Betriebssysteme

12. Tutorium - Files and Directories

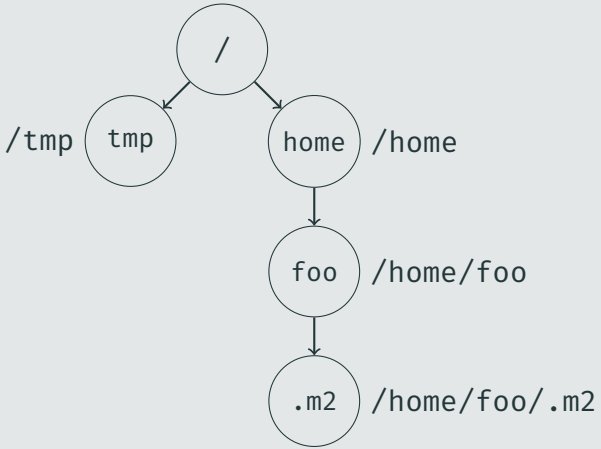
Peter Bohner

31. Januar 2024

ITEC - Operating Systems Group

Files And Directories

Paths (Linux)



Reading A File

What are the two basic access methods (patterns) for reading a file?

- **Sequential Access**
  Accessed in order, reading sequential bytes. Writes append at the end.
- **Random Access**
  Reading and writing at arbitrary positions, programmer needs to specify where to write/read

Which Library Functions and System Calls do we have to access files?

Opening the file

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

`fread` / `read`

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

`fread` / `read`

Writing to a file

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

`fread` / `read`

Writing to a file

`fwrite` / `write`

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

`fread` / `read`

Writing to a file

`fwrite` / `write`

Flushing dirty buffers

Which Library Functions and System Calls do we have to access files?

Opening the file

`fopen` / `open`. Provide mode as argument (read/write, where, create, etc.)

Closing a file

`fclose` / `close`

Reading from a file

`fread` / `read`

Writing to a file

`fwrite` / `write`

Flushing dirty buffers

The library functions sometimes buffer to reduce the amount of syscalls.
`fflush` flushes those buffers.

Random Access

Which system calls / library functions move the „current position pointer“?

Random Access

Which system calls / library functions move the „current position pointer“?

- Linux: `fseek` / `lseek`
- Windows: `SetFilePointerEx` / `SetFilePointer`

Random Access

Which system calls / library functions move the „current position pointer“?

- Linux: `fseek` / `lseek`
- Windows: `SetFilePointerEx` / `SetFilePointer`

What happens when you move the cursor behind the end of the file?

Random Access

Which system calls / library functions move the „current position pointer“?

- Linux: `fseek` / `lseek`
- Windows: `SetFilePointerEx` / `SetFilePointer`

What happens when you move the cursor behind the end of the file?

It creates a hole filled with zeros!

Random Access

Which system calls / library functions move the „current position pointer“?

- Linux: `fseek` / `lseek`
- Windows: `SetFilePointerEx` / `SetFilePointer`

What happens when you move the cursor behind the end of the file?

It creates a hole filled with zeros! Such a file is called a *sparse* file and some file systems might not store empty regions.

Random Access

How could you implement random access without a dedicated file pointer?

Add an offset to the `read` and `write` system call.

Random Access

How could you implement random access without a dedicated file pointer?

Add an offset to the `read` and `write` system call.

+ Save a system call

Random Access

How could you implement random access without a dedicated file pointer?

Add an offset to the `read` and `write` system call.

- \+ Save a system call
- \- Caller need to keep track of the current offset for sequential reads

Random Access

How could you implement random access without a dedicated file pointer?

Add an offset to the `read` and `write` system call.

+ Save a system call
- Caller need to keep track of the current offset for sequential reads

`mmap` the file (works quite well, might cause page faults. Probably faster for large files than read/write calls)

Walking a directory

What system calls do you need to list files in a Linux directory?

Walking a directory

What system calls do you need to list files in a Linux directory?

1. `opendir`

Walking a directory

What system calls do you need to list files in a Linux directory?

1. `opendir`
2. `readdir`:

Walking a directory

What system calls do you need to list files in a Linux directory?

1. `opendir`
2. `readdir`: Returns a `dirent` with a *relative path*

Walking a directory

What system calls do you need to list files in a Linux directory?

1. `opendir`
2. `readdir`: Returns a `dirent` with a *relative path*
3. `closedir`

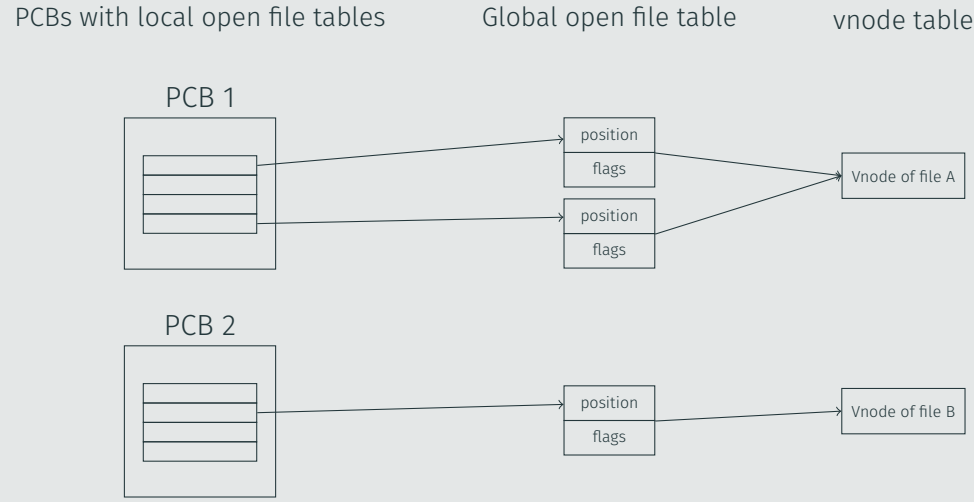
Open Files

Kernel Data Structures For Open Files

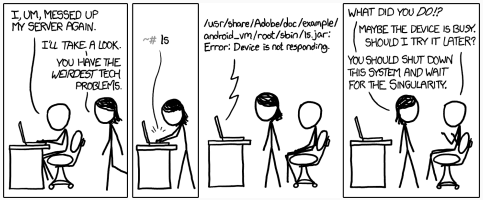
What structures does the kernel use for open files?

Kernel Data Structures For Open Files

What structures does the kernel use for open files?



Ende



XKCD 1084 - Server Problem

F R A G E N?

https://forms.gle/9CwJSKidKibubran9

Bis nächste Woche