



## Prozesszustände

### Prozesszustände in Linux:

1/2)

```
marvinaichinger@LAPTOP-KOQM5K57: ~
top - 18:44:01 up 4 min, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 7 total, 1 running, 6 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.9 us, 2.8 sy, 0.0 ni, 93.1 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st
MiB Mem : 16150.9 total, 8233.4 free, 7693.6 used, 224.0 buff/cache
MiB Swap: 49152.0 total, 49028.1 free, 123.9 used, 8326.8 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0    8936    308    264  S   0.0   0.0   0:00.06 init
    6 root        20   0    8936    212    168  S   0.0   0.0   0:00.01 init
    7 marvina+   20   0   18216   3868   3756  S   0.0   0.0   0:00.13 bash
  104 marvina+   20   0   17500   3264   1500  S   0.0   0.0   0:00.01 nano
  105 root        20   0    8936    224    180  S   0.0   0.0   0:00.00 init
  106 marvina+   20   0   18084   3632   3548  S   0.0   0.0   0:00.04 bash
  121 marvina+   20   0   18908   2136   1512  R   0.0   0.0   0:00.00 top

marvinaichinger@LAPTOP-KOQM5K57: ~/06-prozesse
GNU nano 4.8                                test.txt
Hallo
```

```
marvinaichinger@LAPTOP-KOQM5K57:~$ ps -aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   8936    308 ?        Ssl   18:39   0:00 /init
root         6  0.0  0.0   8936    212 tty1     Ss    18:39   0:00 /init
marvina+    7  0.0  0.0  18216   3868 tty1     S    18:39   0:00 -bash
marvina+   104  0.0  0.0  17500   3264 tty1     S    18:43   0:00 nano test.txt
root       105  0.0  0.0   8936    224 tty2     Ss    18:43   0:00 /init
marvina+   106  0.0  0.0  18084   3632 tty2     S    18:43   0:00 -bash
marvina+   126  0.0  0.0  18880   2032 tty2     R    18:45   0:00 ps -aux
marvinaichinger@LAPTOP-KOQM5K57:~$ ps -aux | grep nano
marvina+   104  0.0  0.0  17500   3264 tty1     S    18:43   0:00 nano test.txt
marvina+   128  0.0  0.0  16208   1288 tty2     S    18:46   0:00 grep --color=auto nano
marvinaichinger@LAPTOP-KOQM5K57:~$ ps
  PID TTY          TIME CMD
  106 tty2        0:00:00 bash
  129 tty2        0:00:00 ps
marvinaichinger@LAPTOP-KOQM5K57:~$ pstree
init--init--bash--nano
   |--init--bash--pstree
   |--{init}
marvinaichinger@LAPTOP-KOQM5K57:~$
```

3)

```
156 marvina+ 20 0 18916 2148 1520 T 0.0 0.0 0:00.01 top
159 marvina+ 20 0 18908 2144 1520 R 0.0 0.0 0:00.01 top
```

4)

```

marvinaichinger@LAPTOP-KOQM5K57: ~
marvinaichinger@LAPTOP-KOQM5K57:~$ top &
[1] 166
marvinaichinger@LAPTOP-KOQM5K57:~$

marvinaichinger@LAPTOP-KOQM5K57: ~
top - 19:00:45 up 21 min, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 7 total, 1 running, 5 sleeping, 1 stopped, 0 zombie
%Cpu(s): 1.9 us, 1.7 sy, 0.0 ni, 96.4 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 16150.9 total, 8312.3 free, 7614.6 used, 224.0 buff/cache
MiB Swap: 49152.0 total, 49023.6 free, 128.4 used, 8405.7 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0    8936    308    264 S   0.0   0.0   0:00.06 init
   105 root        20   0    8936    224    180 S   0.0   0.0   0:00.00 init
   106 marvina+    20   0   18084   3644   3552 S   0.0   0.0   0:00.07 bash
   131 root        20   0    8936    224    180 S   0.0   0.0   0:00.00 init
   132 marvina+    20   0   18216   3868   3644 S   0.0   0.0   0:00.04 bash
   166 marvina+    20   0   18428   1768   1308 T   0.0   0.0   0:00.01 top
   167 marvina+    20   0   18908   2140   1520 R   0.0   0.0   0:00.01 top

```

5)

Mit dem Befehl „fg 1“ kann ein Prozess im Hintergrund wieder in den Vordergrund gebracht werden. Dabei ist immer die Nummer anzugeben, welche beim Starten in den Hintergrund in den [] - Klammern ausgegeben wird. Mit STRG+C kann der Prozess dann beendet werden. Auch möglich ist es den Prozess mit *kill* zu beenden:

```

marvinaichinger@LAPTOP-KOQM5K57: ~/06-prozesse
Tasks: 8 total, 1 running, 7 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.4 us, 0.2 sy, 0.0 ni, 99.3 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st
MiB Mem : 16150.9 total, 8537.1 free, 7389.8 used, 224.0 buff/cache
MiB Swap: 49152.0 total, 49023.6 free, 128.4 used, 8630.5 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
    1 root        20   0    8936    308    264 S   0.0   0.0   0:00.06 init
    6 root        20   0    8936    212    168 S   0.0   0.0   0:00.01 init
    7 marvina+    20   0   18216   3868   3768 S   0.0   0.0   0:00.15 bash
   105 root        20   0    8936    224    180 S   0.0   0.0   0:00.00 init
   106 marvina+    20   0   18084   3636   3448 S   0.0   0.0   0:00.07 bash
   131 root        20   0    8936    224    180 S   0.0   0.0   0:00.00 init
   132 marvina+    20   0   18216   3844   3576 S   0.0   0.0   0:00.04 bash
   154 marvina+    20   0   18908   2144   1520 R   0.0   0.0   0:00.00 top

```

↓

```

marvinaichinger@LAPTOP-KOQM5K57: ~
marvina+ 106 0.0 0.0 18084 3632 tty2 S
marvina+ 126 0.0 0.0 18880 2032 tty2 R
marvinaichinger@LAPTOP-KOQM5K57:~$ ps -aux | gre
marvina+ 104 0.0 0.0 17500 3264 tty1 S
marvina+ 128 0.0 0.0 16208 1288 tty2 S
marvinaichinger@LAPTOP-KOQM5K57:~$ ps
  PID TTY          TIME CMD
   106 tty2        0:00:00 bash
   129 tty2        0:00:00 ps
marvinaichinger@LAPTOP-KOQM5K57:~$ pstree
init--init--bash--nano
   |--init--bash--pstree
   |--{init}
marvinaichinger@LAPTOP-KOQM5K57:~$ pidof top
154
marvinaichinger@LAPTOP-KOQM5K57:~$ kill 154
marvinaichinger@LAPTOP-KOQM5K57:~$

```

6)

Ein Prozess muss nicht ständig Daten verarbeiten oder Berechnungen durchführen. Oft ist es für einen Prozess nötig seine Arbeit niederzulegen und darauf zu warten, bis ein bestimmtes Ereignis passiert. Zum Beispiel, dass eine bestimmte Datei oder irgendwelche Daten aus dem Internet angekommen sind oder, dass der User eine bestimmte Eingabe gemacht hat, usw.

7)

```

marvinaichinger@LAPTOP-KOQM5K57: ~
top - 19:03:41 up 24 min, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 7 total, 2 running, 5 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2.8 us, 8.2 sy, 0.0 ni, 88.7 id, 0.0 wa, 0.4 hi, 0.0 si, 0.0 st
MiB Mem : 16150.9 total, 8322.2 free, 7604.7 used, 224.0 buff/cache
MiB Swap: 49152.0 total, 49023.6 free, 128.4 used, 8415.6 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
 174 marvina+  20   0  15276   812   664  R   19.3   0.0   0:00.78 yes
    1 root      20   0   8936   308   264  S   0.0   0.0   0:00.06 init
 105 root      20   0   8936   224   180  S   0.0   0.0   0:00.00 init
 106 marvina+  20   0  18084  3640  3560  S   0.0   0.0   0:00.07 bash
 131 root      20   0   8936   224   180  S   0.0   0.0   0:00.00 init
 132 marvina+  20   0  18216  3868  3760  S   0.0   0.0   0:00.07 bash
 172 marvina+  20   0  18904  2140  1520  R   0.0   0.0   0:00.03 top

marvinaichinger@LAPTOP-KOQM5K57: ~
top - 19:05:23 up 26 min, 0 users, load average: 0.52, 0.58, 0.59
Tasks: 7 total, 2 running, 5 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4.1 us, 10.2 sy, 0.0 ni, 85.7 id, 0.0 wa, 0.1 hi, 0.0 si, 0.0 st
MiB Mem : 16150.9 total, 8334.2 free, 7592.8 used, 224.0 buff/cache
MiB Swap: 49152.0 total, 49023.6 free, 128.4 used, 8427.6 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
 177 marvina+  20   0  15276   816   664  R  100.0   0.0   0:05.32 yes
    1 root      20   0   8936   308   264  S   0.0   0.0   0:00.06 init
 105 root      20   0   8936   224   180  S   0.0   0.0   0:00.00 init
 106 marvina+  20   0  18084  3644  3560  S   0.0   0.0   0:00.07 bash
 131 root      20   0   8936   224   180  S   0.0   0.0   0:00.00 init
 132 marvina+  20   0  18216  3868  3760  S   0.0   0.0   0:00.07 bash
 172 marvina+  20   0  18904  2140  1520  R   0.0   0.0   0:00.03 top

marvinaichinger@LAPTOP-KOQM5K57: ~$ yes > /dev/null

```

Der Prozess yes ist fast ständig im Running betrieb, weil er immer wieder eine while-Schleife durchgeht. Wenn die ganzen y's auf /dev/null umgeleitet werden, muss der Prozess nicht einmal darauf warten bis, dass die Ausgabe gemacht wurde, deshalb hat er dann eine CPU-Auslastung von 100%.

### Linux Prozesspriorität:

- Eine Prozesspriorität wird verwendet, um jedem Prozess die passende CPU-Leistung zuzuordnen. Der nice-Level ist ein Wert, der vom User gesetzt werden kann, um die Priorität eines Prozesses zu verändern. Dabei sind werte von -20 bis 19 möglich mit -20 als am meisten wichtig. „Nicht-Root-User“ haben nur die Möglichkeit Werte von 0 bis 19 zu setzten. (Priority = NiceLevel + 20)
- Beim Starten kann mit dem Code „nice -n <value> <command>“ ein Befehl mit einem Nice-Level gestartet werden. (z.B.: nice -n 10 yes)  
Während der Befehl läuft kann mit dem Code „renice -n <value> -p <PID>“ der nice-Level eines Prozesses (PID) verändert werden. (z.B.: renice -n 20 -p 156)
- Wenn nur eine 1-Kern CPU zur Verfügung steht und zwei Prozesse laufen, welche viel CPU-Leistung benötigen (z.B.: yes > /dev/null) dann wird die Leistung gerecht aufgeteilt. Das bedeutet, dass in diesem Fall jeder yes-Befehl 50% der CPU-Leistung bekommt. Falls aber es aus irgendeinem Grund nötig ist, dass ein yes-Befehl mehr Leistung bekommen soll, kann mit einem geringerem Nice-Level dies erreicht werden.