Rapport

Introduktion till Linux och små nätverk, 7,5 hp
VT-2025

Projektuppgift

Laboration 5: Processer

av

Sixten Peterson (050402-XXXX)

Akademin för teknik och miljö
Avdelningen för industriell utveckling, IT och samhällsbyggnad

Högskolan i Gävle
S-801 76 Gävle, Sweden

Datorpost:

education@snicon.rip

Innehållsförteckning

1. Inledning

Laborationen utgör det femte momentet i kursen, och består av tre delar som alla ofattar hantering av processer i Linuxmiljö. Syftet med laborationen är att undersöka hur grundläggande systemadministration av processer går till. kan avslutas, hanteras i htop/top samt listas och filtreras.

1.1. Bakgrund

Denna laboration utgör det femte momentet i kursen ”Introduktion till Linux och små nätverk” (DVG001) och består av tre delar. Samtliga delar utförs på labbmiljön som tagits fram under den första laborationen. I sin helhet undersöker laborationen hur processer hanteras i en Linuxmiljö.

Följande delar undersöker:

1. hur processer kan avslutas med `SIGTERM` eller `SIGKILL`. Detta genom användning av två terminalinstanser och `nano`
2. processhantering i htop/top där de inbyggda funktionerna ska brukas för att avsluta en `nano`-process
3. vilket/vilka kommandon som krävs för att spara en lista av alla processer som körts av en användare (`root`) i en textfil

1.2. Syfte

Syfter är att ta reda på hur grundläggande systemadministration av processer går till. Hur avslutas processer samt hur går dessa att lista och filtrera?

2. Planering och genomförande

Genom hela laborationen sker kontinuerlig dokumentation. Ytterligare planeras två SSH-anslutningar upprättas till Linuxmiljön – en för vardera terminal - under utförandet av de praktiska momenten. Vid eventuella problem nyttjas föreläsningsmaterial och internet.

2.1. Planering

Arbetet dokumenteras kontinuerligt i realtid i samband med laborationens utförande för att säkerställa så ackurat information i rapporten som möjligt. Vid eventuella problem eller funderingar nyttjas i första hand föreläsningsmaterialet och i andra hand en sökmotor såsom DuckDuckGo eller Google. Där officiell dokumentation för distributionen finns tillgänglig på internet prioriteras denna högst. Kommunikation upprättas mot labbmiljön som kör Debian 12 Bookworm på en gamal Dell Inspiron 570 genom en SSH-anslutning från min Macbook Pro med macOS Sequoia 15.4.1.

2.2. Genomförande

Genomförandet är uppdelat i tre delar som återspeglar de olika delarna av uppgiftsbeskrivningen. Nedan följer även en redogörelse av det som utförts innan samtliga delar.

I vanlig ordning inleds laborationen med att en anslutning upprättas mot labmiljön genom `ssh hig-25sipe01@192.168.1.250` i ett terminalfönster på min Macbook. Därefter skrevs lösenordet för kontot in. Därpå upprepades samma moment ytterligare en gång för att på så vis upprätta två anslutningar, vilket används senare i laborationen.

Innan själva utförandet av uppgiften utfördes slutligen säkerhetsuppdateringar av labbmiljön i

enlighet med det som detaljeras i boken [1, pp 116, 120], således kördes alltså `sudo apt update` och `sudo apt upgrade.`

2.2.1. Del ett – Processavslutning med SIGTERM

Först startas nano i *term1* med hjälp av `nano`, kommandot kördes i hemkatalogen. Med hjälp av `ps(1)` och växeln `-u` gick det att skriva ut en processtabell som visar alla processer startade av användaren som exekverat kommandot. Alltså kördes kommandot `ps -u` i *term2* av min användare (hig-25sipe01), utdatan skådas i Figur 1.

```
hig-25sipe01@dvg001:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
hig-25s+   18534  0.0  0.0  10844  4668 pts/0    Ss   13:46   0:00 -bash
hig-25s+   18546  0.0  0.0  10844  4804 pts/1    Ss   13:46   0:00 -bash
hig-25s+   42493  0.0  0.0   9704  3696 pts/0    S+   13:52   0:00 nano
hig-25s+   42505  0.0  0.0  13964  4392 pts/1    R+   14:00   0:00 ps -u
```

Figur 1: Processtabellen från körning av ps(1).

I Figur 1 går det att utläsa att processens id (*PID*) är `42493` och således avslutas nano-processen i *term1* genom att i *term2* köra kommandot `kill 42493`. Anledningen till att jag valt att avnända just SIGTERM är det faktum att det inte är något fel på processen, hade processen därmot inte svarat hade `SIGKILL` (`kill -9 <PID>`) som tvingar processen att avsluta kunnat vara ett alternativ. Därpå kördes `ps -u` ytterligare en gång i *term2* för att på så vis kunna bekräfta att processen är avslutad, mycket riktigt återfinns processen inte i den nya processtabellen, se Figur 2. Ytterligare visar *term1* ett meddelande om att processen tagit emot en `SIGHUP` eller `SIGTERM`, se Figur 3.

```
hig-25sipe01@dvg001:~$ kill 42493
hig-25sipe01@dvg001:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
hig-25s+   18534  0.0  0.0  10844  4668 pts/0    Ss+  13:46   0:00 -bash
hig-25s+   18546  0.0  0.0  10844  4804 pts/1    Ss   13:46   0:00 -bash
hig-25s+   42509  200  0.0  13964  4380 pts/1    R+   14:07   0:00 ps -u
```

Figur 2: Processtabellen från andra körningen av ps(1)

```
hig-25sipe01@dvg001:~$ nano
Received SIGHUP or SIGTERM

Buffer written to nano.42493.save
```

Figur 3: Utdata i term1 efter nano-processen avslutats.

2.2.2. Del två – Processavslutning med htop

Till att börja med startades `nano(1)` på nytt i *term1*. I nano skrevs några rader text in i programmet, och därefter förflyttades fokus till *term2* där kommandot `sudo aptitude search htop` användes för att söka efter programmet `htop(1)`. Som Figur 4 visar återfinns ett program vars beskrivning såg rimlig ut för det som programmet ämnas användas för i uppgiften och således installerades det med sudo `aptitude install htop`.

```
hig-25sipe01@dvg001:~$ sudo aptitude search htop
[sudo] password for hig-25sipe01:
p   htop                                                   - interactive processes viewer
```

Figur 4: Sökträff från sökning via aptitude search efter htop.

Med `htop(1)` väl installerat kördes kommandot `htop`. Efter kommandot exekverat möttes jag av ett gränssnitt som bland annat visar olika processer och resursanvändningen på maskinen, se Figur 5. För att sedan avsluta processen navigerade jag med piltangenterna på tangentbordet för att markera nano-processen (PID: 42529). Med processen markerad tryckte jag ner *F9*-tangenten och möttes av en ny meny där jag kunde välja mellan olika signaler, där valde jag `15 SIGTERM` och tryckte på *enter*-tangenten. Efter detta försvann processen från gränssnittet i htop och i *term1* kunde jag återfinna ett identsikt meddelande till det som återfinns i Figur 3

```
...-25sipe01@dvg001: ~ — ssh hig-25sipe01@192.168.1.250   ...sipe01@dvg001: ~ — ssh hig-25sipe01@192.168.1.250   +

  0[|                                          0.7%] Tasks: 20, 0 thr, 68 kthr; 1 running
  1[                                           0.0%] Load average: 0.03 0.03 0.00
Mem[|||||||||                            136M/9.71G] Uptime: 27 days, 22:13:04
Swp[                                        0K/975M]

  Main   I/O
    PID USER       PRI  NI  VIRT   RES   SHR S  CPU%▽MEM%   TIME+  Command
  42671 hig-25sipe  20   0 10728  4100  3292 R   1.3  0.0  0:00.04 htop
      1 root       20   0  163M 11936  9024 S   0.0  0.1  0:25.23 /lib/systemd/systemd --system --deserialize=32
    346 root       20   0  5868  3500  2652 S   0.0  0.0  0:00.13 dhclient -4 -v -i -pf /run/dhclient.enp2s0.pid -lf
    352 root       20   0  9484  2712  2460 S   0.0  0.0  0:05.76 /usr/sbin/cron -f
    353 messagebus 20   0  9392  4884  4248 S   0.0  0.0  0:01.71 /usr/bin/dbus-daemon --system --address=systemd: -
    356 root       20   0 50016  8144  6980 S   0.0  0.1  0:05.88 /lib/systemd/systemd-logind
    370 root       20   0  8748  1056   964 S   0.0  0.0  0:00.00 /sbin/agetty -o -p -- \u --noclear - linux
  18519 root       20   0 17824 10860  9272 S   0.0  0.1  0:00.16 sshd: hig-25sipe01 [priv]
  18522 hig-25sipe 20   0 18968 10468  8752 S   0.0  0.1  0:00.11 /lib/systemd/systemd --user
  18523 hig-25sipe 20   0  164M  3092     0 S   0.0  0.0  0:00.00 (sd-pam)
  18533 hig-25sipe 20   0 17984  6740  4892 S   0.0  0.1  0:00.12 sshd: hig-25sipe01@pts/0
  18534 hig-25sipe 20   0 12240  5548  4000 S   0.0  0.1  0:00.09 -bash
  18539 root       20   0 17824 10792  9200 S   0.0  0.1  0:00.16 sshd: hig-25sipe01 [priv]
  18545 hig-25sipe 20   0 17984  6740  4892 S   0.0  0.1  0:00.26 sshd: hig-25sipe01@pts/1
  18546 hig-25sipe 20   0 10844  4808  3416 S   0.0  0.0  0:00.06 -bash
  31798 root       20   0 26036  5204  4384 S   0.0  0.1  0:00.05 /lib/systemd/systemd-udevd
  31964 root       20   0 15436  8224  7028 S   0.0  0.1  0:00.01 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 sta
  32035 root       20   0 49352 15760 12720 S   0.0  0.2  0:00.06 /lib/systemd/systemd-journald
  42479 ntpsec     RT   0 84888 19260  8852 S   0.0  0.2  0:00.44 /usr/sbin/ntpd -p /run/ntpd.pid -c /etc/ntpsec/ntp
  42529 hig-25sipe 20   0  9704  3640  2868 S   0.0  0.0  0:00.00 nano

F1Help  F2Setup  F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Figur 5: Skärmdump av htop-gränssnittet.

2.2.3. Del tre – Processlistning och filtrering

För denna del var det uppenbart att det går att dela upp processen i två steg: 1) Lista alla processer som körs av root, 2) Spara resultatet i en textfil. I syfte att lista ut hur steg 1 går till använde jag manualen för ps(1) och kom fram till att det gick att använda samma växel som användes under 2.2.1 Del ett. Denna gång specificeras dock ett användarnamn – root. Med denna nyfunna kunskap kördes ps -u root, utdatan listar ett flertal olika processer och ser rimligt ut, exakt vad utdatan innehåller återfinns såklart i filen som medföljer inlämningen men även som bilaga, se Bilaga 1. Eftersom jag vill spara utadatan till en fil valde jag att först skapa en ny katalog med `mkdir lab5` där filen sedan kan lagras. Efter att navigerat in i katalogen med `cd lab5` så körde jag `ps -u root > processer_root.txt` för att på så vis spara ner utdatan till den specificerade filen. Slutligen kördes `cat processer_root.txt` i syfte att bekräfta att filen innehöll processerna, vilket den gjorde.

3. Beskrivning av slutresultat

Med hjälp av kommandon såsom `ps`, `kill`, `htop` och `nano` har en praktisk undersökning av hur processhantering går till i Linux utförts. En process av `nano` har avlsutats genom såväl kommandot `kill` som genom kommandot `htop`, ytterligare har en lista av processer framställts med hjälp av `ps` i kombination med växeln ”*-u*”

4. Diskussion

Under administration av hemmaserverar har jag tidigare hanterat processer utan större problem med hjälp av internet. Däremot har speciellt handledningen för detta moment öppnat ögonen för vad program såsom top och htop kan göra, detta var mycket intressant och lärorikt. Det hade varit kul om uppgiften hade innefattat en del till om användning av htop/top utöver att bara avsluta processer – men samtigt så finns det ju också en manual att djupdyka i. Till en början förväntade jag mig att jag skulle behöva använda grep för att på något vis filtrera ut de processer som startats av en viss användare, men det visade sig att växlar kunde vara till stor hjälp. Allt som allt är jag nöjd med resultaten jag uppnått även om själva uppgiften upplevts som väldigt grundläggande.

5. Slutsatser

Under uppgiftens gång har det blivit klart hur grundläggande systemadministration av processer går till. Genomförandet beskriver väl hur en process kan avslutas. Dessutom visar laborationsrapporten på hur växlar kan utnyttjas för att filtrera processer utefter olika användare.

6. Referenser

Litteraturförteckning

[1]: R. Hertzog, R. Mas, The Debian Administrator's Handbook, Debian Buster from Discovery to Mastery, 2020.

I. Bilaga 1: Processer som körs av root

```
    PID TTY          TIME CMD
      1 ?        00:00:25 systemd
      2 ?        00:00:00 kthreadd
      3 ?        00:00:00 rcu_gp
      4 ?        00:00:00 rcu_par_gp
      5 ?        00:00:00 slub_flushwq
      6 ?        00:00:00 netns
      8 ?        00:00:00 kworker/0:0H-events_highpri
     10 ?        00:00:00 mm_percpu_wq
     11 ?        00:00:00 rcu_tasks_kthread
     12 ?        00:00:00 rcu_tasks_rude_kthread
     13 ?        00:00:00 rcu_tasks_trace_kthread
     14 ?        00:00:00 ksoftirqd/0
     15 ?        00:02:09 rcu_preempt
     16 ?        00:00:17 migration/0
     18 ?        00:00:00 cpuhp/0
     19 ?        00:00:00 cpuhp/1
     20 ?        00:00:17 migration/1
     21 ?        00:00:00 ksoftirqd/1
     23 ?        00:00:00 kworker/1:0H-kblockd
     26 ?        00:00:00 kdevtmpfs
     27 ?        00:00:00 inet_frag_wq
     28 ?        00:00:00 kauditd
     29 ?        00:00:01 khungtaskd
     31 ?        00:00:00 oom_reaper
     32 ?        00:00:00 writeback
     33 ?        00:02:34 kcompactd0
     34 ?        00:00:00 ksmd
     36 ?        00:00:21 khugepaged
     37 ?        00:00:00 kintegrityd
     38 ?        00:00:00 kblockd
     39 ?        00:00:00 blkcg_punt_bio
     40 ?        00:00:00 tpm_dev_wq
     41 ?        00:00:00 edac-poller
     42 ?        00:00:00 devfreq_wq
     44 ?        00:00:00 kswapd0
     50 ?        00:00:00 kthrotld
     52 ?        00:00:00 acpi_thermal_pm
     54 ?        00:00:00 mld
     55 ?        00:00:00 ipv6_addrconf
     58 ?        00:00:00 kworker/0:1H-kblockd
     61 ?        00:00:00 kstrp
     66 ?        00:00:00 zswap-shrink
     67 ?        00:00:00 kworker/u13:0
    137 ?        00:00:00 ata_sff
    139 ?        00:00:00 scsi_eh_0
    140 ?        00:00:00 scsi_tmf_0
    141 ?        00:00:00 scsi_eh_1
    142 ?        00:00:00 scsi_tmf_1
```

```
  143 ?        00:00:00 scsi_eh_2
  144 ?        00:00:00 scsi_tmf_2
  145 ?        00:00:00 scsi_eh_3
  146 ?        00:00:00 scsi_tmf_3
  187 ?        00:00:02 jbd2/sda1-8
  188 ?        00:00:00 ext4-rsv-conver
  287 ?        00:00:00 watchdogd
  296 ?        00:00:11 kworker/1:2H-kblockd
  346 ?        00:00:00 dhclient
  350 ?        00:00:00 nvkm-disp
  352 ?        00:00:05 cron
  356 ?        00:00:05 systemd-logind
  370 tty1     00:00:00 agetty
  380 ?        00:00:00 card0-crtc0
  381 ?        00:00:00 card0-crtc1
  382 ?        00:00:00 card0-crtc2
  383 ?        00:00:00 card0-crtc3
18519 ?        00:00:00 sshd
18539 ?        00:00:00 sshd
31798 ?        00:00:00 systemd-udevd
31799 ?        00:00:00 kworker/0:0-events
31800 ?        00:00:01 kworker/0:3-mm_percpu_wq
31964 ?        00:00:00 sshd
32035 ?        00:00:00 systemd-journal
32036 ?        00:00:02 kworker/1:3-mm_percpu_wq
42682 ?        00:00:00 kworker/u12:2-events_unbound
42688 ?        00:00:00 kworker/u12:1-events_unbound
42699 ?        00:00:00 kworker/1:0-events
42701 ?        00:00:00 kworker/u12:0-events_unbound
42703 ?        00:00:00 kworker/1:1-events_power_efficient
```