CLOUD COMPUTING APPLICATIONS

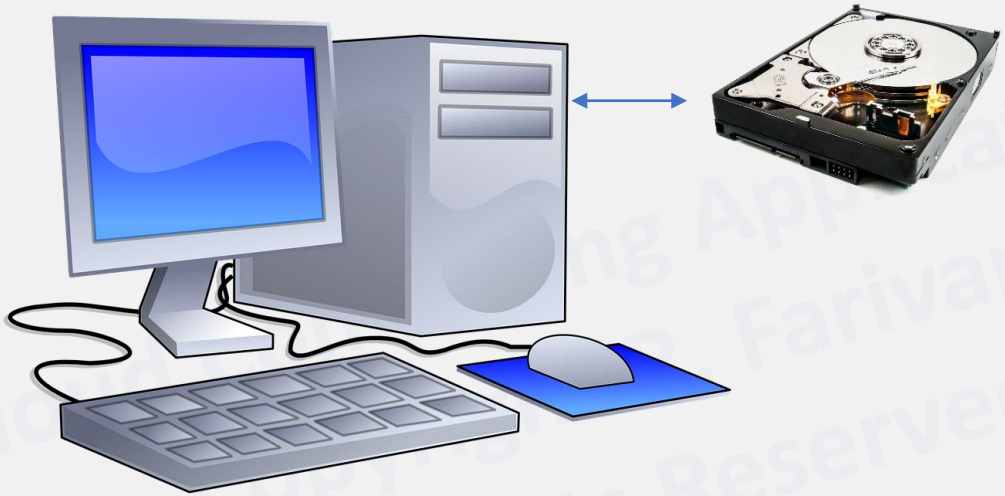
Cloud Storage: Basics

Prof. Reza Farivar

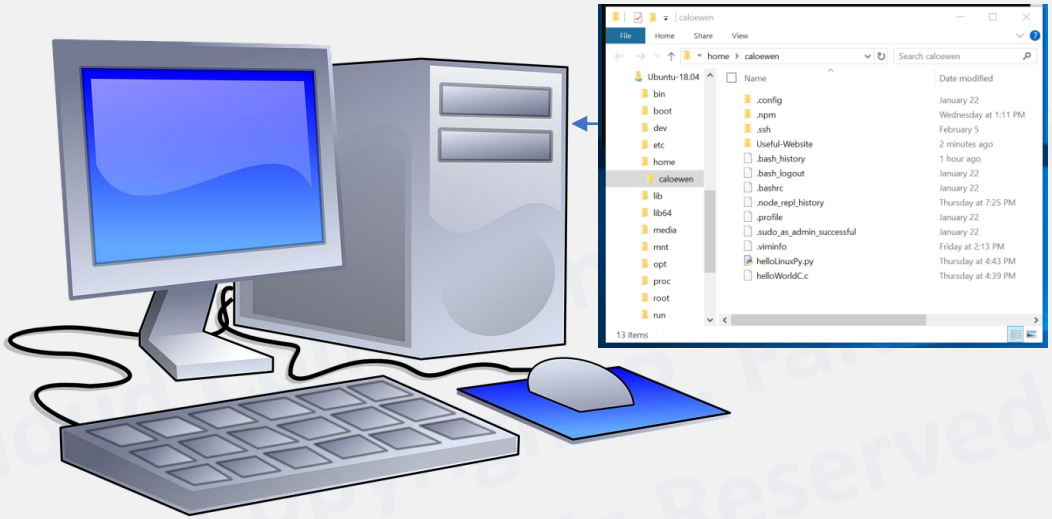
Cloud-based Storage

- Data Storage was one of the first use cases for Cloud Computing
- The competition is very fierce
- The cloud storage models have solidified into a few main categories
- Many different types of cloud storage, sometimes confusing
- We will cover the main categories in this lesson

Back to Basics: Block Storage



File System

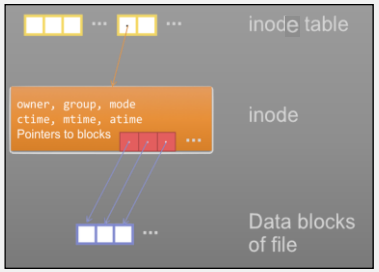


File System

- The middleware between the physical storage device and programs running on top of the OS
  - Block Storage
  - Typically three layers
    - Physical file system
      - Processes physical blocks being read or written
      - Handles buffering and memory management
      - Physical placement of blocks in specific locations on the storage medium
      - Interacts with the device drivers
    - Virtual file system
      - Support for multiple concurrent instances of physical file systems, each of which is called a file system implementation
    - Logical file system
      - File access, directory operations, [and] security and protection
      - Consistency Guarantees

File System

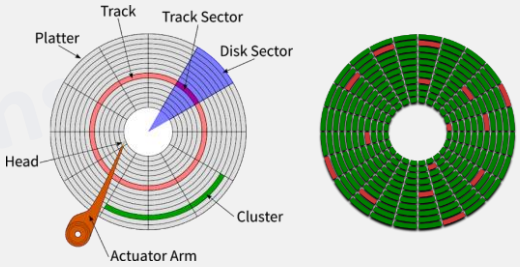
- File systems contain *metadata*
  - There is an *inode* for each file or directory, providing:
    - Locations of the data blocks* for the file
    - Attributes about the file, like “time last accessed” or “owner of the file”
  - The *inode table* records where each *inode* is located, indexed by number.
  - *Directories* are special files listing the names and *inode* numbers of files under the folder.



Internally the data content of a file is stored as a sequence of file system blocks

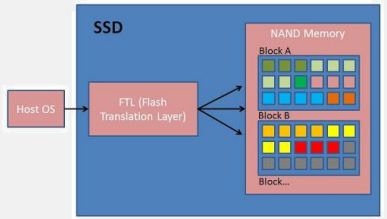
Block Storage

- Internally the data content of a file is stored as a sequence of file system *blocks*
  - Each block is a fixed number of bytes. The last block might not be full.
  - Within a block, all the bytes are sequential. However, within a file, the blocks might not reside sequentially on the disk.
  - Underlying storage systems are usually organized as blocks, and ideally file system blocks are aligned with the storage system's blocks.



Block Storage

- Internally the data content of a file is stored as a sequence of file system *blocks*
  - Each block is a fixed number of bytes. The last block might not be full.
  - Within a block, all the bytes are sequential. However, within a file, the blocks might not reside sequentially on the disk.
  - Underlying storage systems are usually organized as blocks, and ideally file system blocks are aligned with the storage system's blocks.



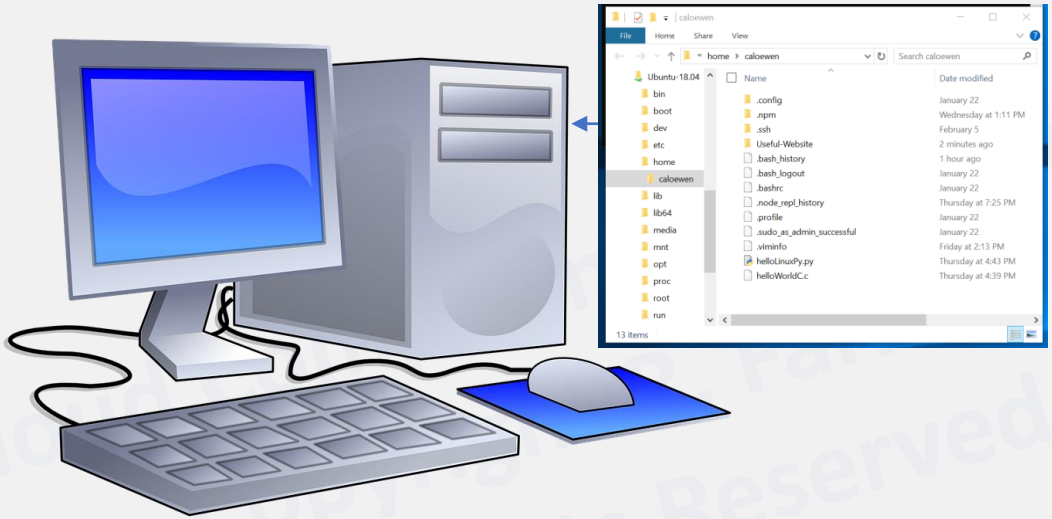
POSIX File System: IEEE Std 1003.1-2017

- Set of operations that the file system has to support
- POSIX file systems are organized as *directories* (folders) containing *files* (documents)
- POSIX file systems are *logically* treated as a sequence of bytes
- POSIX programming API
  - mount, umount the file system
  - open, close file descriptor
  - seek, lseek, fseek to a position in the byte stream
  - write, read to/from file descriptor
  - mkdir, rmdir, creat, unlink, link, symlink
  - fcntl (byte range locks, etc.)
  - stat, utimes, chmod, chown, chgrp
  - Path names are case sensitive, components are separated with forward slash (/).

POSIX Semantics

- POSIX Semantics
  - Define what is and is not guaranteed to happen when a POSIX I/O API call is made.
- *Example:*
  - *After a write() to a regular file has successfully returned:*
    - *Any successful read() from each byte position in the file that was modified by that write shall return the data specified by the write() for that position until such byte positions are again modified.*
    - *Any subsequent successful write() to the same byte position in the file shall overwrite that file data.*
  - Write() is strongly consistent
    - A write() is required to block application execution until the system can guarantee that any other read() call will see the data that was just written

File System



Summary

- Overview of block storage model
- File System Layers
  - Physical file system
  - Virtual file system
  - Logical file system
- POSIX File System
  - API
  - Semantics