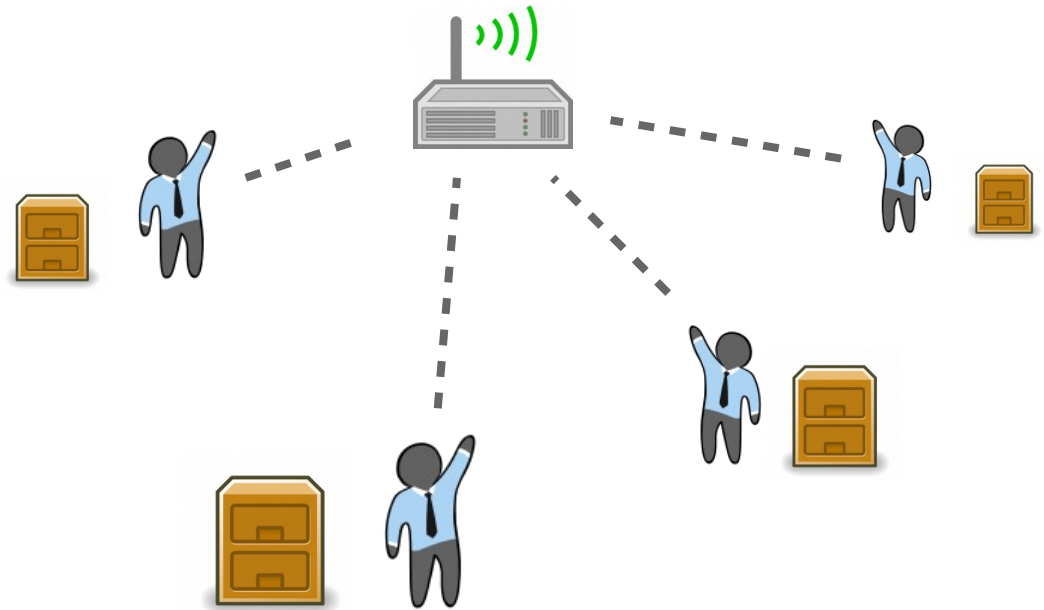


CS 470 Spring 2026

Mike Lam, Professor



Distributed Web and File Systems

A.K.A. "alternative emphases of CS 470"

Content taken from the following:

"Distributed Systems: Principles and Paradigms" by Andrew S. Tanenbaum and Maarten Van Steen (Chapters 11 and 12)

Various online sources

Distributed systems

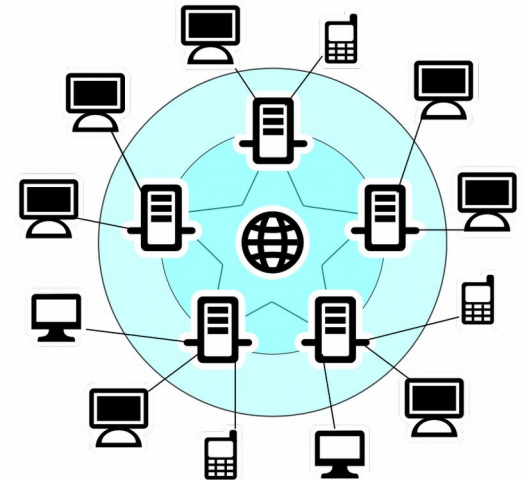
Web Systems

Relevant courses:

- **CS 343. Application Development**
- **CS 347. Web Development**

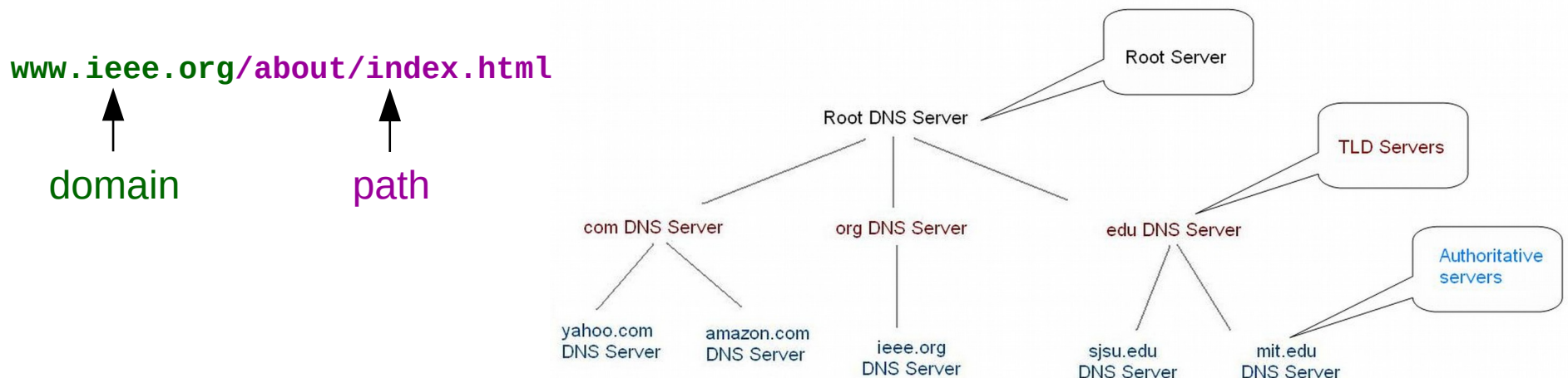
The "Internet"

- **World Wide Web (WWW)**
 - System for sharing information via hyperlinked documents
 - Started as a CERN project by Tim Berners-Lee; now a massive distributed system built on a worldwide network (the "Internet")
- **Issues:**
 - Naming
 - Security
 - Consistency
 - Replication



Naming

- IPv4 and IPv6 addresses for hosts
- Uniform Resource Locator (URL)
 - Unique worldwide name of a document
 - Domain + hierarchical path
- Domain Name Service (DNS)
 - Distributed, hierarchical IP address lookup protocol



Security

- **Secure Socket Layer (SSL) and Transport Layer Security (TLS)**
 - Public-key authentication, symmetric end-to-end encryption
 - **Certificate Authority (CA)** provides centralized key checking
 - Examples: [Comodo/Sectigo](#), [Symantec](#), and [Let's Encrypt](#)
- **HyperText Transfer Protocol (HTTP)**
 - Protocol for browser-server communication
 - Request and response model w/ headers and status codes
 - **HTTPS** is HTTP over SSL/TLS
- **Common Gateway Interface (CGI)**
 - Standardized program execution protocol
 - Somewhat similar to remote procedure calls (RPCs)
- **Denial-of-Service (DoS) or Distributed DoS (DDoS) attacks**
 - Often executed by **botnets** of virus-infected personal computers



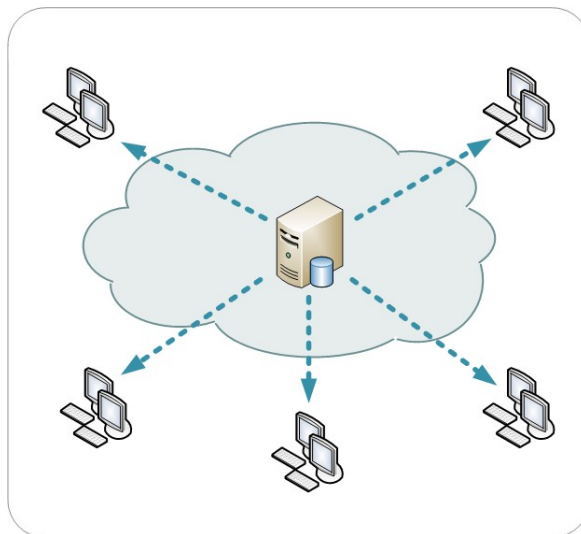
Interchange formats

- The internet is a **very** heterogeneous distributed system
 - Exchanging information can be a challenge
- **HyperText Markup Language (HTML)**
 - Document format for WWW; also SHTML w/ server side includes
- **eXtensible Markup Language (XML)**
 - Generalized HTML; generic data-interchange format
- **Multipurpose Internet Mail Exchange (MIME)**
 - Encoding formats for email messages; facilitates public-key crypto
- **Simple Object Access Protocol (SOAP)**
 - Web services data format
- **JavaScript Object Notation (JSON)**
 - Lightweight data-interchange format

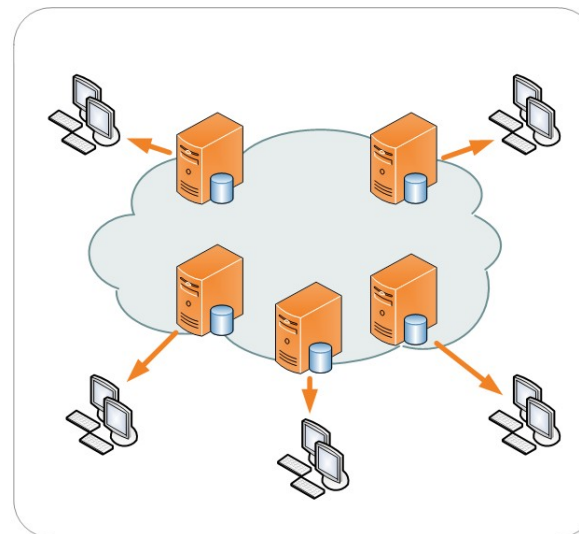
Consistency

- **Content Delivery Networks** (e.g. Akamai)
 - Globally-distributed network of proxy servers
 - Goal: improve data locality
 - Peer-to-peer and private CDNs
- **Firefox / Chrome / Safari / Edge**
 - Graphical interface for HTTP connections
 - Often caches website components locally

Traditional model



CDN model



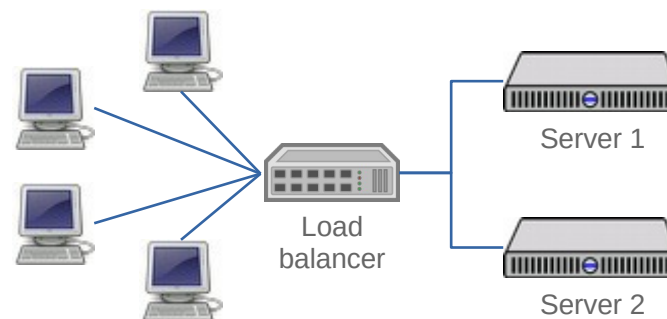
Replication

- **Apache** - Open-source extensible web server
 - **LAMP**: Linux, Apache, MySQL/MariaDB/MongoDB, PHP/Perl/Python
 - Other web servers: **Nginx** & **Microsoft IIS**
- Load balancing
 - Large websites require multiple servers w/ replicated data to provide availability to a massive number of users
 - **Load balancers** ensure that the traffic is distributed evenly

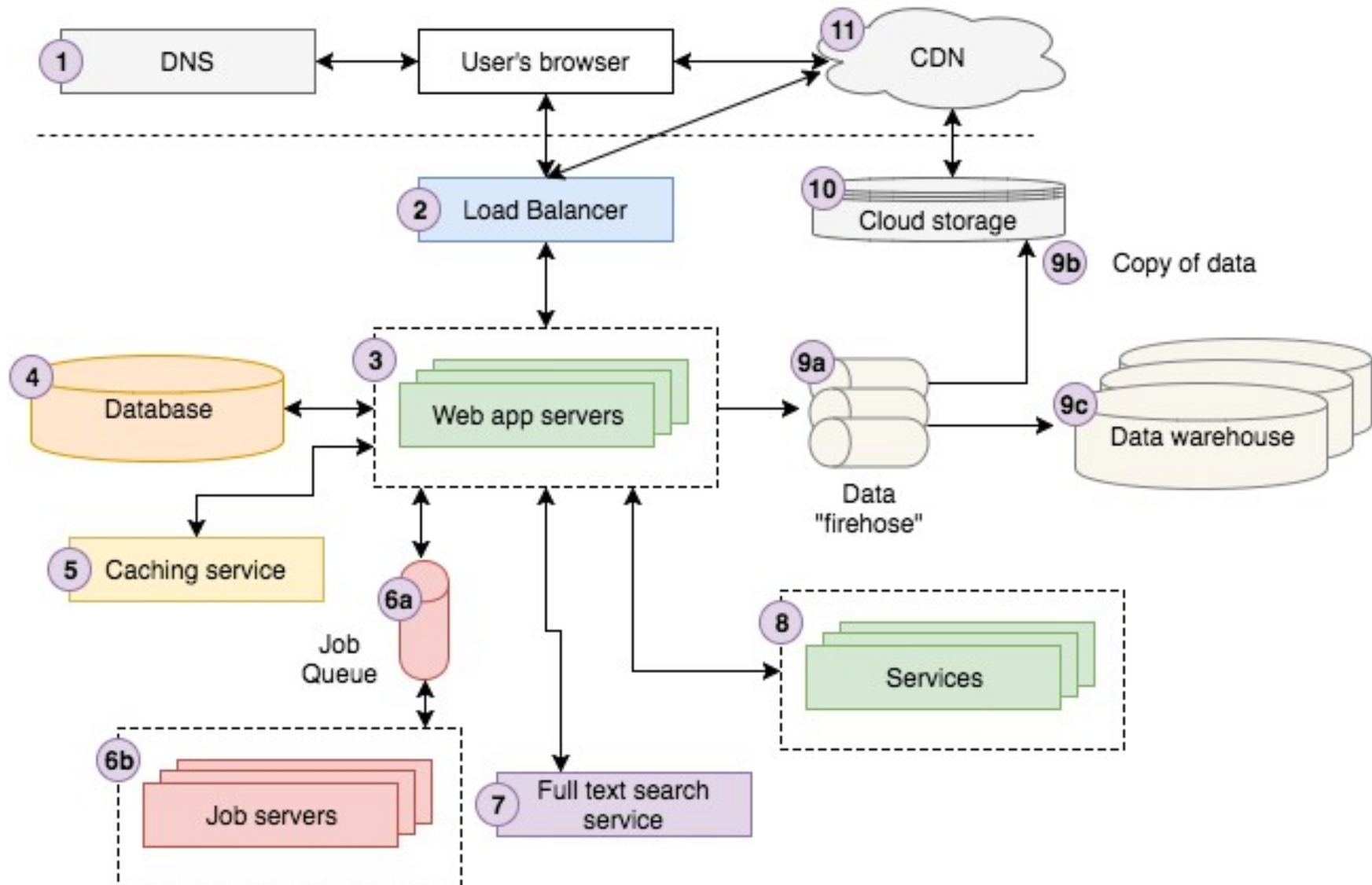


Apache

NGINX



Web systems architecture

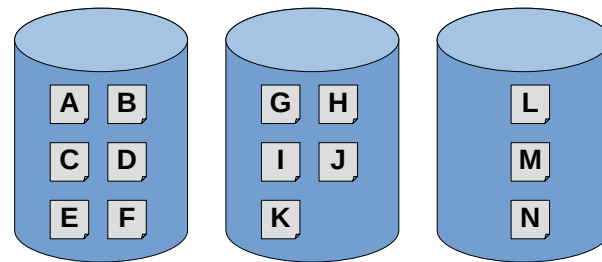


Distributed systems

File Systems

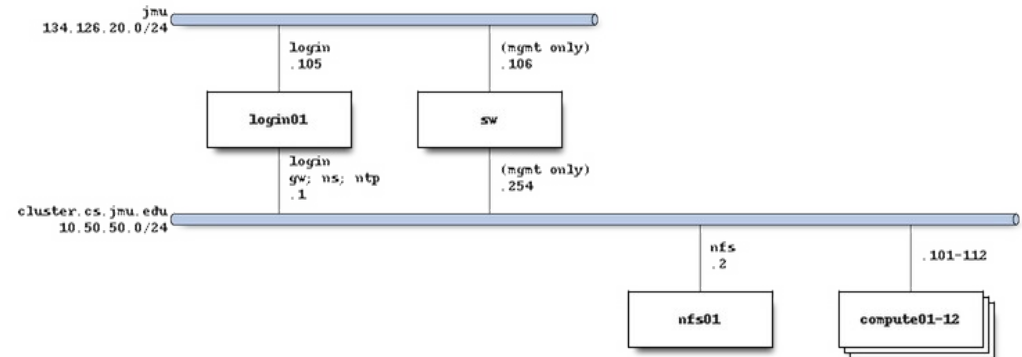
File systems

- **File system**: manages storage of structured data in files
- **Networked file system**: **centralized** storage with export/mount sharing
- **Distributed file system**: **distributed** storage with communication protocol
- **Centralized vs. decentralized**
 - **Asymmetric** (client/server) vs. **symmetric**
- **Issues**:
 - Naming
 - Security
 - Consistency
 - Replication



Networked file systems

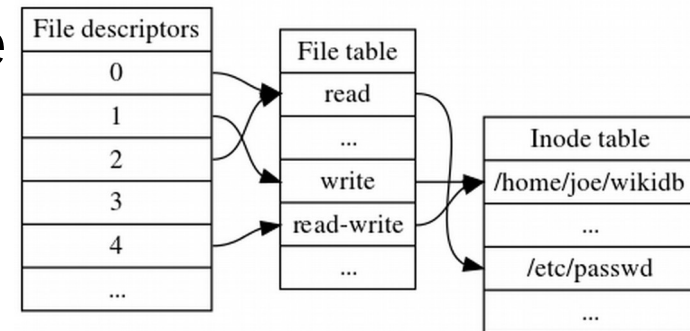
- **Networked file system**: **centralized** storage with export/mount sharing
- **Export**: a file system that is made available to another host
- **Mount**: link to a remote file system in the local file system
 - **File systems table** (fstab) configuration
 - **Static** vs. **automatic** mounting



/etc/fstab on (old) cluster	/dev/mapper/rhel_login01-root	/	xfst	defaults
	/dev/mapper/rhel_login01-swap	swap	swap	defaults
	nfs.cluster.cs.jmu.edu:/nfs/home	/nfs/home	nfs	rw
	nfs.cluster.cs.jmu.edu:/nfs/scratch	/scratch	nfs	rw
	nfs.cluster.cs.jmu.edu:/nfs/shared	/shared	nfs	rw, acl

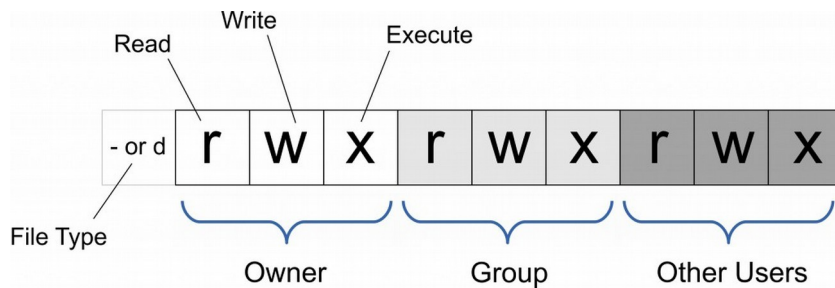
Naming

- Most traditional file systems use hierarchical **file names**
 - **Filesystem Hierarchy Standard** on Unix/Linux-based machines
- **File descriptors / handles**
 - Abstract identifier for an open file
 - In POSIX, positive integers:
 - Standard input: 0
 - Standard output: 1
 - Standard error: 2
- In distributed file systems:
 - **Data-centric** names
 - **Location-centric** names
 - **Name** servers (lookup) vs. **file** servers (access)



Security

- **Authentication**
 - UIDs, LDAP, Kerberos, Active Directory
- **Access control** (authorization)
 - Unix file permissions
 - Access control lists (e.g., POSIX)
 - Client vs. server permissions
- Encryption: security vs. performance tradeoff



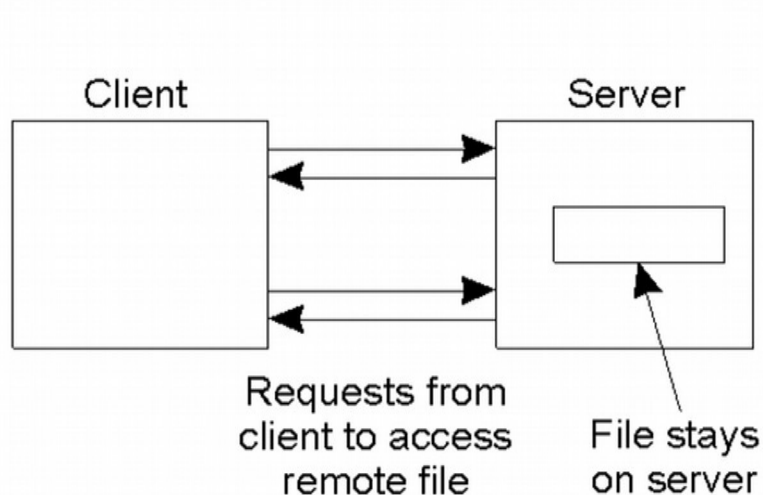
Unix file permissions

```
Alice: read,write;  
Bob: read;  
Admins: read,write;
```

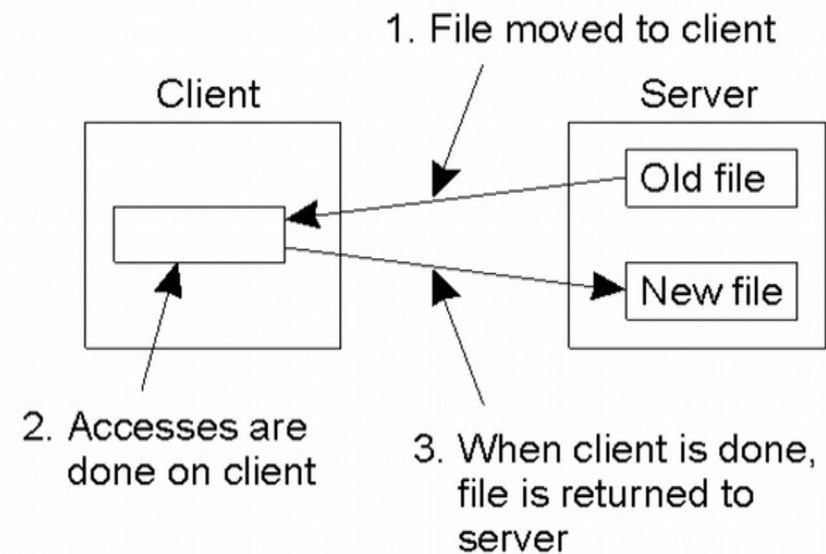
Access control list

Consistency

- **Remote-write** vs. **local-write** model
 - Forward updates immediately vs. buffer and send periodically
 - Also called **remote access** vs. **upload/download**
 - Tradeoff: consistency vs. performance



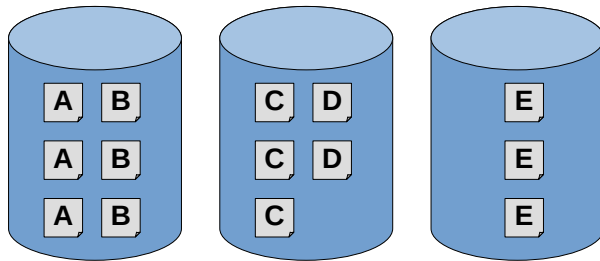
Remote-write



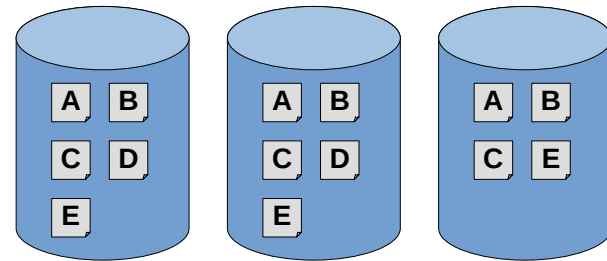
Local-write

Replication

- Client-side replication (**caching**)
 - Provides continued functionality while offline
 - Causes synchronization / consistency problems
 - **Callback** system for updating other clients
- Server-side replication (**mirroring**)
 - Provides **fault tolerance**
 - **Striping**: splitting a file's blocks across multiple servers



no striping



w/ striping

Peer-to-peer file systems

- Characterized by direct communication between clients
 - Centralized (e.g., [Napster](#)) vs. decentralized (e.g., [Bittorrent](#))
 - Anonymized (e.g., [Freenet](#)) via large-scale distributed caching with encryption and hash-based keys to locate data
- Raises many social and ethical issues
 - Censorship, activism, and free speech
 - Privacy and security
 - Illegal activity and law enforcement



Distributed systems

(Particular) File Systems

Network File System (NFS)

- Basic file sharing protocol for local networks
 - Based on remote procedure calls (RPCs)
 - Provides shared storage and reliability in presence of failures

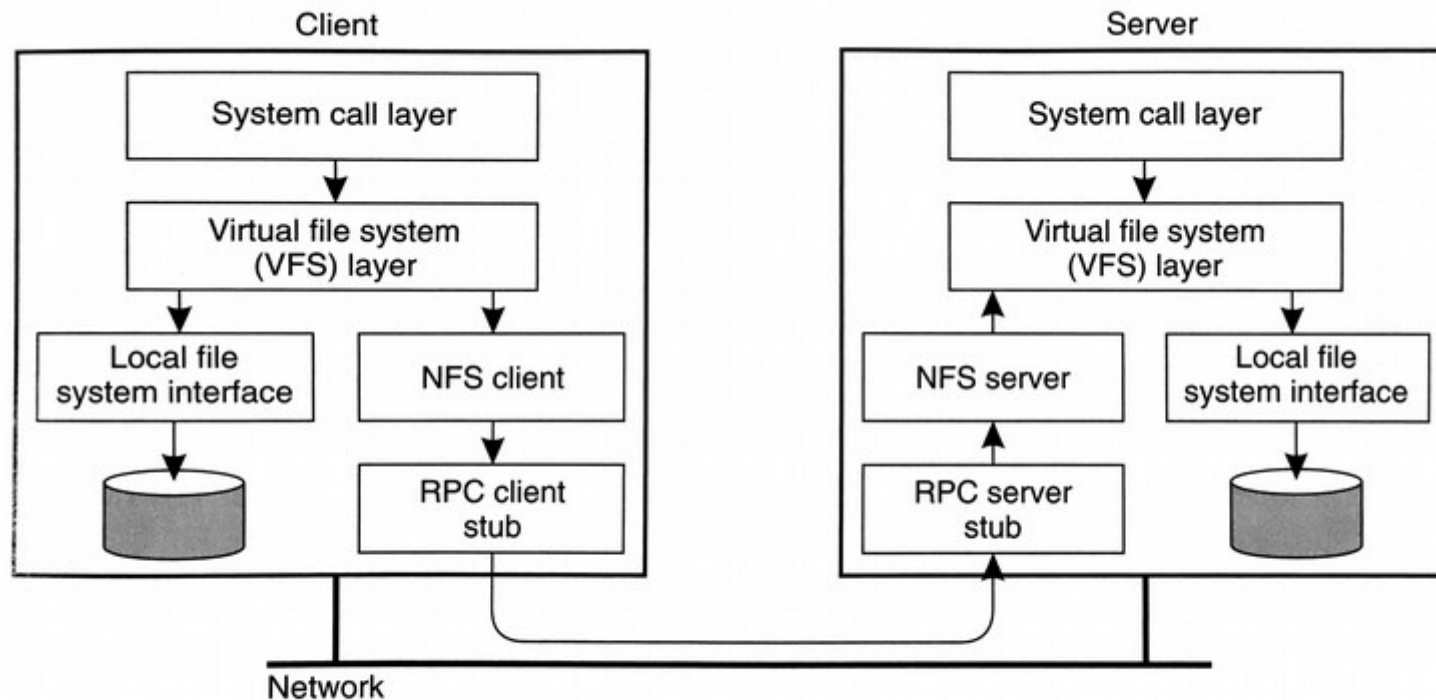


Figure 11-2. The basic NFS architecture for UNIX systems.

Network File System (NFS)

- Developed by **Sun** in 1984
 - Originally an in-house solution (v.1), now an open standard
- **NFSv2** released in 1989
 - UDP-based **stateless** protocol
 - No built-in locking
- **NFSv3** released in 1995
 - 64-bit and TCP support
- **NFSv4** released in 2000
 - Adds **stateful** protocol and compound RPCs
 - Better access control (finer granularity)
 - New security features (including encrypted traffic)
 - **pNFS**: scalable access to files distributed on multiple servers

Andrew File System (AFS)

- Developed at CMU in early 1980s
 - (Named after Andrew Carnegie and Andrew Mellon)
- Improved on NFS in terms of scalability and security
- **Weak consistency** model
 - Each file is locked when opened
 - Modifications are performed and buffered locally
 - Updates are only sent to the server when a file is closed
 - Server uses callbacks to update other clients
- **Kerberos**-based **access control lists**
 - Lookup / insert / delete / administer
 - Read / write / lock
- Heavily influenced development of **NFSv4**

NFSv4

- **Access control lists**

- Type: **A** = allow, **D** = deny, **U** = audit, **L** = alarm
- Flags: **g** = group, **d** = directory-inherit, **f** = file-inherit
- Permissions: **r** = read, **w** = write, **a** = append, **x** = execute, **d** = delete
- Permissions (cont.): **c** = read-ACL, **C** = write-ACL, **o** = write-owner
- Policy of "default-deny"

```
A::OWNER@:rwatTnNcCy
A::alice@nfsdomain.org:rxtncy
A::bob@nfsdomain.org:rwadtTnNcCy
A:g:GROUP@:rtncy
D:g:GROUP@:waxTC
A::EVERYONE@:rtncy
D::EVERYONE@:waxTC
```

Distributed systems

(Particular) Parallel File Systems

Google File System (GFS)

- Reliable **asymmetric** distributed file system on commodity hardware
 - Each file is split into **chunks** with unique chunk IDs (chunks can be replicated)
 - Master stores metadata tracking each file and its chunks (and where they are)
 - Basis for **BigTable**, backing store for the original **MapReduce**

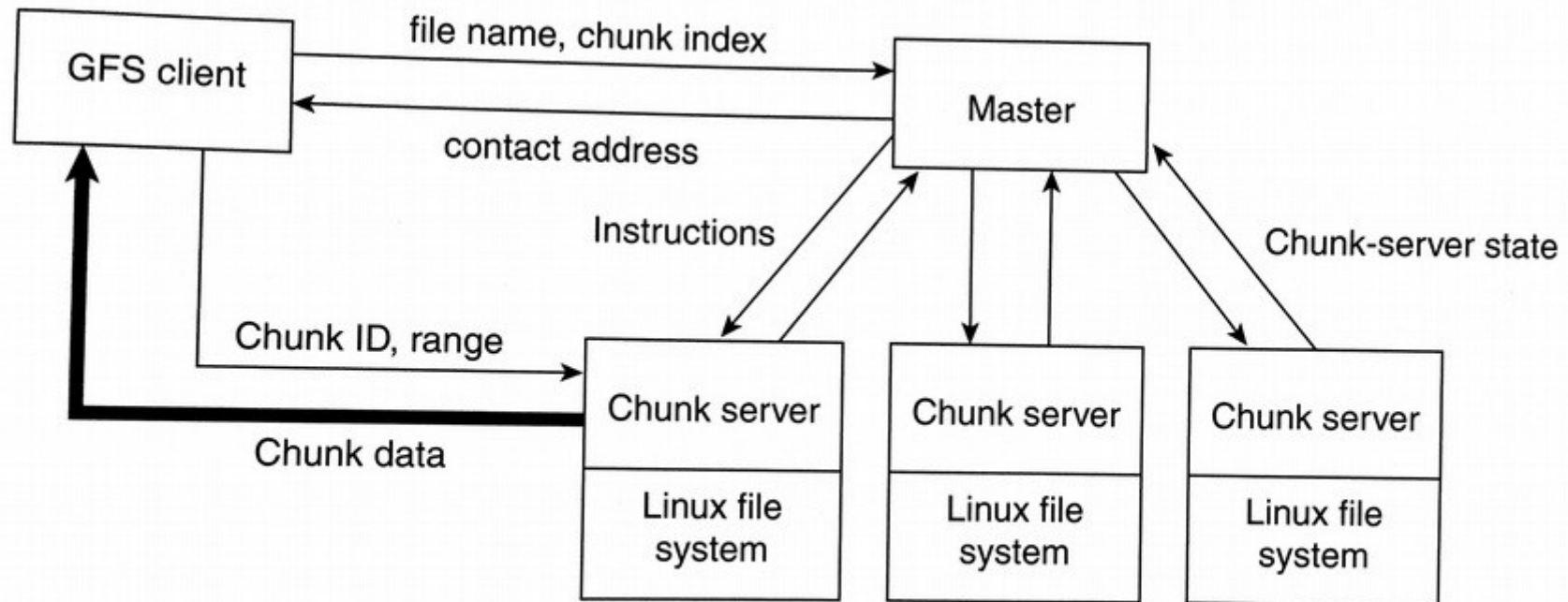


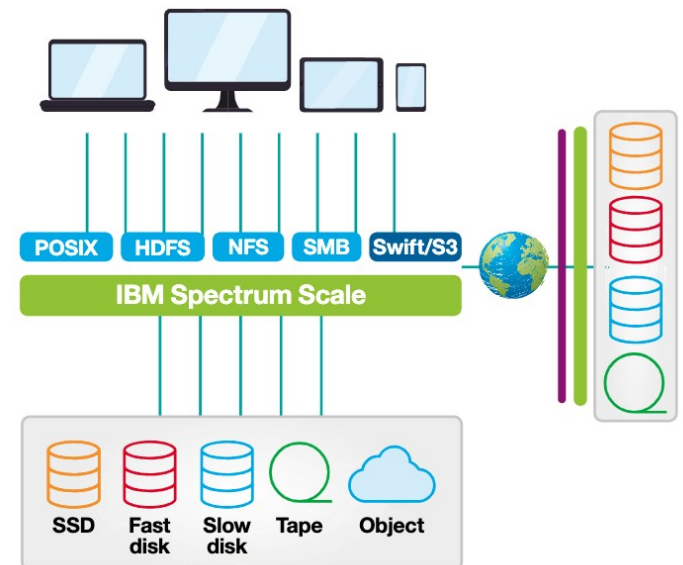
Figure 11-5. The organization of a Google cluster of servers.

Lustre

- High-performance parallel file system
 - Initially a research project; later owned by [Sun/Oracle](#)
 - Now open source, maintained by a collection of organizations
 - Multiple lower-level interconnects: [Ethernet](#), [Infiniband](#)
 - Used by many supercomputer installations
 - E.g., [Sequoia](#) and [Titan](#)
- Three functional units
 - Metadata server (MDS) – names, layout, permissions
 - Object storage server (OSS) – stores file data
 - Clients – connect to servers

GPFS / Spectrum Scale

- The **General Parallel File System (GPFS)** was developed by IBM and released in 1998
 - Re-branded as **IBM Spectrum Scale** in 2015
 - Used in many supercomputer installations
 - E.g., **Sierra** and **Summit**
- Industrial HPC file system
 - Reads and writes happen in parallel
 - Distributed metadata; no single point of failure
 - Availability and fault tolerance mechanism
 - Full POSIX, NFS, and SMB compatibility
 - Multiple backing stores



Rabbit

- **Rabbit**: a new “near-local” storage approach for El Capitan
 - Essentially a “transient Lustre file system” across compute nodes

Rabbit modules are a 4U near node local storage solution



- All in one solution: Rabbit- 4U
 - Houses 18 SSD's (16+ 2 spares) that attach to Rabbit-S board
 - Locates Storage Processor (AMD Epyc CPU) on Rabbit-P board
- Compute blades direct attached to Rabbit-S through bulkhead cables
- Rabbit-S to Rabbit-P board connections are internal (no external cables)

