DM510: Recap and Exam Preparation

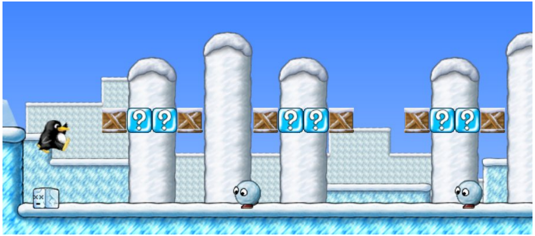
Lars Rohwedder

Disclaimer

These slides contain (modified) content and media from the official Operating System Concepts slides: https://www.os-book.com/OS10/slide-dir/index.html

Today's lecture

- Q & A
- Operating Systems Speedrun and quiz
- Course Evaluation (during break)



Looking back . . . and forward

What you have learned:

- Major concepts of operating systems (textbook chapters) ✓
- Linux & terminal skills ✓
- System/kernel programming skills (programming projects) ✓

Looking back . . . and forward

What you have learned:

- Major concepts of operating systems (textbook chapters) ✓ exam relevant
- Linux & terminal skills ✓ not exam relevant
- System/kernel programming skills (programming projects) ✓ a bit exam relevant

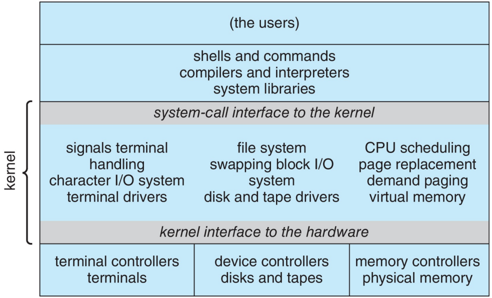
Exam

- 180 minutes
- digital
- closed book
- 5-6 main questions (each on different topic, with subquestions)
- one question testing rudimantary understanding of programming projects

Speedrun & QUIZ

System services

- The services that an operating system provides
- How these services are accessed (system call interface), experienced also in project
- Several internal architectures of operating systems



(the users)

shells and commands
compilers and interpreters
system libraries

system-call interface to the kernel

signals terminal handling
character I/O system
terminal drivers

file system
swapping block I/O system
disk and tape drivers

CPU scheduling
page replacement
demand paging
virtual memory

kernel interface to the hardware

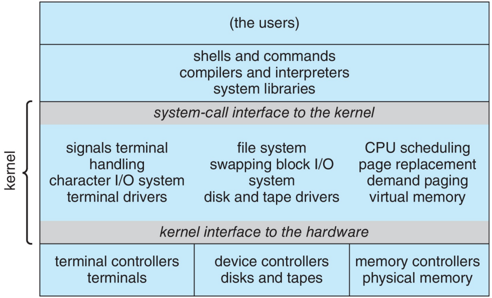
terminal controllers
terminals

device controllers
disks and tapes

memory controllers
physical memory

System services

- The services that an operating system provides
- How these services are accessed (system call interface), experienced also in project
- Several internal architectures of operating systems



(the users)

shells and commands
compilers and interpreters
system libraries

system-call interface to the kernel

signals terminal handling
character I/O system
terminal drivers

file system
swapping block I/O system
disk and tape drivers

CPU scheduling
page replacement
demand paging
virtual memory

kernel interface to the hardware

terminal controllers
terminals

device controllers
disks and tapes

memory controllers
physical memory

??? QUIZ ???

- Compare making a call to the shared library (e.g. `sqrt()` in cmath) to a system call. Internally, what are the differences?
- For convenience, a system call usually has a wrapper function in a shared library. Why isn't the entire implementation of the system call in the shared library?

Processes

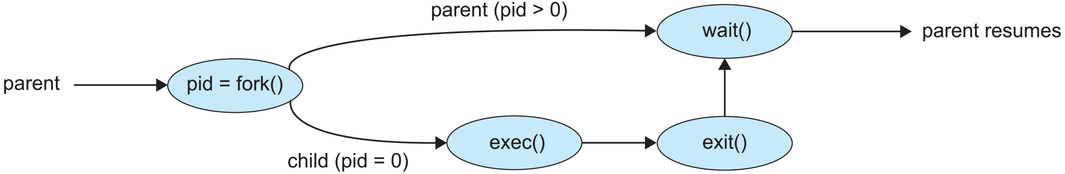
- Process states, data, and context switches
- Process creation and termination
- Shared memory and message passing

Processes

- Process states, data, and context switches
- Process creation and termination
- Shared memory and message passing

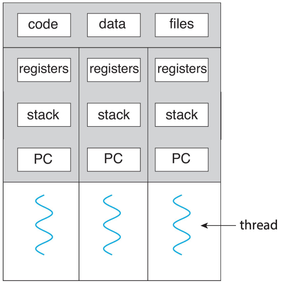
??? QUIZ ???

The image below depicts how a terminal process executes a program (synchronously). Since there is no parallelism anyway, wouldn't it be better (more efficient) to call `exec()` directly without creating an extra process? Explain why this is not viable.



Threads

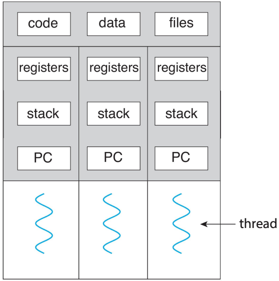
- Data and task parallelism, Amdahl's law
- User and kernel threads
- Implicit multi-threading



multithreaded process

Threads

- Data and task parallelism, Amdahl's law
- User and kernel threads
- Implicit multi-threading



multithreaded process

??? QUIZ ???

The interpreter of a high-level programming language could have information about the upcoming instructions of a thread. Why could this be a reason for user threads managed by the interpreter as opposed to using kernel threads?
Hint: think about the advantages to CPU scheduling.

CPU scheduling

- Scheduling criteria
- Preemptive, non-preemptive scheduling
- Various examples of scheduling algorithms
- Multi-core and realtime scheduling

Synchronization

- Race conditions
- Mechanisms: critical section, mutexes, semaphores, monitors
- Implementation: atomic instructions, spinlocks, block/wakeup system calls

Synchronization

- Race conditions
- Mechanisms: critical section, mutexes, semaphores, monitors
- Implementation: atomic instructions, spinlocks, block/wakeup system calls

??? QUIZ ???

The following code shows one thread writing elements into a shared buffer and once the buffer is full, the other process consumes all of them at once. For the consumer there are two alternatives given (`thread2()` and `thread2_alt()`).
Explain for both alternatives whether they exhibit race conditions when used together with `thread1()`.

Deadlocks

- Necessary conditions for deadlocks
- Deadlock prevention
- Deadlock avoidance (Banker's algorithm)
- Deadlock recovery

Deadlocks

- Necessary conditions for deadlocks
- Deadlock prevention
- Deadlock avoidance (Banker's algorithm)
- Deadlock recovery

??? QUIZ ???

Can the following code lead to a deadlock?

- If not, which necessary condition for deadlocks is not satisfied?
- If yes, explain why and propose a modification that prevents the deadlock. Which necessary condition is not satisfied in your modification?

Main memory

- Physical, logical addresses
- Contiguous allocation
- Paging: page tables, swapping, page faults, page replacement

Main memory

- Physical, logical addresses
- Contiguous allocation
- Paging: page tables, swapping, page faults, page replacement

??? QUIZ ???

Shared library code is only loaded into main memory once, even if several processes use it. Explain the functionality of the read-only bit in the page table and why it is crucial to the aforementioned optimization.

Mass storage

- HDD and NVM
- HDD scheduling
- RAID

Mass storage

- HDD and NVM
- HDD scheduling
- RAID

??? QUIZ ???

- Explain the purpose(s) of RAID
- Consider four equal sized HDDs configured either with RAID 1 (mirrored disks) or RAID 4 (block-interleaved parity). Which one has the higher risk of data loss? Which one has higher storage capacity? Explain your answer.

I/O systems and networks

- Memory-mapped I/O, direct memory access, device drivers
- Programming project on driver kernel module
- Network protocols: Ethernet, IP, TCP, UDP

I/O systems and networks

- Memory-mapped I/O, direct memory access, device drivers
- Programming project on driver kernel module
- Network protocols: Ethernet, IP, TCP, UDP

??? QUIZ ???

Should the TCP protocol be implemented in hardware, driver, or user space? Justify your answer.

File systems

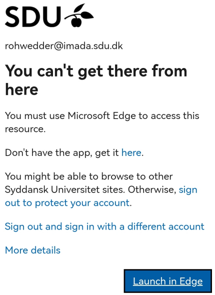
- Files, directory structures, file system mounting
- Implementation: inodes, block allocation, recovery from failure
- Facilitated in programming project

Security and protection

- Security violations,
  four layers of security
- Malicious programs
- Cryptography
- Protection

Security and protection

- Security violations, four layers of security
- Malicious programs
- Cryptography
- Protection



SDU

rohwedder@imada.sdu.dk

You can't get there from here

You must use Microsoft Edge to access this resource.

Don't have the app, get it here.

You might be able to browse to other Syddansk Universitet sites. Otherwise, sign out to protect your account.

Sign out and sign in with a different account

More details

??? QUIZ ???

HTTP requests include information about the browser in the `user-agent` header. For example:

User-Agent: Mozilla/5.0 (iPad; U; CPU OS 3_2_1 like Mac OS X; en-us)
AppleWebKit/531.21.10 (KHTML, like Gecko) Mobile/7B405

Out of “security concerns”, Microsoft wants to limit access to some websites to users of their own browser (Edge). This is done using the user-agent header.

- Explain the masquerading attack and how Microsoft’s approach is vulnerable against it.
- Microsoft could distribute a private key together with the Edge software. Would this solve the vulnerability? If yes, explain how. If no, explain the problems.

Virtual machines

- Type-0, type-1, type-2 hypervisors
- Trap-and-emulate, binary translation
- Containerization