Systemd.journald: pregledavanje binarnih logova

Dok su tradicionalni Unixoidi prepuštali pojedinim servisima i aplikacijama da sami zapisuju vlastite logove, dnevnike, systemd uvodi jedinstven sustav kojem je prvenstven cilj prevazilaženje razlika u zapisima, bolja kontrola i preglednost logova. Servis koji se time bavi je journal daemon ili journald, koji je sad zadužen za logiranje svega, od kernela do korisničkih aplikacija. Time se standardizira način na koji se logovi prikupljaju i analiziraju.

Binarni način zapisivanja omogućava da se zapisi prikazuju na različite načine, na primjer u syslog formatu, ili kao JSON java objekti za grafičke prikaze. Journald čuva kompatibilnost sa starim syslogom, što je korisno ako na primjer koristite centralni syslog server na kojem prikupljate logove se svih servera. No dozvoljava da centralni log server koristi journald, a dozvoljva i kombiniranje oba sustava.

Analiza logova različitih servisa na istom serveru kao i zapisa sa različitih servera, koji mogu biti u različitim vremenskim zonama, zahtijeva prije svega sinhronizaciju vremena. Mi admini stare garde, odgojeni na Unixu, podešavamo hardverski sat u računalu na univerzalno vrijeme, UTC, a onda zadavanjem lokalne vremenske zone omogućujemo da sat pokazuje lokalno vrijeme. Na MS Windowsima nije tako, tamo se računalo podešava na lokalno vrijeme, a zadavanje vremenske zone omogućava određivanje univerzalnog vremena. Da bi provjerili konfiguraciju na svom računalu, koristimo alat **timedatectl**:

```
$ timedatectl status
      Local time: ?et 2017-11-30 17:12:20 CET
  Universal time: ?et 2017-11-30 16:12:20 UTC
        RTC time: ?et 2017-11-30 16:12:20
       Time zone: Europe/Zagreb (CET, +0100)
 Network time on: yes
 NTP synchronized: yes
 RTC in local TZ: no
```

Ovdje je jasno pokazano da je lokalno vrijeme sat vremena ispred univerzalnog, znači da sunce kod nas ranije izlazi nego Greenwichu. Uključena je sinkronizacija vremena preko NTP protokola, a hardverski sat (RTC) je podešen na UTC, a ne na lokalno vrijeme.

Pogledajmo kako su definirane vremenske zone:

```
$ timedatectl list-timezones
Africa/Abidjan
Africa/Accra
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmara
Africa/Bamako
Africa/Bangui
Africa/Banjul
Africa/Bissau
...
```

Prekinuli smo ispis jer je predugačak, ali očito je da se zona određuje parom Kontinent/Glavni grad. Ako pustimo ispis dalje, pojavait će se Europe/Zagreb, kao što nam je pokazala prethodna naredba.

Vremensku zonu podesimo ovako:

```
$ timedatectl set-timezone Europe/Zagreb
```

Nećemo ništa pokvariti ako zadamo vremensku zonu Europe/Sarajevo, jer je to ista zona, CET, odnosno UTC +1, samo ćemo se pretvarati da smo u susjednoj državi. :)

Journald može prikazivati vrijeme zapisa kao lokano, ili univerzalno, kako vam drago. Podrazumijeva se da će pokazivati lokalno vrijeme, ako se drugačije ne zada:

```
$ journalctl --utc
```

Ako natipkamo journalctl bez parametara, dobit ćemo sve zapise, počev od najstarijih prema najnovijima. Ali ako želimo vidjeti samo događaje nakon zadnjeg podizanja sustava, koristit ćemo parametar -b (boot):

```
$ journalctl -b
-- Logs begin at ?et 2017-11-30 16:54:56 CET, end at ?et 2017-11-30 17:35:44 CET. --
Stu 30 16:54:56 l33t systemd-
journald[228]: Runtime journal (/run/log/journal/) is 4.6M, max 37.3M, 32.6M free.
Stu 30 16:54:56 l33t kernel: microcode: CPU0 microcode updated early to revision 0x4,
 date = 2013-06-28
Stu 30 16:54:56 l33t kernel: Initializing cgroup subsys cpuset
Stu 30 16:54:56 l33t kernel: Initializing cgroup subsys cpu
Stu 30 16:54:56 l33t kernel: Initializing cgroup subsys cpuacct
Stu 30 16:54:56 l33t kernel: Linux version 4.4.0-101-generic (buildd@lcy01-amd64-006)
 (gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubun
Stu 30 16:54:56 l33t kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.4.0-101-generic
 root=UUID=ab4ae64d-d5e1-4fd0-967a-45766a96e524
Stu 30 16:54:56 l33t kernel: KERNEL supported cpus:
Stu 30 16:54:56 l33t kernel:   Intel GenuineIntel
Stu 30 16:54:56 l33t kernel:   AMD AuthenticAMD
Stu 30 16:54:56 l33t kernel:   Centaur CentaurHauls
Stu 30 16:54:56 l33t kernel: x86/fpu: Legacy x87 FPU detected.
Stu 30 16:54:56 l33t kernel: x86/fpu: Using 'lazy' FPU context switches.
Stu 30 16:54:56 l33t kernel: e820: BIOS-provided physical RAM map:
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x0000000000000000-0x000000000009fbff] usable
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000000e0000-0x00000000000fffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x0000000000100000-0x00000000ba366fff] usable
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000ba367000-0x00000000bb4e6fff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000bb4e7000-0x00000000bb766fff] ACPI NVS
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000bb767000-0x00000000bb7fefff] ACPI data
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000bb7ff000-0x00000000bb7fffff] usable
Stu 30 16:54:56 l33t kernel: BIOS-
```

```
e820: [mem 0x00000000bb800000-0x00000000bbffffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000e0000000-0x00000000efffffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000fed10000-0x00000000fed13fff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000fed19000-0x00000000fed19fff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000fed1b000-0x00000000fed1ffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x00000000ffd00000-0x00000000ffffffff] reserved
Stu 30 16:54:56 l33t kernel: BIOS-
e820: [mem 0x0000000100000000-0x0000000137ffffff] usable
Stu 30 16:54:56 l33t kernel: NX (Execute Disable) protection: active
Stu 30 16:54:56 l33t kernel: SMBIOS 2.6 present.
Stu 30 16:54:56 l33t kernel: DMI: Hewlett-
Packard HP EliteBook 2740p/7007, BIOS 68COU Ver. F.04 03/08/2011
Stu 30 16:54:56 l33t kernel: e820: update [mem 0x00000000-0x00000fff] usable ==> rese
rved
Stu 30 16:54:56 l33t kernel: e820: remove [mem 0x000a0000-0x000fffff] usable
Stu 30 16:54:56 l33t kernel: e820: last_pfn = 0x138000 max_arch_pfn = 0x400000000
Stu 30 16:54:56 l33t kernel: MTRR default type: uncachable
Stu 30 16:54:56 l33t kernel: MTRR fixed ranges enabled:
Stu 30 16:54:56 l33t kernel:   00000-9FFFF write-back
Stu 30 16:54:56 l33t kernel:   A0000-BFFFF uncachable
Stu 30 16:54:56 l33t kernel:   C0000-FFFFF write-protect
Stu 30 16:54:56 l33t kernel: MTRR variable ranges enabled:
```

Ovaj nam isječak islustrira činjenicu koju smo naveli u prethodnom članku, naime da journald bilježi zapise iz rane faze bootanja, koje smo nekada morali "loviti" po ekranu. Zapis počinje vremenom bootanja, a zatim navodi koliko su veliki logovi i koliko je ograničenje za maksimalnu veličinu datoteka, što nas opet podsjeća da nam više ne treba logrotate, jer journald sam brine da ne zaguši diskove.

O pojedinoj distribuciji ovisi da li se logovi čuvaju "trajno", ili se brišu kod gašenja sustava. Podsjetimo se, dnevnik ostaje nakon gašenja ako kreirate direktorij u kojeg se sprema datoteka:

```
$ sudo mkdir -p /var/log/journal
```

ili to možete zadati u konfiguraciji:

```
$ visudo /etc/systemd/journald.conf
[Journal]
Storage=persistent
```

Čuvanje starih logova ima više smisla na serverima nego na desktopima, gdje logovi nisu persistent (baš kao i na Windowsima :). Provjerimo na mom notebooku:

```
$ journalctl --list-boots
 0 ab1b4e0ce9fc496fbcfcfbb18011c5a4 ?et 2017-11-30 16:54:56 CET—?et 2017-11-30 18:21:
44 CET
```

Ovo će izlistati jedan redak za svako podizanje sustava. Na serveru bi bilo više redaka, ovdje samo jedan. Iza rednog broja vidi se jedinstveni ID svakog boota. Ako biste na serveru željeli vidjeti zapise od zadnjeg boota, trebalo bi ih zatražiti ovako:

```
$ journalctl -b -1
```

ili ovako:

```
$ journalctl -b ab1b4e0ce9fc496fbcfcfbb18011c5a4
```

Ako pretpostavljate da se nešto dogodilo između jedan i dva poslije ponoći, možete izrezati taj dio loga koristeći parametre --since i --until.

```
$ journalctl --since "2017-11-29 01:00" --until "2017-11-29 02:00"
```

Treba li reći da je maska za datum YYYY-MM-DD HH:MM? Ili naški GGGG-MM-DD SS:MM :). (Samo da netko ne pomisli da SS predstavlja sekunde.)

Ponekad će nam zatrebati filtriranje zapisa po procesu, korisniku ili grupi. U tom slučaju koristit ćemo slijedeće parametre:

```
$ journalctl _PID=1234
$ journalctl _UID=1000
$ journalctl _GID=1000
```

To je zgodno ako znate ID nekog procesa, korisnika grupe. Ali ako ne znate PID procesa, poslužite se imenom korisnika koji je vlasnik procesa.

```
$ journalctl -u clamav
```

Poruke koje zapisuje kernel dosad bi izlistali pomoću naredbe dmesg, a sad to možemo i ovako:

```
$ journalctl -k
```

ili

```
$ journalctl --dmesg
```

Dosad ste već primijetili da ne morate, kao nekad, iza naredbe dodati | less, ili | more, jer journalctl automatski koristi pager. No ako želite ispis spremiti u datoteku, isključit ćete pager:

```
$ journalctl --no-pager
```

Journal preuzima syslogove razine poruka:

0: emerg
1: alert
2: crit
3: err
4: warning
5: notice

6: info
7: debug

To omogućuje filtriranje na primjer samo poruka o greškama:

```
$ journalctl -p err
```

Time bi trebali dobiti poruke od treće do sedme razine.

Kod brzog pregledavanja logova vjerojatno ćete htjeti samo osnovne informacije, bez suvišnih detalja:

```
$ journalctl --no-full
```

A kad pronađete nešto sumnjivo, možete zatražiti puni zapis

```
$ journalctl -a
```

Naravno, ograničit ćete ispis na određen proces ili vremensko razdoblje:

```
$ journalctl -a -u XXX --since "YYYY-MM-DD" --until "YYYY-MM-DD HH:SS"
```

Nekad smo koristili naredbu tail da bi vidjeli samo najsvježije zapise, a sad to činimo ovako:

```
$ journalctl -t
```

ili ovako, ako želimo vidjeti zadnjih 10 redaka:

```
$ journalctl -n 10
```

U prošlom smo nastavku objasnili kako se ograničava prostor koji zapisi zauzimaju i vrijeme čuvanja, ali što ako se nađete u kritičnoj situaciji u kojoj je disk zapunjen, pa morate hitno nešto pobrisati? Ako želite žrtvovati logove, pogledajte koliko diska vam troše:

```
$ journalctl --disk-usage
```

A onda pobrišete starije zapise, zadavši veličinu loga koju želite:

```
$ sudo journalctl --vacuum-size=100M
```

Sve u svemu, zanimljivo. Mislim da ćemo se brzo navići na journal, ako već nismo.

čet, 2017-11-30 19:20 - Aco Dmitrović**Kuharice:** Linux [1]
Kategorije: Operacijski sustavi [2]
Vote: 0

No votes yet

story_tag: Linux [3]
journald [4]
systemd [5]

Source URL: https://sysportal.carnet.hr/node/1777

Links
[1] https://sysportal.carnet.hr/taxonomy/term/17
[2] https://sysportal.carnet.hr/taxonomy/term/26
[3] https://sysportal.carnet.hr/taxonomy/term/119
[4] https://sysportal.carnet.hr/taxonomy/term/158
[5] https://sysportal.carnet.hr/taxonomy/term/120