CDN System Design Interview Questions & Answers

Comprehensive Guide

Table of Contents

1️⃣ CDN Basics

1. What is a CDN, and how does it work?

A **Content Delivery Network (CDN)** is a **distributed network of servers** that work together to deliver content to users efficiently. The primary goal of a CDN is to **reduce latency and load times** by caching content in multiple geographically distributed servers called **edge servers** or **PoPs (Points of Presence)**.

How It Works:

1. A user requests content (e.g., an image, video, or webpage).
2. The request is directed to the nearest CDN **edge server** based on factors like **geographic location, network latency, and server load**.
3. If the requested content is already **cached** on the edge server (**Cache Hit**), it is delivered instantly.

4. If the content is not cached (**Cache Miss**), the request is forwarded to the **origin server**, fetched, and stored at the edge server for future requests.

---

2. Why do we need CDNs in system design?

Without a CDN, content is served directly from the **origin server**, which leads to:

- **High latency** due to geographic distance between the user and the server.
- **Overloaded origin servers**, causing slower responses and downtime.
- **Bandwidth constraints**, leading to slow page load times and higher operational costs.
- **Security risks**, including DDoS attacks and malicious traffic.

CDNs **solve these problems** by **distributing** traffic across multiple edge locations, **caching frequently accessed content**, and **protecting against cyber threats**.

---

3. What are the key benefits of using a CDN?

✅ **Reduced Latency** – Faster content delivery by serving from the closest PoP.
✅ **Lower Bandwidth Costs** – Caching reduces requests to the origin, minimizing data transfer costs.
✅ **Increased Availability & Load Balancing** – Distributes traffic across multiple servers, preventing overload.
✅ **Enhanced Security** – Protects against **DDoS attacks**, **traffic filtering**, and **SSL/TLS encryption**.

2️⃣ CDN Architecture & Components

4. Explain the difference between an origin server and an edge server in a CDN.

- **Origin Server:** The central server that hosts the original content. It is responsible for serving content **when a cache miss occurs**.
- **Edge Server (PoP):** A geographically distributed server that caches content closer to users to reduce latency.

💡 **Analogy:** The **origin server** is like a **main warehouse**, while **edge servers** are **local distribution centers** that store frequently accessed goods.

---

5. What is a PoP (Point of Presence) in a CDN?

A **PoP (Point of Presence)** is a **CDN data center** that contains **edge servers** to store and serve cached content to users. The more PoPs a CDN has, the **faster and more reliable** the content delivery.

---

6. How does request routing work in a CDN?

CDNs use various **routing strategies** to direct user requests to the optimal edge server:

- **Geo-Based Routing:** Users are directed to the closest PoP.
- **Latency-Based Routing:** Requests go to the PoP with the lowest network latency.
- **Load-Aware Routing:** Traffic is balanced across multiple PoPs to prevent overload.

3️⃣ Caching Strategies & Content Delivery

7. What is a cache hit vs. cache miss, and how does a CDN handle them?

- **Cache Hit:** The requested content is found in the CDN cache and served immediately.
- **Cache Miss:** The content is not in the cache, so it is fetched from the origin, stored at the edge, and then served to the user.

---

8. Explain cache expiration and TTL (Time-To-Live) in a CDN.

- **TTL (Time-To-Live):** Defines how long content stays cached before it expires.
- **CDN Cache Expiration:** Once TTL expires, the CDN fetches updated content from the origin.

---

9. What are cache invalidation strategies, and why are they important?

- **Manual Purge:** Manually removes outdated content.
- **Stale-While-Revalidate:** Serves stale content while fetching fresh content in the background.
- **Versioning:** Appends version numbers (e.g., `image_v2.png`) to force cache updates.

4️⃣ Load Balancing & Failover Handling

10. How do CDNs use load balancing to improve reliability?

CDNs distribute traffic across multiple PoPs using:
✅ **Round-robin balancing**
✅ **Latency-based routing**
✅ **Geo-based routing**

---

11. What happens if a CDN PoP fails?

The CDN automatically **reroutes traffic** to the next best PoP, ensuring uninterrupted service.

5️⃣ CDN Optimization Techniques

12. What compression and minification techniques do CDNs use?

CDNs optimize content delivery using:

- **Gzip & Brotli Compression** (reduces file size).
- **Minification of CSS/JS** (removes unnecessary characters).
- **Image Optimization** (WebP, AVIF formats).

6️⃣ CDN Security & Challenges

13. How does a CDN protect against DDoS attacks?

- **Rate Limiting:** Blocks excessive requests from a single source.
- **Traffic Filtering:** Identifies and drops malicious requests.
- **Anycast Routing:** Distributes attack traffic across multiple PoPs to absorb the impact.

---

14. What is SSL/TLS offloading, and why is it useful?

SSL/TLS Offloading means the CDN **handles encryption** at the edge, reducing the burden on the origin server.

7️⃣ Advanced CDN Topics

15. How does a multi-CDN architecture work?

A **multi-CDN strategy** uses multiple CDN providers for **better redundancy, performance, and failover handling**.

---

16. How would you design a CDN for a large-scale video streaming platform?

Key design considerations:

- **Segmented Caching** (store different video chunks at different PoPs).
- **Adaptive Bitrate Streaming** (adjusts video quality based on user bandwidth).
- **Load Balancing** (distributes viewers across multiple servers).

Final Thoughts

Understanding **CDN architecture, caching, security, and optimization techniques** will help you design **high-performance, scalable systems**.

💡 **Pro Tip:** Always think about **latency, redundancy, and scalability** when discussing CDNs in system design interviews!