Monitorowanie systemu GNU/Linux przy pomocy Syslog

Grzegorz Jacek Nalepa

17.01.2000, Kraków, *Revision* : 1.7

Streszczenie

Artykuł opisuje system Syslog będący podstawowym narzędziem służącym do monitorowania systemu GNU/Linux. Opisana jest architektura Sysloga i jego konfigurowanie. Zaprezentowane są sposoby rotowania plików rejestrowych, które tworzy Syslog. Na zakończenie wspomniane są pakiety alternatywne do Syslog.

Spis treści

[1]Tekst ukazał się w: *Magazynie NetForum*, nr 3/2000, wydawanym przez Lupus.

[2]Kontakt z autorem: mail:gjn@agh.edu.pl

[3]Tytuł angielski: *Monitoring a GNU/Linux System with Syslog*

[4]Tekst jest rozpowszechniany na zasadach licencji *GNU Free Documentation License*, której pełny tekst można znaleźć pod adresem: http://www.gnu.org/copyleft/fdl.html

1. Wstęp

Bezpieczeństwo systemu komputerowego zależy nie tylko od tego, jak jest zabezpieczony przed intruzami. Bardzo ważne dla bezpieczeństwa jest to, czy administrator ma dostępne informacje o funkcjonowaniu systemu. Dobre narzędzie do monitorowania pracy systemu jest niezbędne do jego użytkowania i zabezpieczania.

W systemie GNU/Linux, podobnie jak w systemach Unix, stosuje się pakiet Syslog, który umożliwia monitorowanie systemu. Jest to program o bardzo dużych możliwościach, lecz wymagający, podobnie jak inne narzędzia systemowe, odpowiedniej konfiguracji.

2. Jak działa Syslog

Program Syslog jest jednym z najważniejszych narzędzi systemowych. Umożliwia rejestrowanie zdarzeń zachodzących w systemie przy pomocy zcentralizowanego mechanizmu. Pozwala na rejestrowanie informacji pochodzących z trzech źródeł: zgłoszeń przekazywanych przez bibliotekę systemową, oraz informacji pochodzących od jądra systemu.

2.1. Architektura

Przyjmowaniem informacji zgłaszanych przez bibliotekę systemową zajmuje się demon `syslogd`. Aplikacje, najczęściej różne demony, przekazują informacje poprzez standardową funkcję `syslog(3)`. Są one następnie przekazywane do demona `syslogd`, który w zależności od ich typu i priorytetu, zapisuje je w odpowiednich plikach, tak zwanych *plikach rejestrowych* (ang. *log files*). Każdy wpis do pliku rejestrowego dokonywany przez `syslogd` zawiera informacje dotyczące czasu jego powstania, maszyny i procesu.

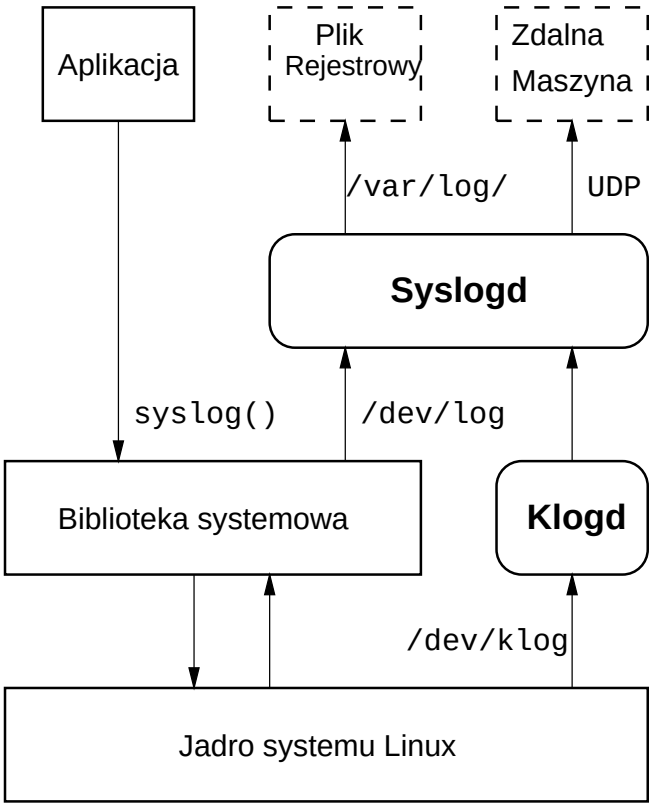
Raporty od jądra systemu Linux zbiera demon `klogd`. Ponieważ mają one inną postać niż te, które przekazuje biblioteka systemowa, zostają poddane obróbce przed przesłaniem ich do `syslogd`. Ta obróbka polega na opatrzeniu ich informacjami o typie i priorytecie wiadomości zgodnymi z `syslogd`.

Działanie opisanego systemu jest przedstawione na Rysunku 1. Jak widać Syslog ma jeszcze jeden interfejs umożliwiający wymianę zbieranych informacji. Obsługuje mianowicie gniazda UDP umożliwiające przesyłanie wiadomości pomiędzy różnymi maszynami w sieci. O wykorzystaniu tego mechanizmu będzie jeszcze mowa dalej.

2.2. Współpraca z innymi programami

Na wspomnianym rysunku widać, że podstawowy mechanizm generowania raportów zapewnia aplikacjom biblioteka systemowa. Udostępnia one trzy funkcje: `openlog()`, `syslog()` i `closelog()`, które umożliwiają odpowiednio: otwarcie komunikacji (stworzenie deskryptora), przesłanie wiadomości i zamknięcie komunikacji. Z tych funkcji korzystają demony i aplikacje pracujące w systemie.

Warto zauważyć, iż możliwe jest przesyłanie wiadomości do Syslog również z poziomu skryptów, na przykład skryptów w sh. Wygodnie jest do tego wykorzystać program `logger`, który wysyła informacje poprzez syslogd. Natomiast w skryptach napisanych w języku Perl można użyć pakietu `Sys::Syslog`, który jest interfejsem do opisanych powyżej wywołań funkcji systemowych.



Rysunek 1: Architektura Syslog

3. Konfigurowanie Syslogd

Syslog by spełniać dobrze swoje funkcje powinien być uruchamiany jako jeden z pierwszych demonów. Uruchamia się go przeważnie przez wywołanie ze skryptów startowych podczas uruchamiania systemu. W dystrybucjach Debian/GNU i RedHat skrypt nazywa się po prostu `syslogd` i znajduje się w katalogu `/etc/init.d` lub `/etc/rc.d/init.d`. Skrypt, oprócz standardowych opcji `start` i `stop` przyjmuje również `restart` i `reload`, które powodują odpowiednio zatrzymanie i ponowne uruchomienie Syslog, oraz powtórne przeczytanie pliku konfiguracyjnego. Skrypt zatrzymuje i uruchamia demon `syslogd` oraz `klogd`.

4. Składnia pliku konfiguracyjnego

Cała konfiguracja Syslog znajduje się w pliku `/etc/syslog.conf`, opisany w (`syslog.conf(5)`). Jest to plik tekstowy, zawierający reguły według których Syslog sortuje informacje i zapisuje je do różnych plików rejestrowych. Składnia pliku jest następująca: każda linijka jest osobną regułą, na którą składają się dwa pola: selektor (ang. *selector*) i działanie (ang. *action*).

Pole „selektor” ma dwie części funkcja (ang. *facility*) i priorytet (ang. *priority*) rozdzielone kropką i określa rodzaj rejestrowanego komunikatu, gdzie:

- „funkcja” wskazuje na źródło komunikatu i może mieć następujące wartości: auth, authpriv, cron, daemon, kern, lpr, mail, mark, news, syslog, user, uucp, local0–local7.
- „priorytet” określa jego stopień ważności (od najniższego): debug, info, notice, warning, err, crit, alert, emerg.

Dodatkowo w polu „selektor” można stosować następujące znaki:

- przecinek (`,`) – grupuje kilka typów „funkcji", którym odpowiada to samo „działanie", w tym przypadku „priorytet" jest ignorowany,
- średnik (`;`) – pozwala na grupowanie kilku selektorów, którym odpowiada to samo działanie.
- gwiazdka (`*`) – może zastępować pola „funkcja" oraz „priorytet" i oznacza „dla każdej wartości tego pola".

Trzeba pamiętać, że „priorytet" oznacza wszystkie komunikaty o priorytecie nie niższym niż podany, na przykład priorytet `err` oznacza komunikaty o priorytecie od `err` do `emerg`. Możliwe jest jednak podawanie innego rodzaju zakresów priorytetów, wykorzystując znak równości (=) i wykrzyknika (!). Oznaczają one odpowiednio komunikaty o dokładnie podanym priorytecie i zanegowanie podanego przedziału priorytetów. Podanie zamiast priorytetu słowa `none` oznacza, że żadne komunikaty z podanego źródła nie będą uwzględniane. Wykorzystanie tych operatorów jest pokazane dalej.

Pole „działanie" wskazuje co ma się dziać z komunikatami określonymi przez selektor reguły:

- nazwa pliku – komunikaty są dopisywane do podanego pliku, poprzedzenie nazwy znakiem minus powoduje wywołanie `sync()` po dopisaniu każdego komunikatu,
- nazwa terminala – umożliwia przesyłanie komunikatów na wybrane terminale,
- |nazwa potoku (ang. *named pipe*) – komunikaty są przesyłane do kolejki FIFO (na przykład stworzonej przy pomocy `mkfifo`), skąd mogą być odbierane przez inny program,
- @nazwa maszyny – Syslog będzie przesyłał komunikaty do innej maszyny w sieci (na port 514/UDP),
- nazwa użytkownika – komunikaty będą wyświetlane na terminalu podanego użytkownika, oczywiście jeżeli jest zalogowany. Można podać listę nazw użytkowników rozdzielonych przecinkami, lub podać gwiazdkę (`*`) na oznaczenie wszystkich zalogowanych użytkowników.

Linia zaczynająca się od znaku `#` jest traktowana jako komentarz i ignorowana.

Znając składnię pliku konfiguracyjnego można podać przykłady konfiguracji Syslogd odpowiadające potrzebom konkretnego systemu. Pisząc, lub modyfikując, plik konfiguracyjny dla Syslog po raz pierwszy należy zadbać o to by wszystkie kategorie komunikatów były rejestrowany. Doświadczony administrator może z czasem świadomie zrezygnować z rejestrowania pewnych kategorii. Nie jest to jednak zalecane i w czasach gdy twarde dyski są nie są drogie warto rejestrować wszystko, a co najwyżej częściej usuwać starsze fragmenty mniej potrzebnych plików rejestrowych.

4.1. Organizacja plików rejestrowych

Tradycyjnie pliki rejestrowe przechowuje się w katalogu `/var/log`. Dla większości podstawowych źródeł komunikatów warto przeznaczyć osobne pliki, których nazwa odpowiada funkcji Syslog:

```
syslog.*        -/var/log/syslog
cron.*           /var/log/cron.log
daemon.*        -/var/log/daemon.log
lpr.*            /var/log/lpr.log
```

```
user.*          /var/log/user.log
uucp.*          /var/log/uucp.log
local.*         /var/log/local.log
```

Informacje dotyczące bezpieczeństwa i autoryzacji warto umieścić w osobnym podkatalogu.

```
auth,authpriv.*   -/var/log/security/auth.log
```

Po każdym wpisie do pliku `auth.log` będzie wywoływana funkcja `sync()`, aby upewnić się, że dane znalazły się natychmiast na dysku.

Jeżeli na maszynie pracuje serwer pocztowy komunikaty pochodzące od niego można podzielić na kilka grup.

```
mail.info          /var/log/mail/mail.info
mail.warn          /var/log/mail/mail.warn
mail.err          -/var/log/mail/mail.err
```

Poniższy przykład pokazuje jak można dokładnie rozdzielić grupy komunikatów w zależności od priorytetu.

```
kern.*               /var/log/kernel/kern.all
kern.=debug          /var/log/kernel/kern.debug
kern.info;kern.!err  /var/log/kernel/kern.info-warning
kern.err             /var/log/kernel/kern.err-emerg
```

Wszystkie informacje od jądra są przekazywane do pliku `kern.all`. Kolejne reguły wysyłają do określonych plików odpowiednio: te i tylko te, które mają priorytet `debug`, komunikaty o priorytecie od `info` do `warning`, oraz te o priorytecie od `err` do `emerg`.

Często spotykanym rozwiązaniem jest również zapisywanie osobno wszystkich wiadomości o priorytecie `debug`. W poniższym przykładzie wykluczone są komunikaty pochodzące z `auth`, `authpriv`, `mail` i `news`.

```
*.debug;\
auth,authpriv.none\
news.none;mail.none   -/var/log/debug
```

Czasami, dla upewnienia się, że rzeczywiście wszystkie kategorie komunikatów są rejestrowane, można je przekazywać do jednego pliku. Należy jednak pamiętać, że w trakcie pracy systemu jego długość będzie bardzo szybko rosła.

```
*.*             /var/log/all
```

Dla administratora często pracującego przy konsoli maszyny może być przydatne bezpośrednie wyświetlanie komunikatów na terminalu, lub przy pomocy polecenia `xconsole`.

```
*.warn       /dev/tty9
*.info       |/dev/xconsole
```

Informacje o zakłóceniu pracy systemu mogą być na tyle poważne, że powinni je natychmiast otrzymać wszyscy zalogowani użytkownicy systemu.

```
*.emerg                         *
```

Z punktu widzenia bezpieczeństwa warto przechowywać komunikaty Syslog poza dyskiem maszyny na której pracuje Syslog, aby nie mogły zostać na skutek awarii lub włamania do systemu. Można do tego wykorzystać zdalną maszynę, na której pracuje Syslog, lub w praktyce każdy inny program nasłuchujący na porcie 514/UDP, na który Syslog przesyła komunikaty.

```
*.warn                  @bezpieczna.maszyna.net
```

Na koniec, warto pamiętać, że jeżeli ma się dostępną nieużywaną drukarkę, najlepiej taką która nie drukuje na pojedynczych kartkach papieru (na przykład igłową), to można ją wykorzystać do drukowania komunikatów Syslog, przesyłając je bezpośrednio na port drukarki (na przykład `/dev/lp0`). Jeżeli dojdzie do awarii systemu lub włamania, raz wydrukowane informacje nie przepadną tak szybko jak te, które znajdują się na dysku.

5. Rotacja plików rejestrowych z wykorzystaniem Cron

Pliki rejestrowe generowane przez Syslog zawierają wiele informacji o pracy systemu, które są ważne dla administratora. Jednak wiele z tych informacji jest potrzebnych jedynie przez określony czas. Z upływem czasu ilość informacji zbieranych przez Syslog jest tak duża, że może nawet zająć całe miejsce w systemie plików. Aby temu zapobiec warto nie tyle wyłączyć rejestrowanie pewnych, pozornie nieistotnych zdarzeń, ile zadbać o okresowe usuwanie starszych informacji. Takie porządkowanie plików rejestrowych nazywa się często rotacją (ang. *rotation*).

Przez rotację plików rejestrowych rozumie się cykliczne usuwanie starszych fragmentów plików. W większości dystrybucji GNU/Linux istnieją dodatkowe narzędzia ułatwiające rotację logów. W dystrybucji Debian/GNU 2.x z pakietem Sysklog współpracują narzędzia `savelog`. W RedHat 5.x do rotacji logów używa się programu `logrotate`.

Dla każdego pliku (`plik`) poddawanego rotacji tworzy się pliki o nazwach: `plik.0 plik.1 ...plik.n-1 plik.n` . Limit ilości plików (liczba n) podaje się programowi służącemu do rotacji logów. Rotację plików rejestrowe można przedstawić w postaci prostego algorytmu:

1. nazwa pliku o najwyższym numerze, `plik.k`, jest zmieniana na `plik.k+1` jeżeli $k < n$, w przeciwnym przypadku plik jest usuwany,
2. dla wszystkich plików o numerach $0 < j < n$ zmienia się nazwy z `plik.j` na `plik.j+1`,
3. aktualny `plik` jest przenoszony do `plik.0`,
4. tworzony jest nowy `plik` o długości 0 bajtów,
5. zmieniane są odpowiednio prawa dostępu do `plik plik.0`.

Ponieważ rotacji powinno się dokonywać systematycznie, w tym celu wykorzystuje się system Cron, umożliwiający cykliczne wykonywanie dowolnych poleceń. Jak wiadomo, konfigurację Cron przeprowadza się przy pomocy polecenia `crontab`, które tworzy lub modyfikuje plik `crontab` (znajdujący się w katalogu `/var/spool/cron`) należący do wywołującego je użytkownika. Oprócz tego, system Cron korzysta z głównego pliku konfiguracyjnego – `/etc/crontab`.

W dystrybucjach systemu GNU/Linux cron jest najczęściej skonfigurowany w ten sposób, że plik `/etc/crontab` zawiera wpisy cyklicznie uruchamiające przy pomocy polecenia `run-parts` skrypty z katalogów: `/etc/cron.daily` (raz dziennie), `/etc/cron.weekly` (raz na tydzień) i `/etc/cron.monthly` (raz na miesiąc).

Rotację plików rejestrowych Syslog najczęściej wykonuje się więc przy pomocy skryptów umieszczonych we wspomnianych powyżej katalogach. Przeważnie wykonuje się ją codziennie przez skrypt znajdujący się w katalogu `/etc/cron.daily`. Można się zdecydować na dokonywanie rotacji niektórych plików w cyklu tygodniowym. Planując cykl rotacji plików rejestrowych należy wziąć pod uwagę jak często każdy plik rejestrowy powinien być poddawany rotacji i ile starszych fragmentów pliku (z ilu cykli) powinno być przechowywanych.

Jeżeli używa się narzędzia operującego na pojedynczym pliku, takiego jak `savelog` należy napisać prosty skrypt uruchamiający to narzędzie dla wszystkich, lub wybranych plików rejestrowych. Przykład takiego skryptu jest następujący:

```
cd /var/log
for LOG in `syslogd-listfiles`
do
   if [ -f $LOG ]; then
      savelog -g adm -m 640 -u root -c 14 $LOG >/dev/null
   fi
done
/etc/init.d/sysklogd reload
```

W przypadku programu `Logrotate` należy skonfigurować go, edytując w miarę potrzeb plik `/etc/logrotate.conf`, a następnie umieścić w katalogu `/etc/logrotate.d` pliki konfiguracyjne, odpowiadające różnym kategoriom plików rejestrowych. Przykład wpisu do takiego pliku może być następujący:

```
/var/log/ppp.log {
        notifempty
        nocompress
        weekly
}
```

Trzeba pamiętać, że po każdej przeprowadzonej rotacji demon `syslogd` powinien być uruchomiony ponownie, w celu otwarcia plików po rotacji. W przypadku skryptu dla polecenia `savelog` trzeba o to zadbać samemu, natomiast program `logrotate` robi to automatycznie.

6. Większe możliwości monitorowania

Syslog jest narzędziem uniwersalnym i zcentralizowanym. Korzyści płynące z tej centralizacji można zauważyć gdy używa się dodatkowych programów monitorujących pewne zdarzenia.

6.1. Monitorowanie TCP/IP

Przykładem prostego narzędzia wzbogacającego możliwości monitorowania systemu jest program IPlogger, standardowo obecny w dystrybucji Debian/GNU. Składa się on z dwóch demonów: `tcplogd` i `icmplogd`, które monitorują ruch z wykorzystaniem protokołów TCP i ICMP [1]. Obydwa demony generują komunikaty o wartościach pól funkcja i priorytet odpowiednio daemon i notice. W związku z czym w pliku `/etc/syslog.conf` można umieścić następujący wpis:

```
daemon.notice              /var/log/security/iplogger
```

Spowoduje to pojawianie się w pliku `/var/log/security/iplogger` komunikatów typu:

```
Jan 15 21:13:07 enterprise icmplogd: ping from localhost [127.0.0.1]
Jan 15 21:13:32 enterprise tcplogd: smtp connection attempt
                from gjn@localhost [127.0.0.1]
```

[1] Na dzień dzisiejszy (marzec 2001) obydwa programy są zastąpione przez bardziej uniwersalny *IPPL* (http://larve.net/ippl)

6.2. Współpraca z IPChains

Innym przykładem wykorzystania Syslog może być monitorowanie pracy firewall'a stworzonego przy pomocy pakietu IPChains. Dla każdej reguły filtrowania przekazywanej IPChains można ustawić flagę `-l`, która powoduje wysyłanie przez jądro komunikatu w przypadku pojawienia się pakietu odpowiadającego danej regule. Komunikat jest przesyłany przez `klogd` do `syslogd`. Jeżeli w pliku `/etc/syslog` dopisze się linię:

```
kern.=info      /var/log/security/packetfilter
```

komunikaty od IPchains znajdą się w podanym pliku.

Jeżeli chce się zablokować obsługę protokołu ICMP (blokowanie wszystkich typów komunikatów ICMP nie jest jednak przeważnie zalecane) oraz rejestrować wszystkie pakiety ICMP można dodać poniższą regułę:

```
ipchains -A input -p icmp -l -j DENY
```

Przykład komunikatu generowanych przez tę regułę jest podany poniżej:

```
Jan 18 21:28:05 enterprise kernel: Packet log: input DENY lo
   PROTO=1 127.0.0.1:8 127.0.0.1:0 L=84 S=0x00 I=1403 F=0x0000 T=64 (#1)
```

Jego dokładny opis można znaleźć w IPChains-HOWTO.

6.3. Monitorowanie instalowania programów

Innym, dość nietypowym, przykładem wykorzystania Syslogd może być monitorowanie instalacji programów przy pomocy narzędzia Installwatch. Installwatch przechwytuje wywołania funkcji systemowych operujących na systemie plików i rejestruje je przy pomocy Syslog. Dopisanie linijki:

```
user.=debug     /var/log/installwatch
```

do pliku konfiguracyjnego Syslog umożliwi przechwytywanie tych komunikatów, które mają postać podobną do:

```
Jan 16 21:52:30 enterprise test-installwatch:
   3 creat /tmp/installwatch-test #success
Jan 16 21:52:30 enterprise test-installwatch:
   0 chmod /tmp/installwatch-test 00600 #success
Jan 16 21:52:30 enterprise test-installwatch:
   0 unlink /tmp/installwatch-test #success
```

Na marginesie warto zauważyć, że Installwatch w połączeniu z programem Instmon jest doskonałym narzędziem do kontrolowania ręcznych instalacji oprogramowania.

7. Alternatywy dla Syslogd

System Syslog jest używany w środowisku systemów Unix od dawna. Jest to narzędzie sprawdzone, lecz posiadające pewne ograniczenia. Poważnym ograniczeniem często okazuje się prosty system sortowania komunikatów. Syslog ma tylko 12 kategorii komunikatów i osiem poziomów ich ważności. Nie ma również wbudowanych mechanizmów sortowania komunikatów na podstawie ich treści – musi być to realizowane przez zewnętrzne programy. Syslog nie posiada również mechanizmów kontroli dostępu przy przesyłaniu komunikatów przez sieć. Zostały podjęte próby napisania narzędzi, które usuwają te ograniczenia.

7.1. Secure Syslog

Pierwszy z nich to program SecureSyslog. Program jest dostępny pod adresem http://www.core-sdi.com/ssyslog. Twórcy programu skupili się na rozszerzeniu Syslog o możliwość bezpiecznego, zdalnego audytu. Audyt jest dokonywany na zdalnej zaufanej maszynie, na którą przesyłane są informacje od Syslog. SecureSyslog wykorzystuje specjalny protokół autoryzacji – PEO-1. Komunikacja pomiędzy osoba przeprowadzającą audyt a maszyną na której pracuje SecureSyslog jest szyfrowana przy pomocy algorytmu Blowfish.

7.2. Syslog-NG

Drugim, zasługującym na szczególne zainteresowanie, jest Syslog-NG (Next Generation). Został napisany w firmie BalaBit Computing i jest dostępny pod adresem http://www.balabit.hu/products/syslog-ng. Syslog-NG jest lepiej konfigurowalny, umożliwia sortowanie wiadomości na podstawie ich zawartości, ma mechanizmy umożliwiające kontrolę integralności plików rejestrowych oraz ich szyfrowanie, posiada także rozbudowane funkcje przesyłania komunikatów przez sieć. Syslog-NG pracuje na platformach Linux, BSD, Solaris.

8. Podsumowanie

Konfigurowanie Syslog nie jest procesem trywialnym, lecz dzięki dobrej dokumentacji nie jest trudne. By wykorzystać w pełni możliwości jakie daje ten pakiet warto użyć narzędzi pomocniczych takich jak Cron. Jednak czas poświęcony na konfigurację Syslog zwraca się potem w trakcie codziennego administrowania systemem.