

# Virtual Memory with Linux

([Usage hints](#) for this presentation)

IT Systems, Summer Term 2024

Dr. Jens Lechtenbörger ([License Information](#))

Chair of Data Science: Machine Learning and Data Engineering (Prof. Gieseke)  
Dept. of Information Systems

# 1. Looking at Memory with Linux

(Specifics of Linux are not part of learning objectives. However, the following illustrates shared memory, and the pseudo-filesystem `/proc` will be revisited in other presentations.)

# 1.1. Linux Kernel: `/proc/<pid>/`

- `/proc` is a pseudo-filesystem
  - See <https://man7.org/linux/man-pages/man5/proc.5.html>
    - (Specific to Linux kernel; incomplete or missing elsewhere)
  - “Pseudo”: Look and feel of any other filesystem
    - Subdirectories and files
    - However, files are no “real” files but meta-data
  - Interface to internal **kernel data structures**
    - One subdirectory per process ID
    - OS identifies process by integer number
    - Here and elsewhere, `<pid>` is meant as **placeholder** for such a number

# 1.1.1. Video about `/proc`



## Speaker notes

This video, “Looking at /proc” by [Jens Lechtenbörger](#), shares the presentation’s license terms, namely [CC BY-SA 4.0](#).

The video shows some aspects of the `/proc` filesystem related to memory management, which are described in more abstract form on subsequent slides.

# 1.1.2. Drawing about /proc

an amazing directory: /proc <sup>JULIA EVANS @b0rk</sup>

Every process on Linux has a PID (process ID) like 42.  In /proc/42, there's a lot of VERY USEFUL information about process 42	<b>/proc/PID/cmdline</b> command line arguments the process was started with	<b>/proc/PID/exe</b> symlink to the process's binary magic: works even if the binary has been deleted!
	<b>/proc/PID/envIRON</b> all of the process's environment variables	<b>/proc/PID/status</b> Is the program running or asleep? How much memory is it using? And much more!
<b>/proc/PID/fd</b> Directory with every file the process has open! Run <code>\$ls -l /proc/42/fd</code> to see the list of files for process 42.  These symlinks are also magic & you can use them to recover deleted files ♥	<b>/proc/PID/stack</b> The kernel's current stack for the process. Useful if it's stuck in a system call	and <b>more</b> :D Look at <b>man proc</b>  for more information!
	<b>/proc/PID/maps</b> List of process's memory maps. Shared libraries, heap, anonymous maps, etc.	

/proc

Figure © 2018 Julia Evans, all rights reserved; from julia's drawings. Displayed here with personal permission.

# 1.1.3. Drawing about man pages

man pages = awesome  
(sometimes. Quality may vary ☹)

JULIA EVANS  
@b0rk

I found out I can get documentation for programs (like grep) with **man grep**!

but that's not all!! lots of other things have man pages too!

man pages are split up into 8 sections

① ② ③ ④ ⑤ ⑥ ⑦ ⑧

/usr/share/man/man5 has section 5 on my machine.

GREAT →

- ① programs  
\$man grep  
\$man ls
- ② system calls  
\$man sendfile
- ③ C functions  
\$man 3 printf  
\$man fopen
- ④ devices  
\$man null  
for /dev/null docs
- ⑤ file formats  
\$man sudoers  
for /etc/sudoers  
\$man proc
- ⑥ games  
(not very useful)  
man sl is good if you have sl though
- ⑦ miscellaneous  
\$man 7 pipe  
\$man 7 symlink  
(these are cool!)
- ⑧ sysadmin programs  
\$man apt  
\$man chroot

Man pages are amazing

Figure © 2016 Julia Evans, all rights reserved; from julia's drawings.  
Displayed here with personal permission.

## 1.2. Linux Kernel Memory Interface

- Memory allocation (and much more) visible under `/proc/<pid>`
- E.g.:
  - `/proc/<pid>/pagemap`: One 64-bit value per virtual page
    - Mapping to RAM or swap area
  - `/proc/<pid>/maps`: Mapped memory regions
  - `/proc/<pid>/smaps`: Memory usage for mapped regions
- Notice: Memory regions include **shared** libraries that are used by lots of processes



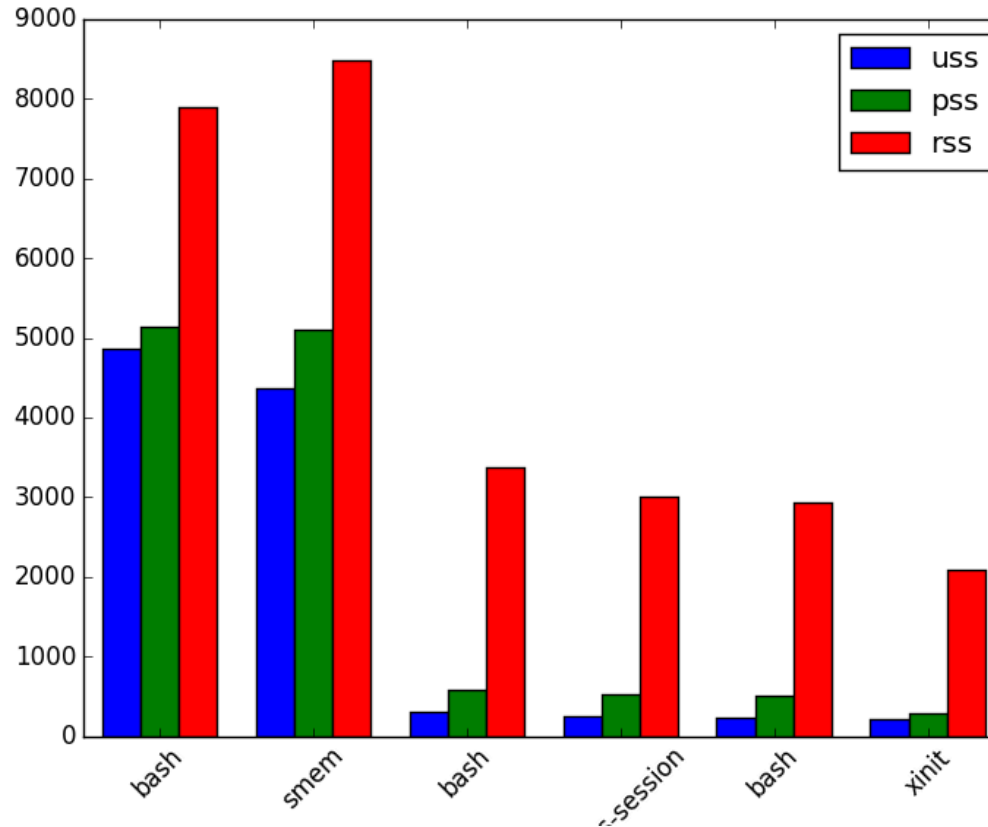
# 1.3. GNU/Linux Reporting: smem

- User space tool to read smaps files: smem
  - See <https://linoxide.com/memory-usage-reporting-smem/>
- Terminology
  - **Virtual set size** (VSS): Size of virtual address space
  - **Resident set size** (RSS): Allocated main memory
    - Standard notion, yet overestimates memory usage as lots of memory is shared between processes
      - Shared memory is added to the RSS of every sharing process
  - **Unique set size** (USS): memory allocated exclusively to process
    - That much would be returned upon process' termination
  - **Proportional set size** (PSS): USS plus “fair share” of shared pages
    - If page shared by 5 processes, each gets a fifth of a page added to its PSS

## 1.3.1. Sample smem Output

```
$ smem -c "pid command uss pss rss vss" -P "bash|xinit|emacs"
PID Command                USS      PSS      RSS      VS
765 /usr/bin/xinit /etc/X11/Xse 220      285     2084    1595
1390 /bin/bash -c libreoffice5.3 240      510     2936    1318
826 /bin/bash /usr/bin/qubes-se 256      524     3008    1320
750 -su -c /usr/bin/xinit /etc/ 316      587     3368    2163
1251 bash                    4864     5136     7900    2602
2288 /usr/bin/python /usr/bin/sm 5272     6035     9432    2468
1145 emacs                    90876    93224    106568  66276
```

# 1.3.2. Sample smem Graph



```
smem --bar pid -c "uss pss rss" -P "bash|xinit"
```

“Screenshot of smem” under CC0 1.0; from GitLab

# License Information

Source files are available on GitLab (check out embedded submodules) [↗](#) under free licenses [↗](#). Icons of custom controls are by [@fontawesome](#) [↗](#), released under CC BY 4.0 [↗](#).

Except where otherwise noted, the work “Virtual Memory with Linux”, © 2017-2022, 2024 [Jens Lechtenbörger](#) [↗](#), is published under the [Creative Commons](#) license CC BY-SA 4.0 [↗](#).

## Speaker notes

This presentation is distributed as Open Educational Resource under freedom granting license terms.

