University of Pisa

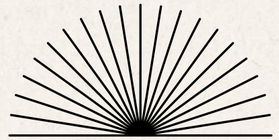
HOW NETFLIX AND YOUTUBE SCALE THEIR SYSTEMS

A Survey on Scalable Architectures in Modern Streaming Platforms

MATRICOLA:
702814

PRESENTED BY:
Zhanarys Zadagerey

PRESENTED TO:
Prof. Patrizio Dazzi

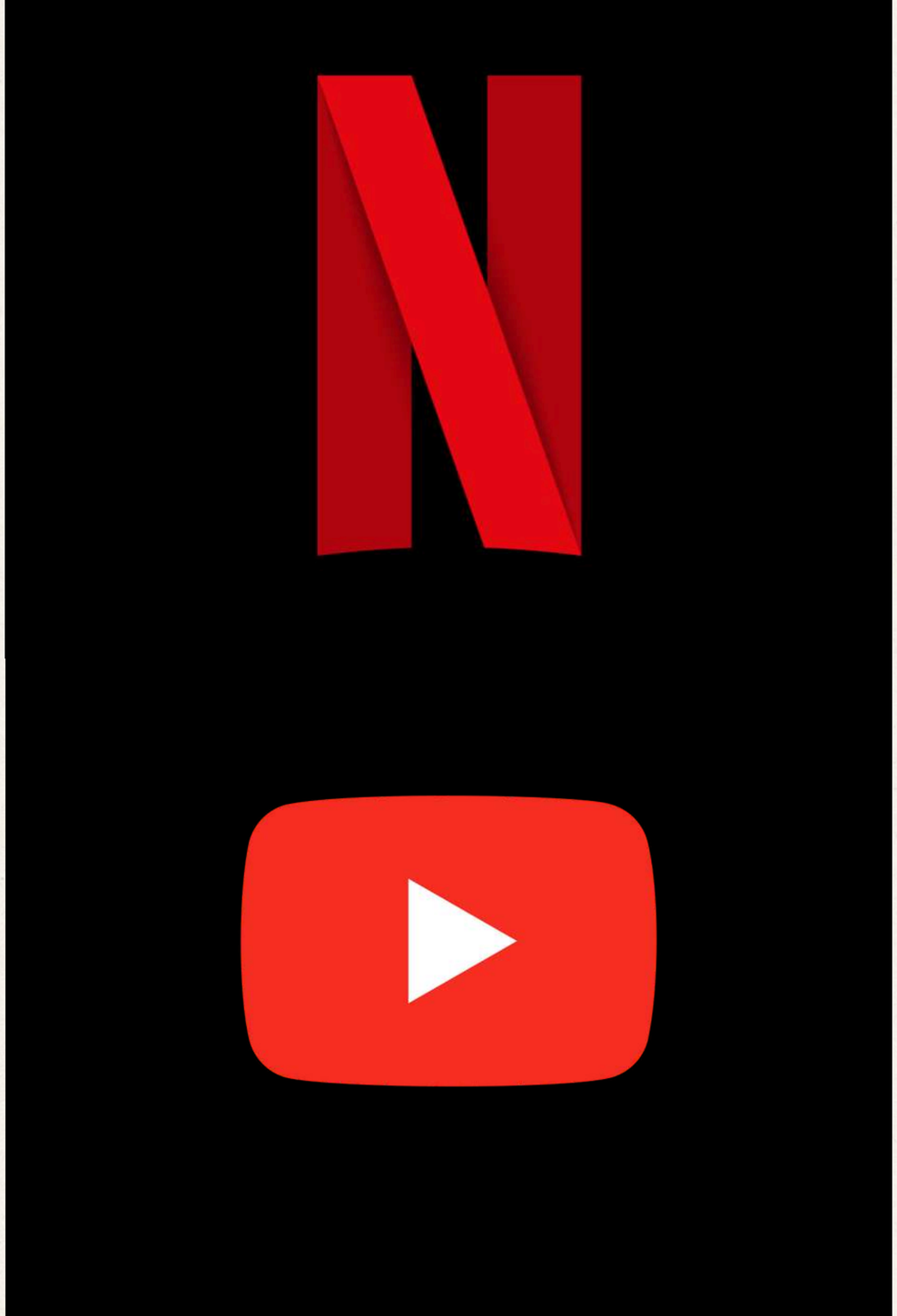


Motivation

01 Streaming platforms serve billions of users

02 Very strict latency and availability requirements

03 Global scale and unpredictable traffic



Scalability & Core Concepts

In streaming systems, scalability refers to the ability to manage growing workloads by adding more resources, while maintaining consistent performance and without needing major architectural changes.

Horizontal vs vertical scaling

- Load balancing
- Autoscaling
- Replication
- Caching
- Fault tolerance

These are the main concepts that was discussed in the first part of the report, Youtube & Netflix use them in different ways, which we will see further.

Why Streaming Is Difficult ?

These platforms must operate with **very low latency.**

Streaming workloads are interactive and time-sensitive, so the system must maintain a high level of **Quality of Service (QoS)** at all times.

Streaming **traffic is also highly unpredictable** and can increase suddenly.

Streaming services must deliver content to users **around the world**, across different time zones and network conditions such as 5G, mobile networks, Wi-Fi or fiber connections.

With millions of users active at the same time, system **failures** are unavoidable

Case Study: Netflix

High-level Architecture

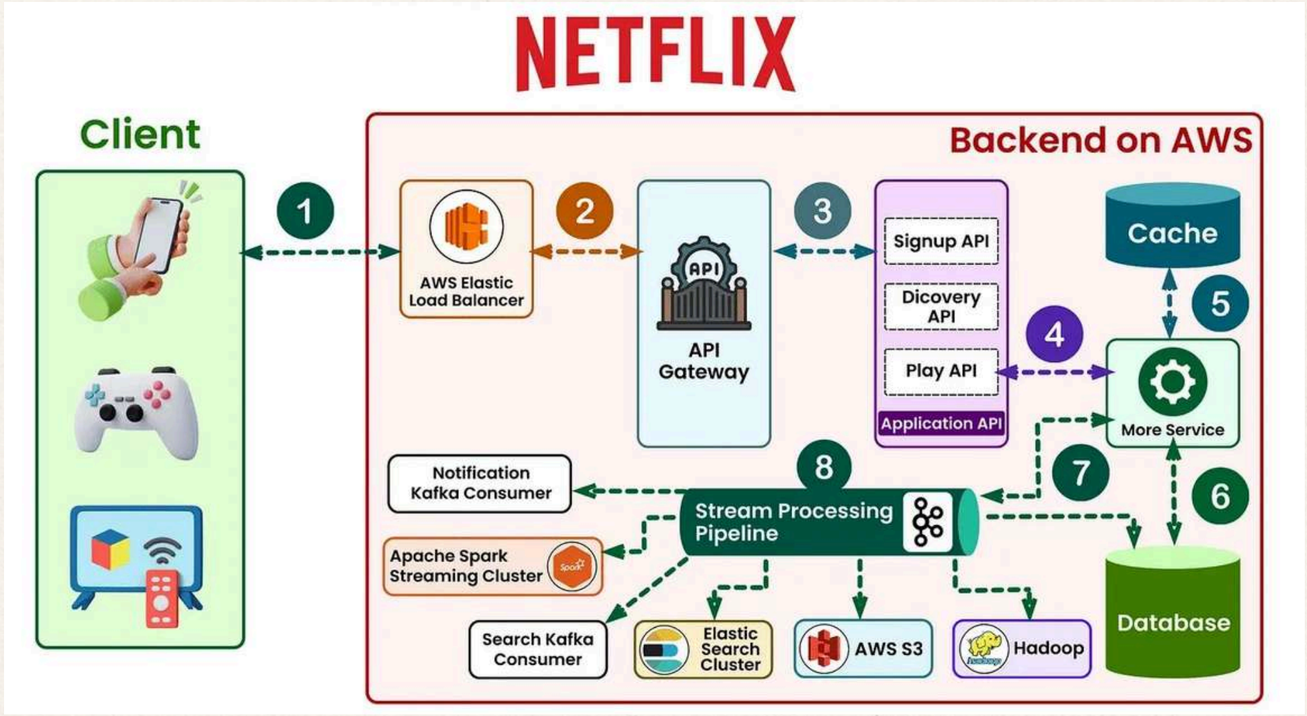
Netflix is built as a large-scale distributed system based on a **cloud-native microservices architecture** running on **Amazon Web Services** across multiple geographic regions. **Amazon S3** serves as a storage layer

Elastic Compute Cloud (EC2) is used to provide scalable virtual machines. **Elastic Load Balancing** is employed to distribute incoming traffic across frontend services and prevent overload.

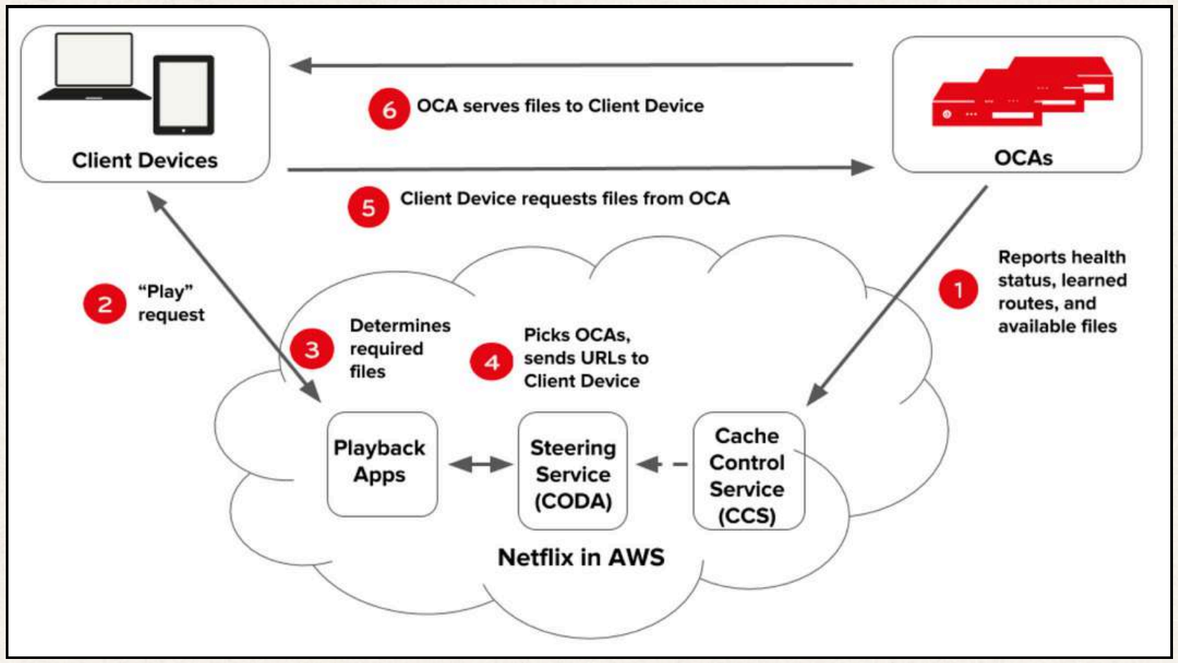
For real-time data streaming, Netflix relies on platforms such as **Kafka** and **Amazon Kinesis.**

For large-scale data storage, Netflix uses **Cassandra**

Netflix also uses **AWS Lambda** to run serverless functions



The following diagram illustrates how the playback process works:



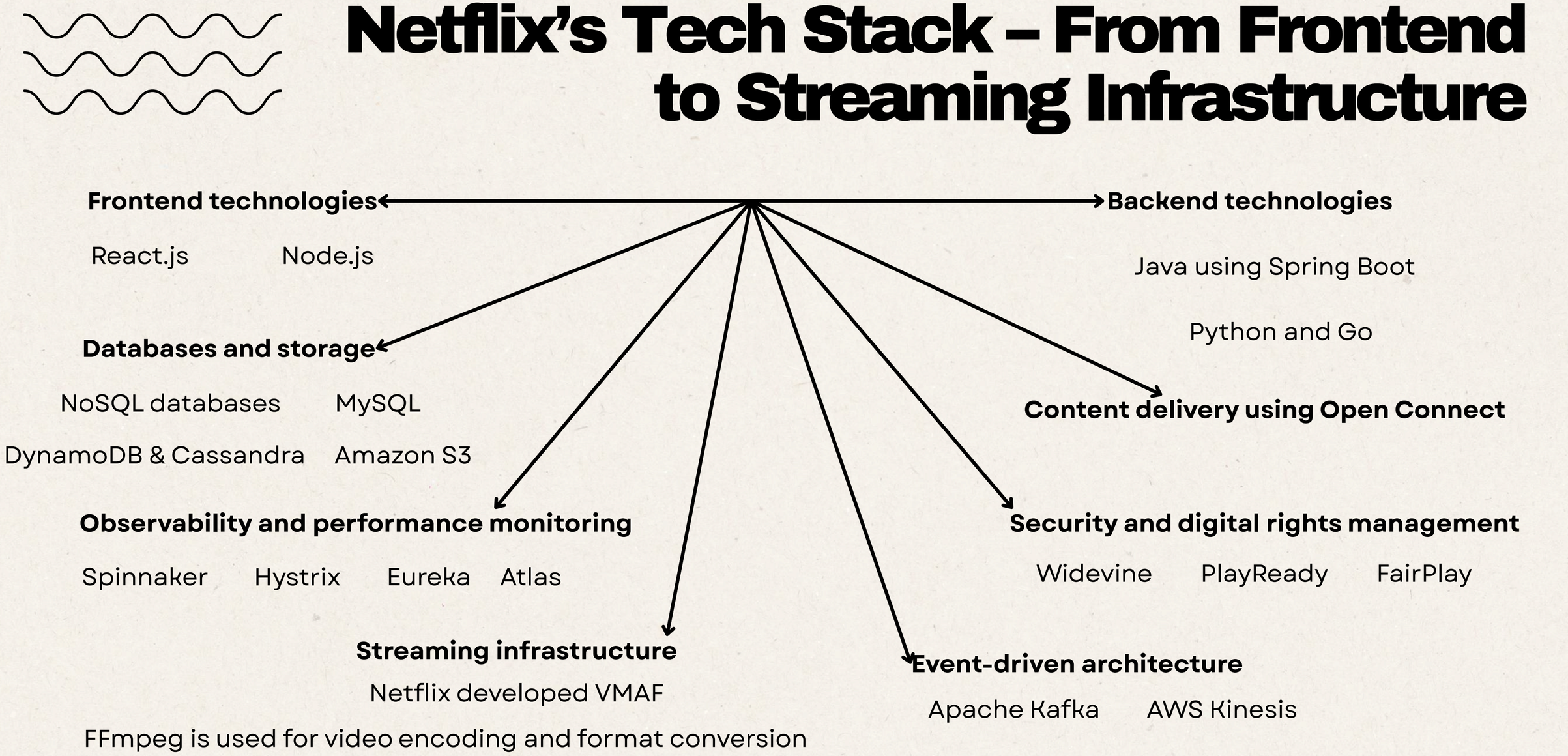
Content Delivery at Scale – Open Connect

Netflix developed its own content delivery network called **Open Connect**. This system is responsible for delivering Netflix movies and television shows to users around the world in an efficient and reliable way.

The core components of Open Connect are specialized servers known as **Open Connect Appliances**.
Open Connect Appliances are deployed globally in two main ways.

1. Netflix installs them at internet exchange points in major markets
2. Netflix provides these appliances free of charge to qualifying ISPs, which then deploy them directly within their own networks.

Netflix's Tech Stack – From Frontend to Streaming Infrastructure



Resilience and Fault Tolerance – Chaos Engineering

Netflix maintains high availability through the use of **Chaos Engineering**, an approach that focuses on improving system resilience by intentionally testing how systems behave when failures occur.

Chaos Monkey is a well-known open-source tool created by Netflix to apply Chaos Engineering principles in distributed systems.

Purpose of Chaos Monkey:

Resilience testing

Encouraging redundancy

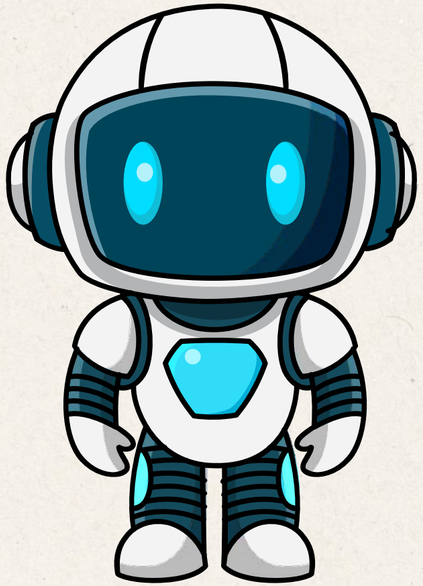
Continuous improvement

Identifying weaknesses

Promoting a resilience-focused culture

Building confidence

Personalisation & AI – The Brain Behind Netflix Recommendations

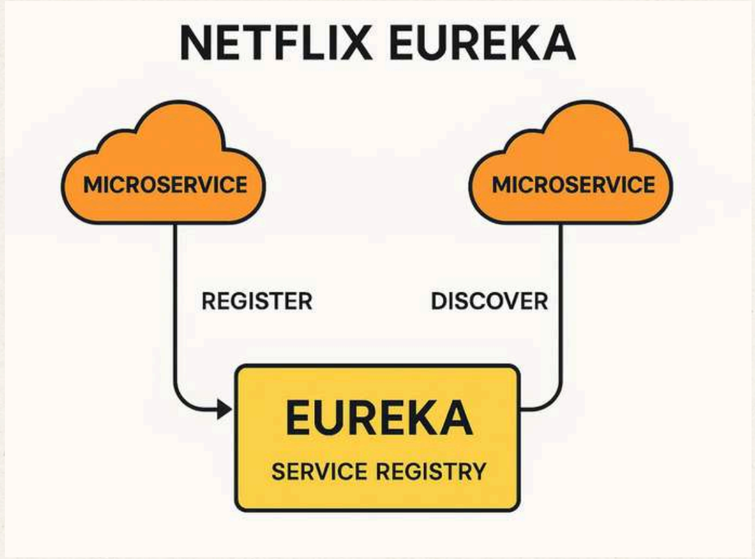


Netflix relies heavily on **artificial intelligence** and **machine learning** to deliver personalized recommendations to its users.

One of the core techniques used is **collaborative filtering**

Content-Based filtering which focuses on analysing the properties of the content itself.

Netflix also uses **deep learning models** to capture more complex viewing patterns



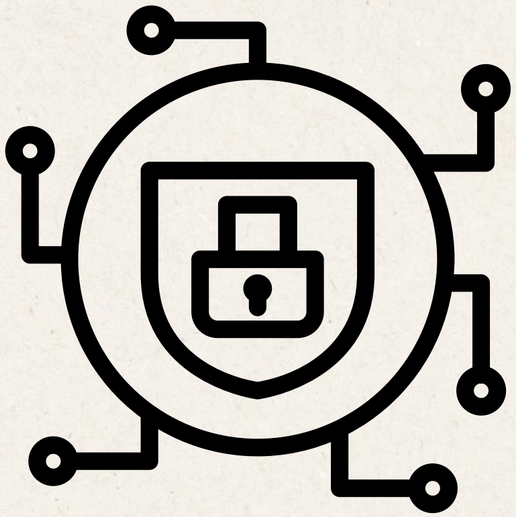
Monitoring

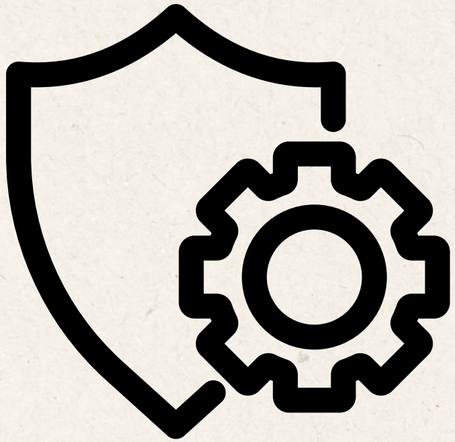
Atlas – Netflix's real-time telemetry platform for **tracking system health**.

Eureka – Service Discovery Tool for Microservice Communication.

Hystrix – Circuit Breaker Library for Preventing Cascading Failures

Spinnaker is an open-source, multi-cloud continuous delivery platform used to deploy software changes quickly and reliably.





Security & Privacy

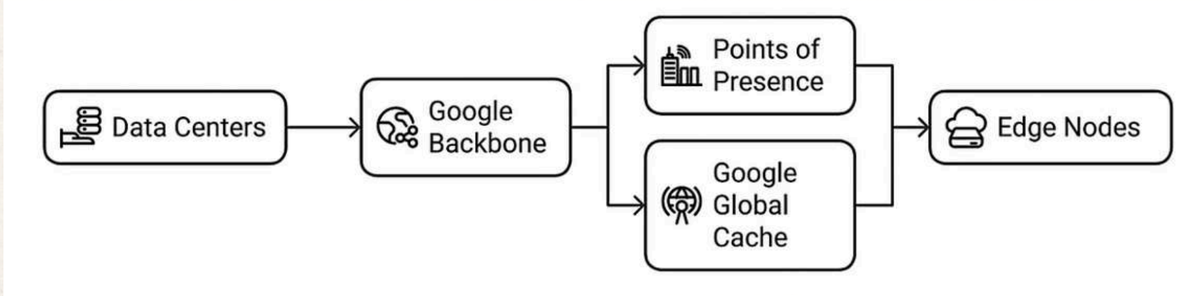
- **End-to-End Encryption:** Ensures that user information and streamed content are protected from unauthorized access during transmission.
- **Multi-Factor Authentication (MFA):** Adds an extra layer of protection to user accounts, reducing the risk of account takeovers.
- **Access Control and Role-Based Policies:** Limits internal access to sensitive systems, allowing employees to access only what is necessary for their roles.
- **Digital Rights Management (DRM):** Uses encryption and watermarking technologies to prevent illegal copying and distribution of content.
- **Bot Detection and Fraud Prevention:** Detects and blocks automated attacks such as credential stuffing and misuse related to account sharing.

Case Study: Youtube

YouTube, the world's largest video-sharing platform, serves billions of hours of video content daily to over 2 billion users worldwide.

At the core of this system is **Google's Content Delivery Network (CDN)**, specifically the **Media CDN**, built on the same technology that powers YouTube.

Google's Global Infrastructure



Google's private backbone network spans millions of kilometers of fiber-optic cables, including several submarine cables such as Curie, which connects the United States and Chile, and Equiano, linking Europe and Africa.

Points of Presence (PoPs) are located in over 200 cities across more than 90 countries.

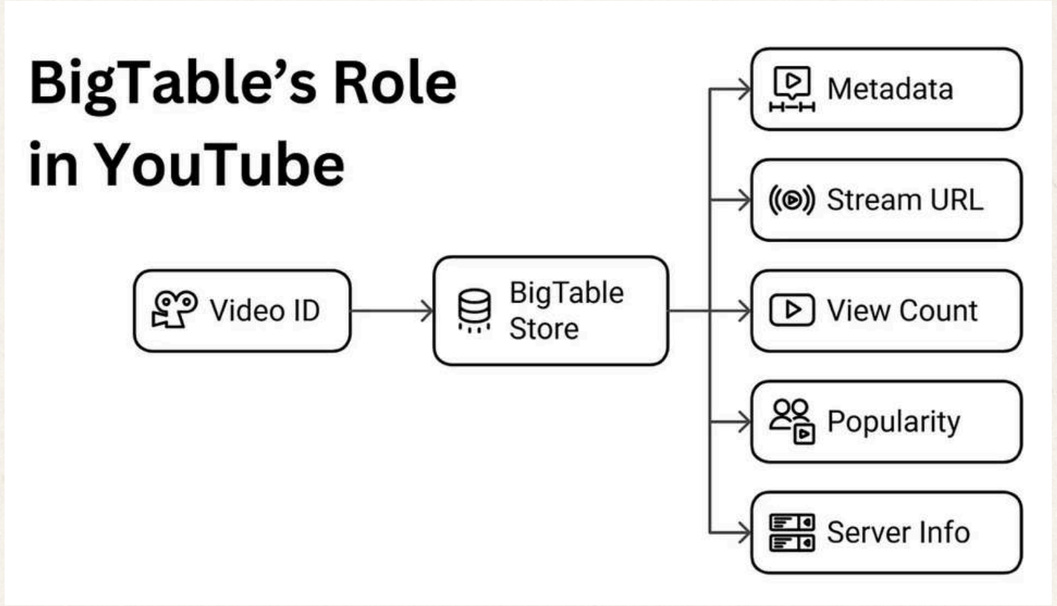
Google Global Cache (GGC) are caching servers installed directly inside ISP networks.

Google deploys **edge nodes** that handle tasks such as request routing, load balancing, and caching decisions in real time.

The Role of Google BigTable in YouTube's Infrastructure

YouTube

Google Bigtable plays an important role in YouTube's infrastructure by providing a highly scalable and low-latency storage system for large amounts of metadata.
It is a distributed NoSQL database built to handle very high read and write throughput, managing petabytes of data across thousands of machines.

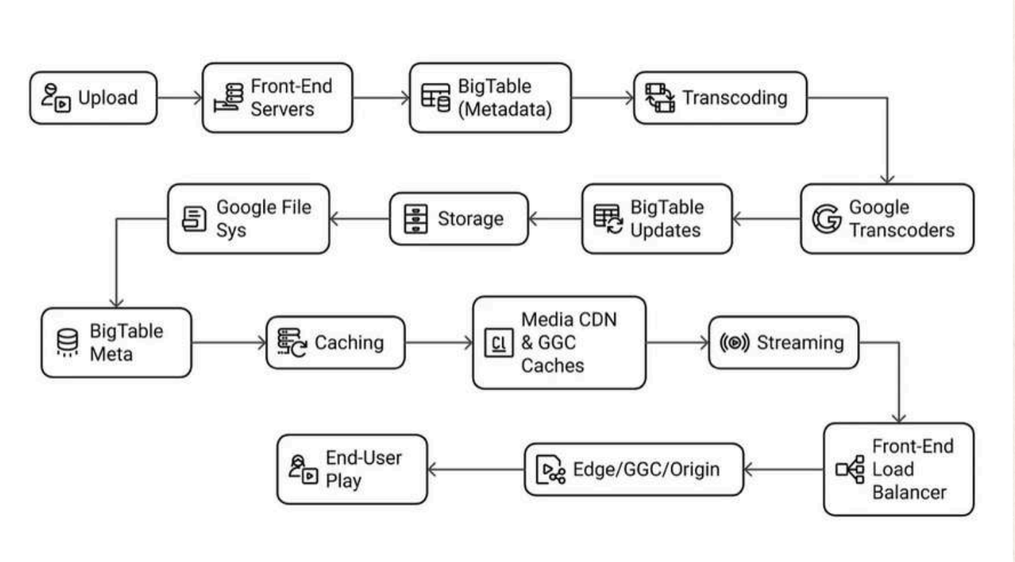


Metadata storage: Video metadata is stored and indexed using the video ID, enabling fast and efficient lookups.
Content retrieval: The stored metadata helps locate the actual video files and identify available cached copies within the delivery infrastructure.

Server selection: BigTable supports CDN and load-balancing logic by providing information about server availability and geographic proximity.
Dynamic updates: It continuously records changing data, such as view counts and popularity signals, which influence caching and delivery decisions.

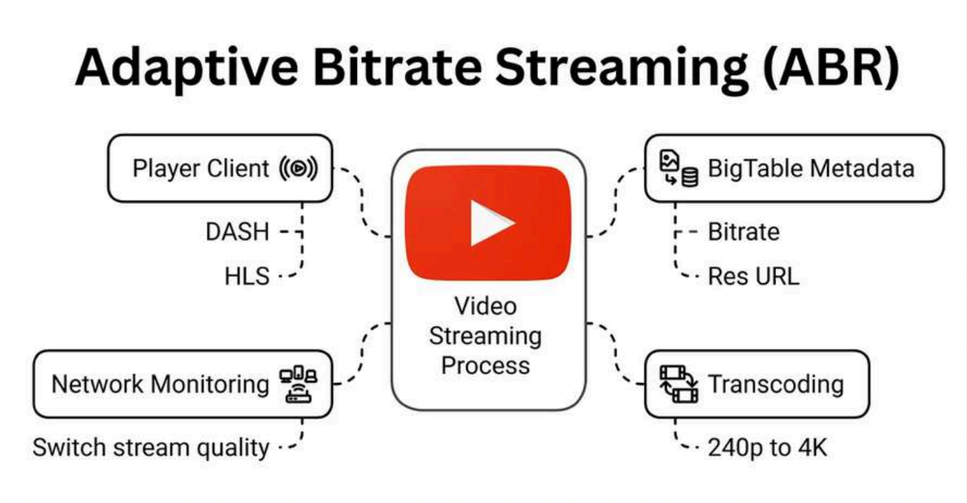
YouTube's Video Processing

When a video is uploaded to YouTube, it goes through a structured processing pipeline that prepares it for storage and delivery to viewers.
This process is often referred to as the video lifecycle and includes five main stages: upload, transcoding, storage, caching, and streaming.



1. **Upload:** A user uploads a video through YouTube's web or mobile application
2. **Transcoding:** After the upload is complete, the video is processed by Google's transcoding infrastructure
3. **Storage:** Once transcoding is finished, the video files are stored in the Google File System, a distributed file system designed to handle very large amounts of data.
4. **Caching:** Videos that become popular are cached closer to users on Media CDN edge servers and Google Global Cache nodes, which are deployed in thousands of locations worldwide
5. **Streaming:** When a user plays a video, YouTube's front-end servers query BigTable to retrieve the required metadata and determine the best source for streaming, whether that is an edge server, a GGC node, or a central data center.

Adaptive Bitrate Streaming (ABR)



How ABR works

Encoding: During the transcoding phase, each video is converted into several versions with different resolutions and bitrates

Dynamic adjustment: The YouTube video player uses streaming protocols such as (DASH) or (HLS) to continuously monitor network speed and device performance.

Implementation: When a user starts watching a video, BigTable provides metadata about all available stream versions.

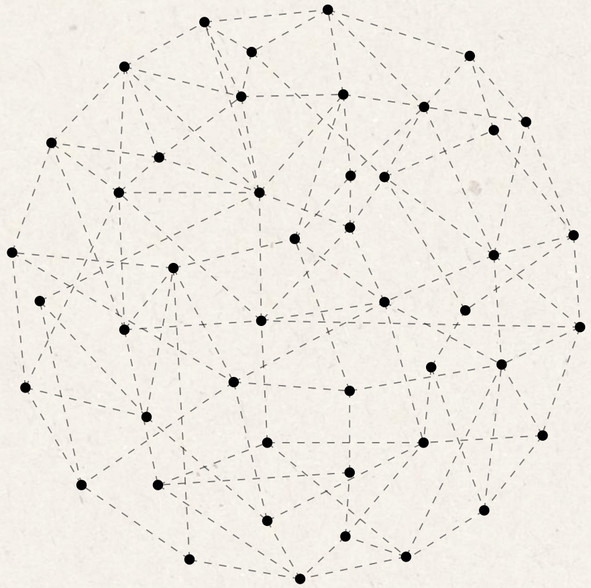
Benefits: ABR helps avoid buffering on unstable connections and uses bandwidth more efficiently.

Network Optimization Technologies

QUIC Protocol: YouTube uses Google's QUIC (Quick UDP Internet Connections) protocol to improve video streaming performance. Compared to traditional TCP, QUIC provides several important improvements:

- Faster connection setup
- More efficient congestion control
- Better performance on unstable or lossy networks
- Support for multiple streams without head-of-line blocking

YouTube relies on Google's **private fiber optic network**, which spans more than 1.5 million miles of cable worldwide.



Storage Infrastructure & YouTube's CDN Operations:

Technical Details of Media CDN

1. Distributed File System: YouTube stores its large video library, consisting of billions of terabytes, using Google's distributed file system built on consumer- and business-grade hard drives.

2. Modular Data Centers: Most YouTube data is stored in Google's modular data centers. These are portable, self-contained units that can be deployed quickly where more capacity is required.

YouTube's CDN Operations: YouTube's content delivery network, based on Google's Media CDN, improves video delivery through several mechanisms.

1. Content Distribution: Popular videos & Less popular videos

2. Load Balancing and Server Selection: Server selection uses metadata stored in BigTable

3. Content Propagation: Newly uploaded videos are first stored on main servers

Advanced Techniques & Conclusion

1. Caching and Latency Reduction

2. Fault Tolerance and Chaos Testing

3. Security and DDoS Protection

In Conclusion, YouTube’s capacity to deliver billions of videos each day is supported by Google’s advanced CDN infrastructure.

Comparative Analysis Table

Aspect	**Netflix**	**YouTube**
Core architectural model	Cloud-native microservices architecture deployed on AWS. Each service (authentication, recommendations, playback)	Distributed service-oriented architecture built on Google's internal infrastructure (Google Cloud and custom systems).
Hosting and infrastructure	Hosted entirely on public cloud (Amazon Web Services), relying on EC2, S3, DynamoDB, etc., for compute, storage,	Hosted on Google's private, global infrastructure. Core services and data stores run on Google's internal platforms
Service communication and orchestration	Services communicate via APIs (REST/gRPC) with centralized service discovery and client-side load balancing	Services communicate through internal RPC, distributed pipelines, and Google's internal infrastructure protocols.
Scaling strategy	Horizontal auto-scaling of stateless microservices using AWS auto-scaling and regional replication.	Horizontal scaling via Borg (Google's container manager) and global infrastructure, with dynamic allocation of
Content delivery and caching	Uses proprietary CDN ("Open Connect") placed at ISP edge locations for efficient content delivery; caches popular	Uses Google's global CDN (Google Global Cache) with multi-layer caching: edge, regional, and origin caches to
Video ingestion and processing	Videos are pre-encoded and stored in cloud; Netflix creates multiple bitrate versions ahead of time.	Continuous upload pipeline: users upload videos which are immediately transcoded into multiple formats and stored;
Fault tolerance and resilience	Designed for failure with fault isolation per microservice; uses chaos engineering tools (e.g., Chaos Monkey) to	High redundancy via data replication and global load balancing. YouTube's distributed system redirects traffic
Data consistency approach	Eventual consistency for most distributed services, with stronger local consistency where required (in caches and	Eventual consistency for global caches and metadata; strong consistency enforced selectively where needed
Analytics and machine learning	Extensive use of ML for personalization, content recommendation, and quality-of-experience optimization.	Uses large-scale ML for ranking, search quality, recommendation, content moderation, and ad targeting

Challenges

Traffic spikes

Consistency vs availability

Observability complexity

Cost management

Even with advanced architectures, scaling remains challenging. Trade-offs are always present.

Conclusion

Netflix and YouTube are real-world distributed systems

Same principles, different implementations

Course concepts apply directly to industry systems

To conclude, scalable distributed computing is not abstract theory. It is actively used in systems that we interact with every day.

Thank you

Questions ?

References that was used during the **Report preparation & Presentation** are written in the References section of Report with links.

CONTACTS:

Student: Zhanarys Zadagerey

E-mail z.zadagerey@studenti.unipi.it

Phone +39 351 492 1652