

Prozessmanagement

Table of Contents

1. Prozesszustände in Linux	1
1.1. Suchen und starten Sie einen Prozess, der relativ lange oder permanent läuft, z.B.:	1
2. Starten Sie eine zweite Shell und analysieren Sie die laufenden Prozesse und deren Zustände in Ihrem System	2
3. Halten Sie den langen/permanenten Prozess an und analysieren Sie, welchen Prozesszustand er jetzt aufweist.	3
4. Schicken Sie den Prozess in den Hintergrund. Welcher Prozesszustand wird jetzt angezeigt?	4
4.1. Prozesse automatisch in den Vordergrund oder Hintergrund schicken.	4
5. Holen Sie Ihren Prozess wieder in den Vordergrund und schließen Sie den Prozess in der Shell.	5
6. Diskutieren und begründen Sie, warum ein nicht angehaltener Prozess nicht immer den Status running hat.	5
7. Tool YES	5
8. Recherchieren Sie den Unterschied zwischen einer Prozesspriorität und einem NiceLevel in Linux.	7

1. Prozesszustände in Linux

1.1. Suchen und starten Sie einen Prozess, der relativ lange oder permanent läuft, z.B.:

- Download eines sehr großen Files mit wget

Beispiel: Download mit Linux Mint download

```
root@Stevan:~# wget https://mirrors.layeronline.com/linuxmint/stable/21/linuxmint-21-cinnamon-64bit.iso
```

- Anzeige in top

Prozess wget läuft...

```
top - 19:10:16 up 4 min, 0 users, load average: 0.00, 0.00, 0.00
Tasks: 10 total, 1 running, 9 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.1 us, 0.3 sy, 0.0 ni, 99.4 id, 0.2 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 6874.0 total, 6317.5 free, 304.3 used, 252.1 buff/cache
MiB Swap: 2048.0 total, 2048.0 free, 0.0 used, 6348.8 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
81	root	20	0	14400	6388	5704	S	6.6	0.1	0:00.34	wget
1	root	20	0	2272	1528	1436	S	0.0	0.0	0:00.00	init
4	root	20	0	2272	4	0	S	0.0	0.0	0:00.00	init
7	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
8	root	20	0	2292	104	0	S	0.0	0.0	0:00.01	init
9	root	20	0	10072	4996	3300	S	0.0	0.1	0:00.05	bash
62	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
63	root	20	0	2292	104	0	S	0.0	0.0	0:00.00	init
64	root	20	0	10072	4964	3268	S	0.0	0.1	0:00.03	bash
79	root	20	0	10872	3656	3144	R	0.0	0.1	0:00.00	top

2. Starten Sie eine zweite Shell und analysieren Sie die laufenden Prozesse und deren Zustände in Ihrem System

- `ps -aux | grep nano`

```
root@Stevan:~# ps -aux | grep nano
root      87  0.0  0.0  8572  3752 pts/0    S+   19:13   0:00 nano
root     129  0.0  0.0  8164   652 pts/1    S+   19:15   0:00 grep --color=auto nano
```

- `pstree`

wget-Prozess wird angezeigt

```
root@Stevan:~# pstree
init--init--{init}
      |init--init--bash--wget
      |init--init--bash--pstree
      |--{init}
```

- `top`

```
top - 19:10:16 up 4 min, 0 users, load average: 0.00, 0.00, 0.00
Tasks: 10 total, 1 running, 9 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.1 us, 0.3 sy, 0.0 ni, 99.4 id, 0.2 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 6874.0 total, 6317.5 free, 304.3 used, 252.1 buff/cache
MiB Swap: 2048.0 total, 2048.0 free, 0.0 used, 6348.8 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
81	root	20	0	14400	6388	5704	S	6.6	0.1	0:00.34	wget
1	root	20	0	2272	1528	1436	S	0.0	0.0	0:00.00	init
4	root	20	0	2272	4	0	S	0.0	0.0	0:00.00	init
7	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
8	root	20	0	2292	104	0	S	0.0	0.0	0:00.01	init
9	root	20	0	10072	4996	3300	S	0.0	0.1	0:00.05	bash
62	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
63	root	20	0	2292	104	0	S	0.0	0.0	0:00.00	init
64	root	20	0	10072	4964	3268	S	0.0	0.1	0:00.03	bash
79	root	20	0	10872	3656	3144	R	0.0	0.1	0:00.00	top

3. Halten Sie den langen/permanenten Prozess an und analysieren Sie, welchen Prozesszustand er jetzt aufweist

- Der Prozess wird mit STRG+Z angehalten

```
linuxmint-21-cinnamon-64bit.i 1%[ ] 28.88M 8.55MB/s eta 4m 35s ^
Z
[1]+ Stopped wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso
```

- Verhalten bei angehaltenem Prozess

Unter **[wget]** kann man sehen, dass der Prozess wirklich angehalten wurde. Die Laufzeit wurde gestoppt.

```
top - 19:22:08 up 16 min, 0 users, load average: 0.02, 0.02, 0.00
Tasks: 10 total, 1 running, 8 sleeping, 1 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 6874.0 total, 3794.0 free, 336.1 used, 2743.8 buff/cache
MiB Swap: 2048.0 total, 2048.0 free, 0.0 used, 6283.1 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	2272	1528	1436	S	0.0	0.0	0:00.00	init
4	root	20	0	2272	4	0	S	0.0	0.0	0:00.00	init
7	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
8	root	20	0	2292	104	0	S	0.0	0.0	0:00.10	init
9	root	20	0	10072	5000	3300	S	0.0	0.1	0:00.05	bash
109	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
110	root	20	0	2292	104	0	S	0.0	0.0	0:00.00	init
111	root	20	0	10072	4960	3256	S	0.0	0.1	0:00.04	bash
140	root	20	0	14400	6244	5560	T	0.0	0.1	0:00.25	wget
141	root	20	0	10872	3704	3188	R	0.0	0.1	0:00.00	top

4. Schicken Sie den Prozess in den Hintergrund. Welcher Prozesszustand wird jetzt angezeigt?

- Ein Prozess kann mit **[bg wget ...]** in den Hintergrund geschickt werden

```
root@Stevan:~# bg wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso
[1]+  wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso &
-bash: bg: https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso: no such job
root@Stevan:~#
Redirecting output to 'wget-log'.

root@Stevan:~# bg wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso
-bash: bg: job 1 already in background
```

- Ein Prozess wird mit **[&]** sofort in den Hintergrund geschickt:

```
wget -nv http://download.example.org/linux_image_1.iso &
```

Prozesse, die im Hintergrund laufen benutzen, immer noch die Standard-ein und Ausgabe der Shell.

Prozesse im Hintergrund laufen zu lassen ist nur sinnvoll, wenn der Prozess lange braucht(Downloads,...).

- Ansicht in der top

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
140	root	20	0	14400	6244	5560	S	4.3	0.1	0:02.83	wget
1	root	20	0	2272	1528	1436	S	0.0	0.0	0:00.00	init
4	root	20	0	2272	4	0	S	0.0	0.0	0:00.00	init
7	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
8	root	20	0	2292	104	0	S	0.0	0.0	0:00.14	init
9	root	20	0	10072	5000	3300	S	0.0	0.1	0:00.07	bash
109	root	20	0	2276	100	0	S	0.0	0.0	0:00.00	init
110	root	20	0	2292	104	0	S	0.0	0.0	0:00.00	init
111	root	20	0	10072	4960	3256	S	0.0	0.1	0:00.04	bash
143	root	20	0	10872	3772	3260	R	0.0	0.1	0:00.00	top

- jobs

Mit dem Command **[jobs]** lassen sich die derzeitig laufenden Prozesse nachvollziehen

```
root@Stevan:~# jobs
[1]+  Running                  wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso &
```

4.1. Prozesse automatisch in den Vordergrund oder Hintergrund schicken

- Vordergrund
 - **fg %id**

- Hintergrund
 - `bg %id`

5. Holen Sie Ihren Prozess wieder in den Vordergrund und schließen Sie den Prozess in der Shell.

Prozess wird in den Vordergrund geschoben

```
root@Stevan:~# fg %1
wget https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso
--2022-12-11 19:45:54-- https://muug.ca/mirror/linuxmint/iso/stable/21/linuxmint-21-cinnamon-64bit.iso
Resolving muug.ca (muug.ca)... 208.81.1.244, 185.159.196.2, 198.182.167.1, ...
Connecting to muug.ca (muug.ca)|208.81.1.244|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2449342464 (2.3G) [application/octet-stream]
Saving to: 'linuxmint-21-cinnamon-64bit.iso'

linuxmint-21-cinnamon-64bit.i  7%[==>          ] 165.90M  9.33MB/s   eta 4m 57s |
```

6. Diskutieren und begründen Sie, warum ein nicht angehaltener Prozess nicht immer den Status running hat.

Da ein anderer Prozess[B] eine höhere Priorität als der Prozess[A] hat. Der Prozess[B] wird abgearbeitet, danach bekommt Prozess[A] den Status running.

7. Tool YES

`yes y`

Prozessorauslastung & Zustände

- Ohne Umleitung

```

1 [ 0.0%] 5 [||| 4.1%]
2 [|| 2.0%] 6 [ 0.0%]
3 [| 0.7%] 7 [ 0.0%]
4 [ 0.0%] 8 [ 0.0%]
Mem[||||| 400M/6.71G] Tasks: 10, 2 thr; 1 running
Swp[ 0K/2.00G] Load average: 0.27 0.16 0.06
Uptime: 00:53:48

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
224 root 20 0 7232 580 512 S 7.4 0.0 0:00.97 yes y
8 root 20 0 2292 104 0 S 0.7 0.0 0:01.14 /init
6 root 20 0 2272 1528 1436 S 0.0 0.0 0:00.00 /init
1 root 20 0 2272 1528 1436 S 0.0 0.0 0:00.00 /init
5 root 20 0 2288 68 68 S 0.0 0.0 0:00.00 plan9 --control-socket 5 --log-level 4 --server-fd 6 --pi
4 root 20 0 2288 68 68 S 0.0 0.0 0:00.00 plan9 --control-socket 5 --log-level 4 --server-fd 6 --pi
7 root 20 0 2276 100 0 S 0.0 0.0 0:00.00 /init
9 root 20 0 10720 5804 3440 S 0.0 0.1 0:00.71 -bash
109 root 20 0 2276 100 0 S 0.0 0.0 0:00.00 /init
110 root 20 0 2292 104 0 S 0.0 0.0 0:00.02 /init
111 root 20 0 10072 4960 3256 S 0.0 0.1 0:00.05 -bash
225 root 20 0 8156 3728 3068 R 0.0 0.1 0:00.01 htop

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit

```

- Mit Umleitung

Der 4te Prozessorkern wird extremst belastet

```

1 [ 0.0%] 5 [ 0.0%]
2 [ 0.0%] 6 [ 0.0%]
3 [ 0.0%] 7 [ 0.0%]
4 [||||||||||||||||||||||||||||||||||||| 100.0%] 8 [ 0.0%]
Mem[||||| 405M/6.71G] Tasks: 10, 2 thr; 2 running
Swp[ 0K/2.00G] Load average: 0.11 0.11 0.05
Uptime: 00:56:06

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
226 root 20 0 7232 516 448 R 100.0 0.0 0:08.15 yes y
6 root 20 0 2272 1528 1436 S 0.0 0.0 0:00.00 /init
1 root 20 0 2272 1528 1436 S 0.0 0.0 0:00.00 /init
5 root 20 0 2288 68 68 S 0.0 0.0 0:00.00 plan9 --control-socket 5 --log-level 4 --server-fd 6 --pi
4 root 20 0 2288 68 68 S 0.0 0.0 0:00.00 plan9 --control-socket 5 --log-level 4 --server-fd 6 --pi
7 root 20 0 2276 100 0 S 0.0 0.0 0:00.00 /init
8 root 20 0 2292 104 0 S 0.0 0.0 0:01.48 /init
9 root 20 0 10720 5804 3440 S 0.0 0.1 0:00.71 -bash
109 root 20 0 2276 100 0 S 0.0 0.0 0:00.00 /init
110 root 20 0 2292 104 0 S 0.0 0.0 0:00.03 /init
111 root 20 0 10072 4960 3256 S 0.0 0.1 0:00.05 -bash
227 root 20 0 8156 3672 3012 R 0.0 0.1 0:00.00 htop

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit

```

Die Prozessorauslastung ist so hoch da das dev-Verzeichnis ein besonderes ist. Das Verzeichnis /dev/ besteht aus Dateien, die Geräte darstellen, die an das lokale System angeschlossen sind.

Das /dev/null "Verzeichnis" ist eine Art Vakuum, welches alle Schreibvorgänge verwirft, also zu 0 bytes macht.

`dd > /dev/null 500MB` werden auf vom Prozessor auf 0 Bytes geschrieben

Der Prozessor wird dementsprechend durch das dauerhafte umleiten und verwerfen überfordert, da zwei Prozesse unheimlich schnell aufeinander treffen.

8. Recherchieren Sie den Unterschied zwischen einer Prozesspriorität und einem NiceLevel in Linux.

Der Nice-wert gibt die Prozesspriorität an.

Ein Prozess ist dann "nice", wenn dieser weniger Rechenzeit verbraucht und anderen Prozessen mehr Rechenzeit lässt.

Prozesse ohne spezielle Handhabung haben eine normale Priorität, das heißt sie haben einen nice-wert von 0.

- Die Nice-Priorität kann beim Start vom Prozess mit verändert werden:

```
nice -15 foo
```

Je höher der nice-wert, desto höher die Priorität

- Mit dem Tool re-nice oder mit top lassen sich Prozessprioritäten während des Laufens verändern

```
sudo renice nice-value pid
```

```
sudo renice 6 3244
```

- Praktisches Beispiel für Prozesspriorisierung

Beispiel:

Man besitzt einen Leistungsschwachen PC und muss eine schwierige Primzahl berechnung in 2h fertig bekommen. Nebenbei möchte man sich aber ein Video herunterladen. Das Herunterladen des Videos hat aber eine höhere Prozesspriorität, daher bleibt weniger Leistung für die Berechnung übrig.

Hierbei kann man das Tool renice verwenden, um die Priorisierung zu ändern. Nun wird hat das Berechnen eine höhere priorität und wird zuerst bearbeitet.