CDN's Authorization Framework and Content

Linfeng Li, Jaoudat Karime, Claudio Jimenez

March 2021, CS 540

Introduction

Content Delivery Networks (CDN) are today's digital distribution systems. They distribute software orvideo and are deeply embedded in the body of the internet. They are growing in demand. Video streaming services account for around 80% of internet traffic today. Customers of these networks want their moviesand series streamed fast and at the highest image definition available. They also want their software to be downloaded fast and secure.A CDN network has usually many server clusters close to the users. These clusters are composed of an authorization server and many content servers. The content servers are usually replicated many times within the network so if one of them becomes unavailable clients are able to get the content from another.

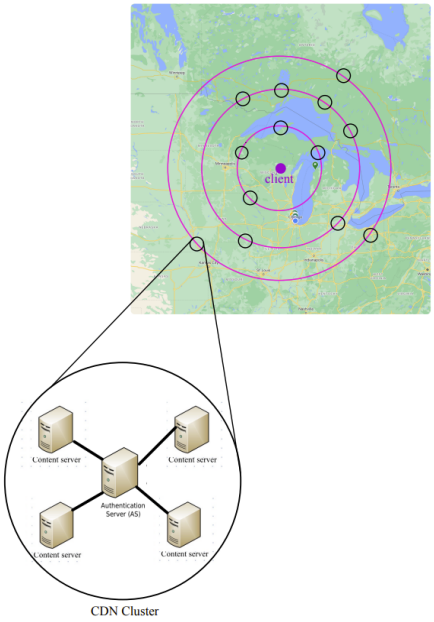

Figure 1. Geographic representation of a CDN network

Description of the Problem

In the current framework most CDNs use credentialing authority for token creation and authentication. A protocol with similar functionality to Kerberos protocol.The authorization handshake between client requests for content and the content being served involves 3pipe-lined events:

1. Client authentication: The client sends a message to the authorization server (AS) for login authorization. The AS checks login info with a key distribution center, located in the same host, and replies to the AS. The AS sends two messages to the client with an encrypted key and clients login/address information.
2. Client Service Authorization: The client sends now 2 messages to the AS with an encrypted key and an authenticator. The AS replies with another 2 messages containing a client to server ticket and session key.
3. Client Service Request: At this point the client has enough information to authenticate itself with the content server(CS). The client sends 2 messages to the CS with data received from the previous step anda new generated authenticator key. At last upon receiving these messages the CS verifies the request and authentication key and serves the content.

As seen in the handshake routine there are many messages being sent between the client, AS and CS (See figure 2). Another problem arises from the ability of clients to get content from different content network clusters. Sometimes the client requests multiple content servers in different clusters for different parts of the content depending on bandwidth and network traffic delay. Every time a new request is made to a new AS cluster the client must be re-authenticated creating more overhead. Extrapolating these problems to millions of connection requests in today's CDN's, and the ability of the network layer to lose or corrupt packages. We can show that protocols like kerberos although well suited for secure service connections, they ultimately increase network congestion and delay in serving the content.

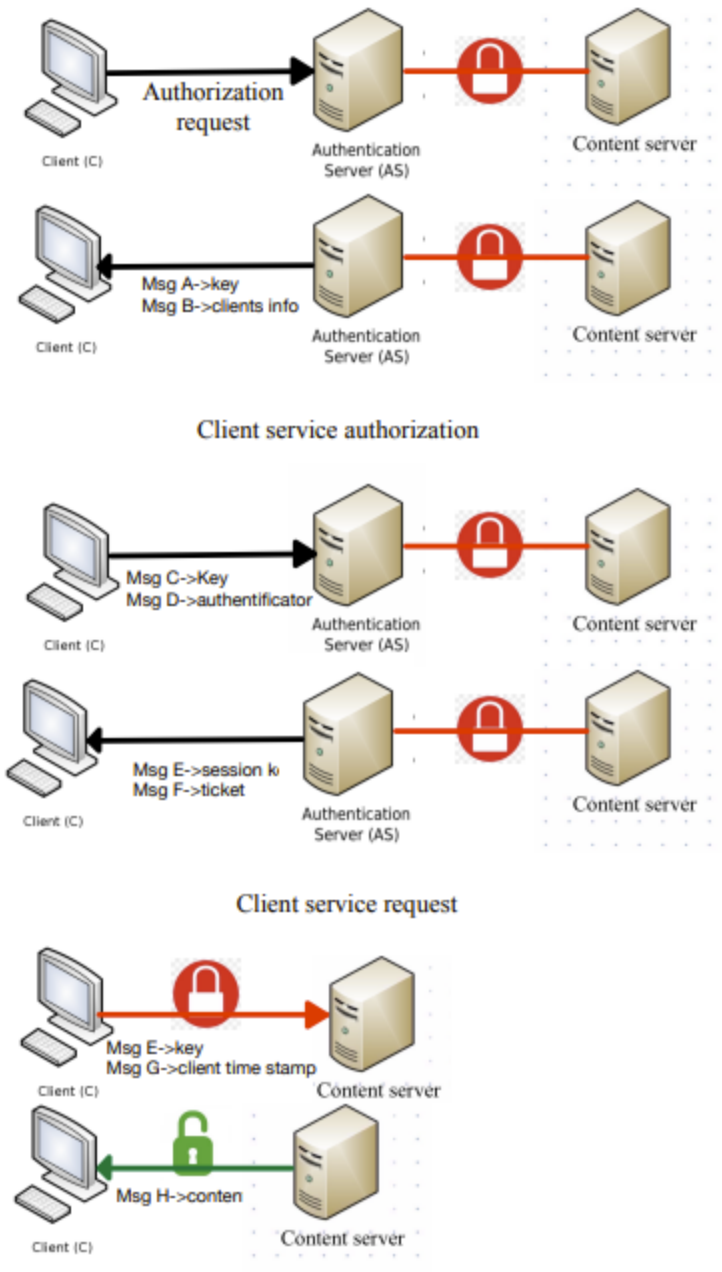

Figure 2. Client authentication to the AS

Why is it important to solve and what are the benefits?

In today's streaming environments content delivery networks are everywhere. Whether it be online shopping or video streaming it is an integral part of the world's daily internet usage. It helps increase the speed at which a business/e-commerce can load their website onto a device. According to an article by Globaldotswho state CDNs can provide a website "Increased revenue by 1% for every 100ms of improvement to your page load time" (1). In layman's terms an increase in load time costs the business that uses a CDN to display their web-page money. As previously stated, part of the process of getting connected to these services is allowing the user to be authenticated and possibly re-authenticated when switching CDNs clusters. Our ability to cut out re-authentication would help to decrease latency associated with higher traffic and allow for companies to retain users as well as increase their earnings.Benefits of this implementation on the user's side would be decreased loading time and latency when bandwidth becomes scarce. For example, the current implementation for video streaming allows a user today on the same page without the need for hard refresh although this causes the user to experience an extended buffering time. Our ability to decrease the buffering associated caused by CDN switching will help increase application quality experience.

Proposed Framework Solution

Therefore we propose a solution to solve both the authentication complexity in the existing CDNs and also try to implement a new approach to reduce network latency and improve user experience.First, we are going to improve network latency by using a shared backbone between CDN clusters. The backbone service will provide functionalities where the users' authentication secrets are stored and shared. Whenever a user is trying to login into an authentication server and is ready to receive content. Our authentication server will acknowledge this connection by putting the user in the access allowed list alongwith a validation timestamp and share it with a set of predefined authentication servers through the backbone. Consequently, the user will be granted access to contents on the server as well as receiving an

accesskey. When the connection quality from the current content server drops. The user then will be able to access contents from CDN clusters that were sharing the same backbone; using the same access key. And the user only needs to be re-authenticated when the timestamp associated with the user in the shared backboneexpires.This way, the re-authentication step is significantly reduced by utilizing the backbone and the userexperience is therefore optimized (See figure 3).

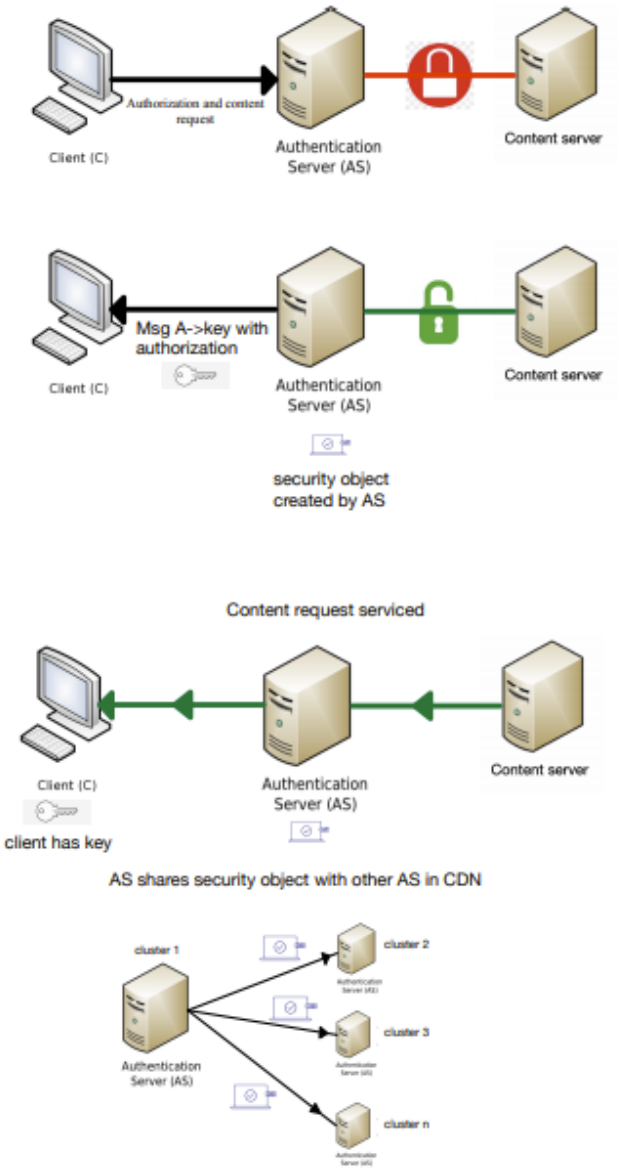

Figure 3. Client authentication to the AS

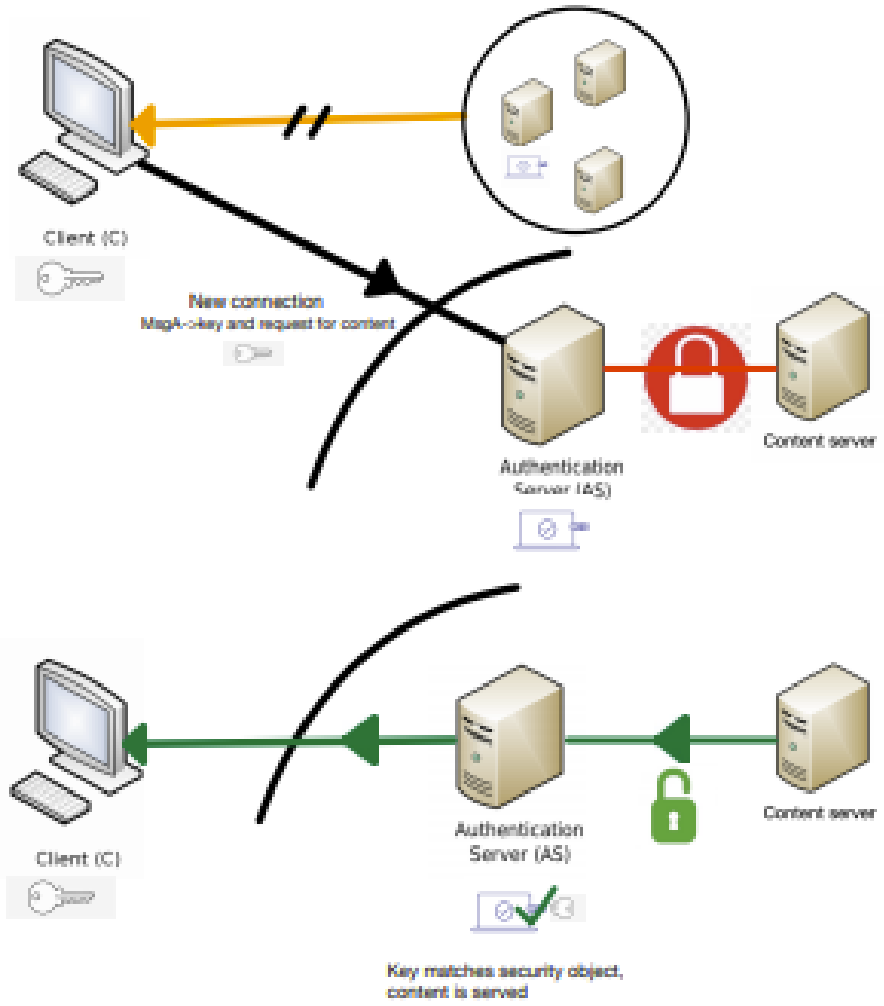

Figure 4. Client authentication to the AS

Implementation Timeline

Week	Tasks	Dues
Week 1: 3/15 - 3/21	• Form Group • Discuss the topic • Finalize the topic • Plan road Map	Milestone 1 due 3/17
Week 2: 3/22 - 3/28	• Start GitHub Repository • Divide Coding Tasks in accordance to diagrams	
Week 3: 3/29 - 4/4	• Basic / Abstract Architecture is complete • Develop and add more specificity	
Week 4: 4/5 - 4/11	• Begin developing general and edge test cases • Fix failing test cases	
Week 5: 4/12 - 4/18	• Update the Readme.md Document with the current specifications • Ensure Readme.md correctly corresponds with code functionality	
Week 6: 4/19 - 4/25	• Begin detailed analysis about the approach • Gather Statistics on how our implementation fairs to the current implementation	
Week 7: 4/26 - 5/2	• Start Documentation of the results • Design Philosophy • Implementation and statistics outcome explanation	
Week 8: 5/3 - 5/5	• Present	Deadline 5/5 @ 9 PM

Table 1.

Use Cases

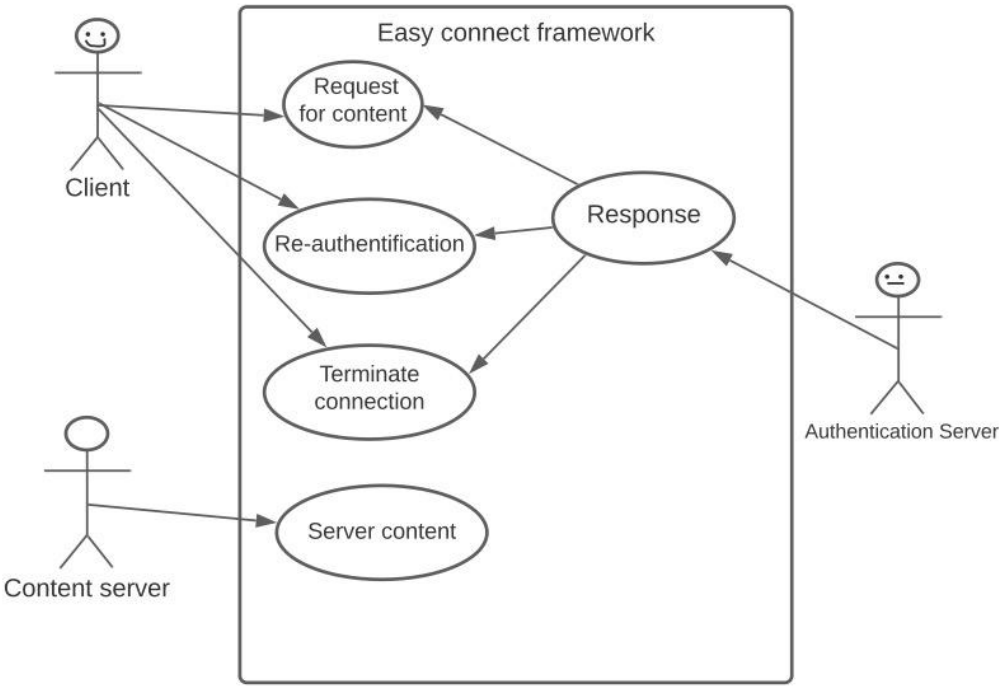

Figure 5. Use-case diagram

Use case ID: Re-Authentication Name: Re-authenticate a User
Pre-conditions: Client has been authenticated by the authentication server.
post-conditions: The timestamp in the security object is updated to a future time.
Initiated by: Client
Triggering Event: The timestamp in the security object is about to expire, but the client has just initiated a new content request.
Additional Actors: n/a

Sequence of Events:

1. The client receives the security object and starts accessing content through a fetch request using the security object.
    a. The Authentication Server receives the content request and compares the timestamp in the security object with the current timestamp on the Authentication Server.
    b. If the timestamp is after the current time, the authentication server will authorize the content access request and pass the content back, along with an updated future timestamp.
    c. The authentication server publishes the updated security object to nearby CDN clusters.

Alternatives:

1. If the timestamp is after the current time, the user will be required to go through the login process again.

Exceptions: n/a

Table 2.

Use case ID: content_request Name: Client request content

pre-conditions: The client is logged into the system and has chosen content to be served.

post-conditions: Client's machine receives the content requested

Initiated by: Client

Triggering Event: Client clicks on content (Ex. movie, software) it wants.

Additional Actors: Authentication server, content server

Sequence of Events:

1. Client requests content from the Cluster through a given application
    a. Authentication server responds by sending the client,
        i. Security key
        ii. Content Manifest: contains addresses of replicated content servers that have the requested content within the network.
2. Client saves security Key and Content Manifest.
3. Clients connects to content server
    a. Content server serves content.

Alternatives:

Client connection slows or brakes. Client needs to be re-authenticated (refer to use case ID: re-authentication.

Exceptions: Authentication fails, content server fails to send content

Table 3.

Basic Class Overview

Class Name	Description
SecurityObject	Is the object that will hold an encoded hash key as well as a time object for when the client was authenticated.
Time	Sole purpose is to hold the encoded time of when the user was granted access but the Authentication Server

Client	Class that is associated with the user in getting the response from the Authentication Server. Clients are also allowed to ask for a response which is basically asking to connect to the Authentication Server.
Response	Response holds the security key that was assigned to the client by the authentication server.
Authentication Server	Is the class that is associated with holding the database of all the clients and their security keys. Authentication servers are also responsible for sending a client's security object to other Authentication servers within one degree distance of the client.
Content Server	Class that is associated with the ability to stream content to the client.

Table 4.

Sequence Diagrams

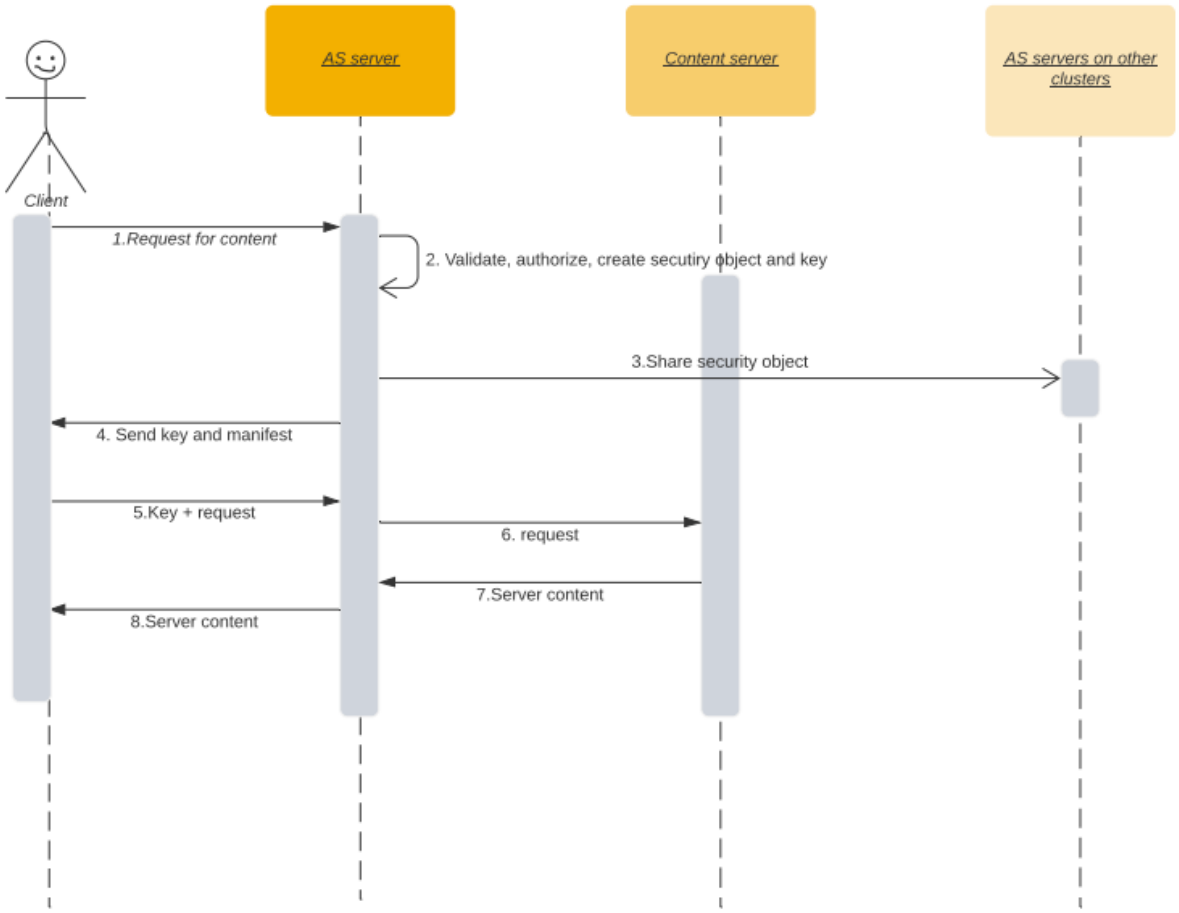

Figure 6. Client content request sequence diagram

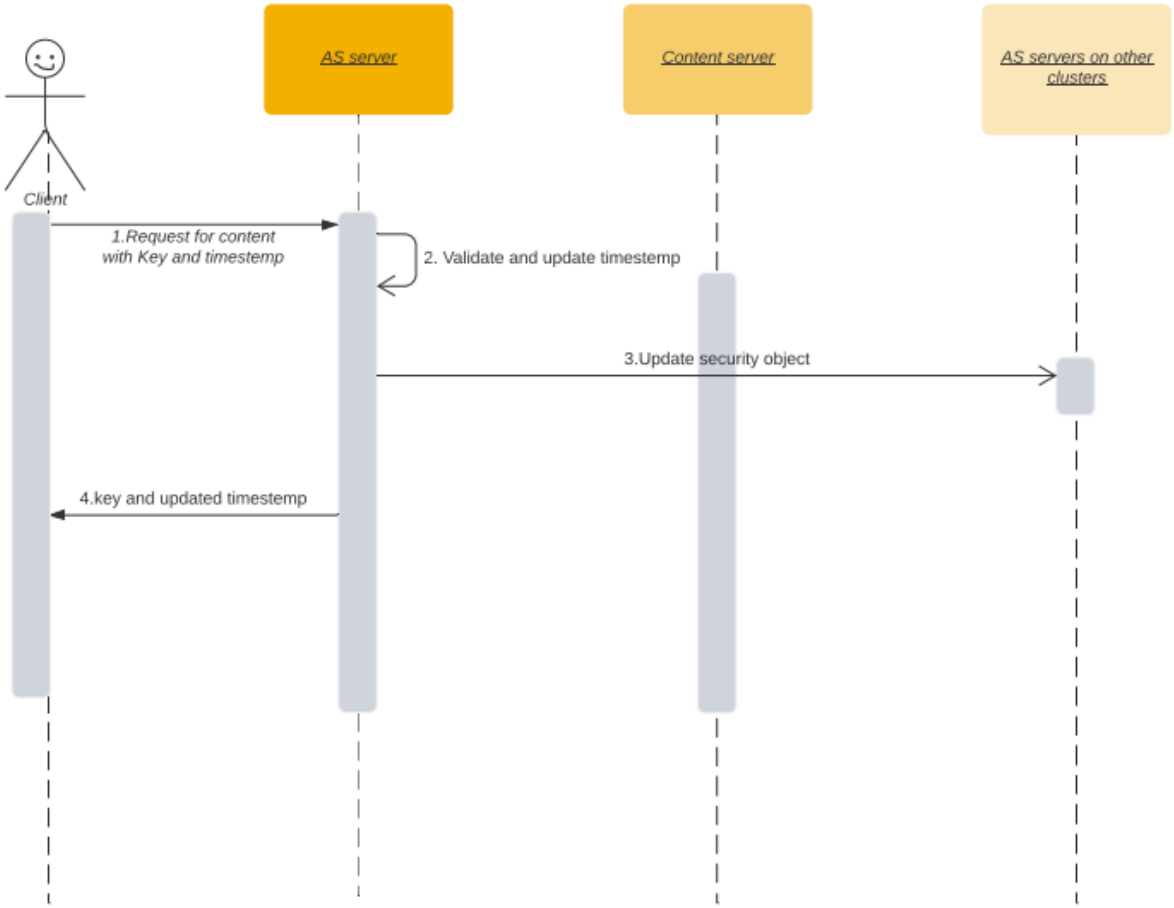

Figure 7. Re-authorization sequence diagram

References

[1] GlobalDots. “Content Delivery Network (CDN) Explained.” Globaldots, www.globaldots.com/content-delivery-network-explained