

su, sudo

# Nowa tożsamość

Lepiej być ostrożnym, niż później żałować. Pamiętaj o tej zasadzie – nawet jeżeli masz w systemie dostęp do konta root, to aby zapobiec przypadkowym awariom, rozsądnie jest używać uprawnień administracyjnych tylko tymczasowo. Polecenia `su` i `sudo` pozwalają szybko przełączyć się na innego użytkownika z wiersza poleceń.

HEIKE JURZIK

Uprawnienia root-a są potrzebne do czynności administracyjnych, jednak nie ma potrzeby, aby przez cały czas być super-użytkownikiem. Bardziej wskazane jest używanie konta `root` tylko do wybranych działań administracyjnych, po czym powrót do uprawnień normalnego użytkownika. Dwa polecenia: `su` i `sudo` pozwalają na przełączanie konta (identyfikatora – ID) innego użytkownika.

## Su

Polecenie `su` ('substitute user') pozwala przełączać ID z linii komend. Polecenie wywołuje w tle nową powłokę, używając ID nowego użytkownika (**UID**) i grupy ID (**GID**). Kiedy uruchamiasz `su`, żeby zostać administratorem lub innym uprzywilejowanym użytkownikiem, musisz też znać hasło do tego konta.

Podstawowa składnia polecenia to `su [-] [nazwa_użytkownika]` -- ale istnieją subtelne różnice w zależności od tego, czy wpiszesz znak minus czy nie. Znak ten (lub w zastępstwie parametr `-l` albo `--login`) gwarantuje, że jesteś aktualnie zalogowany, tak więc ustawiane są odpowiednie zmienne środowiskowe, powłoka i odpowiedni katalog użytkownika. Zmienne środowiskowe pozostają niezmienione, jeżeli opuścisz znak minus – będzie to oznaczać, że nowy użytkownik nie posiada uprawnień do bieżącego katalogu (rys 1).



Rysunek 1. Bez odpowiedniego zalogowania nie uzyskasz żadnych uprawnień.

Jeżeli nie określisz nazwy użytkownika domyślnie, zakłada się, że jest to konto administratora `root`, wynika to ze złego założenia, że `su` jest skrótem od `superuser`. Domyślnie polecenie `su` nie pozwala nowemu użytkownikowi na uruchomienie aplikacji X-ów. Zewnętrzni użytkownicy muszą najpierw uzyskać zezwolenie na używanie X serwera jako standardowego – oznacza to konieczność edycji pliku `Xauthority` w katalogu domowym użytkownika (zobacz też `man xauth`). Aby zezwolić użytkownikowi `root` na uruchamianie programu w X-ach na Xterm należącym do użytkownika `petronella`, musimy wydobyć klucz z `Xauthority` i dodać do `Xauthority` administratora, a następnie przededefiniować zmienną `DISPLAY` (listing 1).

Program `su` pozwala także użyć innego konta do uruchomienia pojedynczego polecenia. Aby to zrobić, określ parametr `-c` (`--command`):

```
huhn@asteroid:~$ su -c „less
/var/log/messages”
Password:
```

Użycie polecenia `su` jest zapisywane w logu. W zależności od używanej dystrybucji, wpisy te są umieszczane w `/var/log/auth.log` (e.g. Debian) lub `/var/log/messages` (SuSE Linux). Błędne próby są łatwe do znalezienia, co pozwala administratorowi szybko zorientować się, kto próbował korzystać z cudzych uprawnień w celu wejścia na roota:

```
Dec 22 14:50:50 asteroid su
PAM_unix[2108]: authentication failure;
(uid=500) -> root for su service
Dec 22 14:50:52 asteroid su
```

```
su[2108]: pam_authenticate: Authentication failure
Dec 22 14:50:52 asteroid su
su[2108]: - pts/8 huhn-root
```

Użytkownik z uprawnieniami administratora nie musi wprowadzać hasła po wpisaniu polecenia `su`. Może on przyjąć każdą tożsamość, by sprawdzić modyfikacje z perspektywy innego użytkownika.

## Użycie sudo

Polecenie `sudo` pozwala uniknąć ujawniania hasła roota, co jest zrozumiałe ze względu na bezpieczeństwo. Polecenie działa tak, jak sugeruje nazwa: `sudo` jest skrótem od substitute user i pozwala nadać pojedynczemu użytkownikowi lub grupie uprawnienia administratora na ograniczony czas lub ograniczone do określonego zadania. Użytkownik może teraz zwyczajnie wpisać swoje hasło i uruchomić uprzywilejowane polecenie.

Administrator musi utworzyć listę użytkowników posiadających zezwolenie na wykonanie określonych uprzywilejowanych poleceń w pliku `/etc/sudoers`. Pracując jako `root`, należy zredagować plik poleceniem `visudo`. Program ten oferuje możliwości znane z edytora `vi` i kilka dodatkowych funkcji. Edytor `visudo` blokuje dostęp do pliku `/etc/sudoers`, nie pozwalając na edycję w tym samym czasie przez wielu użytkowników. Wychodząc z programu `visudo` sprawdza użyta składnię, informując o znalezionych błędach:

```
>>> sudoers file: syntax error, line 20 <<<
What now?
```

Mamy trzy wyjścia: nacisnąć `e` i powrócić do edycji, `x` – aby anulować zmiany i opuścić edytor lub `Q`, aby jednak zapisać zmiany do pliku.

W `/etc/sudoers` istnieje domyślny wpis `root ALL=(ALL) ALL`. Pozwala to użytkownikowi z prawami roota robić wszystko, ale przecież root może wszystko również bez `sudo`. Jeżeli potrzebujemy nadać nieograniczone prawa root-a innemu użytkownikowi, wystarczy skopiować tą linię i zamienić `root` na nazwę tego użytkownika.

## Listing 1: Ustawienie zmiennej DISPLAY.

```
petronella@asteroid:~$ xauth extract
key $DISPLAY
huhn@asteroid:~$ su -
Password:
asteroid:~# xauth merge /home/huhn/key
asteroid:~# export DISPLAY=:0.0
```

nika. Po zapisaniu pliku użytkownik ten będzie mógł wykonywać polecenia administracyjne, używając przedtem *sudo*, przykładowo:

```
huhn@asteroid:~$ sudo
/sbin/shutdown
Password:
```

Jeżeli użytkownik nie ma pozwolenia na użycie *sudo*, otrzyma on komunikat: *huhn is not in the sudoers file. This incident will be reported..* Ten domyślny sposób działania możemy zmienić w pliku */etc/sudoers*. Ostrzeżenie ze szczegółami dotyczącymi użytkownika, który próbował użyć *sudo*, może zostać wysłane do administratora e-mailem (rys 2). Aby uchronić się przed tym, nieuprzywilejowani użytkownicy mogą wpisać *sudo -l* i wyświetlić sobie listę dozwolonych komend.

## Szczegółowy nadzór

Sekcja *Host alias specification* w */etc/sudoers* umożliwia określenie, na którym komputerze, jaka komenda *sudo* może zostać użyta. Możemy użyć *Host Alias* i podając ich nazwy lub zakres IP utworzyć grupę komputerów. Opcja ta ma sens jedynie wtedy, gdy będziemy stosować centralnie zarządzaną konfigurację *sudo* dla wielu komputerów.

Sekcja *User Alias* pozwala grupować użytkowników, którzy wymagają posiadania tych samych uprawnień. Najpierw definiujemy typ aliasu (n.p. *User Alias*), następnie nazwę dla aliasu (mogącą zawierać duże litery, podkreślenia i cyfry), przyporządkowanie wskazywane przez znak = i na końcu oddzieloną przecinkiem nazwę użytkownika. Dodajemy więc użytkowników *huhn* i *petronella* do grupy, której wolno zamknąć system:

```
# User alias specification
User_Alias SHUTTERSDOWN = petronella, huhn
```

Następnym krokiem będzie zdefiniowanie aliasu do polecenia *shutdown* w sekcji *Cmd alias specification*. Aby to zrobić, wpisujemy bezwzględną ścieżkę do potrzebnego programu:

```
# Cmd alias specification
Cmd_Alias DOWN = /sbin/shutdown
```

Aby przekazać do *sudo*, że *SHUTTERS-DOWN* mają pozwolenie na użycie tego polecenia, potrzebny będzie jeszcze jeden wpis poniżej *User privilege specification*:

```
SHUTTERSDOWN ALL = DOWN
```

Użytkownicy w grupie *SHUTTERS-DOWN* mogą teraz zamknąć system na komputerze wpisując *sudo /sbin/shutdown*. Istnieje jednak łatwiejszy sposób, aby nadać uprawnienia pojedynczemu użytkownikowi do wykonania określonej komendy. Dla przykładu wpis:

```
huhn ALL = /usr/sbin/visudo
```

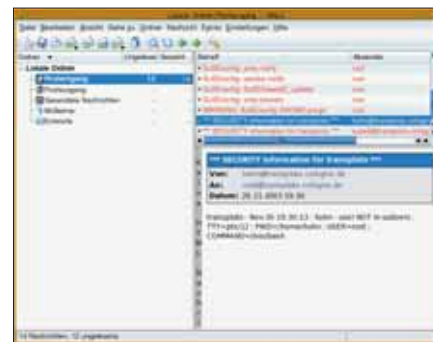
nadaje użytkownikowi *huhn* pozwolenie na edycje pliku */etc/sudoers*, używając do tego polecenia *sudo /usr/sbin/visudo*.

## Zabronione, czy dostępne dla wszystkich?

Prosty wpis w */etc/sudoers* pozwala nam odebrać przywileje poszczególnym użytkownikom. Użyjemy tutaj następującej składni:

```
SHUTTERSDOWN ALL = DOWN
petronella ALL = !DOWN
```

Ważne jest, aby określać te wyjątki bezpośrednio po regule, ponieważ plik jest odczytywany z góry na dół. Pozwoli to nam utrzymywać zezwolenie na wykonywanie innych poleceń dla grupy *SHUTTERS-DOWN*, nie zezwalając jednocześnie użytkownikowi *petronella* na zamknięcie systemu. Jeżeli *petro-*



Rysunek 2. Bezpieczeństwo będzie nagrodzone – *sudo* informuje o nieupoważnionym dostępie.

*nella* spróbuje zamknąć system, otrzyma wiadomość: „Sorry, user *petronella* is not allowed to execute *'/usr/sbin/visudo'* as root on *asteroid.linux-magazine.com.*”

W celu ukrycia zachęty do wprowadzenia hasła dla pojedynczego lub wielu poleceń, wystarczy ustawić parametr *NOPASSWD*:

```
SHUTTERSDOWN ALL=NOPASSWD:DOWN
```

Zamiast obniżać poziom bezpieczeństwa, co jest efektem stosowania *sudo*, można go zwiększyć, zmuszając użytkowników do wpisywania swoich hasel za każdym razem, kiedy będą uruchamiać *sudo*. Domyślnie *sudo* działa jak system biletów czasowych, oznacza to, że na przykład osierocona powłoka *roota* na konsoli nie pozwoli na pracę na konsoli. Domyślny termin ważności biletu to 15 minut. Można ustawić ten czas na 0 minut, dodając do pliku */etc/sudoers*:

```
Defaults timestamp_timeout = 0
```

## Opcjonalnie

Program *sudo* dysponuje również kilkoma parametrami do sterowania z linii komend. Prawdopodobnie najważniejszy z nich to *-s*, pozwalający uruchomić główną powłokę. Wystarczy po prostu wpisać *sudo s*, żeby umożliwić administratorowi uruchamianie programów w X-ach, bez potrzeby ręcznego konfigurowania dostępu do X serwera. Parametr *-L* wylistuje wszystkie opcje w pliku */etc/sudoers*. Jeżeli chcesz przedłużyć ważność sesji *sudo* bez uruchamiania polecenia, wystarczy wpisać *sudo -v*. W przypadku, gdy czas ważności sesji się skończył, zostaniesz zapytany o hasło. Można także wyłączyć sesję wpisując *sudo k*. Parametr *-b* uruchamia polecenie w tle, chociaż nie będzie można do niego wrócić posługując się standardowym poleceniem *fg*. ■

## SŁOWNICZEK

**UID:** Każdego użytkownika można zidentyfikować po jego UID („User IDentification number”), który jest unikalnie przyporządkowany kontu użytkownika. Swoją własny UID można łatwo znaleźć pisząc *echo \$UID*.

**GID:** Poza UID użytkownicy posiadają tak zwany GID („Group IDentification number”), który wskazuje na przynależność do danej grupy. Wchodzący w skład grupy mogą posiadać wspólne uprawnienia. Polecenie *id* pokazuje aktualny UID i GID użytkownika.