System operacyjny Linux

Paweł Rajba

pawel.rajba@continet.pl
http://kursy24.eu/

Zawartość modułu 10

- Usługi w Linuxie, czyli skrypty rc
- Rdzenne usługi systemowe
  - init
  - inet, xinet
  - syslog
  - cron

Skrypty rc

- Zarządzanie usługami (demonami) w systemie
- Skrypty znajdują się w katalogu /etc/init.d
- Każdy skrypt akceptuje parametry start i stop
- Skrypt może też pokazywać informacje o usłudze
- Katalogi /etc/init.d/rcX.d/ (X=0,...6)
- Zawartość katalogów /etc/init.d/rcX.d/
  - pliki K[0-9]{2}*, np. K10sshd
  - pliki S[0-9]{2}*, np. S05network

Skrypty rc

- Tworzenie skryptu rc
  - Skrypt powinien akceptować i interpretować parametry start i stop. Może też akceptować inne parametry, np. restart, status.
  - Polecenia przydatne przy tworzeniu takich skryptów
    - startproc
    - killproc
    - checkproc

Skrypty rc

- startproc – uruchamia proces
  - Składnia
    - startproc [-u user] [-g grupa] [-l logfile] [-q] [-d] [-t sek] program [argumenty]
  - Opcje
    - -u user – określa, kto będzie właścicielem procesu
    - -g grupa – określa, w której grupie będzie proces
    - -l logfile – określa, gdzie będą przekierowywane wszystkie komunikaty (zamiast na ekran)
    - -q – to samo co -l /dev/null

Skrypty rc

- killproc – wyłącza proces
  - Składnia
    - killproc [-SYGNAŁ] [-p pidfile] program
  - Opcje
    - -SYGNAŁ – wysłany zostanie określony sygnał, domyślnie SIGTERM
    - -p pidfile – określa plik z PID procesu
  - Przykłady
    - killproc -TERM /usr/sbin/sendmail (wyłącza wszystkie)
    - killproc -p /var/myrun/lpd.pid -TERM /usr/sbin/lpd

Skrypty rc

- checkproc – sprawdza pid procesu
  - Składnia
    - checkproc [-v] [-p pidfile] program
  - Opcje
    - -v – wersja gadatliwa (wypisuje PID)

Skrypty rc

- Przykład: tworzymy własną usługę
  - Najpierw tworzymy skrypt, który będziemy uruchamiać i zatrzymywać:

```
#!/bin/bash
i=0
while [ 1=1 ]
do
  echo $i
  i=$[$i+1];
  sleep 1;
done
```


Skrypty rc

- Przykład: tworzymy własną usługę c.d.
  - Sprawdzamy działanie poleceń:

```
startproc -l /sciezka/usluga.log /sciezka/usluga.sh
checkproc -v usluga.sh
killproc -TERM usluga.sh
```

  - Tworzymy właściwy skrypt rc

```
#!/bin/bash
case "$1" in
  start)
    echo "Uruchamiamy usluge"
    startproc -l /path/usluga.log /path/usluga.sh ;;
  stop)
    echo "Stopping usluga"
    killproc usluga.sh ;;
esac
```


Skrypty rc

- Przykład: tworzymy własną usługę c.d.
  - Uruchamiamy usługę: demon start
  - Patrzymy na zawartość pliku log:
    tail -f /path/usluga.log
  - Patrzymy na wynik polecenia ps -f
  - Próbujemy uruchomić usługę z innym właścicielem
  - Zatrzymujemy usługę
  - Patrzymy na wynik polecenia ps -f

Proces init

- Pierwszy proces w systemie (PID=1)
- Zarządza poziomem pracy systemu, zwanym też poziomem uruchomieniowym (ang. runlevel)
- Działa przez cały czas działania systemu
- Konfiguracja procesu w pliku /etc/inittab

Proces init

- Poziomy pracy w linux-ie
  - 0 — zatrzymanie systemu,
  - 1 — tryb jednoużytkownikowy (bez sieci),
  - 2 — tryb wieloużytkownikowy (bez NFS),
  - 3 — pełny tryb wieloużytkownikowy,
  - 4 — nieużywane,
  - 5 — tryb 3 + logowanie w systemie graficznym,
  - 6 — przeładowanie systemu,
  - S, s – tryb single (nie potrzebuje pliku inittab)

Proces init

- Plik konfiguracyjny /etc/inittab
  - Każdy wiersz jest postaci
    - id:poziomy:akcja:proces
  - Znaczenie pól
    - id — niepowtarzalna sekwencja 1–4 znaków identyfikująca wpisy,
    - poziomy — poziomy pracy, na których proces powinien zostać wywołany; jeśli wpis jest pusty, oznacza to wszystkie poziomy; jeśli chcemy kilka poziomów na raz, możemy po kolei je wypisać np. 123,
    - akcja — określa warunki, przy których proces zostanie uruchomiony
    - proces — określa program do uruchomienia procesu

Proces init

- Plik konfiguracyjny /etc/inittab
  - Niektóre akcje
    - respawn — jeśli proces zakończy swoje działanie, bedzie uruchamiany ponownie,
    - wait — proces zostanie uruchomiony raz po przejściu na określony poziom pracy, a init poczeka na jego zakończenie
    - once — proces zostanie uruchomiony raz po przejściu określonego poziomu pracy, a init nie będzie czekał na jego zakończenie,
    - boot — proces będzie uruchamiany podczas startu systemu; pole poziomu pracy będzie ignorowane,

Proces init

- Plik konfiguracyjny /etc/inittab
  - Niektóre akcje c.d.
    - bootwait — znaczenie takie jak boot, przy czym init poczeka na jego zakończenie
    - off — nic nie robi,
    - ondemand — proces zostanie uruchomiony, gdy pojawi się poziom na zadanie (poziomy te to a, b, c), natomiast nie następuje zmiana poziomu pracy,
    - initdefault — domyślny poziom dla init; jeśli nie zostanie podany, użytkownik będzie o niego zapytany,
    - sysinit — proces będzie uruchomiony przed wpisami boot i bootwait; pole poziomu pracy będzie ignorowane,

Proces init

- Plik konfiguracyjny /etc/inittab
  - Niektóre akcje c.d.
    - powerwait — proces będzie uruchomiony przy kłopotach z zasilaniem; init poczeka na zakończenie procesu,
    - powerfail — znaczenie takie jak powerwait, przy czym init nie będzie czekał na zakończenie procesu
    - powerfailnow — proces będzie uruchomiony, gdy zasilanie awaryjne jest prawie wyczerpane,
    - powerokwait — proces zostanie uruchomiony, gdy zasilanie zostanie przywrócone,
    - ctrlaltdel — proces zostanie uruchomiony, jeśli zostanie naciśnieta kombinacja klawiszy ctrl-alt-del.

Proces init

- Sterowanie akcjami związanymi z zasilaniem
  - Odpowiednia akcja jest wywoływana po otrzymaniu sygnału SIGPWR i na podstawie zawartości pliku /etc/powerstatus
  - Znaczenie zawartości pliku /etc/powerstatus
    - F(AIL) – brak zasilania i aktywuje się UPS; wykonywane są powerwait i powerfail
    - O(K) – zasilanie zostało przywrócone; wykonywany jest powerokwait
    - L(OW) – brak zasilania i stan baterii jest bliski wyczerpania; wykonywany jest powerfailnow

Proces init

- telinit – program do zmiany poziomu pracy
  - Wysyła sygnał SIGTERM, a po 5 sekundach SIGKILL.
  - Zwykle tego polecenia się jawne nie używa.
  - Składnia
    - telinit [-t sekundy] poziom
  - Opcje
    - -t – określa czas pomiędzy wysłanie sygnałów SIGTERM i SIGKILL; domyślnie 5 sekund

Proces init

- telinit, przykład
  - Do pliku /etc/inittab dopisujemy wiersz
    - `99:a:ondemand:echo "1" > /root/telinit.txt`
  - Wydajemy polecenie
    - `roadrunner:~ # telinit a`

Proces xinet

- Zasada działania
  - inetd
    - wszystkie usługi włączane są przy starcie systemu
    - procesy są uśpione do momentu pojawienia się żądania na odpowiednim porcie
  - xinetd
    - nasłuchuje na wszystkich obsługiwanych portach
    - w momencie pojawienia się żądania włączana jest odpowiednia usługa

Proces xinet

- Ze względu na swoją rolę serwery inetd i xinetd są też nazywane superserwerami
- Serwer xinetd jest nowszą wersją inetd
- Konfiguracja
  - plik /etc/xinetd.conf
  - katalog /etc/xinetd.d

Proces xinet

- Elementy konfiguracji
  - Blok defaults

```
defaults
{
        log_type            = FILE /var/log/xinetd.log
        log_on_success      = HOST EXIT DURATION
        log_on_failure      = HOST ATTEMPT
        only_from           = localhost
        instances           = 30
        cps                 = 50 10
}
```


Proces xinet

- Elementy konfiguracji:
  - Dyrektywa includedir
    ```
    includedir /etc/xinetd.d
    ```
  - Definicja pojedynczej usługi
    ```
    service <nazwa_usługi>
    {
        <atrybut> = <wartość> <wartość> ...
        ...
    }
    ```


Proces xinet

- Rodzaje usług:
  - INTERNAL – usługi obsługiwane bezpośrednio przez xinetd: echo, time, daytime, chargen oraz discard.
  - RPC – usługa typu RPC
  - UNLISTED – usługa nie wymieniona w żadnym z plików /etc/rpc lub /etc/services
  - Pozostałe – wszystkie pozostałe usługi np. ftp, telnet

Proces xinet

- Wybrane atrybuty
  - id
    - pozwala jednoznacznie zidentyfikować usługę
    - domyślnie id usługi jest takie samo jak jej nazwa
    - przydatny, jeśli dana usługa używa kilku różnych protokołów i wtedy będzie opisana za pomocą kilku wpisów
  - type
    - dozwolona jest dowolna kombinacja następujących wartości:
      - RPC (usługa RPC)
      - INTERNAL (usługa obsługiwana przez xinetd)
      - UNLISTED (usługa nie jest wpisana w żadnym z plików /etc/rpc i /etc/services)

Proces xinet

- Niektóre atrybuty c.d.
  - disable
    - określa, czy usługa jest wyłączona
    - wartości yes (wyłączona) lub no (włączona),
  - socket_type
    - dostępne wartości to
      - stream — usługa oparta na strumieniach (np. np. protokole TCP),
      - dgram — usługa oparta na datagramach (np. na protokole UDP),
      - raw — usługa operuje bezpośrednio protokole IP,

Proces xinet

- Niektóre atrybuty c.d.
  - protocol
    - określa protokół wykorzystywany przez usługę
    - protokół powinien być zdefiniowany w pliku /etc/protocols,
  - wait
    - wartości yes lub no
      - jeśli zostanie ustawione na yes, to w danym momencie będzie obsługiwane tylko jedno połączenie *(single-threaded)*
      - jeśli zostanie ustawione na no, to obsługiwanych może być kilka połączeń równocześnie *(multi-threaded)*,

Proces xinet

- Niektóre atrybuty c.d.
  - user
    - określa UID dla procesu serwera; możemy podać zarówno nazwę użytkownika (zalecane) lub UID
    - jeśli podamy nazwę, musi być wpisana w pliku /etc/passwd,
  - group
    - określa GID dla procesu serwera; możemy podać zarówno nazwę użytkownika (zalecane) lub GID
    - jeśli podamy nazwę, musi być wpisana w pliku /etc/group,

Proces xinet

- Niektóre atrybuty c.d.
  - instances — określa maksymalną obsługiwaną liczbę połączeń (uruchomionych serwerów); domyślnie wartość UNLIMITED (ma sens, jeśli atrybut wait jest ustawiony na no); ma spore znaczenie w obronie przed atakami DoS
  - server — nazwa programu do obsługi połaczenia,
  - server_args — argumenty, które należy przekazać do programu określonego w server,

Proces xinet

- Niektóre atrybuty c.d.
  - only_from — określa adresy, z których usługa jest dostępna; możemy podać:
    - adres ip lub podsieć np. 212.102.149.0,
    - nazwę komputera np. foo.bar.net
  - no_access — określa adresy, dla których usługa jest zablokowana,
    - wartość podajemy jak w only_from

Proces xinet

- Dostęp do usługi: only_from i no_access
  - Jeśli żaden z atrybutów nie jest ustawiony, dostęp do usługi mają wszystkie hosty
  - Jeśli ustawimy only_from, dostęp do usługi mają tylko wymienione na liście hosty,
  - Jeśli ustawimy no_access, blokuje dostęp tylko dla hostów określonych przez ten atrybut
  - Jeśli ustawimy only_from i no_access, wyższy priorytet ma wpis dokładniej precyzujący hosta (np. 192.168. i 192.168.0.1)

Proces xinet

- Niektóre atrybuty c.d.
  - access_times – określa godziny i minuty, w których usługa jest dostępna
    - format hh:mm-hh-mm
    - przykład:
      access_times = 8:00-16:30 21:00-23:00

Proces xinet

- Niektóre atrybuty c.d.
  - log_type — określa, gdzie trafia komunikaty generowane przez usługę
    - SYSLOG rodzaj [priorytet] — określa, że komunikaty zostaną obsłużone przez demon syslog; dostępne rodzaje to: daemon, auth, authpriv, user, mail, lpr, news, uucp, ftp, local0-7, a dostępne priorytety to: emerg, alert, crit, err, warning, notice, info, debug; jesli nie podamy priorytetu, to domyślnie bedzie info,
    - FILE plik [miękki_limit [twardy_limit]] — określa, że komunikaty trafią do pliku plik; możemy podać dwa limity: miękki i twardy. Przy przekroczeniu miękkiego będzie raz wygenerowany komunikat informujący o tym fakcie, natomiast przy osiągnięciu rozmiaru zdefiniowanego przez twardy limit, zapisywanie komunikatów zostanie wstrzymane,

Proces xinet

- Niektóre atrybuty c.d.
  - log_on_success — określa, które informacje maja być zapisane w przypadku udanego połączenia
    - PID — id procesu,
    - HOST — adres zdalnego hosta,
    - USERID — nazwę zdalnego użytkownika,
    - EXIT — kod wyjścia lub sygnał zakończenia procesu,
    - DURATION — czas trwania procesu,
    - TRAFFIC — ilość wymienionych danych,

Proces xinet

- Niektóre atrybuty c.d.
  - log_on_failure — określa, które informacje maja być zapisane w przypadku, gdy serwer nie może być uruchomiony; dostępne są: HOST, USERID i ATTEMPT (wynika z pozostałych)
  - port — określa port dla usługi; jeśli usługa jest wymieniona w pliku /etc/services, porty powinny się zgadzać,

Proces xinet

- Niektóre atrybuty c.d.
  - bind, interface — jako parametr bierze adres IP i pozwala przydzielić usługę do konkretnego interface-u, czyli, przykładowo, serwer telnetu może być dostępny tylko z sieci wewnętrznej,
  - cps — bierze dwie liczby: pierwsza określa maksymalna liczbę połączeń na sekundę, druga precyzuje, na ile sekund usługa będzie wyłączona w przypadku przekroczenia liczby połączeń określonej przez pierwsza liczbę.

Proces xinet

- Do określenia usługi wystarczy określić:
  - socket_type
  - user
  - server
  - wait

Proces xinet

- Przykład
  - Włączamy i testujemy usługę echo
  - Sprawdzamy stan poleceniem
    - netstat -at
  - Łączymy się poleceniem
    - telnet localhost 7

Proces syslog

- Dawniej systemem do rejestracji był syslog
  - w części dystrybucji dalej jest
- Obecnie do rejestracji komunikatów w systemie używany jest syslog-ng (wersje 1.6 i 2.0)
  - Strona domowa: http://www.balabit.com/products/syslog_ng/
- Komunikaty są określone przez dwa parametry
  - pochodzenie komunikatu (facility)
  - priorytet, czyli znaczenie komunikatu (level)

Na podstawie dokumentacji i artykułu http://pl.docs.pld-linux.org/uslugi_syslog-ng.html

Proces syslog

- Pochodzenia komunikatów
  - auth, authpriv — wiadomości uwierzytelniania,
  - cron — wiadomości generowane przez crona i at,
  - deamon — wiadomości od demonów usług,
  - ftp — wiadomości od demona ftp,
  - kern — wiadomości od jądra systemu,
  - lpr — wiadomości od demona drukowania,
  - mail — wiadomości od systemu pocztowego (łącznie z rejestracja przyjscia maili),

Proces syslog

- Rodzaje komunikatów c.d.
  - news — wiadomości od systemu NNTP,
  - syslog — wiadomości od systemu syslog,
  - user — wiadomości od programów użytkowników,
  - uucp — wiadomości od podsystemu UUCP (Unix to Unix CoPy),
  - local0-local7 — zarezerwowane do zastosowań lokalnych; możliwe do dowolnego zastosowania przez administratora.

Proces syslog

- Priorytety
  - debug — informacje do usuwania usterek,
  - info — wiadomość informacyjna,
  - notice — ważna zdarzenie (niekoniecznie złe),
  - warning — ostrzeżenie,
  - err — błąd,
  - crit — zdarzenie krytyczne,
  - alert — należy podjąć niezwłoczne działanie,
  - emerg — sytuacja awaryjna, system uszkodzony.

Proces syslog

- Konfiguracja ogólna
  - plik /etc/sysconfig/syslog
- Konfiguracja logowania
  - plik /etc/syslog-ng/syslog-ng.conf.in
- Dodatkowe informacje
  - man syslog-ng.conf

Proces syslog

- Konfiguracja syslog-ng
  - Mamy trzy rodzaje obiektów
    - Źródła - wskazują miejsca pochodzenia komunikatów
    - Filtry - pozwalają selekcjonować dane
    - Cele - wskazują sposób i miejsce magazynowania logów
  - Konfiguracja polega na ich zdefiniowaniu i połączeniu w reguły

Proces syslog

- Konfiguracja syslog-ng
  - Definiowanie źródeł
    - source nazwa { źródło(opje) }
  - Wybrane źródła
    - internal – komunikaty demona syslog-ng
    - tcp – komunikaty od innych komputerów w sieci (TCP)
    - udp – komunikaty od innych komputerów w sieci (UDP)
  - Przykłady
    - source src { internal(); }; source udp { udp(); };
    - source tcp { tcp(ip(192.168.0.10) port(1514) max-connections(20)); };

Proces syslog

- Konfiguracja syslog-ng
  - Definiowanie filtrów
    - filter nazwa { rodzaj(wartość); };
    - Możemy używać operatów logicznych and, or i not
  - Wybrane filtry
    - facility – pochodzenie zdarzenia: auth, cron, mail, itd.
    - level – priorytet: emerg, alert, crit, itd.
    - host – filtrowane po nazwie hosta
    - program – filtrowane po nazwie programu
    - match – filtrowanie po całym komunikacie
      - host, level, match – można używać wyrażeń regularnych

Proces syslog

- Konfiguracja syslog-ng
  - Przykłady filtrów
    - filter f_important { level(err, crit, alert) };
    - filter f_daemon { facility(daemon); };
    - filter f_mainhost { host("pimpus"); };
    - filter f_su_sudo { program("^su|sudo$"); };
    - filter f_iptables { facility(kern) and match("IN=") and match("OUT="); };

Proces syslog

- Konfiguracja syslog-ng
  - Definiowanie celów
    - destination nazwa { cel(miejsce); }
  - Wybrane cele
    - file - plik tekstowy / urządzenie znakowe (/dev/)
    - usertty - ekran terminala wskazanego użytkownika
    - tcp - komunikaty do loghosta (TCP)
    - udp - komunikaty do loghosta (UDP)

Proces syslog

- Konfiguracja syslog-ng
  - Przykłady celów
    - destination all { file(“/var/log/allmessages”) }
    - destination auth { file("/var/log/auth"); };
    - destination console { file("/dev/tty12"); };
    - destination root { usertty("root"); };
    - destination loghost { udp("192.168.0.250"); };

Proces syslog

- Konfiguracja syslog-ng
  - Tworzenie reguł
    - log { source(źrodlo); filter(f1); ...; filter(fN); destination(cel) }
      - gdzie N>=0 (czyli możemy w ogóle nie podawać filtrów)
  - Przykłady
    - log { source(src); destination(fileall); };
    - log { source(src); filter(f_important); destination(loghost); };

Proces syslog

- Program logger
  - Służy do tworzenia komunikatów
  - Składnia
    - logger [-t tag] [-p facility.level] komunikat
- Przeglądamy pliki
  - /etc/sysconfig/syslog
  - /etc/syslog-ng/syslog-ng.conf.in
  - /var/log/messages
  - /var/log/*

Proces syslog

- Narzędzia do analizy logów
  - Analog
    http://www.statslab.cam.ac.uk/~sret1/analog/
  - LogSurfer
    http://www.cert.dfn.de/eng/logsurf/
  - Swatch
    ftp://sierra.stanford.edu/swatch/
  - Webalizer
    http://www.mrunix.net/webalizer/
  - XLogMaster
    http://www.gnu.org/software/xlogmaster/

Proces cron

- Służy do cyklicznego lub jednorazowego wykonania polecenia
- Proces crond sprawdza pliki konfiguracyjne co minutę i ustala, czy ma coś w danym momencie do wykonania

Proces cron

- Konfiguracja użytkownika
  - plik /var/spool/cron/tabs/nazwa_użytkownika
- Konfiguracja systemowa
  - plik /etc/crontab
  - katalogu /etc/cron.d
    - rozszerzenie pliku /etc/crontab
  - katalogi
    - /etc/cron.daily
    - /etc/cron.hourly
    - /etc/cron.monthly
    - /etc/cron.weekly

Proces cron

- Konfiguracja pliku użytkownika
  - Każdy wiersz pliku konfiguracyjnego składa się z następujących pól:
    - minuta (0–59),
    - godzina (0–23),
    - dzień (1–31),
    - miesiąc (1–12),
    - dzień tygodnia (0–7; 0 lub 7 to niedziela),
    - komenda do wykonania.
- Konfiguracja pliku systemowego
  - Dodatkowo jest kolumna określająca kto będzie właścicielem uruchomionego procesu

Proces cron

- Konfiguracja pliku użytkownika c.d.
  - Wartości pól:
    - Możemy użyć znaku *, który dopasowuje się do wszystkich wartości (min-max),
    - Możemy podawać listę wartości, oddzielając je przecinkami np. 12,13,14,
    - Możemy również definiować zakresy wraz z krokiem np. 12-16/2 (12,14,16).

Proces cron

- crontab – zarządzanie plikiem konfiguracyjnym użytkownika
  - Składnia
    - crontab [-u użytkownik] {-e | -l | -r }
  - Opcje
    - -u użytkownik — określa użytkownika,
    - -e — wejście do edycji pliku crontab,
    - -l — wyświetla bieżącą konfiguracje,
    - -r — usuwa zawartość pliku crontab.

Proces cron

- Pliki określające, kto może korzystać z polecenia crontab
  - /etc/cron.allow
  - /etc/cron.deny
- Co musi być, żebyśmy mieli dostęp do polecenia crontab:
  - jeśli plik allow istnieje, musimy być w nim wymienieni
  - jeśli plik deny istnieje, nie możemy być w nim wymienieni

Proces cron

- Rozważania...
  - Jeśli istnieją oba pliki, i w obu jesteśmy wpisani, będzie mieć dostęp
  - Jeśli istnieje pusty plik allow, nikt nie ma dostępu

Proces cron

- Przykłady wpisów konfiguracji:
  - ```
    # kopia.sh będzie uruchamiane codziennie 10 minut po północy
    10 0 * * * $HOME/bin/kopia.sh
    ```
  - ```
    # o 10.45 dziesiątego każdego miesiąca będzie uruchamiane
    # generowanie raportu miesiecznego
    45 9 10 * * $HOME/bin/raport_miesieczny.sh
    ```
  - ```
    # wysyłanie maila do zajączka w dni robocze o 7 rano
    0 7 * * 1-5 echo "Wstawaj!" | mail -s "Pobudka" zajączek
    ```
  - ```
    # dotykanie pliku codziennie o nieparzystych
    # godzinach przez cała dobę
    0 1-23/2 * * * touch /var/mail/pawel
    ```
  - ```
    # wysyłanie maila co niedziele o 11.30
    30 11 * * sun echo -s "Juz czas!" | mail -s "Odjazd" zenon
    ```


    ```