poszczególne uprawnienia użytkowników na liście ACL.

Ramka 3. Dziedziczone uprawnienia użytkownika do plików.

```
01 volker@maus:~> getfacl directory
02 # file: directory
03 # owner: volker
04 # group: users
05 user::rwx
06 group::r-x
07 other:r-x
08 default:user::rwx
09 default:user:robin:rw-
10 default:group::r-x
11 default:mask:rwx
12 default:other:r-w
13
14 volker@maus:~> touch directory/file
15 volker@maus:~> getfacl
   directory/file
16 # file: file
17 # owner: volker
18 # group: users
19 user::rwx
20 user:robin:rw-
21 group::r-x
22 mask:rwx
23 other:r-x
```

Ramka 4. Komunikaty polecenia *getfacl*.

```
# file: file
# owner: volker
# group: users
user::rwx
user:robin:rwx   #effective:rw-
group::r-x
mask:rw-
other:r-x
```

Ramka 5. Lista możliwych opcji ACL dla *setfacl*.

```
u[ser]::permissions
u[ser]:other-userid:permissions
g[roup]::permissions
g[roup]:other-groupid:permissions
m[ask]:permissions
o[ther]:permissions
d[efault]:u[ser]::permissions
d[efault]:other-userid:permissions
d[efault]:g[roup]::permissions
d[efault]:g[roup]:other-groupid:
permissions
```

Elementy w nawiasach nie są konieczne.

W opisywanym przypadku należy określić parametr *mask*, aby przeliczyć ponownie wpis użytkownika inny wpis nie będzie brany pod uwagę.

Jako, że w tak dużej ilości wpisów dla poszczególnych plików na liście ACL łatwo o pomyłkę, powinniśmy przekazać plik z ustawionymi parametrami do polecenia *setfacl*. Aby to zrobić, musimy określić parametr *--set-file* i wskazać plik z parametrami *setfacl*.

Plik z parametrami musi być w tym samym formacie, co dane wyjściowe polecania getfacl. Kolejność poszczególnych wpisów na liście ACL nie jest istotna, *setfacl* poradzi sobie z tym bez problemów (ułatwia to ręczną edycję wpisów).

Wpisanie -, zamiast wskazania pliku, oznacza dla *setfacl* akceptację wpisów ze standardowego miejsca. Przydaje się to wtedy, gdy zachodzi konieczność przeniesienia wpisów ACL z jednego pliku do drugiego (w naszym przykładzie pliki *xyz* i *abc*). Poniższe polecenie wykonuje opisane zadanie:

```
root@maus:/# getfacl -R ➥
--skip-base / > /backup.acl
```

Rysunek 2 pokazuje sposób, w jaki proces uzyskuje dostęp do obiektu dzięki ACL.

Problemy z kopiami zapasowymi

Jak już wspomniano wcześniej, listy ACL można dołączyć do plików i katalogów. Dotyczy to również urządzeń i potoków nazwanych (*named pipes*), choć niewielu użytkowników korzysta z tych możliwości. Przyjrzyjmy się tymczasem ważniejszej sprawie: kopiom zapasowym. Jeżeli zamierzasz wykonywać kopie zapasowe, musisz pamiętać, że standardowe narzędzia (dotyczy to również programu TAR) nie potrafią zapisywać informacji o listach kontroli dostępu.

Dzięki nowemu standardowi POSIX systemy operacyjne UNIX zyskują nowe możliwości - między innymi obsługę formatu PAX, wprowadzoną w 2001 roku właśnie w celu rozwiązania problemu archiwizacji ACL. Programem, który obsługuje format PAX jest np. *star tape archiver* [1].

Jest jednak sztuczka, która nie wymaga używania formatu PAX, a jednocześnie umożliwia utworzenie kopii zapasowej list ACL. Użytkownik root powinien w tym celu wykonać następujące polecenie:

```
root@maus:/# getfacl -R ➥
--skip-base / > /backup.acl
```

Spowoduje to odszukanie przez *getfacl* list kontroli dostępu, zaczynając od katalogu głównego, i zapisanie znalezionych informacji w pliku o nazwie */backup.acl*. Parametr *R* powoduje przeszukanie całego drzewa systemu plików, natomiast *--skip-base* powoduje ignorowanie domyślnych list kontroli dostępu (tzn. dziewięciu domyślnych bitów uprawnień systemu Unix). Dzięki temu TAR może wykonać kopie zapasowe list ACL. Aby odzyskać tak zapisane listy kontroli dostępu, należy wykonać następujące polecenie:

```
root@maus:linux# zcat ../linux-➥
a.b.cacl-x.y.z.diff.gz | patch -p1
```

Instalacja obsługi ACL

Żeby móc korzystać z list kontroli dostępu, niezbędne są cztery pakiety: *e2fsprogs* (zwykle znajduje się w większości dystrybucji Linuksa), *libattr*, *libacl* oraz *acl* i poprawka jądra dostępna ze strony [2].

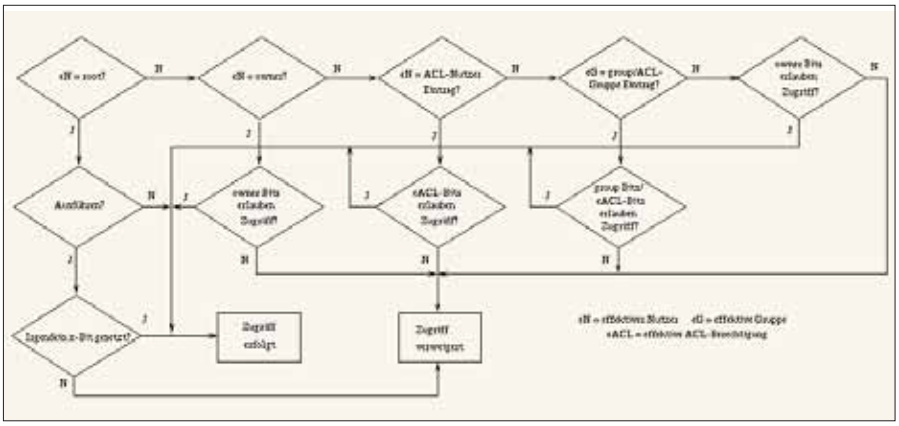
Najpierw należy zainstalować poprawkę jądra Linuksa. Rozpakuj plik źródłowy do katalogu /usr/src/linux. Zapisz poprawkę w katalogu */usr/src/* i przejdź do katalogu */usr/src/linux* oraz wykonaj polecenie:

```
root@maus:linux# zcat ../linux-➥
a.b.cacl-x.y.z.diff.gz | patch -p1
```

Litery *a.b.c.* oznaczają numer ściągniętej przez ciebie poprawki jądra. Teraz kolej na konfigurację jądra i włączenie opcji list ACL w pliku *.config* (w bieżącym katalogu):

```
CONFIG_FS_POSIX_ACL=y
CONFIG_EXT3_FS_XATTR=y
CONFIG_EXT3_FS_POSIX_ACL=y
CONFIG_EXT2_FS_XATTR=y
CONFIG_EXT2_FS_POSIX_ACL=y
```

Następnie, zgodnie ze standardową procedurą kompilujemy i instalujmy nowe jądro. Po wykonaniu tych czynności można zainstalować pakiety RPM oraz inne biblioteki i narzędzia do obsługi ALC. Szczegółowy opis znajdziesz na stronie [3]. Ostatnim krokiem jest modyfikacja pliku */etc/fstab*. Do opcji *mount* należy dopisać słowo kluczowe *acl* – choć jeśli korzystasz z systemu plików Xfs lub Jfs, nie musisz dodawać tej opcji. Od tej chwili twój system jest gotowy do korzystania z potężnego mechanizmu list kontroli dostępu – ACL. ■



Rysunek 2. Mechanizm działania list praw dostępu – ACL.

SŁOWNICZEK

Potok nazwany (named pipe) potok z nadaną nazwą jest specjalnym rodzajem pliku tymczasowego wykorzystującego metodę kolejkowania FIFO podczas operacji odczytu i zapisu.

INFO

[1] Program Star tape: http://acl.bestbits.at/download.html#Star

[2] Poprawka ACL dla jądra systemu Linux: http://acl.bestbits..at/download.html

[3] Opis ACL krok po kroku: http://acl.bestbits.at/steps.html

AUTOR

Volker Schmitt jest matematykiem i pracuje dla dużej firmy ubezpieczeniowej. Ma doświadczenie w programowaniu wielozadaniowym maszyn mainframe typu PL/I. W czasie wolnym Volker zajmuje się NQC i LEGO Spybot.