A Crash Course on Content-Delivery Networks (CDN)

BYTEBYTEGO
JUN 06, 2024 · PAID

In the era of the modern web, latency can directly impact an organization's bottom line.

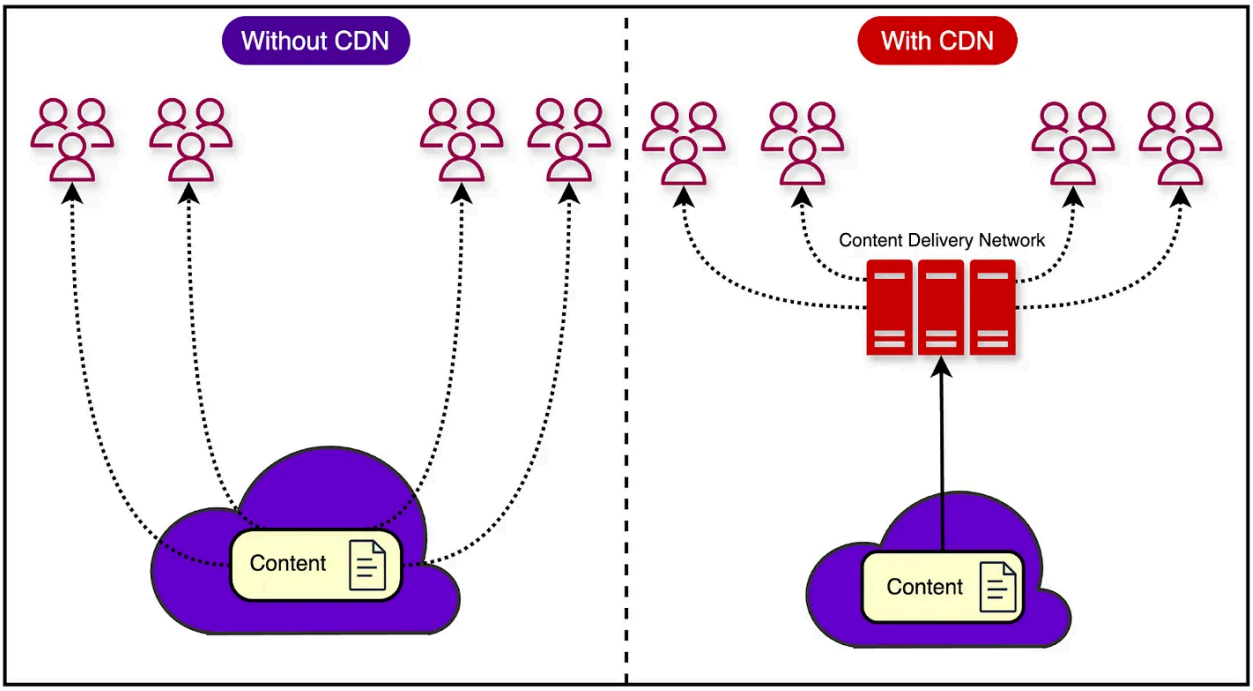
Here are some stats from a study by Portent:

- A page that loads in 1 second has a 3x higher conversion rate than a page that takes 5 seconds to load.
- E-commerce conversion rates decrease by 0.3% for every second of load time.
- The optimal loading time to maximize sales is less than 2 seconds.

One of the most effective strategies to reduce latency is using a Content-Delivery Network or a CDN.

A CDN is a geographically distributed network of servers that work together to deliver fast and reliable content delivery across the globe.

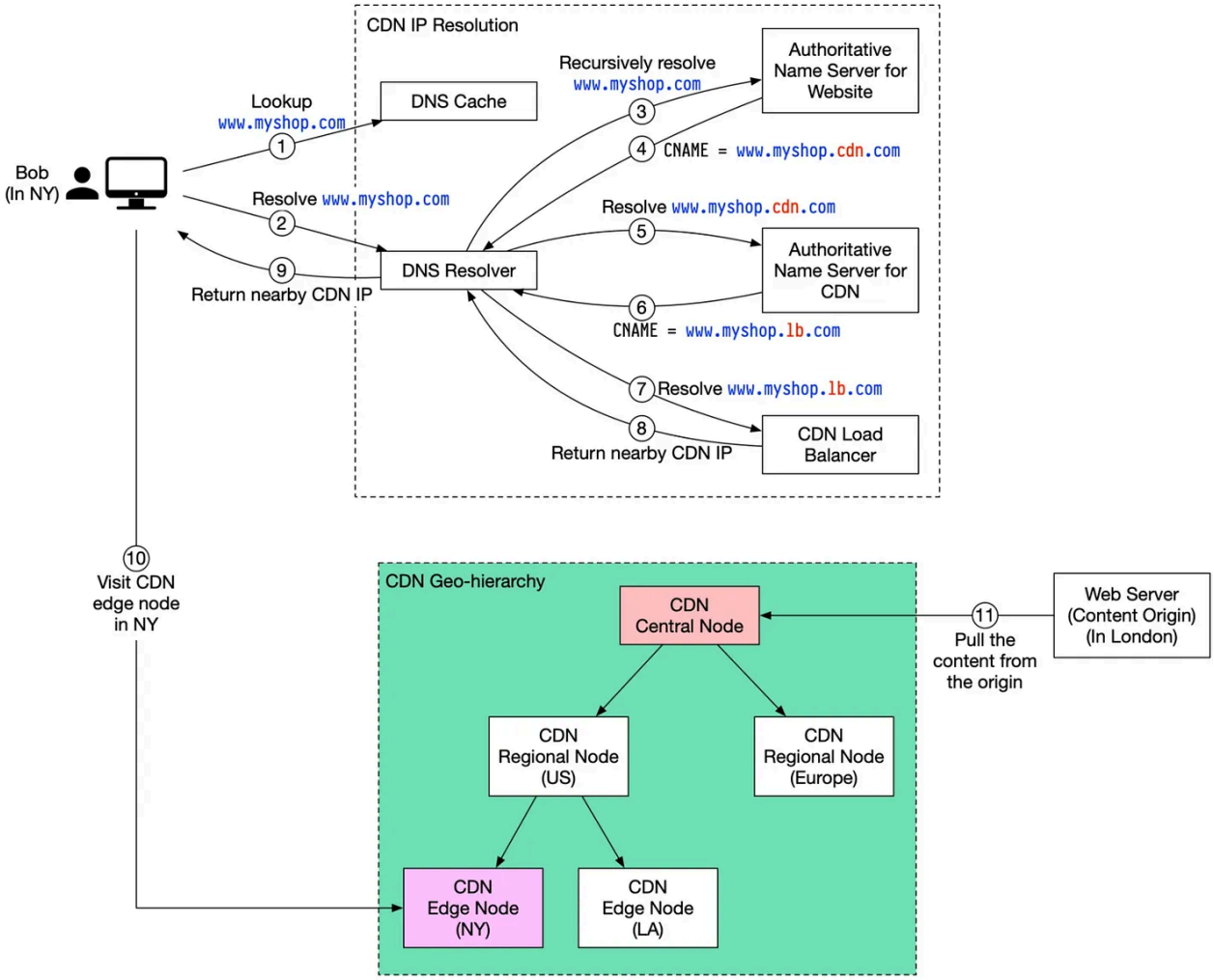
When a user requests content from a website or application that uses a CDN, the request is redirected to the nearest CDN server, which serves the content to the user. This reduces the distance data travels and improves overall performance.



Think of a CDN like an ATM. If money were available only from one bank branch in town, everyone would have to make a time-consuming trip to that branch. With ATMs in every locality, everyone has fast and easy access to money.

The market size for CDN-related solutions is expected to reach nearly $38 billion by 2028, with companies like Akamai, Cloudflare, and Amazon CloudFront investing heavily in improving their CDN offerings.

How does CDN work



Why the Need for a CDN?

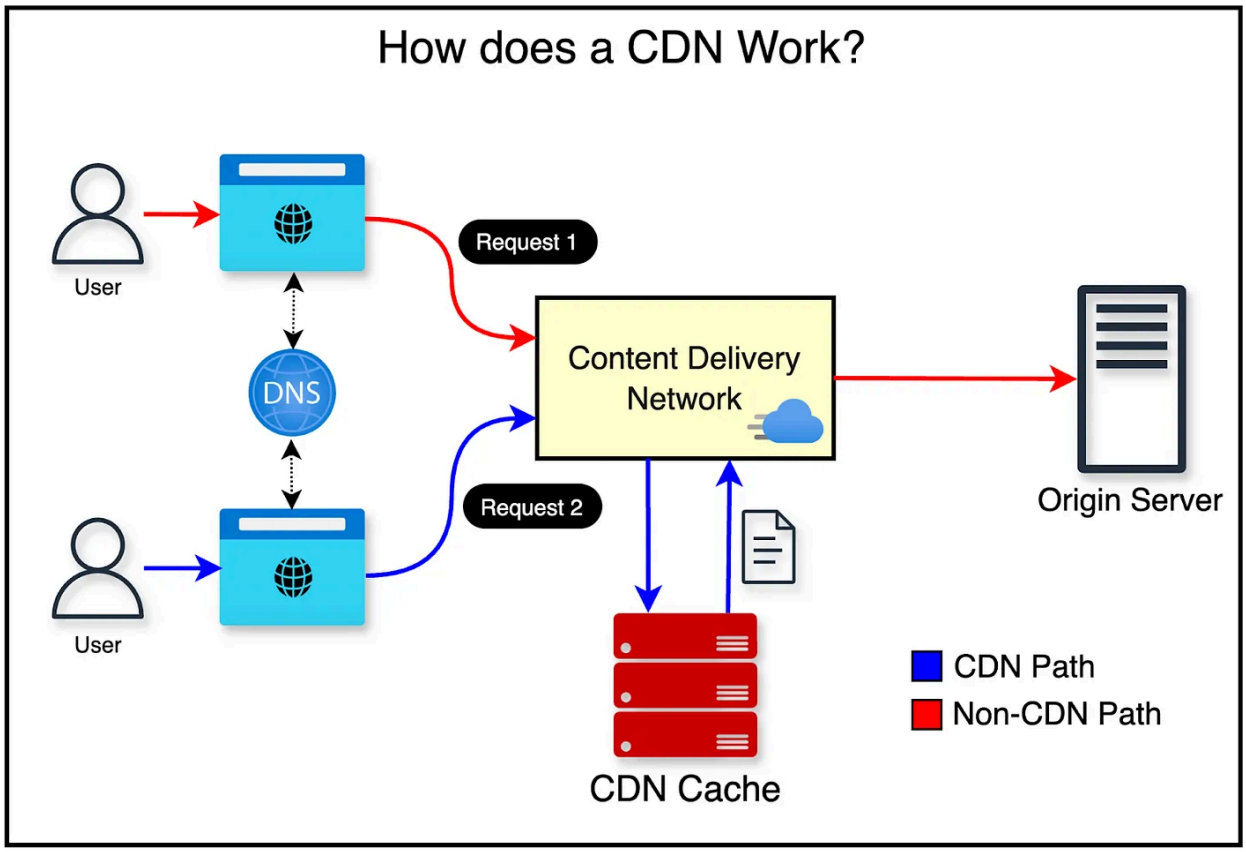
CDNs provide several key benefits:

- **Improved Performance:** By serving content from a server closer to the user, CDNs significantly reduce latency and improve page load times.
- **Increased Availability:** CDNs distribute the load across multiple servers, making sure that content can be accessed even if some servers fail.
- **Reduced Bandwidth Costs:** By caching content on the edge servers, CDNs reduce the amount of data that needs to be transferred from the origin server. This reduces bandwidth costs for the content provider.
- **Enhanced Security:** Many CDNs offer features like DDoS protection, SSL/TLS encryption, and Web Application Firewall (WAF) to protect websites and applications from security threats.

- **Scalability:** CDNs can handle sudden traffic spikes, making it easier for websites and applications to scale as needed.

How Does a CDN Work?

The diagram below shows the content delivery process in detail.



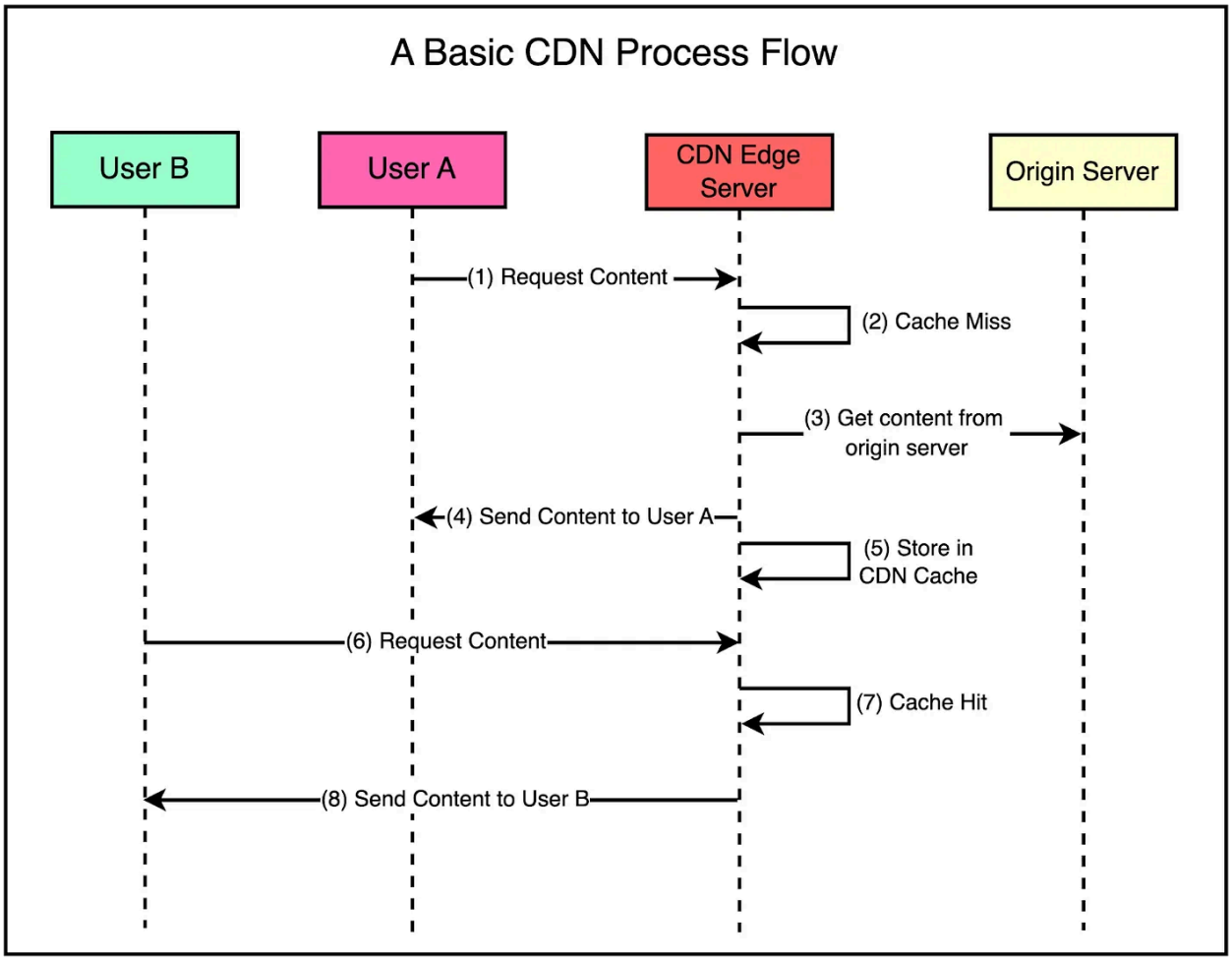
Here's what happens in each step:

- **DNS Resolution:** When a user requests content from a website or application served by a CDN, their device sends a DNS query to resolve the domain name to an IP address. The CDN's DNS system handles the query. It determines the optimal edge server to serve the content based on factors such as geographic proximity and server load.
- **Request Routing:** After DNS resolution, the user's device sends the content request to the IP address of the edge server. We will explore more about request routing in a later section.
- **Cache Lookup:** The edge server receives the request and checks its local cache for the requested content. If the content is found in the cache (a cache hit) and is

still valid (TTL has not expired), the edge server serves the content directly from the cache to the user.

- **Origin Fetch:** If the requested content is not found in the cache (a cache miss) or the cached content has expired, the edge server forwards the request to the origin server. The origin server processes the request and sends the requested content back to the edge server.
- **Caching and Delivery:** The edge server receives the content from the origin server, stores a copy in its local cache, and delivers the content to the user. It also assigns a Time-to-Live (TTL) value to the content. The cached content is now available for future requests by the same or different users.
- **Optimization and Security:** Throughout the content delivery process, CDNs may apply various optimizations and security measures to protect the content. This includes techniques like compression, minification, SSL/TLS encryption, and DDoS mitigation.

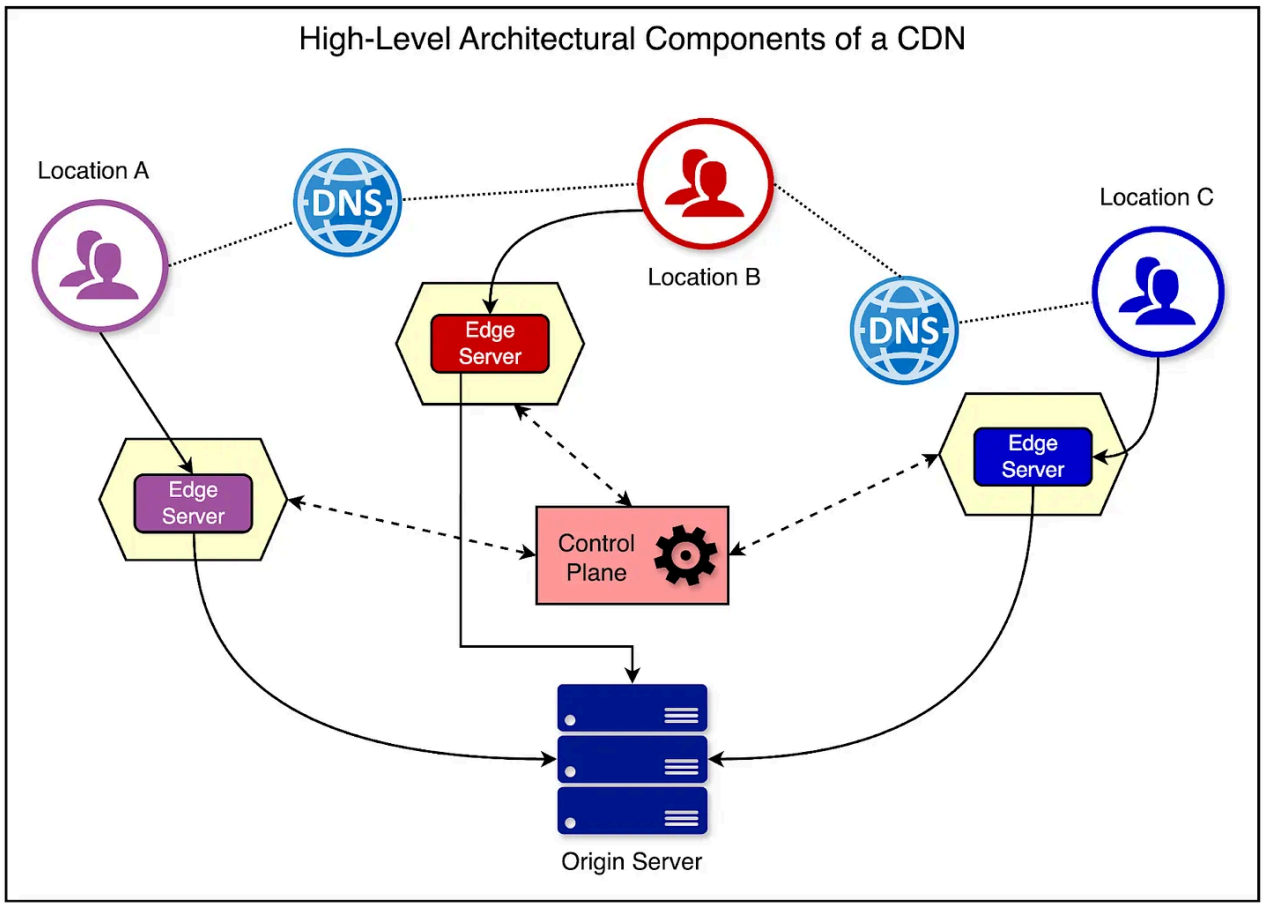
The sequence diagram below shows a basic CDN flow.



CDN Architecture and Components

A typical CDN architecture consists of several key components:

- Origin Server
- CDN Edge Servers
- Domain Name System (DNS)
- CDN Control Plane
- Monitoring and Analytics



Let's take a deeper look at each component and its role in the overall architecture of a CDN.

Origin Server

The origin server is the primary source of the content that the CDN distributes.

It hosts the authoritative version of the content and ensures that the CDN has access to the most up-to-date version. The origin server can be a physical server, a cluster of servers, or even a cloud-based service.

When a user requests content that is not available on the CDN edge servers, the CDN fetches the content from the origin server, caches it, and then serves it to the user.

CDN Edge Servers

CDN edge servers, also known as Points of Presence (PoPs), are the backbone of the CDN infrastructure.

These servers are strategically located in data centers around the world to ensure that content is delivered quickly and efficiently to users in different geographic regions.

The primary role of CDN edge servers is to cache and serve content to the users. When a user requests content, the request is routed to the nearest edge server, which then serves the content from its cache.

Edge servers perform critical functions, such as:

- **Load balancing:** Distributing incoming requests across multiple servers to ensure optimal performance and availability.
- **Content optimization:** Compressing, minifying, and optimizing content to reduce file sizes and improve load times.
- **Security:** Implementing security measures like DDoS protection, SSL/TLS encryption, and Web Application Firewalls (WAF) to protect the content and users.

DNS

The DNS plays a crucial role in directing user requests to the appropriate CDN edge server.

When a user requests content from a website or application that uses a CDN, the DNS resolves the domain name to the IP address of the nearest edge server.

CDN providers typically use specialized DNS services, such as Global Server Load Balancing (GSLB) or Anycast DNS. We will look at both in the next section.

CDN Control Plane

The CDN control plane is the central management system that oversees the operation of the CDN.

It is responsible for configuring and managing the edge servers, monitoring performance, and ensuring that the CDN operates efficiently. The control plane typically includes tools and interfaces for:

- Provisioning and configuring edge servers
- Managing content distribution and caching policies
- Monitoring real-time performance and traffic metrics
- Troubleshooting and resolving issues
- Reporting and analytics

Monitoring and Analytics

CDN providers use sophisticated monitoring and analytics tools to track the performance of the CDN.

These tools collect data on various metrics, such as:

- Cache hit ratios
- Response times
- Error rates
- Bandwidth usage
- Geographic distribution of traffic

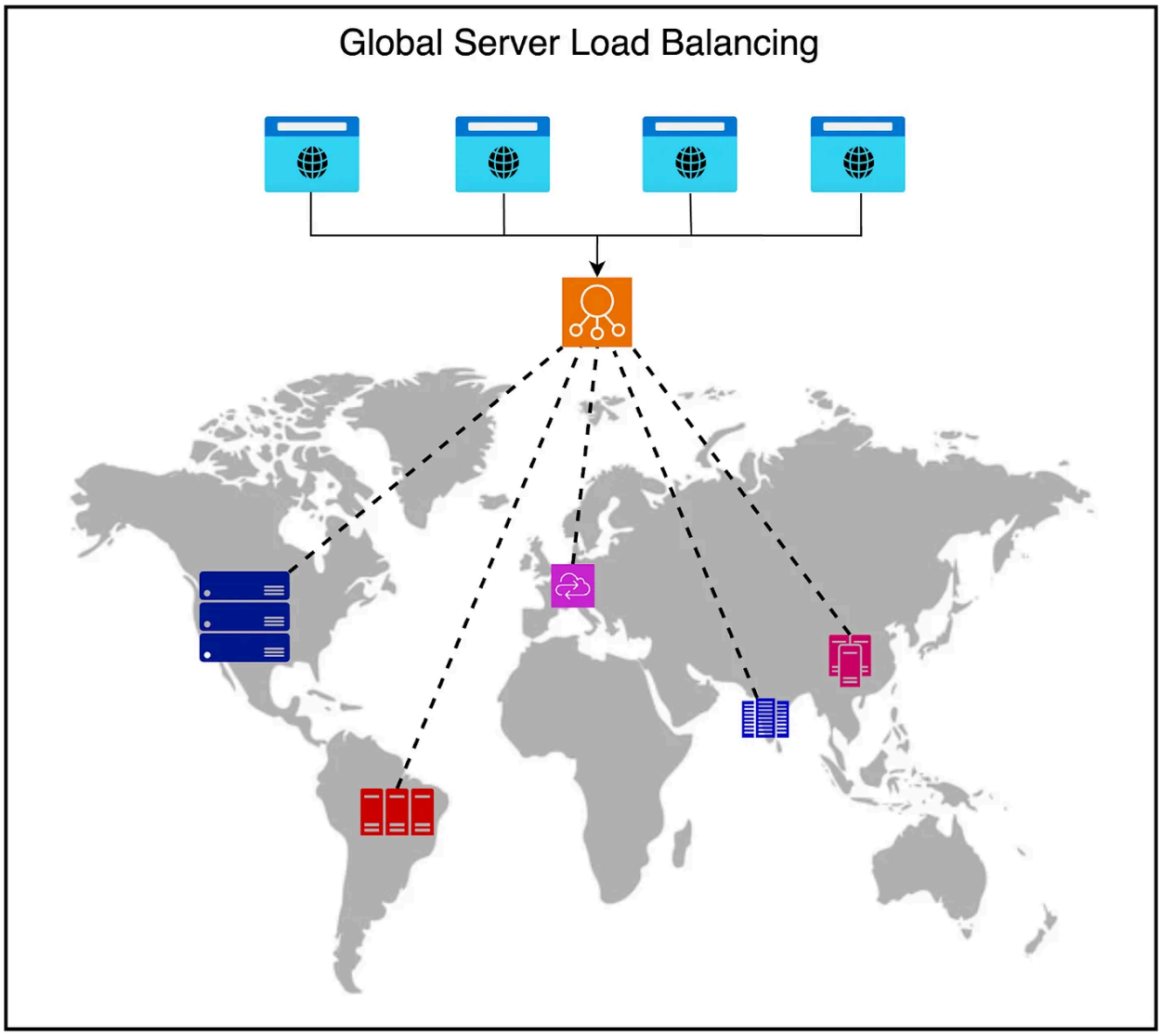
This data is used to identify performance bottlenecks, optimize content delivery, and ensure that the CDN meets the needs of the content providers and their users.

CDN Request Routing

There are some key concepts to understand when it comes to request routing in a CDN.

Global Server Load Balancing

GSLB is a technique used by CDN providers to distribute incoming traffic across multiple geographically dispersed servers or data centers.



The primary goal of GSLB is to ensure that user requests are routed to the server that can provide the best performance and availability. It typically considers the following factors:

- Geographic proximity
- Server load
- Network conditions

For a CDN, the GSLB system is usually integrated into the CDN's overall DNS infrastructure, which works alongside the standard DNS hierarchy, which includes:

- Root DNS servers
- Top-Level Domain (TLD) servers
- Authoritative DNS servers

In a CDN setup, the GSLB operates at the authoritative DNS level. Here's how it works:

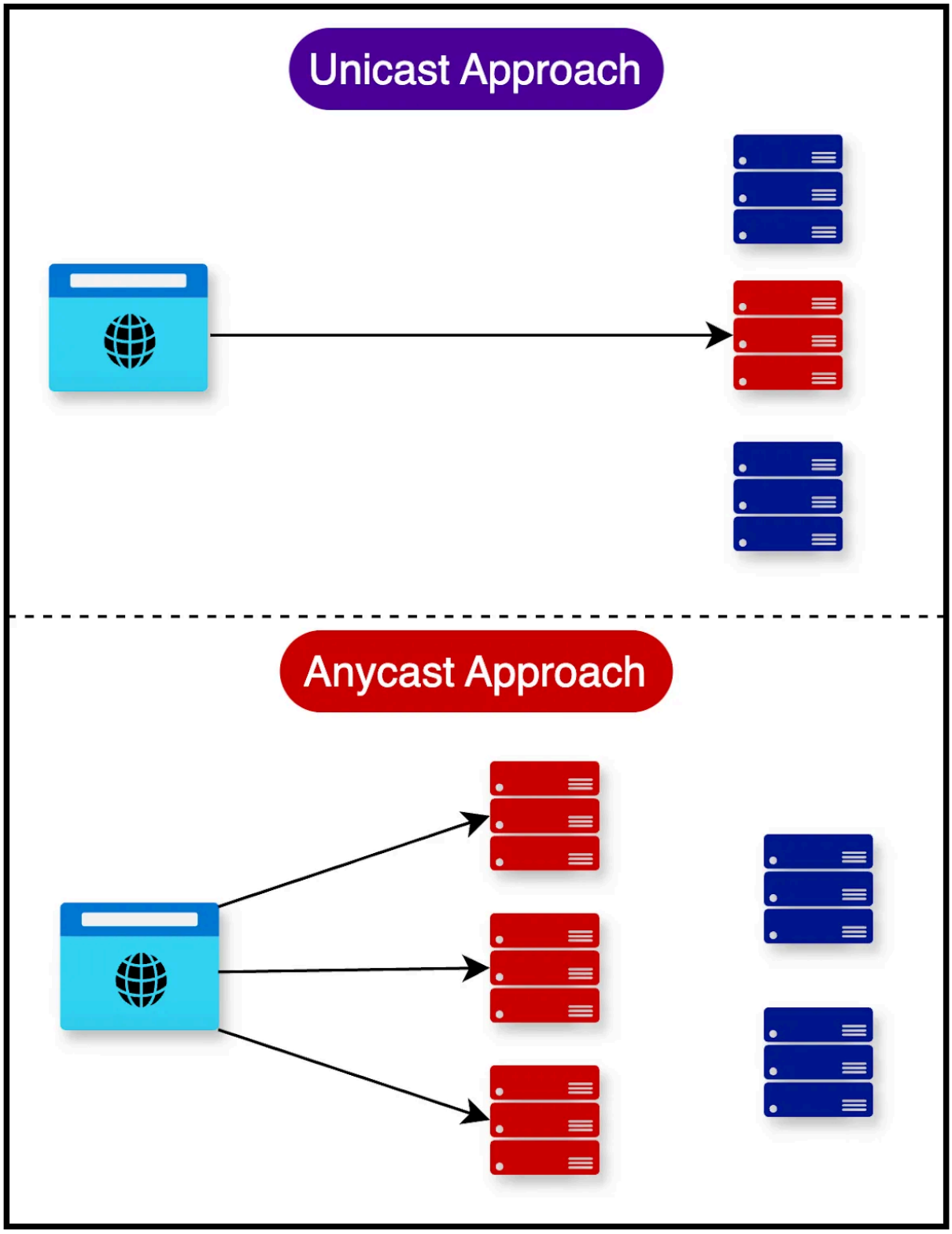
- The user's device sends a DNS query to resolve the domain name to an IP address.
- The query is sent to the user's local DNS resolver, typically provided by their Internet Service Provider (ISP).
- If the local DNS resolver doesn't have the IP address cached, it starts the recursive DNS resolution process by querying the root DNS servers.
- The root DNS servers respond with the IP addresses of the relevant TLD DNS servers.
- The TLD DNS servers respond with the IP addresses of the authoritative DNS servers for the domain.
- In a CDN setup, the authoritative DNS servers are managed by the CDN provider and integrate the GSLB system.
- The GSLB receives the DNS query and analyzes it based on factors like the user's geographic location, server load, and network conditions. Based on the analysis, it selects the optimal CDN edge server to handle the request and returns its IP address to the local DNS resolver.
- The DNS resolver returns the IP address to the user's device, which sends the content request to the selected CDN edge server.

Anycast DNS

Anycast DNS is a networking technique that allows multiple servers to share the same IP address.

In CDNs, Anycast DNS typically routes incoming traffic to the nearest data center with the capacity to process the request efficiently. With an Anycast network, instead of one server handling the load of all traffic, the load is spread across other available data centers.

The diagram below shows the difference between Anycast and Unicast:



Here's how it works:

- The CDN provider assigns the same IP address to multiple edge servers located in different geographic regions.
- When a user's device sends a DNS query to resolve the domain name to an IP address, the query is routed to a DNS server using standard DNS routing protocol.
- The DNS server responds with the shared IP address of the CDN's edge servers.
- The user's device sends the content request to the shared IP address.
- The request is automatically routed to the nearest edge server based on the network topology and routing protocols.

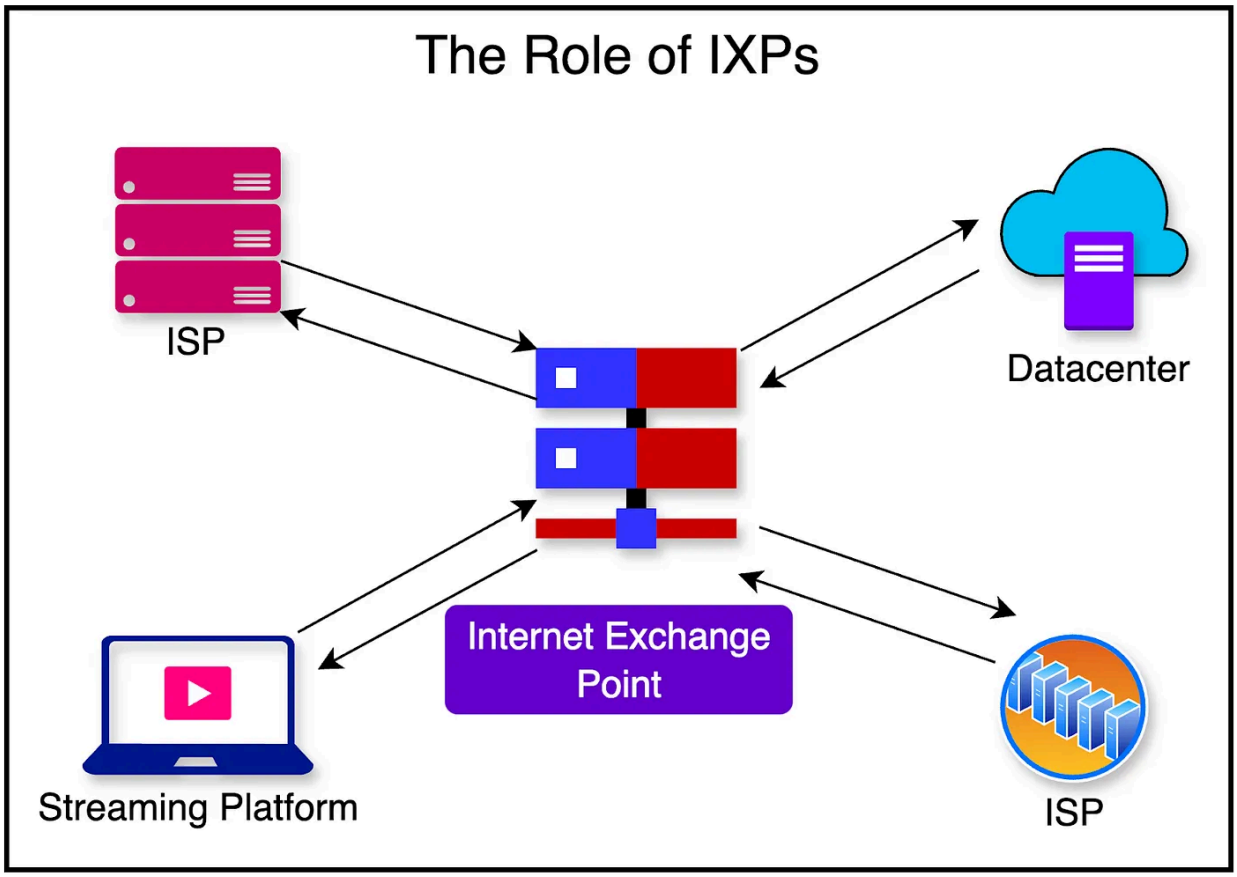
A CDN using Anycast is also great at preventing DDoS attacks since it increases the surface area of the receiving network.

Unfiltered denial-of-service traffic from a distributed botnet is absorbed by each of the CDN's data centers. The larger the network size, the harder it becomes to launch an effective DDoS against the CDN.

The Role of Internet Exchange Points

Internet Exchange Points (IXPs) are physical infrastructure facilities where multiple networks can connect and exchange data traffic.

They enable internet service providers, content providers, and other network operators to exchange traffic directly between their respective networks.



Without IXPs, traffic going from one network to another might rely on an intermediary network, Potentially causing “tromboning”, where traffic from one city destined for another ISP in the same city may end up traveling vast distances.

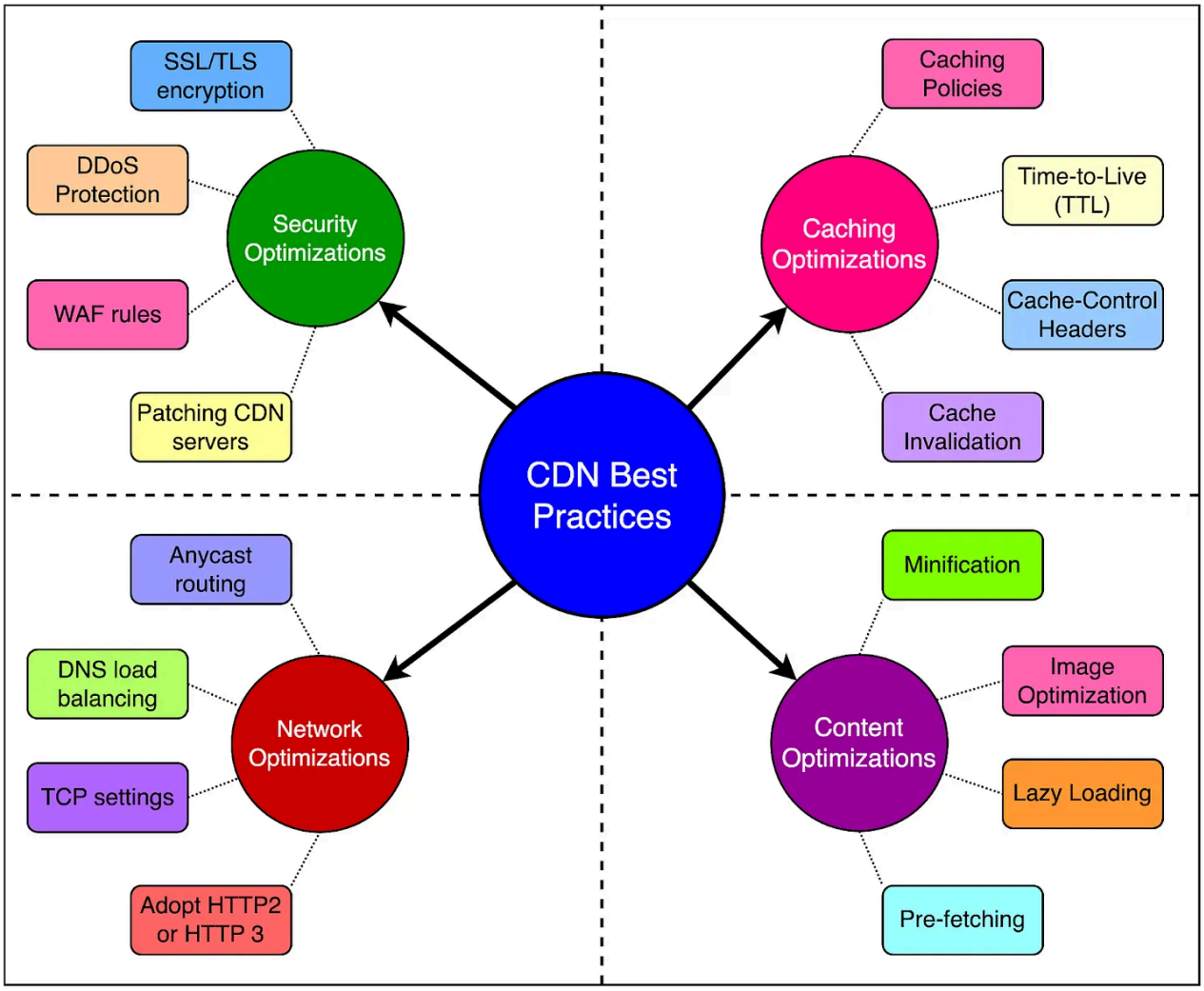
But how do CDNs leverage IXPs?

CDN providers establish a presence at major IXPs to improve their connectivity and reach.

IXPs allow CDNs to exchange traffic directly with ISPs and other networks, minimizing the distance and time required for data to travel between the CDN and end users. A CDN with an IXP presence optimizes the path through which data flows within its network.

Best Practices to Optimize CDN Performance

Optimizing CDN performance is crucial for delivering fast, reliable, and efficient content to users worldwide.



Here are some best practices:

1 - Caching Optimization

- Implement effective caching policies to maximize cache hit ratios and minimize the need to fetch content from the origin server.
- Set appropriate Time-to-Live (TTL) values for cached content based on its update frequency and importance.
- Use cache-control headers to specify caching behavior, such as cache expiration and validation.
- Implement cache purging or invalidation mechanisms to remove stale or outdated content from the cache when necessary.

2 - Content Optimization

- Minify and compress text-based assets, such as HTML, CSS, and JavaScript files, to reduce their size and improve loading times.
- Optimize images by compressing them, using appropriate formats (e.g., JPEG, PNG, WebP), and serving responsive images based on device capabilities.
- Implement lazy loading techniques to load non-critical resources only when needed, improving initial page load times.
- Use content pre-fetching or pre-rendering to proactively load and cache content that users are likely to request.

3 - Network Optimization

- Implement Anycast routing to ensure that user requests are automatically routed to the nearest available CDN server, reducing latency.
- Use DNS load balancing techniques to distribute traffic across multiple CDN servers and ensure optimal server selection.
- Optimize TCP/IP settings to improve network performance, such as enabling TCP Fast Open and optimizing TCP congestion control algorithms.
- Implement HTTP/2 or HTTP/3 protocols to enable features like multiplexing, header compression, and server push to reduce latency and improve performance.

4 - Security Optimization

- Implement SSL/TLS encryption to secure communication between users and the CDN to protect sensitive data and ensure privacy.
- Enable DDoS protection mechanisms to mitigate the impact of distributed denial-of-service attacks on CDN performance.
- Implement Web Application Firewall (WAF) rules to protect against common web vulnerabilities and threats.
- Regularly update and patch CDN servers and software to address security vulnerabilities and maintain a secure infrastructure.

Can a CDN Deliver Dynamic Content?

Dynamic content refers to content that is generated or personalized in real-time based on user interactions, database queries, or other dynamic factors.

Some examples of dynamic content include:

- Personalized product recommendations on an e-commerce website.
- Content displayed in a user's social media feed.
- Personalized news aggregators

Delivering dynamic content through a CDN is challenging because it requires real-time processing and cannot be easily cached like static content.

CDNs can employ various techniques to optimize dynamic content delivery:

- **Caching Dynamic Content:** CDNs support caching of dynamic content at the edge servers using specific headers such as 'Cache-Control' or 'Expires'.
- **Edge-Side Includes (ESI):** ESI allows dynamic content to be assembled at the CDN edge servers by combining cached static fragments with real-time dynamic elements.
- **Serverless Computing:** Some CDNs offer serverless computing capabilities, allowing dynamic content generation and processing to be executed at the edge, closer to the users. For example, Cloudflare Workers is a serverless execution environment that runs custom JavaScript code at the edge.

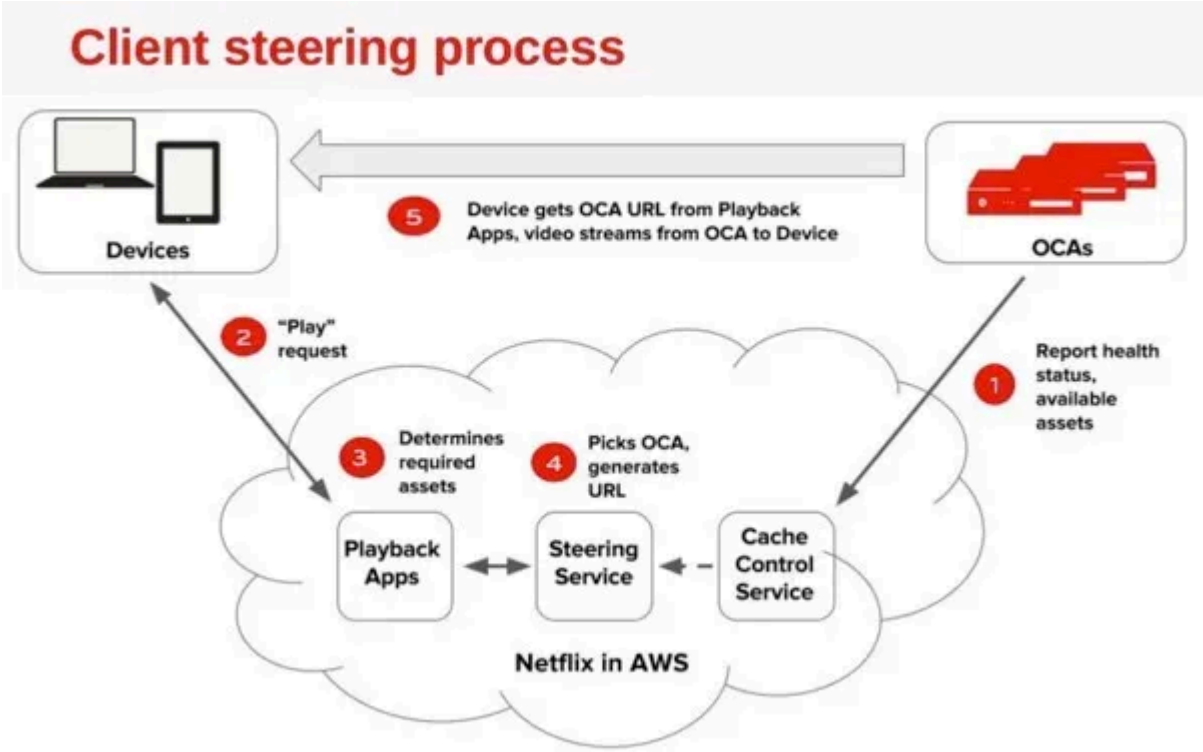
A Real-World Example of CDN

Let's now look at a prominent real-world example where the use of CDN played a key role in the scaling journey of Netflix.

Netflix debuted its streaming service in 2007. In the early days, they had 35 million members across 50 countries, streaming more than a billion hours of video each month.

In 2009, Netflix used a 3rd party CDN to keep the costs down and invest their time on higher-priority projects. However, in 2012, Netflix built its own dedicated CDN to maximize network efficiency and viewing experience.

The CDN is called Open Connect.



The Role of OCAs at Netflix

Here's how it works on a high level:

- Netflix deploys a network of purpose-built server appliances called Open Connect Appliances (OCAs) within Internet Service Provider (ISP) networks and at Internet Exchange Points (IXPs). These appliances are strategically placed closer to end users to reduce the distance data travels.
- OCAs are designed to store and serve Netflix content efficiently, with high storage capacity and an optimized software stack for video delivery.
- Netflix's content library is encoded into multiple bitrates and formats to accommodate different network conditions and device capabilities.
- The encoded content is distributed to OCAs based on popularity, regional demand, and other factors. OCAs cache the most frequently requested content locally.
- When a user requests to watch a Netflix title, the request is directed to the nearest OCA based on the user's location and network topology. If the content is found in the cache, the OCA immediately starts streaming the video to the user. It also dynamically adjusts the video bitrate and quality in real-time, depending on changes in network bandwidth.

Summary

Content Delivery Networks have become a key component of modern web applications.

We've discussed many aspects of CDN, such as:

- The need for a CDN and how it works.
- The overall architecture and components of a CDN.
- Best practices of using a CDN.
- The various components of CDN's request routing process.

Here are a few important learning points to take away from this article:

- CDNs improve website performance, reduce latency, and handle high traffic loads by distributing content across a network of edge servers located closer to the users.
- CDN architecture consists of key components such as origin servers, edge servers, and the DNS system, which work together to efficiently deliver content to users.
- CDNs employ various request routing techniques, including GSLB, Anycast, and leveraging IXPs, to ensure that user requests are directed to the optimal edge server for faster content delivery.
- Optimizing CDN performance involves best practices such as content caching, compression, minification, and using techniques like ESI and serverless computing for delivering dynamic content at the edge.
- Implementing a CDN can help businesses scale their content delivery and expand their global reach.

References:

- Site Speed is Still Impacting Your Conversion Rate
- Content Delivery Network Market Size

131 Likes · 7 Restacks

2 Comments

Write a comment...

Paul Davies Jun 7

One thing that is not covered, is how does authentication and authorisation would work with CDN.

LIKE (1) REPLY SHARE

Elbeast 18 hrs ago

Very nice article. Would like to see more about routing IXPs and POPs.

LIKE REPLY SHARE