Linux - prawa dostępu, skrypty powłoki, dowiązania

ROOT

Wyróżniającym się spośród wszystkich użytkowników w Linuksie jest *root*, czyli superużytkownik. *Root* posiada specjalne przywileje do każdego katalogu, pliku i konfiguracji. Pełni on rolę administratora całego systemu.

Prawa dostępu w Linuksie

W systemie operacyjnym Linux każdy plik i katalog posiada zestaw praw określający, kto ma dostęp do pliku i jakie ma prawa. Każdy plik lub katalog może mieć prawo:

r - czytania (*read*),
w - pisania (*write*),
x - wykonywania (*eXecute*).

W przypadku katalogu "eXecute", czyli uruchomienie, oznacza możliwość przeglądania jego zawartości.

Każde z tych praw dostępu reprezentowane jest odpowiednią literą i posiada przypisany odpowiedni parametr cyfrowy.

Litera	Znaczenie	Parametr liczbowy
r	prawo odczytu	4
w	prawo zapisu	2
x	prawo uruchomienia	1
-	brak praw dostępu	0

Do każdego pliku lub katalogu możemy wyszczególnić trzy zestawy takich praw:

- prawa właściciela
- prawa grupy
- prawa pozostałych użytkowników

Do wyświetlenia praw dostępu do plików można posłużyć się znanym poleceniem:

```
ls -l
```

Pierwsza kolumna składająca się z dziesięciu znaków opisuje prawa dostępu, przy czym:

- pierwszy znak określa rodzaj pliku (np. - (minus) oznacza plik, d oznacza katalog, l dowiązanie itd).
- znak drugi, trzeci i czwarty określają prawa właściciela do pliku,
- znak piąty, szósty i siódmy określają prawa grupy do której należy użytkownik,
- znak ósmy, dziewiąty i dziesiąty - prawa innych użytkowników systemu.

Dodając do siebie odpowiednie parametry, zestaw trzech praw możemy przedstawić za pomocą jednej cyfry.

Oto kilka najczęściej spotykanych kombinacji:

Prawa	Wartość	Znaczenie
---	0	brak praw
--x	1	prawo do uruchomienia
-w-	2	prawo zapisu
-wx	3	prawo zapisu i uruchomienia
r--	4	prawo do odczytu
r-x	5	prawa do odczytu i uruchomienia
rw-	6	prawa do odczytu i zapisu
rwx	7	prawa do odczytu zapisu i uruchomienia

W ten sposób za pomocą trzech cyfr możemy przedstawić całkowity (tzn. trzy zestawy) zbiór praw dostępu do pliku (pierwsza cyfra - prawa właściciela, druga cyfra - prawa grupy, trzecia - prawa dla pozostałych użytkowników).

Prawa dostępu	Wartość liczbowa	Znaczenie
-rw-------	600	Prawa do odczytu i zapisu tylko dla właściciela pliku
-rw-r--r--	644	Prawa odczytu i zapisu dla właściciela oraz odczytu dla wszystkich pozostałych użytkowników.
-rw-rw-rw-	666	Prawa odczytu i zapisu dla wszystkich użytkowników.
-rwx------	700	Wszystkie prawa (odczyt, zapis, uruchomienie) tylko dla właściciela pliku.
-rwxr-xr-x	755	Prawa do odczytu, zapisu i uruchomienia dla właściciela pliku oraz odczytu i uruchomienia dla wszystkich innych użytkowników
-rwxrwxrwx	777	Wszystkie prawa dla wszystkich użytkowników (ustawienie niebezpieczne)
-rwx--x--x	711	Prawa odczytu zapisu i uruchomienia dla właściciela pliku oraz prawo uruchomienia dla pozostałych użytkowników.
drwx------	700	Dotyczy prawa zapisu i odczytu w katalogu tylko przez właściciela. Katalogom zawsze musi być ustawione prawo dostępu x.
drwxr-xr-x	755	Do takiego katalogu wszystkie prawa ma właściciel, a pozostali użytkownicy mogą tylko odczytać jego zawartość.
drwx--x--x	711	Wszystkie prawa ma właściciel. Katalog z takimi prawami dostępny jest także dla wszystkich pozostałych użytkowników, lecz jego zawartość jest przed nimi ukryta (polecenie ls nie wyświetli listy plików umieszczonych w tak oznaczonym katalogu). Aby odczytać plik użytkownik musi znać jego nazwę.

Zmiana właścicieli plików

Do zmiany właściciela pliku służy polecenie **chown**.

Należy mieć uprawnienia do takich zmian lub zalogować się jako *root* i napisać:

```
chown  nazwa_właściciela  nazwa_pliku
```

Ponieważ nie mamy praw root'a, nie możemy tego wykonać.

Zmiana praw dostępu do plików

Do zmiany praw dostępu do plików w systemie Linux służy polecenie **chmod**.

Składnia polecenia przy wykorzystaniu metody cyfrowej:

```
chmod  prawa_dostępu  nazwa_pliku
```

gdzie *prawa_dostępu* to jedna z opisanych wyżej trzycyfrowych liczb.

Przykładowo aby zmienić prawa dostępu do pliku readme.txt na -rw------ (prawo odczytu i zapisu tylko dla właściciela pliku) musimy napisać:

```
chmod  600  readme.txt
```

Gdybyśmy chcieli zmienić prawa dostępu do wszystkich plików w danym katalogu i jego podkatalogach należy skorzystać z opcji –R.

Przykładowo - aby ustawić zestaw uprawnień 644 dla wszystkich plików w katalogu /home/waldek/pub należy napisać:

chmod -R 644 /home/waldek/pub

Inną metodą nadawania uprawnień jest metoda symboliczna, polega na zdefiniowaniu odpowiedniego ciągu pojedynczych liter. W tabelce poniżej przedstawione są znaczenia poszczególnych liter.

Kto	Działanie	Uprawnienie
u użytkownik	+ nadać	r odczyt
g grupa	- zabrać	w zapis
o inni	= przypisać	x wykonanie
a wszyscy		

Składnia polecenia przy wykorzystaniu tego sposobu:

chmod [ugoa][+ - =][rwxs] nazwa_pliku

Uwaga: nawias kwadratowy opuścić, elementy w nawiasie [ ] są do wyboru.

Przykłady:

$ chmod +x playit - plik *playit* może być uruchomiony przez dowolnego użytkownika

$ chmod u+x playit - daje właścicielowi pliku *playit* prawo jego wykonania

$ chmod a–w readme - odbiera wszystkim prawo do zapisu do pliku *readme*

$ chmod a+r * - daje wszystkim użytkownikom prawo do odczytu do wszystkich plików znajdujących się w danym katalogu.

Podobnie można nadawać i odbierać prawa dla katalogów. Prawo r dla katalogu odpowiada za listowanie zawartości katalogu, prawo w jest oczywiste (zapis w katalogu, tworzenie plików, zmiany nazw itp.), od prawa x zależy możliwość uczynienia katalogu katalogiem bieżącym (można lub nie można "wejść" do katalogu").

Pliki wykonywalne - skrypty powłoki

Plik **wykonywalny** (np. plik tekstowy, którego treścią będzie polecenie zakładania katalogu):

cat> nazwa_pliku
mkdir NOWY
CTRL+D

Można w pliku umieścić ciąg wielu poleceń. Należy ndać plikowi *nazwa_pliku* prawa wykonywania dla siebie. Udokumentować działanie (także np. po przelogowaniu na innego użytkownika). Uwaga: wykonanie pliku odbywa się po wpisaniu w wierszu poleceń jego nazwy w postaci:

./nazwa_pliku - *wykonaj plik nazwa_pliku*

UWAGA: W terminalu webminal.org: sh ./nazwa_pliku

Po wykonaniu pliku sprawdzić, czy utworzony został katalog NOWY.

Dowiązania

Dowiązanie to pierwszym przybliżeniu inna nazwa tego samego pliku (pewna analogia do *skrótów* w *Windows*). Dowiązania dzielimy na:

- dowiązanie miękkie, inaczej zwane symbolicznymi wykonywane poleceniem:

ln –s nazwa pliku nazwa_dowiazania

- dowiązanie sztywne wykonywane poleceniem:

ln nazwa_pliku nazwa_dowiazania

Dowiązanie symboliczne to skojarzenie nazwy pliku z istniejącą ścieżką dostępu do pliku. Pozwala zastąpić długą, pełną ścieżkę dostępu do pliku kilkuliterowym "skrótem", który jest niewielkim plikiem, na przykład:

cat > mojplik1
Cos zapisane w pliku
CTRL+D

```
ln -s mojplik1 skrsym
ls -l
```

Teraz można wyświetlić treść pliku przez skrót:

```
cat skrsym
```

Jeśli usuniemy plik *mojplik* to dowiązanie przestanie "działać".

Każdą nazwę pliku skojarzoną wprost z i-węzłem nazywamy dowiązaniem sztywnym, często dowiązaniem (ang. *link*). Zatem każdy plik ma co najmniej jedno sztywne dowiązanie, gdyż musi mieć co najmniej jedną nazwę. Nazwa potrzebna jest tylko człowiekowi, system identyfikuje pliki w oparciu o niepowtarzalny (wewnątrz konkretnego systemu plików) numer i-węzła. i-węzeł nie przechowuje nazwy pliku, do przechowywania nazwy pliku służą katalogi.

i-węzeł jest strukturą, w której jądro systemu przechowuje informacje o pliku. Każdy plik ma dokładnie jeden unikatowy i-węzeł. Dowiązanie sztywne jest inną nazwą tego samego pliku. Możemy zatem tworzyć wiele nazw tego samego fizycznego plik, co daje oszczędność miejsca na dysku.

```
ln mojplik1 skrszt
cat skrszt
```

Można się przekonać, że *mojplik* i *skrszt* prowadzą do tego samego i-węzła (pięciocyfrowe liczby na początku każdego wiersza) poleceniem:

```
ls -li
```

Jeśli usuniemy plik *mojplik*:

```
rm skrszt  mojplik
```

przekonamy się, że przez dowiązanie sztywne (inną nazwę) mamy nadal dostęp do pliku:

Różnice:

- Dowiązanie sztywne zachowuje pełne prawa dostępu i właścicieli oryginału.
- Dowiązanie symboliczne oferuje pełne prawa wszystkim, a jego właścicielem jest ten co je utworzył.

.

Porównajmy informacje o plikach *mojplik1* i *nowyszt* - mają te same numery i-węzłów i identyczne prawa dostępu. Porównajmy też *mojplik1* z *nowysym*. Plik *nowysym* posiada inny numer i-węzła oraz inne, pełne prawa dostępu i tylko jedno dowiązanie, inną długość i datę utworzenia lub modyfikacji. Można go traktować jak zupełnie inny plik. O tym, że jest to dowiązanie symboliczne, informuje litera "`l`" umieszczona przed prawami dostępu oraz strzałka wskazująca na *mojplik1*, czyli obiekt, do którego utworzone jest dowiązanie symboliczne.

Zmienimy teraz nazwę pliku mojplik1:

```
$ cat mojplik
Cos zapisane w pliku
$ cat nowyszt
Cos zapisane w pliku
$ cat nowysym
Cos zapisane w pliku
$ mv mojplik inny
$ ls
inny  kat2  mojplik2  nowysym  nowyszt
$ cat inny
Cos zapisane w pliku
$ cat nowyszt
Cos zapisane w pliku
$ cat nowysym
cat:  nowysym: No such file or directory
```

Efekt - dowiązanie sztywne związane z "fizycznym" plikiem (i-węzłem) 32847 nie zareagowało na zmianę nazwy *mojplik1* na *inny*. Natomiast dowiązanie symboliczne po zmianie nazwy trafia "w próżnię", gdyż nazwa *mojplik1* już nie istnieje.

Inne operacje plikowe

Porównywanie treści plików

Polecenie **cmp** o postaci:

$ **cmp** p1 p2

znajduje pierwszy bajt różnicy w treści obu plików p1 i p2. Jeśli chcemy znaleźć wszystkie różnice używamy opcji –l:

$ **cmp** –l p1 p2

Używając opcji `-l` (*list*) powodujemy, że polecenie **cmp** nie spowoduje zatrzymania przy pierwszej napotkanej różnicy, lecz doprowadzi porównanie plików do końca, z wylistowaniem wszystkich napotkanych różnic (kody ASCII znaków).

Wyszukiwanie plików

Polecenie **find** prowadzi wyszukiwanie według zadanych kryteriów od wskazanego katalogu "w dół", uwzględniając wszystkie podkatalogi (rekurencyjnie). Jeżeli wskazanym katalogiem jest /, wówczas przeszukiwany jest cały system plików.

Przykłady:

$ **find** ~ -name passwd
$ **find** ~ -name "p*"
$ **find** / -name "p*"

W powyższym zapisie tylda "~" oznacza katalog domowy użytkownika, czyli punkt rozpoczęcia poszukiwań, -name ta opcja określa wyszukiwanie według nazwy. Jeżeli nie podamy katalogu rozpoczęcia poszukiwań, find automatycznie przyjmie bieżący.

Ćwiczenie

1. Utworzyć nowy plik (pliki). Jakie ma prawa po utworzeniu? Zmieniać opisanymi metodami prawa dostępu do tego pliku.
2. Jak zmiana praw do plików wpływa na możliwość odczytu pliku, zmiany nazwy, usuwania itp.
3. Wykonać zmiany praw dotyczące katalogu (uwaga: bieżącym musi być katalog nadrzędny). Ustawiać tak prawa, żeby:
    a. użytkownik chronił wszystkie pliki tego katalogu przed zmianą treści,
    b. nie miał do niego dostępu żaden inny użytkownik, tylko grupa, wszyscy.
    c. inni mieli prawo dostępu ale nie mieli prawa listowania zawartości.
4. Jakie prawa dostępu ma katalog domowy użytkownika, sprawdzić czy użytkownik-właściciel może te prawa zmieniać i jaki to ma wpływ na dostęp do naszego katalogu przez innych użytkowników.
5. Zmiana właściciela pliku - czy nam się to uda i dlaczego, jeśli nie.
6. Zalogować się na inne konto i wypróbować zmianę praw dostępu do plików o poprzednio zmodyfikowanych prawach.
7. Wypróbować symboliczny sposób zmian praw.
8. Utworzyć do dowolnego pliku dowiązanie sztywne i symboliczne.
9. Sprawdzić co dzieje się jeśli usuniemy dowiązanie sztywne, co gdy symboliczne. Co się stanie z plikami dowiązań gdy usuniemy główny plik. Jakie prawa domyślne otrzymują pliki obydwu typów dowiązań?
10. Przetestować wyszukiwanie plików o zadanym wzorcu nazwy w zakresie katalogu domowego.
11. Utworzyć dwa pliki tekstowe o podobnej treści i sprawdzić efekt porównania ich treści bez opcji **`-l`** i z tą opcją.