Infrastructure Security Using Linux

Computer science / Cybersecurity

Web Hosting and Storage

19 → 20

Web Hosting

Dominance of UNIX/Linux in Web Hosting

- **Platform Market Share**
  - **67%** of top 1M websites run on **Linux or FreeBSD**

    (Source: w3techs.com)
- **Web Server Software**
  - **80%+** use **open source** web servers
    - Examples: Apache, Nginx, Lighttpd

Key Takeaway: UNIX-like systems and open source software dominate the web hosting landscape.

HTTP Overview

- Core protocol for web communication, Stateless **request/response** model, Simplicity hides layers of flexibility and complexity

Table 19.2: HTTP response classes

Code	General indication	Examples
1xx	Request received; processing continues	101 Switching Protocols
2xx	Success	200 OK 201 Created
3xx	Further action needed	301 Moved Permanently 302 Found [a]
4xx	Unsatisfiable request	403 Forbidden 404 Not Found
5xx	Server or environment failure	503 Service Unavailable

Virtual Hosts & SNI

- Originally, **one server = one website** (requests routed via DNS to a single IP on port 80)
- Need arose to host multiple sites on a single server/IP
- Virtual Hosts (HTTP/1.1)
  - Clients send a Host header in HTTP requests
  - Server uses it to distinguish between sites
  - Now the standard method for multi-site hosting
- SNI (Server Name Indication)
  - Allows TLS to support virtual hosts
  - Client sends hostname during handshake
  - Server selects the correct certificate before encryption
  - Transparent in modern clients and servers

Web software basics

Table 19.4: Partial list of HTTP server types

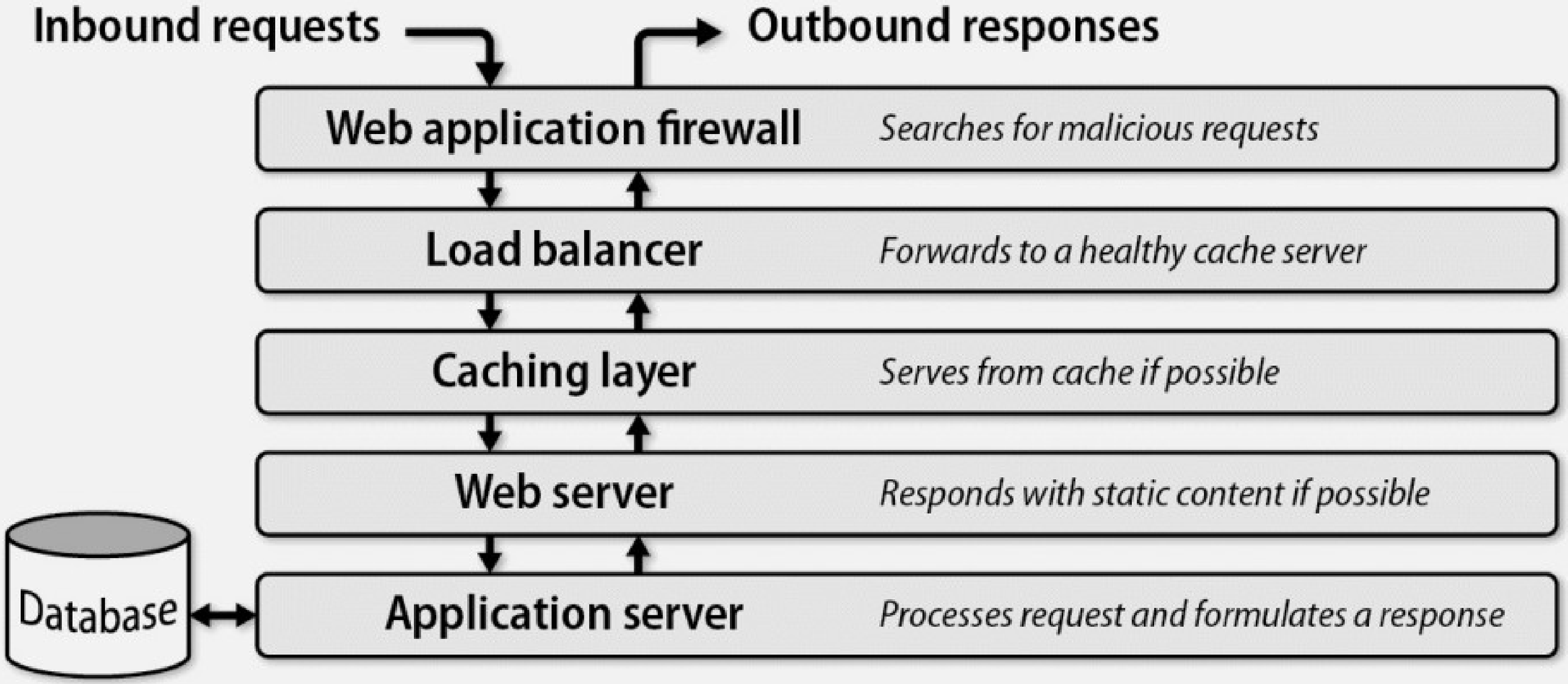
Type	Purpose	Examples
Application server	Runs web app code, interfaces to web servers	Unicorn, Tomcat
Cache	Speeds access to frequently requested content	Varnish, Squid
Load balancer	Relays requests to downstream systems	Pound, HAProxy
Web app firewall [a]	Inspects HTTP traffic for common attacks	ModSecurity
Web server	Serves static content, couples to other servers	Apache, NGINX

a. Often abbreviated WAF

Web Proxies

- A **web proxy** is an intermediary that handles HTTP requests from clients and forwards them to the destination
- Can perform processing: filtering, caching, logging, rewriting, etc.
- **Types of Proxies**
  - **Load Balancer** – Distributes traffic across backend servers
  - **WAF (Web Application Firewall)** – Filters malicious requests
  - **Cache Server** – Stores responses to reduce latency/load
  - **Web Server as Proxy** – Forwards requests to app servers (e.g., via reverse proxy)

Exhibit A: Components of a web application stack



Web Servers & HTTP Proxy Software

- **Core Functions of Web Servers**
  - Serve static content or proxy to app servers
  - Support **virtual hosts**, **TLS**, and **HTTP basic auth**
  - Provide **configurable logging**
  - Route requests based on URL
  - Execute dynamic content via app servers
- **Popular Web Servers**
  - **Apache (httpd)**:
    - Since 1995, modular, flexible, widely used
    - Powerful but complex to configure
  - **NGINX**:
    - Since 2004, lightweight, fast, scalable
    - Excels at static content, reverse proxying, load balancing

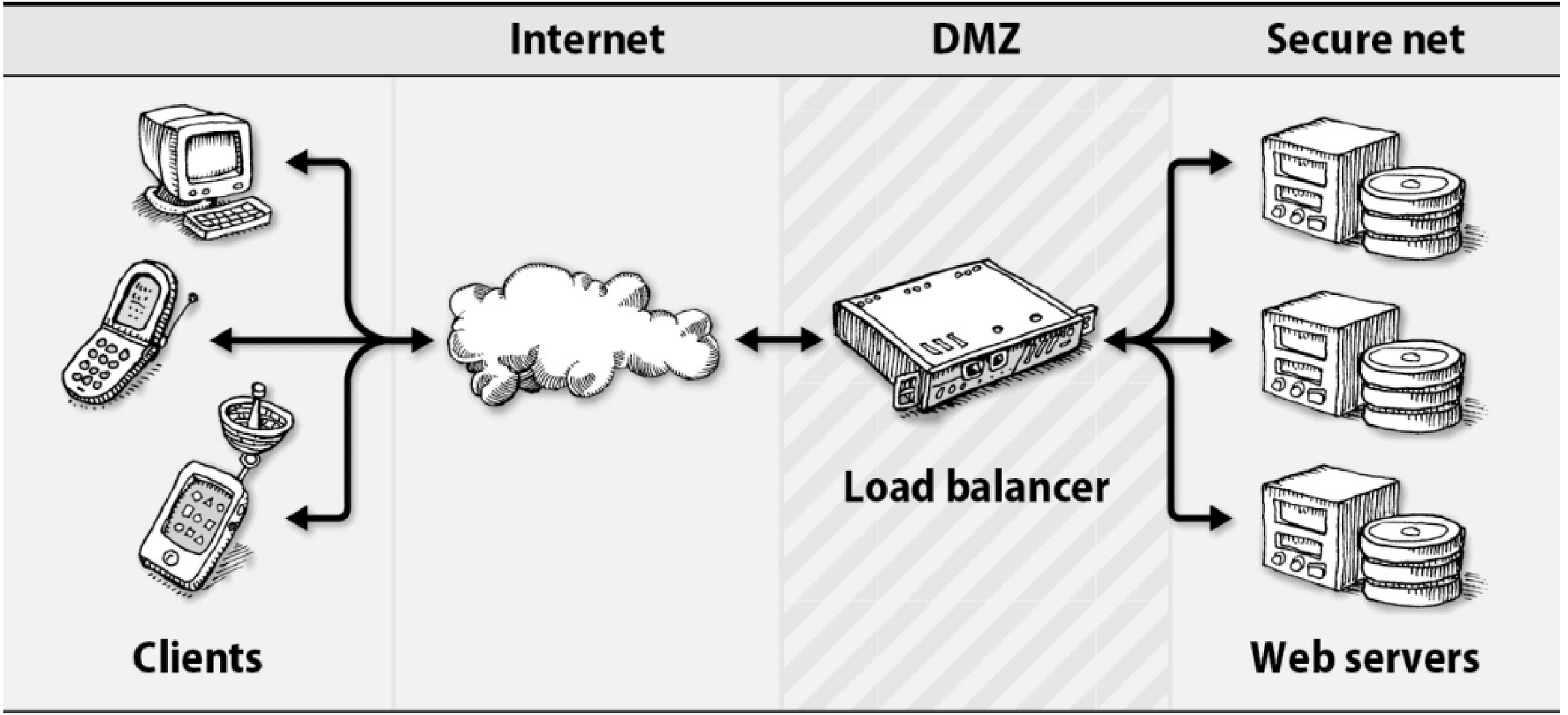
High Availability & NGINX

- **Maximize Availability**
  - Run each layer on multiple nodes
  - Distribute across geographic regions to avoid single points of failure
- **NGINX: More Than a Web Server**
  - Acts as web server, cache, and load balancer
  - With caching enabled, NGINX outperforms stacks of separate VMs
  - Consolidates roles for better efficiency and lower latency
- **Other Options**
  - **Node.js**, **Go**: Built-in HTTP servers, production-ready
  - **H2O**: Modern server optimized for HTTP/2, often faster than NGINX

Load Balancers

- **Single server = single point of failure**
  - No resilience to crashes or maintenance
  - No seamless updates without downtime
- **Load Balancer Role**
  - Distributes incoming requests across multiple backend servers
  - Monitors server health and response quality
  - Ensures high availability, scalability, and fault tolerance
- **Key Benefit:** Continuous service, even during server updates or failures

Exhibit B: The role of a load balancer



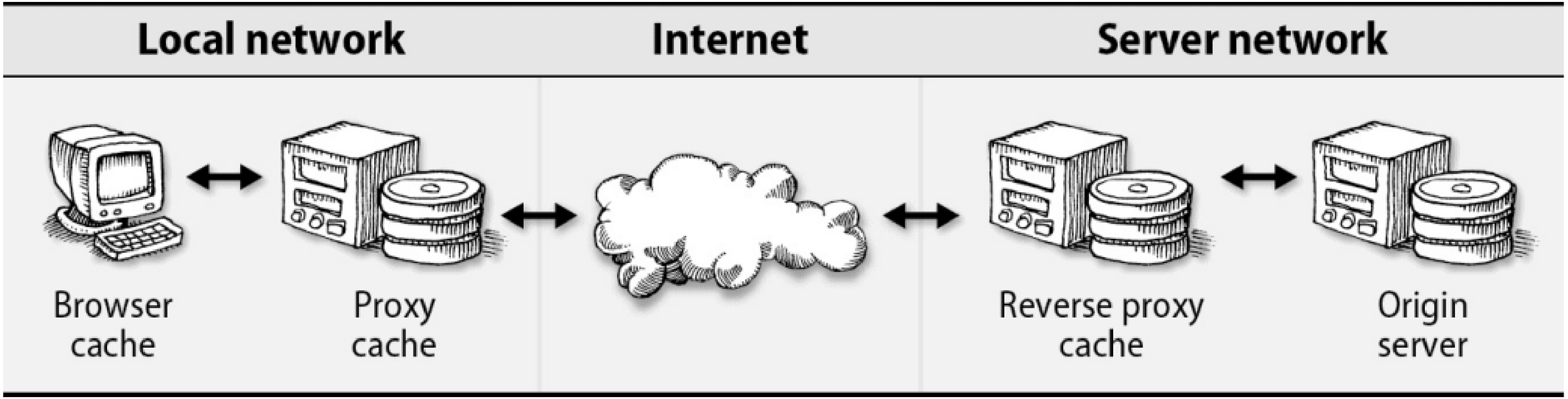
Load Balancers

- **Purpose**: Distribute incoming traffic across multiple servers to ensure **high availability**, **scalability**, and **fault tolerance**
- **Benefits:**
  - Avoid downtime during maintenance
  - Handle more connections than individual web servers
  - Monitor backend health and remove unresponsive nodes
  - Improve security (TLS termination, DMZ isolation)
- **Operation Modes: Layer 4** – Route by IP/port, **Layer 7** – Route based on URL, headers, cookies
- **Deployments**
  - **Commercial** – F5, Citrix (hardware/software)
  - **Cloud** – AWS ELB (uses listeners to proxy TCP/HTTP/HTTPS)

Web Caches – How They Work

- Caches store frequently requested content to reduce server load and improve response times. They sit between clients and origin servers, sometimes caching content in memory.

Exhibit C: Caching players involved in handling an HTTP request



Web hosting in the cloud

- **Platform as a Service (PaaS):** simplifies web hosting by abstracting infrastructure. Developers package code in a defined format and upload it; the provider handles provisioning, deployment, and scaling automatically.
- A DNS endpoint is issued and can be mapped via CNAME to a custom domain.
- **Google App Engine** pioneered PaaS. It supports Python, Java, PHP, and Go, with built-in features like scheduled jobs, log access, messaging, and database services.
- **Heroku** is another major PaaS. It supports Ruby, Node.js, Python, PHP, and Java, with a broad ecosystem of add-ons for databases, caching, and more.

Apache HTTPD

- Widely used web server across UNIX and Linux systems
- Modular architecture enables flexible configuration via dynamic modules
  - Supports custom auth, enhanced security, multiple languages, URL rewriting

Table 19.6: Apache configuration details by platform

	RHEL/CentOS	Debian/Ubuntu	FreeBSD
Package name	httpd	apache2	apache24
Config root	/etc/httpd	/etc/apache2	/usr/local/etc/apache24
Primary config file	conf/httpd.conf	apache2.conf	httpd.conf
Module config	conf.modules.d/	mods-available/ mods-enabled/	modules.d/
Virtual host config	conf.d/	sites-available/ sites-enabled/	Includes/
Log location	/var/log/httpd	/var/log/apache2	/var/log/httpd-*.log
User	apache	www-data	www

Storage

Data Storage Capacity

Floppy Disk
1.4MB

CD-ROM
700MB

DVD
4.7GB

Blu-Ray
25 GB – 128GB

Data storage devices have very different capacities. Over time the capacity has increased which has allowed for more data to be stored:

Increase in storage capacity

Hard Drive
8 TB

Magnetic Tape
Up to 185 TB

Data Storage Systems

- **Slide Title: Data Storage Systems**
  - Modern storage is modular—like Lego blocks—adaptable for speed, durability, or scale
  - Configurable for high-performance databases or long-term archival storage with redundancy and versioning
- **Storage Media**
  - **HDDs**: High capacity, cost-effective
  - **SSDs**: High performance, lower latency
  - **Caching systems**: Bridge speed and capacity using hardware/software layers

Software Stack (Above Hardware)

- Device drivers and partitioning schemes
- RAID for redundancy and performance
- Logical Volume Managers (LVM) for flexible storage
- Network storage virtualization systems
- Filesystem implementations (e.g., ext4, XFS, ZFS)

PCI Express (PCIe)

- High-speed **serial bus** connecting devices to the motherboard
- Commonly used for **SSDs** and other **high-performance peripherals**
- **PCIe vs. SATA**
  - **SATA**: 6 Gb/s (gigabits per second)
  - **PCIe**: Full-width lanes offer **20×+** the speed of SATA
  - Ideal for performance-critical storage solutions

PCI Express (PCIe)

Version	Release Year	Bandwidth per Lane	Total Bandwidth (lane x16)	Notable Features
PCIe 1.0	2003	250 MB/s	4 GB/s	Initial release
PCIe 2.0	2007	500 MB/s	8 GB/s	Doubled bandwidth
PCIe 3.0	2010	985 MB/s	15.75 GB/s	Improved encoding
PCIe 4.0	2017	1.97 GB/s	31.5 GB/s	Doubled bandwidth again
PCIe 5.0	2019	3.94 GB/s	63 GB/s	Improved signal integrity
PCIe 6.0	2022	7.88 GB/s	126 GB/s	Pulse Amplitude Modulation 4-level (PAM4) signaling
PCIe 7.0	Expected 2025	~15.75 GB/s	~252 GB/s	In development

PCIe devices can use different numbers of lanes, typically denoted as x1, x4, x8, or x16.

Disk device files

A newly added disk is represented by device files in **/dev**.

Disk Device Naming Standards in Linux and FreeBSD:

Aspect	Linux	FreeBSD
IDE/PATA Disks	/dev/hda, /dev/hdb, /dev/hdc, etc. (older systems)	/dev/ada0, /dev/ada1, etc.
SATA Disks	/dev/sda, /dev/sdb, /dev/sdc, etc.	/dev/ada0, /dev/ada1, etc.
NVMe Disks	/dev/nvme0n1, /dev/nvme0n2, /dev/nvme1n1, etc.	/dev/nvd0, /dev/nvd1, etc.
Virtual Disks	/dev/vda, /dev/vdb, etc. (for VMs)	/dev/vtbd0, /dev/vtbd1, etc.
Partitions	Appended numbers: sda1, sda2, nvme0n1p1, etc.	Appended letters: ada0p1, ada0p2, nvd0p1, etc.
Software RAID	/dev/md0, /dev/md1, etc.	/dev/mirror/name, /dev/stripe/name, etc.
Logical Volumes (LVM)	/dev/mapper/vgname-lvname	N/A (LVM not natively supported)
ZFS Pools	/dev/zd0, /dev/zd1, etc. (for zvols)	N/A (uses labels instead of device names)

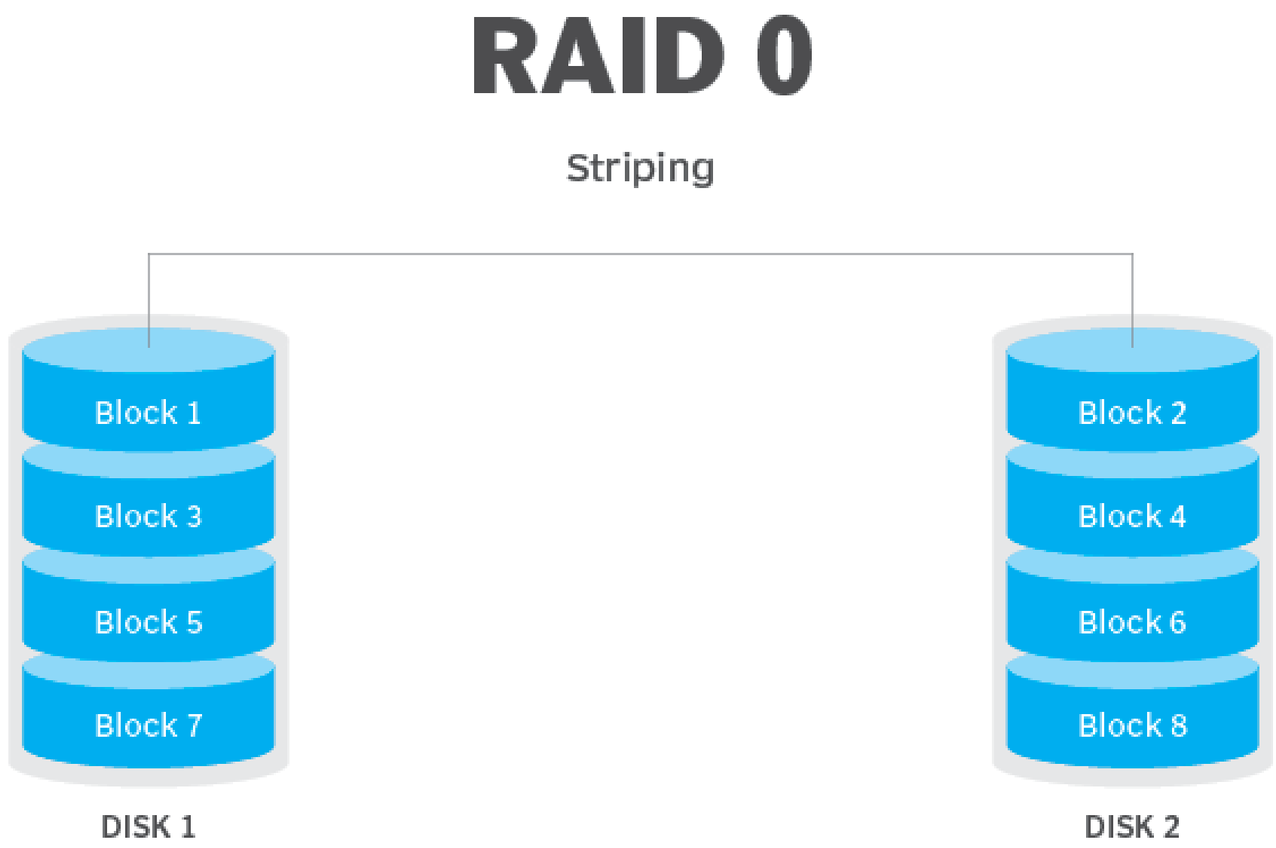
RAID Arrays

- **RAID** (Redundant Array of Independent Disks) combines multiple drives into one virtual device, Improves **data redundancy** and/or **performance (Throughput**: Parallel reads/writes increase speed)

RAID Level	Min Disks	Description	Advantages	Disadvantages
RAID 0	2	Striping without parity	High performance, full capacity	No fault tolerance
RAID 1	2	Mirroring	Simple, good redundancy	50% capacity loss
RAID 5	4	Striping with distributed parity	Good balance of performance and redundancy	Write performance hit
RAID 6	5	Striping with double distributed parity	Can survive two disk failures	More expensive, worse write performance
RAID 10	4	Striped set of mirrors	High performance and good redundancy	50% capacity loss

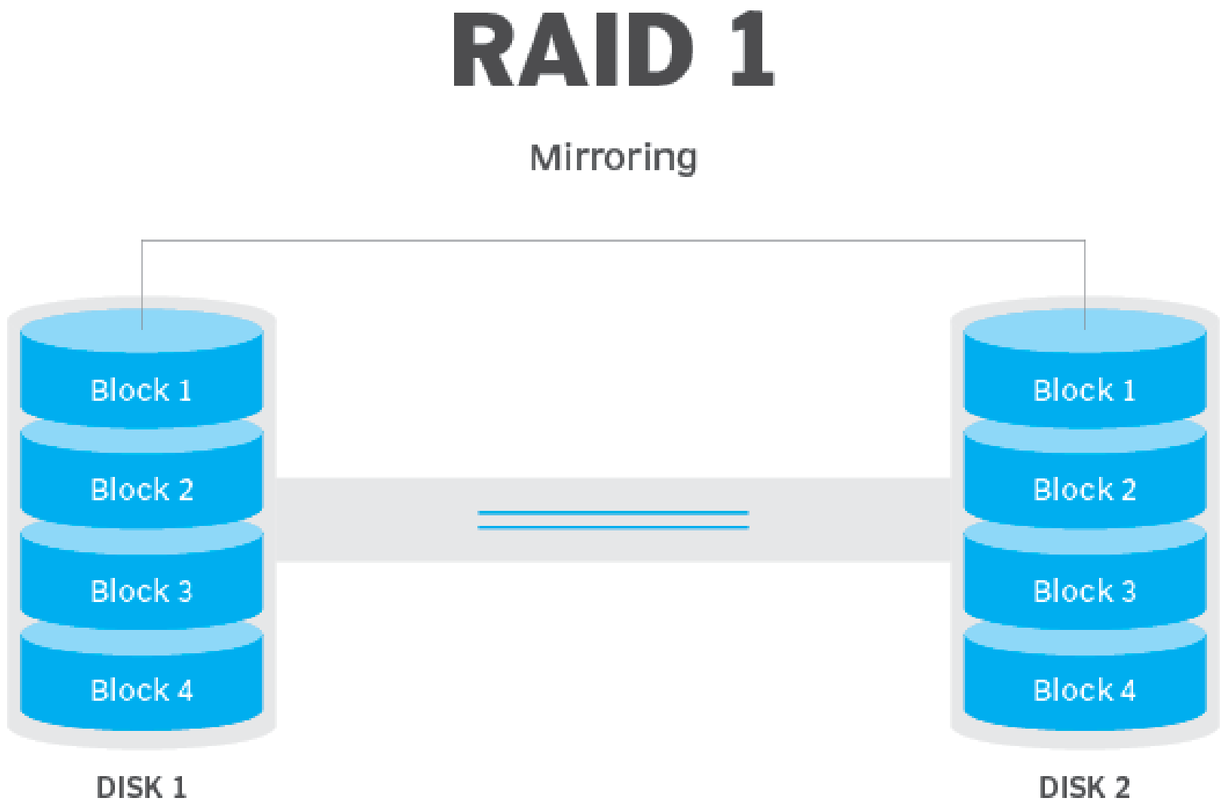
RAID 0

- RAID 0 is based on data striping.
- Advantages:
  - Performance boost for read and write operations
  - Space is not wasted as the entire volume of the individual disks are used up to store unique data
- Disadvantages
  - There is no redundancy/duplication of data. If one of the disks fails, the entire data is lost.



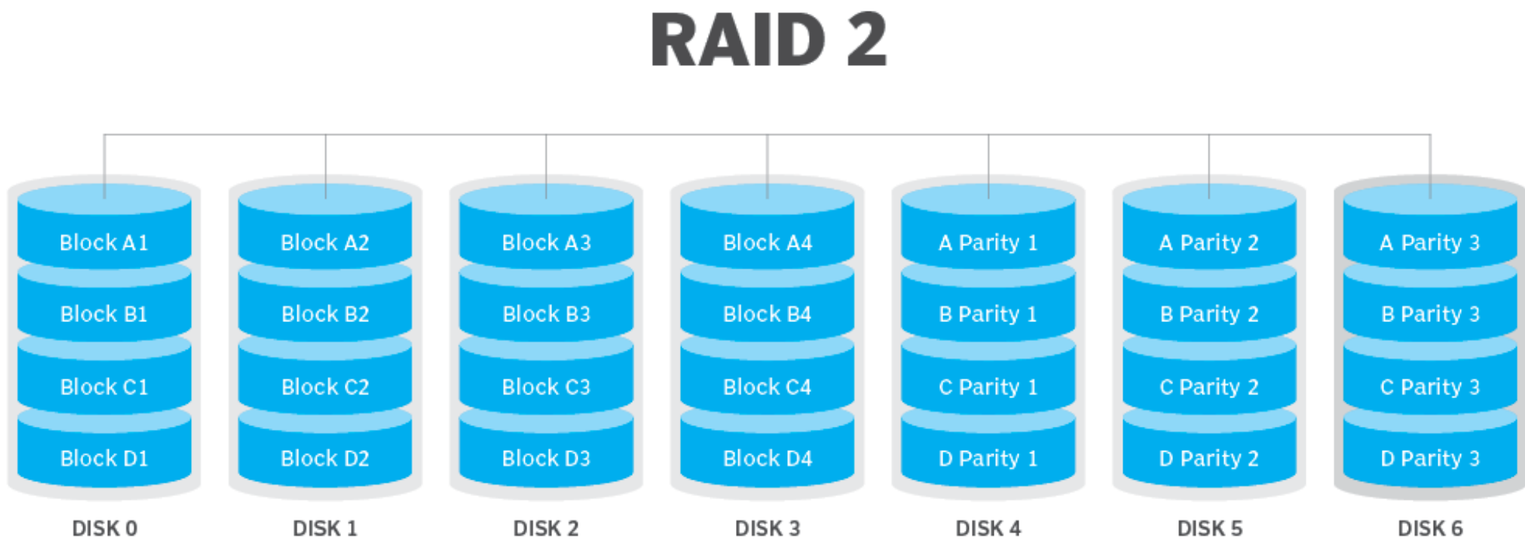
RAID 1

- RAID 1 uses the concept of data mirroring.
- Advantages
  - Data can be recovered in case of disk failure
  - Increased performance for read operation
- Disadvantages
  - Slow write performance
  - Space is wasted by duplicating data which increases the cost per unit memory



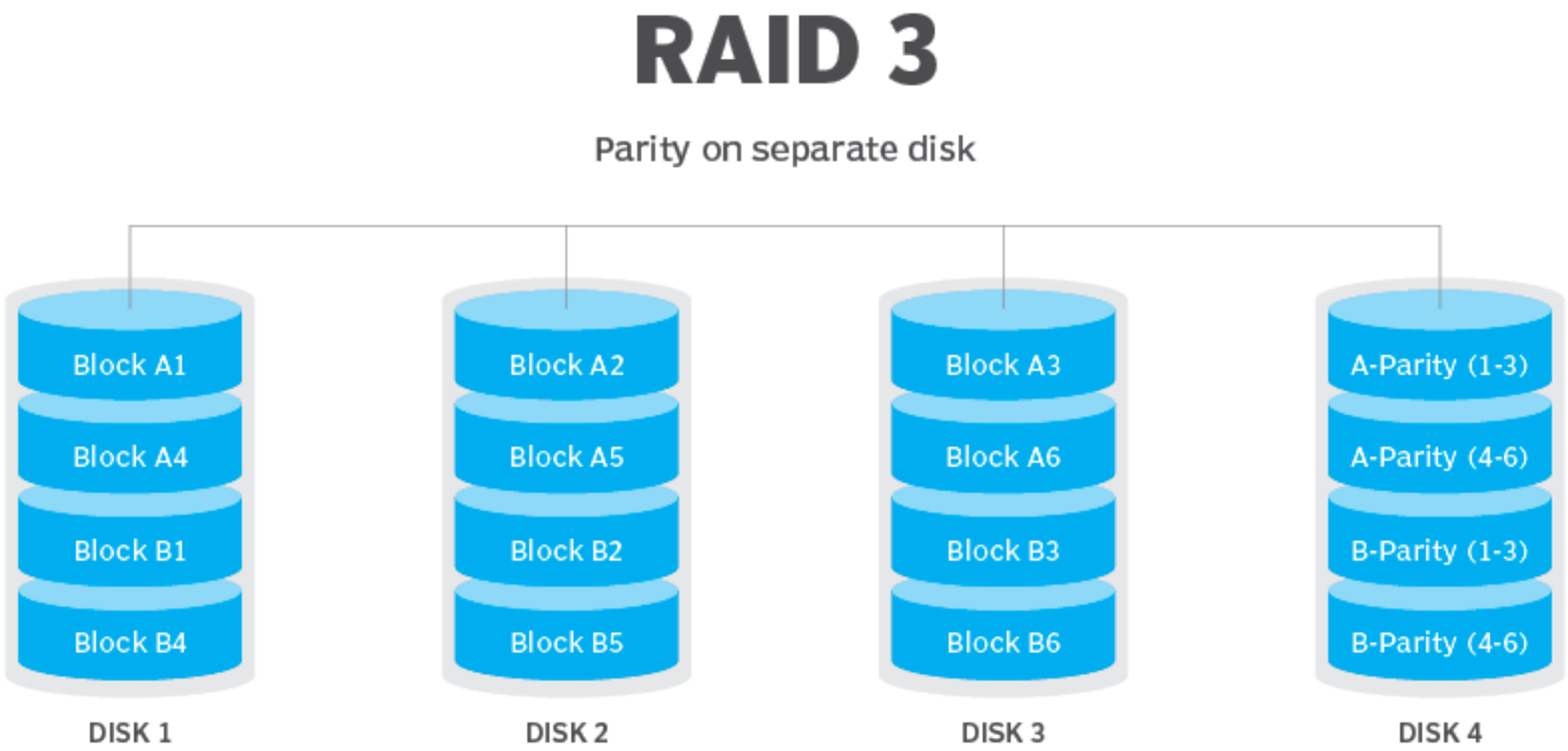
RAID 2: Bit-Level Striping with Dedicated Hamming-Code Parity

- Advantages
  - Reliability.
  - The ability to correct stored information.
- Disadvantages Expensive.
  - Difficult to implement.
  - Require entire disks
  - for ECC.



RAID 3

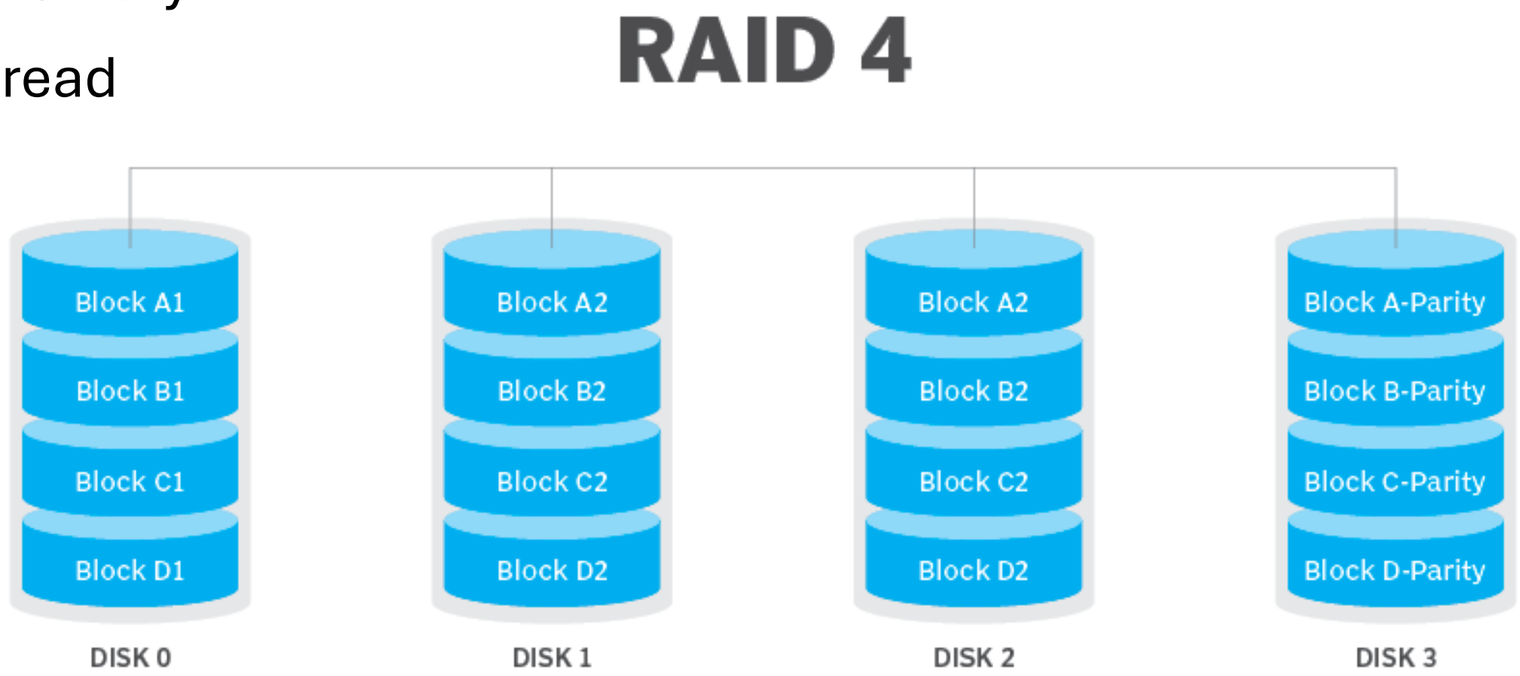
- RAID 3 uses striping and dedicates one drive to store parity information. The embedded ECC information is used to detect errors.



RAID 4

- RAID 4 stripes the data across multiple disks just like RAID 0.
- Advantages
  - Efficient data redundancy in terms of cost per unit memory
  - Performance boost for read operations due to data stripping

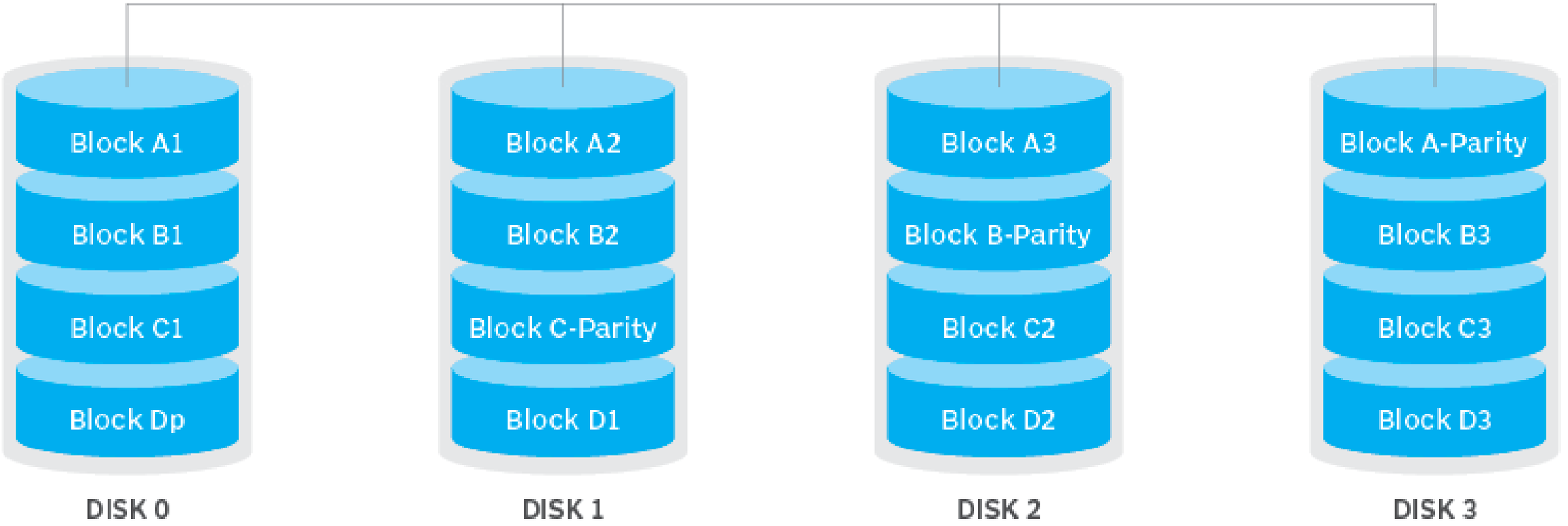
- Disadvantages
  - Write operation is slow
  - If the dedicated parity disk fails, data redundancy is lost



RAID 5

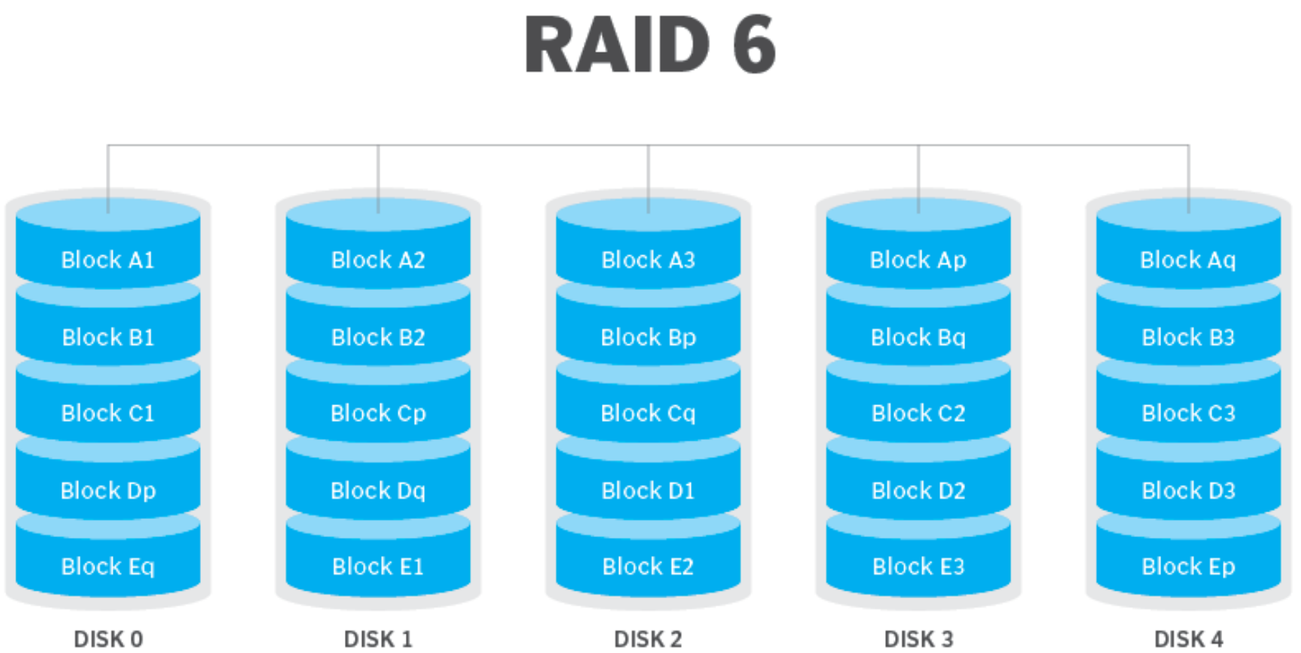
- RAID 5 is very similar to RAID 4, but here the parity information is distributed over all the disks instead of storing them in a dedicated disk.
- Advantages
  - All the advantages of RAID 4 plus increased write speed and better data redundancy
- Disadvantages
  - Can only handle up to a single disk failure

RAID 5


Block A1
Block B1
Block C1
Block Dp
Block A2
Block B2
Block C-Parity
Block D1
Block A3
Block B-Parity
Block C2
Block D2
Block A-Parity
Block B3
Block C3
Block D3
DISK 0
DISK 1
DISK 2
DISK 3

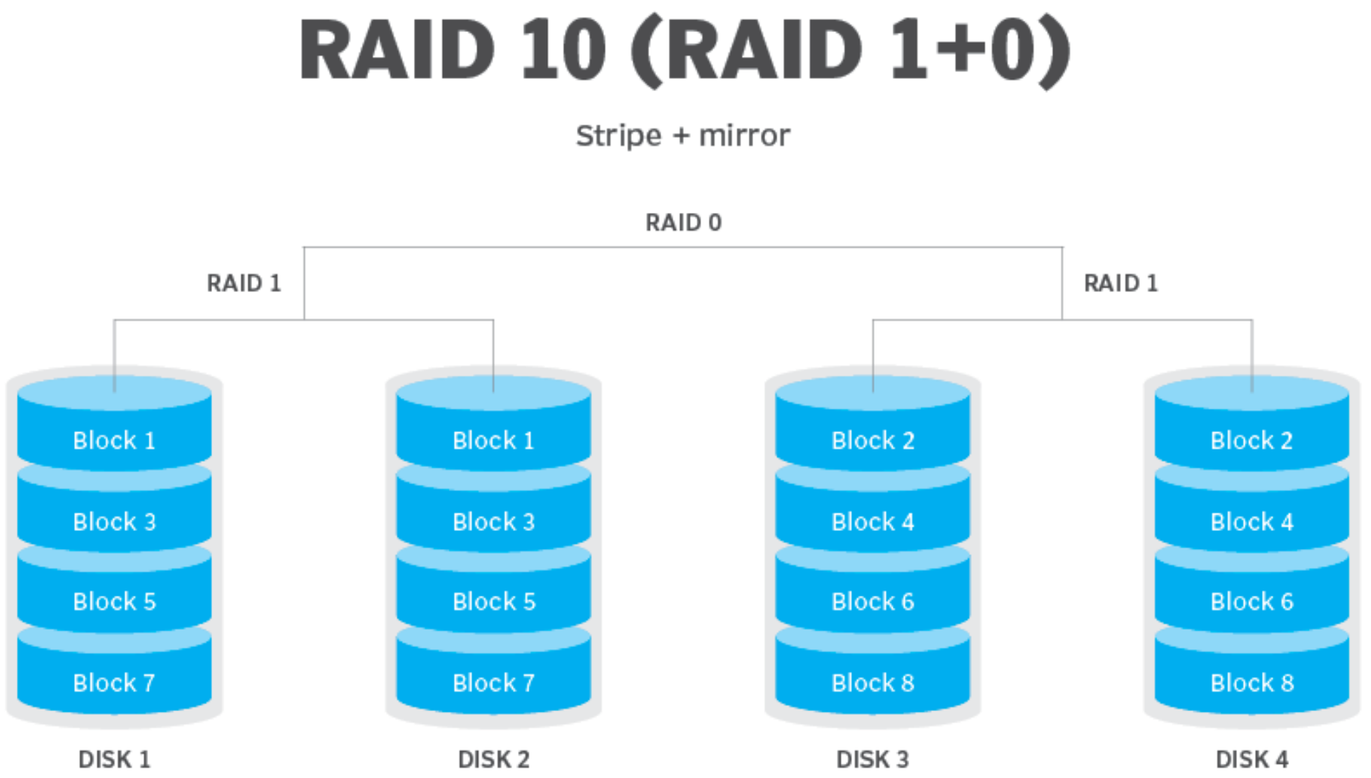
RAID 6

- RAID 6 uses double parity blocks to achieve better data redundancy than RAID 5.
- Advantages
  - Better data redundancy. Can handle upto 2 failed drives
- Disadvantages
  - Large parity overhead



RAID 10 (RAID 1+0)

- RAID 10 combines both RAID 1 and RAID 0 by layering them in opposite order.
- Advantages
  - Very fast performance
  - Redundancy and fault tolerance
- Disadvantages
  - Cost per unit memory is high since data is mirrored



Backup strategy

Build your Backup strategy!

Deploy Open-Source Hypervisor (ProxMox installation on a VM, for knowing it only, it will not be able to support nested VM in most of CPUs)