Kurs administrowania systemem Linux 2019

Lista zadań na pracownię nr 8

Na zajęcia 15 kwietnia 2019

Zadanie 1 (1 pkt). Napisz skrypt, który raz na minutę wypisuje komunikaty do logów systemowych (zob. `logger(1)`) oraz na standardowe wyjście. Uruchom go w tle i zakończ pracę w konsoli. Zobacz, że skrypt zostanie zakończony poprzez wysłanie sygnału `SIGHUP` w chwili zamknięcia sesji. Uruchom go ponownie za pomocą programu `nohup(1)` i zobacz, że teraz będzie działał nadal po zamknięciu sesji. Gdzie są zapisywane komunikaty, które skrypt wypisuje na standardowe wyjście? Jak teraz zakończyć jego działanie? Zapoznaj się z poleceniem `trap` powłoki. Zmień skrypt tak, żeby teraz nie korzystał ze standardowego strumienia wyjściowego, ignorował sygnał `SIGHUP`, a po otrzymaniu sygnału `SIGUSR1` wypisywał do logów komunikat o otrzymaniu tego sygnału i kończył działanie.

Zadanie 2 (1 pkt). Napisz skrypt, którego zadaniem jest bezcelowe zużywanie mocy obliczeniowej procesora, np. poprzez wykonywanie w pętli jakichś obliczeń. Skrypt powinien uruchamiać podaną podczas wywołania liczbę podprocesów tak, aby dało się obciążyć wszystkie rdzenie procesora. Uruchamiaj go z różnymi wartościami *niceness* (użyj polecenia `nice(1)`) lub na bieżąco zmieniaj *niceness* działającego procesu (użyj polecenia `renice(1)`) i zobacz, jaki ma to wpływ na działanie systemu.

Zadanie 3 (1 pkt). Napisz w C prosty program `mystat`, który periodycznie (domyślnie raz na sekundę) odczytuje z pseudopliku `/proc/stat` obciążenie procesora (zob. `proc(5)`) i co jakiś czas (domyślnie co 1 minutę) zapisuje uśrednione/maksymalne/minimalne wartości obciążenia do ustalonego pliku (domyślnie `/var/log/mystat.log`). Podane domyślne wartości powinny być możliwe do zmiany za pomocą opcji `-p` lub `--period`, `-i` lub `--interval` oraz `-f` lub `--logfile` (użyj `getopt_long(3)`).

Zadanie 4 (1 pkt). Użyj programu MRTG (`https://oss.oetiker.ch/mrtg/`) aby na podstawie zebranych przez program `mystat` logów generować czytelne wykresy obciążenia systemu.

Zadanie 5 (1 pkt). Na podstawie programu `mystat` napisz jego zdemonizowaną wersję `mystatd`, zob. `daemon(7)`. Poza funkcjonalnościami odziedziczonymi z programu `mystat` program powinien także przechwytywać sygnał `SIGHUP` i w razie jego otrzymania zamykać i powtórnie otwierać swój plik z logami (co ułatwi rotowanie logów).

Zadanie 6 (3 pkt). Przygotuj odpowiedni skrypt SysV Init `/etc/init.d/mystat`. Skrypt powinien być zgodny z przyjętymi konwencjami (zob. `/etc/init.d/skeleton`), zawierać odpowiednie nagłówki *LSB dependency boot headers* (zob. `insserv(8)`), obsługiwać *pid file* w katalogu `/run`, używać biblioteki `/lib/lsb/init-functions` i programu `start-stop-daemon(8)`. Konfigurację demona (w tym argumenty dla opcji `--period`, `--interval` i `--logfile`) powinny być czytane z pliku konfiguracyjnego `mystat` umieszczonego w katalogu `/etc/`. Poza standardowymi akcjami `start`, `stop`, `restart`, `reload` i `status` skrypt powinien dodatkowo obsługiwać opcję `rotate`, która powoduje wysłanie do demona `mystat` sygnału `SIGHUP`. Za pomocą programu `update-rc.d(8)` zainstaluj serwis `mystat` w SysV Init tak, by był uruchamiany podczas startu systemu. *Uwaga:* to zadanie można wykonać zarówno w systemach korzystających z SysV Init, jak i SystemD. Jeśli korzystasz z OpenRC lub BSD Init, to konieczne będą niewielkie modyfikacje.

Zadanie 7 (1 pkt). Dopisz do `/etc/logrotate.d/` obsługę logów demona `mystat`. System powinien rotować logi raz na dobę i przechowywać logi z ostatniego tygodnia. Pamiętaj o skonfigurowaniu wysyłania sygnału do demona w celu zamknięcia i ponownego otwarcia pliku z logami (zob. `invoke-rc.d(8)`).

Zadanie 8 (1 pkt). Zmodularyzuj swój plik `~/.bashrc`. Załóż w tym celu katalog `~/.bashrc.d/`. Różne fragmenty konfiguracji startowej (konfiguracja zmiennej `PS1`, aliasy, itd.) umieść w osobnych plikach o odpowiednich nazwach i wczytuj je z głównego pliku.

Zadanie 9 (1 pkt). Zapoznaj się z poleceniem `run-parts(8)`. Przygotuj krótkie omówienie jego działania. Dlaczego nie mogliśmy z niego skorzystać w poprzednim zadaniu? Użyj go, aby sprawdzić, jakie skrypty są wykonywane przez demona `crond` raz na dobę.

Zadanie 10 (1 pkt). *Zrób sobie zombie.* Napisz program, który utworzy proces potomny, a następnie nie obsłuży jego zakończenia. Proces potomny powinien się zakończyć. Zobacz, że pozostanie na liście procesów jako *zombie* do czasu, aż zakończy się oryginalny program. Zakończenie procesu potomnego zostanie wówczas obsłużone przez program `init(1)`.

Zadanie 11 (1 pkt). Zapoznaj się z dokumentacją polecenia `top(1)` i przygotuj się do zaprezentowania tego programu podczas zajęć. Przedstaw podstawowe skróty klawiszowe. Pokaż, jak można uzyskiwać różne informacje o zbiorze działających procesów przełączając sposób wyświetlania, sortowania itd. Dostosuj kolorystykę wyświetlanych informacji zgodnie z własnymi upodobaniami. Przygotuj odpowiedni plik `.toprc`.

Zadanie 12 (1 pkt). Zapoznaj się z programami `ps(1)` i `pstree(1)`. Przygotuj krótką demonstrację ich użycia.