

## Harmonogram zadań

Cel ogólny lekcji: Nauczenie się planowania zadań przy pomocy narzędzi cron i at.

Cele szczegółowe lekcji, uczestnik lekcji:

1. Nauczy się, jak skonfigurować i korzystać z narzędzia at w celu zaplanowania zadania na konkretną godzinę.
2. Pozna, jak korzystać z narzędzia cron w celu zaplanowania zadań cyklicznych.
3. Nauczy się konfigurować zadania crontab dla użytkownika oraz dla konta root.
4. Nauczy się, jak przeglądać zaplanowane zadania i usuwać je z listy.
5. Będzie w stanie potwierdzić, czy zaplanowane zadania działają poprawnie, przeglądając odpowiednie pliki z logami.

## Ćwiczenia

Nauczysz się jak planować zadania przy pomocy narzędzi cron i at.

### 1 Harmonogram zadań narzędziem at

Wykonamy następujące czynności:

W X-ach zaloguj się jako ubuntu z hasłem ubuntu

sudo -s hasło ubuntu

Lub w oknie terminala zaloguj się jako root (hasło roota 1234).

Zainstaluj:

**apt install at**

**apt install finger**

Sprawdź, czy usługa at pracuje, poleceniem: **systemctl status atd.service**

```
root@ubuntu-VirtualBox:/home/ubuntu# systemctl status atd.service
● atd.service - Deferred execution scheduler
   Loaded: loaded (/lib/systemd/system/atd.service; enabled; vendor preset: e>
   Active: active (running) since Tue 2023-07-18 14:37:36 CEST; 55s ago
     Docs: man:atd(8)
  Process: 2294 ExecStartPre=find /var/spool/cron/atjobs -type f -name =* -no>
   Main PID: 2295 (atd)
    Tasks: 1 (limit: 9414)
   Memory: 244.0K
      CPU: 7ms
   CGroup: /system.slice/atd.service
           └─2295 /usr/sbin/atd -f
```

(Opcjonalnie) Jeśli usługa nie pracuje (status: unused), uruchamiamy ją poleceniem:

```
systemctl start atd.services
```

Wyświetlamy bieżącą datę i czas poleceniem:

```
date
```

Trzy minuty później ma zostać dopisana w pliku `/var/log/syslog` informacja o tym, kto jest aktualnie zalogowany:

```
at hh:mm
```

```
finger >> /var/log/syslog
```

Wychodzimy z edytora poprzez wciśnięcie Ctrl+d.

Przeglądamy zaplanowane prace dla at poleceniem:

```
atq (lub at -l).
```

Notujemy numer pracy.

Odczekujemy 3 minuty.

Po upływie zaplanowanego czasu - sprawdzamy zawartość końcowej partii pliku `/var/log/syslog` poleceniem:

```
tail /var/log/syslog
```

Widzimy, że została dopisana Informacja o zalogowaniu.

Planujemy to samo zadanie dla at, ale na południe następnego dnia

```
at noon tomorrow
```

```
finger >> /var/log/syslog
```

Wychodzimy z edytora przez Ctrl+d

Planujemy dopisanie bieżącej daty do pliku `/var/log/syslog` na jutro o godzinie 14:00:

```
at 14:00 tomorrow
```

```
date >> /var/log/syslog
```

Wychodzimy z edytora Ctrl + d.

Przeglądamy zaplanowane prace dla at:

**atq** (lub **at -l**)

Zauważamy, że na liście są dwie prace, każda z indywidualnym numerem pracy.

Usuwamy z listy zadanie zaplanowane na jutro o 14:00, poleceniem:

**atrm** job\_number

Przeglądamy zaplanowane prace dla at :

**atq** (lub **at -l**)

Widzimy, że tylko zadanie zaplanowane na 12:00 jest nadal na liście

(zgłoszenie nr1)

Zamykamy okno terminala.

## 2 Harmonogramowanie zadań przy pomocy cron

Wykonamy następujące czynności:

Uruchamiamy terminal.

Sprawdź, czy w naszym systemie usługa działa. W tym celu wykonaj polecenie:

**/etc/init.d/cron status**

```
root@ubuntu-VirtualBox:~# /etc/init.d/cron status
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: e
nabled)
   Active: active (running) since Tue 2023-07-18 14:34:43 CEST; 11min ago
```

```
root@ubuntu-VirtualBox:~# /etc/init.d/cron
●* Usage: /etc/init.d/cron {start|stop|status|restart|reload|force-reload}
```

Jak widać - usługa działa:

Pracę dla crontab zaplanujemy jako użytkownik ubuntu.

wprowadzamy polecenie liniowe:

**crontab -e**

Otwórz plik crontab użytkownika ubuntu w edytorze vi. (2)

klawiszem Ins przełączamy vi w tryb edycji,

**~/users.log**

zapisujemy to zadanie w pliku crontab w postaci:

```
* * * * * finger >> ~/users.log
```

zapisujemy plik crontab i wychodzimy z edytora vi: Esc a następnie **:wq**,

obserwujemy, co się dzieje w pliku users.log przez kilka minut:

```
tail -F ~/users.log (dokumentujemy to jako 1)
```

przerywamy wyświetlanie zawartości pliku > Ctrl+c.

Kasujemy plik crontab dla ubuntu poleceniem:

```
crontab -r
```

Upewniamy się, że plik crontab już nie istnieje:

```
crontab -l
```

Upewniamy się, że zadanie cron, zdefiniowane w tym ćwiczeniu, nie jest już aktywne:

```
tail -F ~/users.log (dokumentujemy to jako 2)
```

Widzimy, że nie są dodawane wpisy do users.log

Kończymy pracę tail kombinacją klawiszy Ctrl+c.

Definiujemy i uruchamiamy zadanie dla cron na konto root:

w oknie terminala uruchamiamy sesję powłoki dla roota: su - (hasło 1234),

wprowadzamy polecenie:

```
crontab -e
```

przechodzimy w tryb edycji vi klawiszem Ins,

Do listy zadań cron roota dodajemy zadanie uruchamiania w każdy wtorek o godzinie 14 programu tar celem utworzenia kopii zapasowej katalogu /etc w katalogu /tmp:

```
0 2 * * 2 tar czvf /tmp/etc.tgz /etc
```

wychodzimy z trybu edycji vi klawiszem Esc, a następnie zapisujemy zmiany w pliku i zamykamy edytor **:wq**,

upewniamy się, że zadanie zostało umieszczone w pliku crontab dla root:

```
crontab -l (dokumentujemy to jako 3)
```

Kasujemy plik crontab użytkownika root poleceniem:

```
crontab -r
```

Upewniamy się, że plik crontab już nie istnieje, poleceniem

```
crontab -l
```

(zgłoszenie nr2)

Zamykamy wszystkie otwarte okna.

### 3 Skrypty w katalogach cron.x

#### 1. Wprowadzenie

Kolejne miejsce związane z tematem to katalogi, w których znajdują się skrypty uruchamiane w określonych okresach.

Sprawdź, czy są to:

- /etc/cron.daily - codziennie
- /etc/cron.hourly - co godzinę
- /etc/cron.weekly - raz w tygodniu
- /etc/cron.monthly - raz w miesiącu

Za pomocą poleceń:

```
ls /etc/cron*
```

```
ls /etc/ -l |grep cron
```

```

root@ubuntu-VirtualBox:~# ls /etc/cron*
/etc/crontab

/etc/cron.d:
anacron e2scrub_all

/etc/cron.daily:
0anacron appport apt-compat cracklib-runtime dpkg logrotate man-db

/etc/cron.hourly:

/etc/cron.monthly:
0anacron

/etc/cron.weekly:
0anacron man-db
root@ubuntu-VirtualBox:~# ls /etc/ -l |grep cron
-rw-r--r-- 1 root root 335 mar 23 2022 anacrontab
drwxr-xr-x 2 root root 4096 lip 15 12:39 cron.d
drwxr-xr-x 2 root root 4096 lut 23 04:59 cron.daily
drwxr-xr-x 2 root root 4096 lut 23 04:57 cron.hourly
drwxr-xr-x 2 root root 4096 lut 23 04:59 cron.monthly
-rw-r--r-- 1 root root 1136 mar 23 2022 crontab
drwxr-xr-x 2 root root 4096 lut 23 04:59 cron.weekly

```

(zgłoszenie nr3)

## 2. Katalog cron.daily

w systemie Ubuntu zawiera skrypty, które są automatycznie wykonywane codziennie przez demon cron. Możesz umieścić własne skrypty w tym katalogu, a zostaną one uruchamiane raz dziennie zgodnie z harmonogramem cron. Oto przykład korzystania z katalogu cron.daily:

1. Otwórz terminal w systemie Ubuntu.
2. Przejdź do katalogu cron.daily za pomocą polecenia:

```
cd /etc/cron.daily
```

3. Utwórz nowy skrypt bash, na przykład my\_daily\_script.sh, za pomocą swojego ulubionego edytora tekstowego:

```
nano my_daily_script.sh
```

4. Wpisz treść skryptu, na przykład:

```
#!/bin/bash
```

```
echo "To jest mój codzienny skrypt!"
```

```
date
```

```
# Dodaj inne polecenia, które mają zostać wykonane codziennie
```

Skrypt może zawierać dowolne polecenia, które chcesz wykonać codziennie.

5. Zapisz i zamknij plik.

6. Upewnij się, że skrypt ma odpowiednie uprawnienia do wykonania:

```
chmod +x my_daily_script.sh
```

7. Skrypt my\_daily\_script.sh zostanie automatycznie wykonany codziennie przez demon cron, o ile masz zainstalowany i uruchomiony daemon cron na swoim systemie.

Należy pamiętać, że skrypty w katalogu cron.daily muszą mieć odpowiednie uprawnienia do wykonania, czyli muszą być wykonywalne. Ważne jest również, aby treść skryptu była poprawnie napisana i testowana, aby uniknąć nieoczekiwanych efektów w systemie.

```
cat my_daily_script.sh
```

(zgłoszenie nr4)

### 3. Katalog /etc/cron.hourly

w systemie Ubuntu zawiera skrypty, które są automatycznie wykonywane co godzinę przez demon cron. Możesz umieścić własne skrypty w tym katalogu, a zostaną one uruchamiane raz na godzinę zgodnie z harmonogramem cron.

Oto przykład korzystania z katalogu /etc/cron.hourly:

1. Otwórz terminal w systemie Ubuntu.

2. Przejdź do katalogu /etc/cron.hourly za pomocą polecenia:

```
cd /etc/cron.hourly
```

3. Utwórz nowy skrypt bash, na przykład my\_hourly\_script.sh, za pomocą swojego ulubionego edytora tekstowego:

```
nano my_hourly_script.sh
```

4. Wpisz treść skryptu, na przykład:

```
#!/bin/bash
```

```
echo "To jest mój skrypt, który wykonuje się co godzinę!"
```

```
date
```

```
# Dodaj inne polecenia, które mają zostać wykonane co godzinę
```

Skrypt może zawierać dowolne polecenia, które chcesz wykonywać co godzinę.

5. Zapisz i zamknij plik.

6. Upewnij się, że skrypt ma odpowiednie uprawnienia do wykonania:

```
chmod +x my_hourly_script.sh
```

7. Skrypt my\_hourly\_script.sh zostanie automatycznie wykonany co godzinę przez demon cron, o ile masz zainstalowany i uruchomiony daemon cron na swoim systemie.

Należy pamiętać, że skrypty w katalogu /etc/cron.hourly muszą mieć odpowiednie uprawnienia do wykonania, czyli muszą być wykonywalne. Ważne jest również, aby treść skryptu była poprawnie napisana i testowana, aby uniknąć nieoczekiwanych efektów w systemie.

```
cat my_hourly_script.sh
```

(zgłoszenie nr5)

#### 4. Katalog cron.monthly

w systemie Ubuntu zawiera skrypty, które są automatycznie wykonywane raz na miesiąc przez demon cron. Możesz umieścić własne skrypty w tym katalogu, a zostaną one uruchamiane raz na miesiąc zgodnie z harmonogramem cron.

Oto przykład korzystania z katalogu cron.monthly:

1. Otwórz terminal w systemie Ubuntu.
2. Przejdź do katalogu cron.monthly za pomocą polecenia:

```
cd /etc/cron.monthly
```

3. Utwórz nowy skrypt bash, na przykład my\_monthly\_script.sh, za pomocą swojego ulubionego edytora tekstowego:

```
nano my_monthly_script.sh
```

4. Wpisz treść skryptu, na przykład:

```
#!/bin/bash
```

```
echo "To jest mój skrypt, który wykonuje się raz na miesiąc!"
```



date

# Dodaj inne polecenia, które mają zostać wykonane raz na miesiąc

Skrypt może zawierać dowolne polecenia, które chcesz wykonywać raz na miesiąc.

5. Zapisz i zamknij plik.

6. Upewnij się, że skrypt ma odpowiednie uprawnienia do wykonania:

chmod +x my\_monthly\_script.sh

7. Skrypt my\_monthly\_script.sh zostanie automatycznie wykonany raz na miesiąc przez demon cron, o ile masz zainstalowany i uruchomiony demon cron na swoim systemie.

Należy pamiętać, że skrypty w katalogu cron.monthly muszą mieć odpowiednie uprawnienia do wykonania, czyli muszą być wykonywalne. Ważne jest również, aby treść skryptu była poprawnie napisana i testowana, aby uniknąć nieoczekiwanych efektów w systemie.

cat my\_monthly\_script.sh

(zgłoszenie nr6)

Po sprawdzeniu przez prowadzącego przywróć pierwszą migawkę.

Podsumowanie:

Po wykonaniu wszystkich czynności z powyższej instrukcji przeczytaj ponownie z zrozumieniem cel ogólny i cele szczegółowe, które znajdują się na pierwszej stronie instrukcji. Jeżeli one zostały niezrealizowane to powtarzaj wykonie tej instrukcji w szkole lub/i w domu do momentu zrealizowania.

Z ćwiczenia na temat planowania zadań przy pomocy narzędzi **cron** i **at** oraz korzystania z katalogów **cron.x** można wyciągnąć następujące wnioski:

1. **at jest narzędziem do jednorazowego planowania zadań na określoną godzinę i minutę**, co jest przydatne do jednorazowych, punktualnych zadań.
2. **cron jest narzędziem do planowania zadań cyklicznych**, takich jak wykonywanie czynności co minutę, godzinę, czy w określony dzień tygodnia lub miesiąca. Może być używane dla zadań powtarzających się.
3. **Konfiguracja i zarządzanie zadaniami at i cron są różne**. **at** służy do jednorazowych zadań i wymaga podania konkretnej daty i godziny, podczas gdy **cron** używa specjalnego formatu czasu i daty w pliku crontab.
4. **cron pozwala na definiowanie zadań jako użytkownik oraz jako użytkownik root**. To ważne ze względu na dostęp do różnych systemowych zasobów.

5. **Zarządzanie zaplanowanymi zadaniami jest kluczowe.** **atq** lub **at -l** pozwalają na przeglądanie listy zadań **at** i usuwanie ich. W przypadku **cron** można użyć **crontab -r** do usunięcia całego pliku **crontab**.
  6. **Katalogi cron.x pozwalają na automatyczne wykonywanie skryptów z określoną częstotliwością.** Mogą być przydatne do wykonywania skryptów codziennie (**cron.daily**), co godzinę (**cron.hourly**), raz na tydzień (**cron.weekly**) lub raz na miesiąc (**cron.monthly**).
  7. **Skrypty w katalogach cron.x muszą być wykonywalne.** Upewnij się, że skrypt ma odpowiednie uprawnienia do wykonania (**chmod +x skrypt.sh**).
  8. **Dobrze jest regularnie monitorować logi zadań cron** (**tail -F ~/users.log**) oraz potwierdzać poprawność działań, szczególnie przy planowaniu zadań na konto **root**.
  9. **Przy planowaniu zadań cyklicznych (cron) należy zwracać uwagę na specyfikację czasu i daty,** aby uniknąć nieoczekiwanych wyników i konfliktów z innymi zadankami.
  10. **Dokładność i ostrożność w tworzeniu i zarządzaniu zadaniami są kluczowe,** aby uniknąć ewentualnych problemów i zapewnić, że zadania będą działać zgodnie z oczekiwaniami.
- Ćwiczenie to daje podstawową wiedzę na temat planowania zadań w systemach Unix/Linux za pomocą narzędzi **at** i **cron**, co może być przydatne w administracji systemami i automatyzacji określonych czynności.