Solution to the Fall 2012 First COSC 6360 Quiz

Jehan-François Pâris
jfparis@uh.edu

UNIX Portability

- Which feature of UNIX made it portable? (10 points)
  - *It was written in a high-level language.*

Soft links (I)

- What are ***UNIX soft links*** ? (10 points)
  - *UNIX soft links are special entities within the file system that point to other files, much like Windows shortcuts.*
  - *They are also called **symbolic links**.*

Soft links (II)

- What is the main reason for having them? (10 points)
  - *They can cross disk partition boundaries, which other links cannot cross.*

VMS

- What was the ***main advantage*** of the VMS page replacement policy over that of Berkeley UNIX? (10 points)

  - *It supported real-time processes because it could allocate a specific number of page frames to any process.*

Page replacement policies (I)

- Why did Babaoğlu and Joy decide ***not*** to use the Sampled Working Set page replacement policy for their implementation of UNIX? (10 points)
  - *That policy required resetting the page referenced bit of all page frames of a process every T milliseconds. Since that operation had to be simulated, this would have caused too many context switches.*

Page replacement policies (II)

- Why did Babaoğlu and Joy decide ***not*** to use the VMS page replacement policy for their implementation of UNIX? (10 points)

  - *It was too difficult to ascertain the right number of page frames to allocate to any given process*.

UNIX file system (I)

- Recall that FFS i-nodes have a fifteenth block address that is never used. Assuming a block size of X KB,
  - What would we gain by using this fifteenth block address to store one extra direct block address? (10 points)
    - *We would be able to access X extra kilobytes directly from the i-node.*

UNIX file system (II)

- Recall that FFS i-nodes have a fifteenth block address that is never used. Assuming a block size of X KB,
  - What would we gain by using this fifteenth block address to store a second single indirect block address? (10 points)
    - *We would be able to **double** the number of blocks that could be accessed with one level of indirection*

    —

UNIX file system (III)

- Which of these two options would you recommend and why? (10 points)

  - *Doubling the number of blocks that can be accessed with one level of indirection is more important than allowing direct access to four extra kilobytes.*

Combining fork() and exec() (I)

- Consider a proposed variant of the UNIX system that would combine the UNIX **fork( )** and **exec( )** into a single system call like MS Windows **CreateProcess( )**.

Combining fork() and exec() (II)

- What would be the main advantage of this approach? (10 *easy* points)

  - *In addition to eliminating two context switches, we would eliminate the wasteful practice of copying the contents of the parent process address space into the child process address space.*

Combining fork() and exec() (III)

- Which UNIX features would have to be completely reimplemented? (10 *less obvious* points)
- *Pipes and all kinds of I/O redirection would have to be completely reimplemented because the parent program would lose the access lose control to the child process from the moment the child process is created.*

UNIX special files (I)

- What are ***UNIX special files*** ? (10 points)
  - *UNIX special files are not files but physical storage devices, such as floppy drives, flash drives and so on.*

UNIX special files (II)

- What is the main reason for having them? (10 points)
  - *Giving file names to these devices allows programmers to access these devices as if they were regular files.*