Cloud Computing Applications and Services

(Aplicações e Serviços de Computação em Nuvem)

Storage

University of Minho
2022/2023

Why are storage systems relevant?

- Cornerstone for data management infrastructures and systems
  - Cloud, HPC, IoT ,...
  - Databases, Analytics, Machine Learning, ...
- Data persistency and availability
- Performance is key!
  - Slow data storage and retrieval translates into slow applications

Storage Types
Archival

- Data is stored for archival purposes
  - Throughput is favoured over latency
    - large amounts of data must be written/read efficiently
  - Sequential workloads
  - Write-once data (typically)

- E.g., Amazon Glacier (cloud service)

Storage Types
Backup

- Data backups of *fresh* data
    - Throughput is favoured over latency
        - large amounts of data must be written/read efficiently
    - Sequential workloads (mainly...)
    - In some cases, data can be updated (sporadically and in-place)
    - In some cases, only *diffs* (modified data) are stored across backups of the same source

- E.g., Amazon S3 (cloud service)

Storage Types

Primary Storage* (not only RAM!)

- Storage support for databases, analytics, VMs ...
    - High-throughput and low-latency is now desirable
        - large amounts of data may be written/read (throughput)
        - small sized writes/reads (potentially over different files) must be efficient (latency)
    - Sequential and random workloads
    - Data and metadata intensive workloads
    - Data can be updated frequently

- E.g., Amazon EBS (cloud service)

*definition taken from: *Paulo, J and Pereira, J. 2014.* ***A Survey and Classification of Storage Deduplication Systems****. ACM Comput. Surv*

Storage Mediums

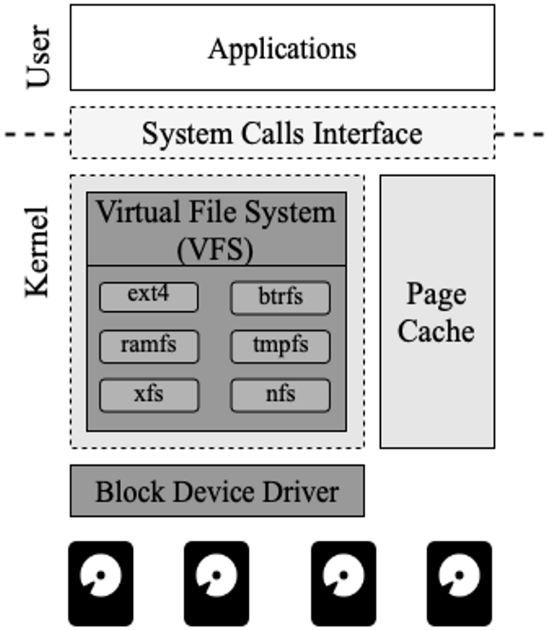
- Tape
  - Archival storage
- HDD
  - Archival, Backup, and Primary storage
- SSD (includes NVMe)
  - Primary storage
- Persistent Memory
  - Primary storage
- RAM
  - Primary storage

Storage Interfaces

- Block Device
  - Data is managed as blocks (e.g., iSCSI, Amazon EBS, Ceph ...)
- File System
  - Data is managed as a hierarchy of files (e.g., Ext4, Lustre, Ceph...)
- Object Storage
  - Data is managed as objects (e.g., Amazon S3, Openstack Swift, Ceph)

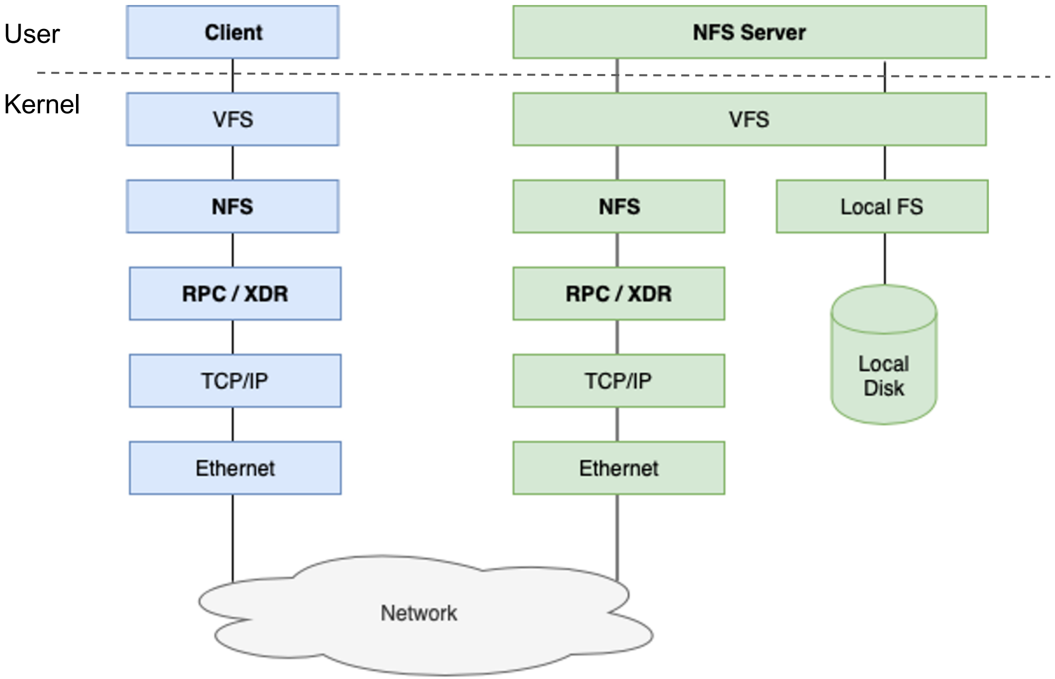
Scope
From Local ...

- Operating System (OS) Block Devices
- OS File Systems
  - e.g., Ext4, ZFS
- Kernel vs User space



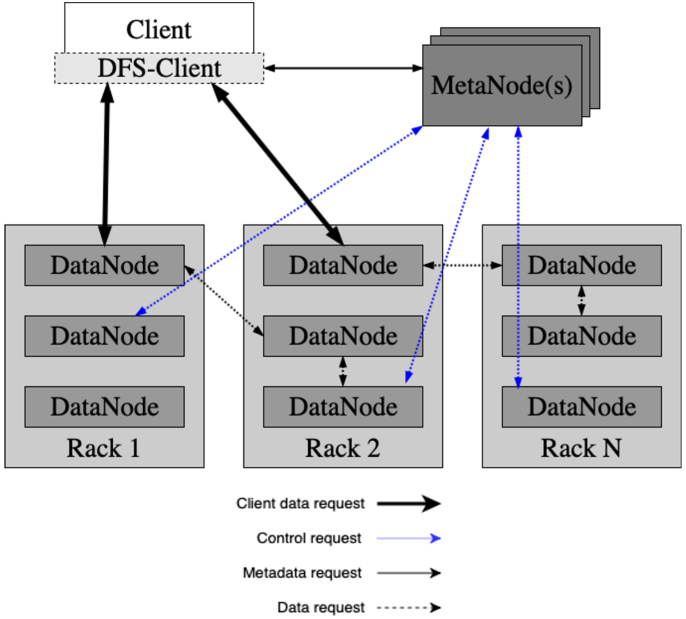
Scope
to Remote ...

- Network Block Devices
  - e.g., iSCSI
- Network File Systems
  - e.g., NFS
- Client-Server paradigm (Network)



Scope
to Distributed... (Data Center)

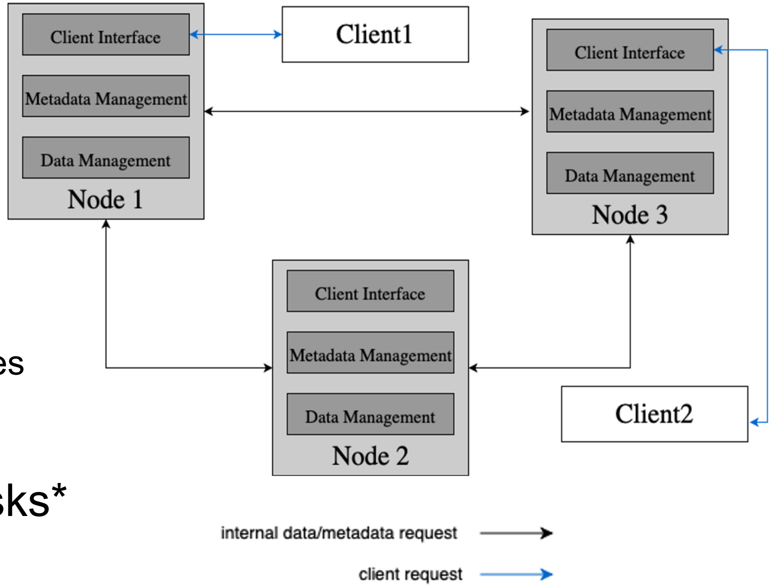
- Large-scale (e.g., Cloud and HPC infrastructures)
  - hundreds to thousands of nodes
- Stable churn*
  - … but nodes still fail
- No single point of failure
  - Data is distributed (replicated) across nodes
  - Metadata is typically managed by independent nodes
- e.g., HDFS, Ceph, Lustre, GPFS



* Nodes entering and leaving the system

Scope
to Highly Distributed... (Peer-to-Peer)

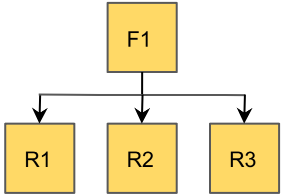
- Very large-scale (e.g., IoT infrastructures)
  - thousands to millions of nodes
- High churn
  - nodes fail and are replaced frequently
- No single point of failure
  - Data and metadata distributed (replicated) across nodes
  - Different nodes can interact with the user application
- e.g., Napster, Gnutella, CFS, Farsite, DataFlasks*



* Maia et al. 2014. **DATAFLASKS: epidemic store for massive scale systems**. Symposium on Reliable Distributed Systems (SRDS) .

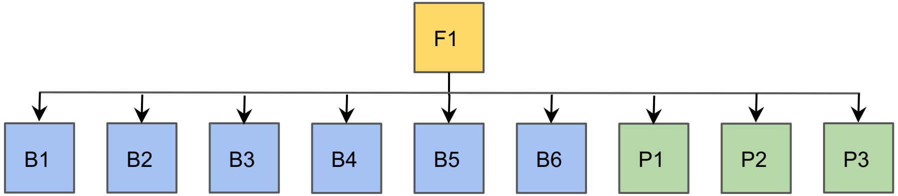
Storage Features (some examples…)
Data Availability

- RAID - Redundant Array of Inexpensive Drives
- Replication
- Erasure-Codes



Replication: Exact Replicas (of F1)

E.g., Replication factor = 3
- Tolerates 2 failures
- 3X storage overhead



Erasure-codes: Original data (F1) is divided into *k* Blocks and *m* Parity blocks

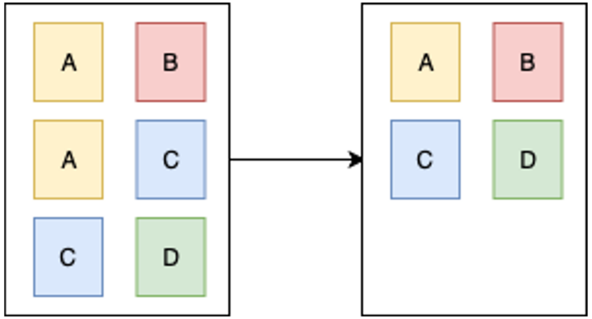
E.g., Reed- Solomon (k = 6, m = 3)
- Tolerates 3 failures
- 1.5 X storage overhead

Storage Features (some examples…)
Performance Optimizations

- Data locality
  - Push computation near to the devices / servers holding data
  - Storage and processing co-location at the same server / device
  - E.g., HBase and HDFS, active storage
- Caching
  - Keep data closer to the client and/or accessible from a faster source
  - Avoid waiting for data to be written/read from local or remote storage
  - E.g., file system page cache, Alluxio (in-memory distributed file system)

Storage Features (some examples…)
Space Efficiency

- Compression
  - Reduces redundant content (e.g., bytes) inside and across files
  - Usually used as a static approach
- Deduplication
  - Eliminates redundant copies at a storage system (e.g., files / blocks)
  - Dynamic technique



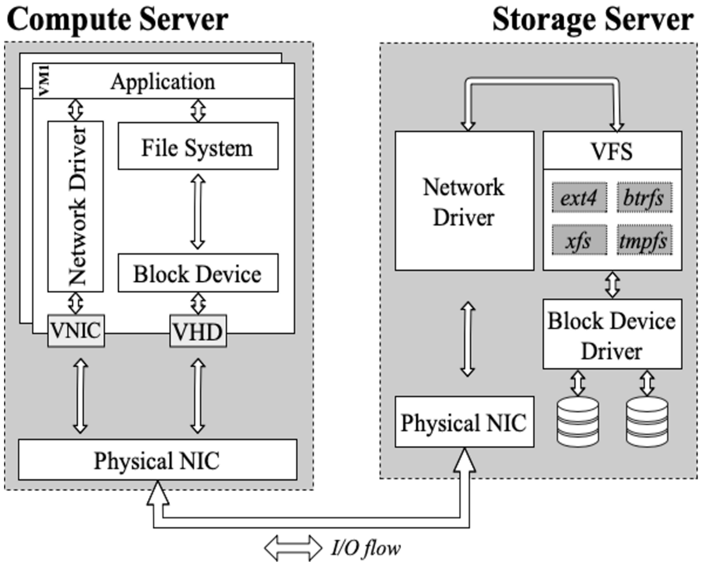
Storage Features (some examples…)
Security

- Data Encryption
  - Encryption at rest
    - Data is encrypted before being stored persistently
  - Encryption in transit
    - Data is encrypted at the client premises before being set through the network (e.g., for remote storage systems)

- Access Control
  - Avoid unauthorized access to users data

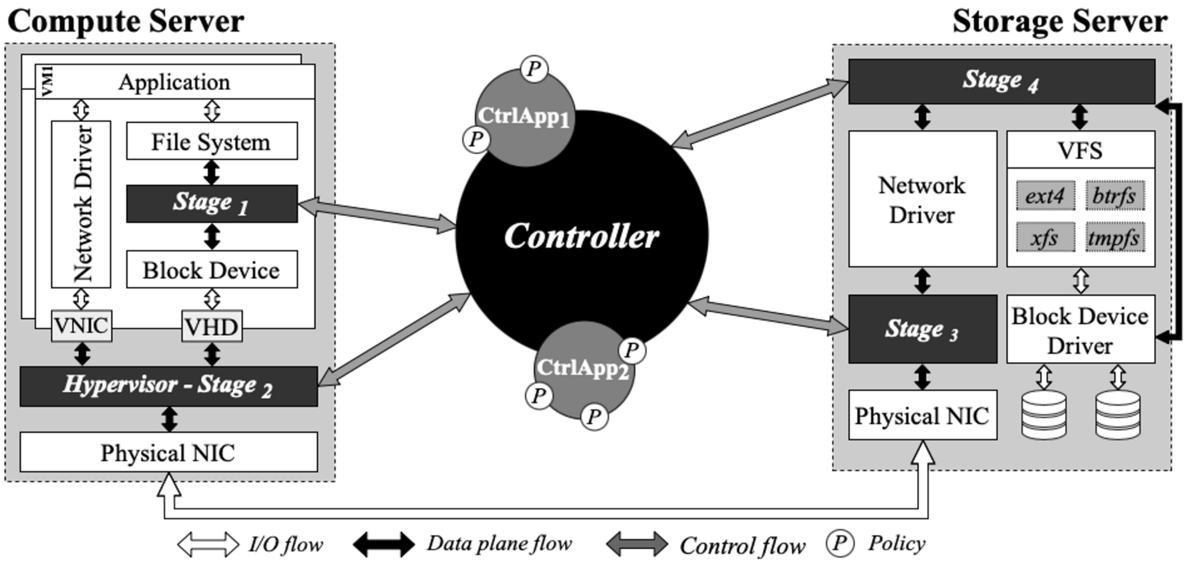
Complex and Monolithic Storage Solutions

- The I/O stack of data centers is long and composed by several components
  - E.g., apps, remote storage, file systems, block devices, disks
- Each providing a strict combination of storage features, however…
- … the best combination of features to be applied depends on the requirements of each application:
  - Small files versus large files
  - Storage access patterns
  - ...



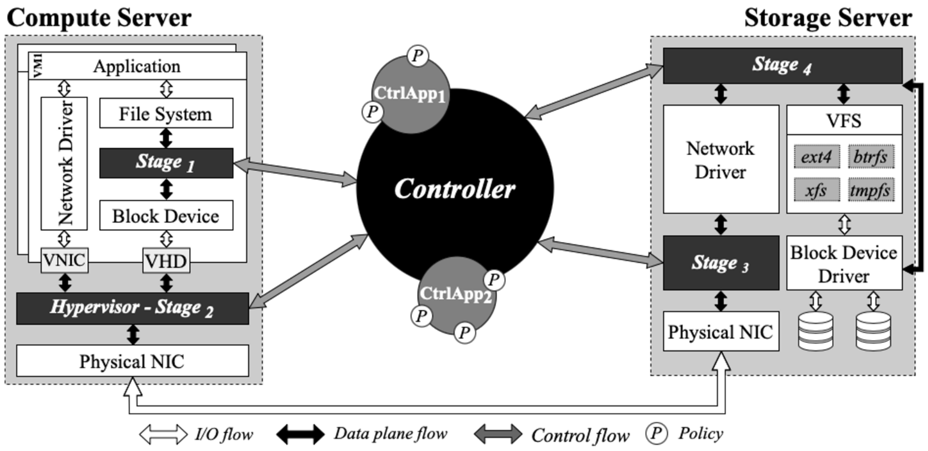
Software-Defined Storage

- Follows the principles of Software-Defined Networks (SDN)
- I/O flow (data plane) is separated from the control flow (control plane)
- Global control of I/O flows (logically centralized)



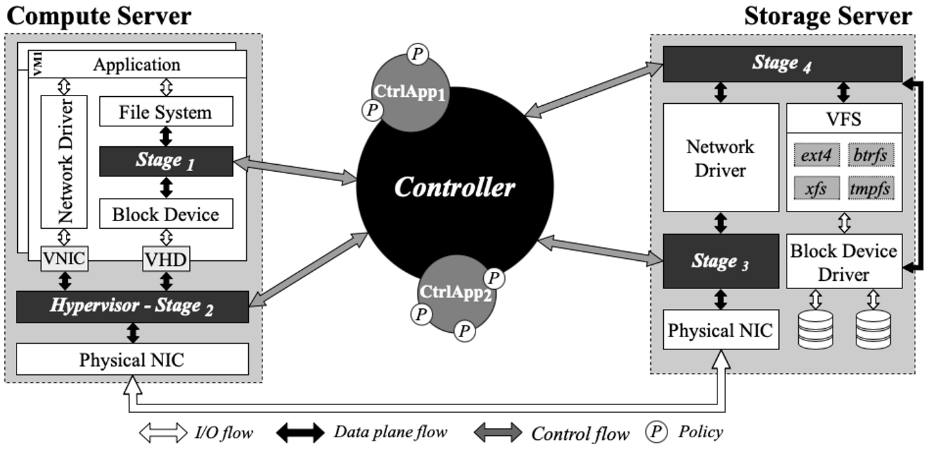
Software-Defined Storage
Data Plane

- Layered approach (stages)
- Each stage handles requests at the I/O path and provides different functionalities
  - E.g., caching, compression, encryption
- Programmable and extensible design



Software-Defined Storage
Control Plane

- Distributed and dependable
- Global visibility of applications, stages and infrastructure resources
- Configures and tunes data plane stages to enforce I/O policies
    - Defined by Control Applications
    - Quality of Service (e.g., I/O fairness or prioritization)
    - Transformations (e.g., encryption, compression)



Further reading

- Macedo R, Paulo J, Pereira J, Bessani, A. 2020. **A Survey and Classification of Software-Defined Storage Systems**. ACM Computing Surveys.
- Paulo J, Pereira J. 2014. **A Survey and Classification of Storage Deduplication Systems**. ACM Computing Surveys.
- Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Carlos Maltzahn. 2006. **Ceph: a scalable, high-performance distributed file system**. Operating Systems Design and Implementation (OSDI).
- Maia F, Matos M, Vilaça R, Pereira JO, Oliveira R, Rivière E. 2014. **DATAFLASKS: epidemic store for massive scale systems.** Symposium on Reliable Distributed Systems (SRDS) .

Questions?