Broadcast and Multicast Communication Enablers for the Fifth-Generation of Wireless Systems

Deliverable D5.3
Application Layer Intelligence

Version v1.0 Date: 2018/11/29

Document properties:

Grant Number:	761498
Document Number:	D5.3
Document Title:	Application layer intelligence
Editor(s):	Baruch Altman (LiveU)
Authors:	David Navratil (NOK); Andrew Murphy, Richard Bradbury (BBC); Steve Appleby, Jon Hart, Tim Stevens, Rory Turnbull, (BT); Maël Boutin, Duy-Kha Chau (Broadpeak); Roman Odarchenko, Rui Andrade Aguiar (Bundeslab); Tuan Tran (Expway); Baruch Altman (LiveU); Peter Sanders (one2many); Francesco De Angelis (EBU); Ioannis Selinis (Surrey U)
Reviewers	Paul Higgs (Chief Strategy Officer, Video Product Line, Huawei Technologies)
Contractual Date of Delivery:	2018/11/30
Dissemination level:	CO[1]
Status:	Final
Version:	1.0
File Name:	5G-Xcast D5.3_v1.0

Abstract

A core principle of the 5G-Xcast approach to content delivery is to insert multicast into an otherwise unicast path. We treat multicast as an internal optimisation capability which should have minimal impact on the uncast logic of the higher protocol layers, including the application layer.

However, a range of optimisation techniques have been deployed over the years with the assumption that the final hop, from edge server to client application, is unicast. The insertion of multicast into this path will break the unicast assumption and therefore has the potential to disrupt these optimisation techniques.

In this document we analyse the impact of the insertion of multicast into the delivery path on such optimisation techniques. Where applicable, we discuss negative impacts, but on the whole, most of the optimisation techniques are still effective and do not prevent the use of MC.

Keywords

5G, QoE, broadcast, Multi-access edge computing, mobile network, MooD, multicast, unicast, Multilink, point-to-multipoint, mABR, converged networks, ROM broadcast, QUIC, MPQUIC, HTTP, HTTPS, HTTP/2, MPEG DASH, Objects content, Encrypted content, CDN, MultiCDN

[1] CO = Confidential, only members of the consortium (including the Commission Services)

PU = Public

Disclaimer

Executive Summary

As a central principle to the approach to exploiting multicast, we stipulate that service requirements are preferentially met without the assistance of special features provided by the network operators, particularly where such features would add complexity to the technical and commercial relationships between different organisations. For example, adaptive streaming is preferred over network QoS, despite the fact that a QoS-based solution could be considered technically superior to adaptation over a best efforts channel.

Note that this principle is not something that a network operator would necessarily advocate. It is rather a pragmatic observation that it is difficult to monetise network technology if it results in more complex cross-organisational interfaces and lower flexibility to test or introduce new capabilities. When developing network solutions we need to take into account that content service provider would rather deploy technology at protocol layers that they have under their direct control.

In this document, we loosely refer to the collection of such techniques that could be applied at higher protocol layers (above layer 3) as 'application-layer intelligence'.

The objective of this document is to identify possible application-layer techniques that might be used so that we can understand whether the point to multipoint content delivery framework will prevent these techniques from operating correctly. Conversely, we will also consider whether the application-layer techniques will prevent the framework from operating effectively.

Many of these techniques concern the dynamics of Quality of Experience control. Within the video encoders, within the adaptation algorithms that run in streaming applications, and even in the TCP behaviour itself, there are semi-independent control loops which can be argued to be optimising their own limited view of QoE. The impact of introducing multicast will be to change the dynamics of the server to client connection in such a way that is very likely to impact the effectiveness of these QoE optimisation techniques. Conversely, the use of such techniques could render the use of multicast less effective than it might otherwise have been.

Another impact to consider relates to trust and security. For the network operator to be able to insert proxies en-route and to gather enough information about a content service to steer traffic appropriately, it will need to have more access to the content stream than would normally be the case in a traditional CDN architecture.

After examination of a number of application layer techniques, we conclude that most still have value when multicast is used. Some may impact the design of the multicast system. For example, careful selection of rates will be necessary to avoid confusing adaptation algorithms.

Content protection poses no problem at all, however transport layer protection does need a specific agreement between the CDN operator and the network operator.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

AF	Application Function
AMF	Access and Mobility Management Function
AR/VR	Augmented Reality / Virtual Reality
AS	Application Server
AUSF	Authentication Server Function
BM-SC	Broadcast Multicast Service Centre
CAPEX	Capital Expenditure
CBC	Cell Broadcast Centre
CBE	Cell Broadcast Entity
CDN	Content Delivery Network
CSP	Content Service Provider
DN	Data Network
DSL	Digital Subscriber Line
EC	European Commission
GCS	Group Communication System
GERAN	GSM EDGE Radio Access Network
GSM	Global System for Mobile Communications
GTP	GPRS Tunnelling Protocol
HTTP	Hypertext Transfer Protocol
IE	Information Element
IoT	Internet of Things
ISD	Inter-Site Distance
ISI	Inter-Symbol Interference
ISP	Internet Service Provider
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LTE	Long-Term Evolution
MCE	Multi-cell/multicast Coordination Entity
MCPTT	Mission-critical push-to-talk
MBMS	Multimedia Broadcast Multicast Service
MEC	Mobile Edge Computing
ML	Multilink
SAI	Service Area Identifier
MBMS-GW	MBMS Gateway
MBSFN	Multicast Broadcast Single Frequency Network
MME	Mobility Management Entity
MNO	Mobile Network Operator
MOS	Mean Opinion Score
NEF	Network Exposure Function
NF	Network Function
NRF	NF Repository Function
NSSF	Network Slice Selection Function
OFDM	Orthogonal frequency-division multiplexing
PCF	Policy Control Function
PDSCH	Physical Downlink Shared Channel
PKI	Public Key Infrastructure
POP	Points of Presence
PTM	Point to Multipoint
PW	Public Warning
QoE	Quality of Experience
QoS	Quality of Service
RAB	Radio Access Bearer
RAN	Radio Access Network
RTT	Round Trip Time
SAI	Service Area Identifier
SC-PTM	Single Cell Point to Multipoint

SIB	System Information Block
SINR	Signal-to-Interference-plus-Noise Ratio
SMF	Session Management Function
TLS	Transport Layer Security
TMGI	Temporary Mobile Group Identity
UDM	Unified Data Management
UDP	User Data Protocol
UDR	Unified Data Repository
UDSF	Unstructured Data Storage network function
UE	User Equipment
UPF	User Plane Function
VoD	Video on Demand

1 Introduction

Earlier in Workpackage 5, we outlined a set of principles (reported in D5.1 [1]), and, building on these, we then described a Content Delivery Framework (reported in D5.2 [2]).

One of the principles that we set out is that service requirements are preferentially met without the assistance of special features provided by the network operators, particularly where such features would add complexity to the technical and commercial relationships between different organisations.

That is, for example, adaptive streaming is preferred over network QoS, despite the fact that a QoS-based solution could be considered technically superior to adaptation over a best efforts channel.

In this document, we loosely refer to the collection of these type of techniques that could be applied at higher protocol layers (above layer 3) as 'application-layer intelligence'.

Note that this principle is not something that a network operator would necessarily advocate. It is rather a pragmatic observation that it is difficult to monetise network technology if it results in more complex cross-organisational interfaces and lower flexibility to test or introduce new capabilities. When developing network solutions we need to take into account that content service provider would rather deploy technology at protocol layers that they have under their direct control.

These observations led us to conclude that we should exploit a point to multipoint network capability as means for internal network optimisation and for the flexible introduction of such services without requiring lower level changes. Content service providers will implement application-layer intelligence as a means to control their content distribution and cost models based on certain assumptions about the way that a network will behave. For example, adaptive streaming algorithms will assume a monotonically decreasing relationship between the time it takes to deliver a video segment and the media bitrate. If the network is carrying out its own independent optimisation, then this could confuse such algorithms.

The objective of this document is to identify possible application-layer techniques that might be used so that we can understand whether the point to multipoint content delivery framework will prevent these techniques from operating correctly. Conversely, we will also consider whether the application-layer techniques will prevent the framework from operating effectively.

The document will start with a short discussion of Quality of Experience (QoE), its main relevant parameters and the main feedback loops that execute between the client application and the content server.

We examine how QoE management provided by examples of such network layer technologies for Unicast (UC) are impacted by the introduction of a point to multipoint capability into the content delivery path for the main use cases as described in Workpackage 2.

Clearly, this document is not an exhaustive evaluation of all technologies, use cases or QoE implications. It does however offer a fresh and challenging way of evaluating the impact of BC/MC on the QoE of UC-designed technologies when the lower Core and RAN are not involved and therefore allows the network provider and (CDN) provider to assess the need for such involvement vs. the advantages in not having it.

2 QoE and QoE Control Logics

If we follow multimedia content from source to consumption by the end user, we will see that there are a number of semi-independent control loops. The goal of each of the control logic is to optimise QoE in view of its own measurements, parameters and criteria. The result of their operation may affect the operations of others, either directly or implicitly. Such loops would include the video compressor, which, for each Group of Pictures will be trying to optimise the rate-distortion trade-off. Other such loops include the Adaptive Bitrate selection by the client player application. Even the TCP congestion algorithm could be considered to be an example of such a loop, attempting to optimise the bitrate share whilst adhering to the principle of “TCP Friendliness”.

These dynamics will be altered with the introduction of multicast in the delivery path. To understand the impact of introducing multicast, it is appropriate to discuss the nature of Quality of Experience and the relationship between the semi-independent logics that seek to optimise their view of it.

2.1 Application layer QoE

The main expressions of consumption QoE can be viewed as:

1. The right video quality according to the right bandwidth that each UE can receive at any point in time. Different multiple devices, screens, connection conditions, congestions, mobility or other conditions impact the viewable quality. This QoE needs to be managed also for MC/BC.
2. Continuous stable and reliable user experience
3. Zero or minimal latency in receiving the video, including live, and to switch between states and contents, even when moving
4. When viewing and experiencing buffering, users might disengage and abandon that video.
5. Seamless transition between internal delivery modes (e.g. between multicast and unicast).
6. Deliver UE-specific content side by side with the generic broadcast, using “objects”.

The definition of a common set of objective metrics is a fundamental step towards an effective analysis of QoE; the Streaming Video Alliance [3] and the by the DASH Industry Forum [4] proposes various basic benchmark parameters that can be evaluated on the user’s device by resorting to JavaScript APIs in HTML5 [5]. In general, such techniques and KPIs are different from the ones used in broadcasting scenarios [1].

The Mean Opinion Score (MOS) [6], [7] estimates the overall QoE as perceived by users. MOS is a value on a predefined scale; the arithmetic mean over all values obtained from a set of experiments is interpreted as the overall quality of the experience. The MOS evaluation can also be carried out algorithmically; video quality metrics [8] and QoS Key Performance Indicators (KPIs) can be combined to realise QoE models that can predict viewers’ level of satisfaction as evaluated in visual experiments (e.g. [7], [9], [10], [11], [12], [13]). A prediction model for QoE, easy to integrate with off-the-shelf video players, will be detailed in deliverable 5.4 [14].

In order to assess the QoE, the application intelligence should be able to perform at least one of the abovementioned quality analysis. This requires a dedicated module on fixed and mobile devices that is able to assess the level of quality as perceived by the user who consumes the content on that specific device.

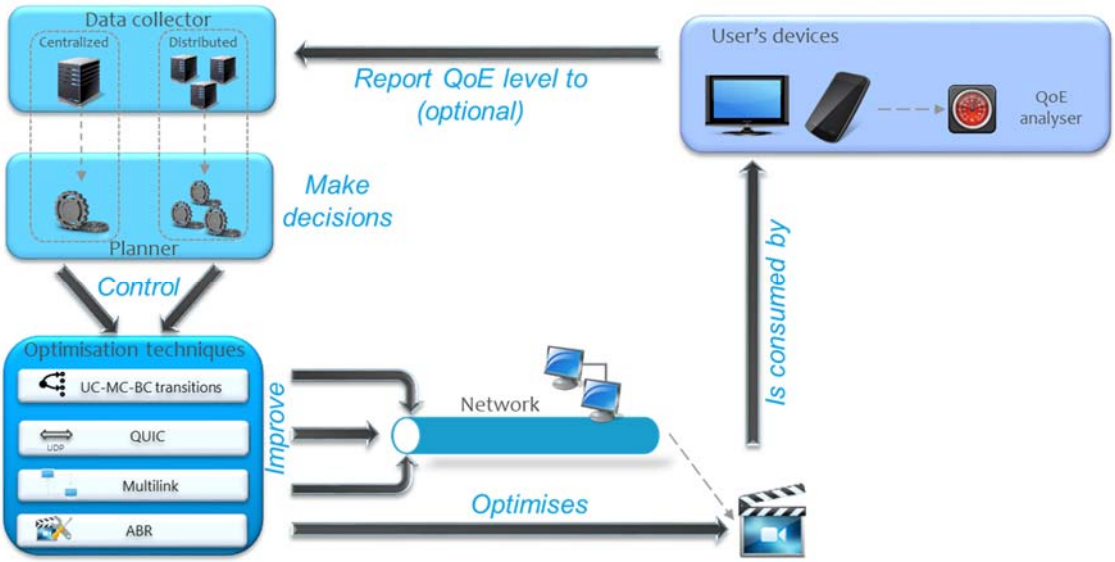

Figure 1 Conceptual view of the QoE assessment process

Figure 1 shows the main phases and activities carried out in the assessment of QoE. On users' devices, specialised applications monitor network and video performances. The perceived QoE level is then collected by i) a single centralised or ii) multiple distributed entities (e.g. Content Service Provider (CSP), Network operators' devices), where it can be aggregated with further additional information measured at different places of the distribution chain (e.g. network, CDNs). The collected information is taken as input by centralised or distributed planners to perform an optimisation of contents and networks. In this respect, it should be noticed that the distribution framework deals with the following entities: users, Network Providers, CDN providers and CSPs. In such a complex and interdependent environment, the high number of actors and entities (more in general the high number of parameters and variables) involved in the communication process implicitly justifies the existence of several control points and logics, which can be implemented in a centralised, distributed or decentralised fashion. The overall stability of the system depends on the stability of such components, which must be rigorously modelled in advance to obtain efficient mechanisms improving QoE in real-time. The outcome of the planning phase results in a collection of activities aiming at improving network performance and content properties; these activities are carried out by resorting to specific optimisation techniques:

- Dynamic transitions between unicast, multicast and broadcast distribution (for example by using MooD).
- QUIC.
- Multilink
- Adaptive Bitrate Streaming

2.2 Application Intelligence Layer Control Logics

As introduced in Section 2.1, several control logics may coexist in the Application Intelligence Layer., Such control loops can implement event-reaction control processes, adaptive mechanisms based on feedback analysis or more advanced (feedback) control loops. When performing its internal activities, a one control loop component or logic may interact with another one either explicitly or implicitly; this means that some events may trigger the execution of several actions in a chain

reaction fashion,. Figure 2 shows the main control logics on the Application Intelligence Layer and helps identifying how they can affect the QoE level:

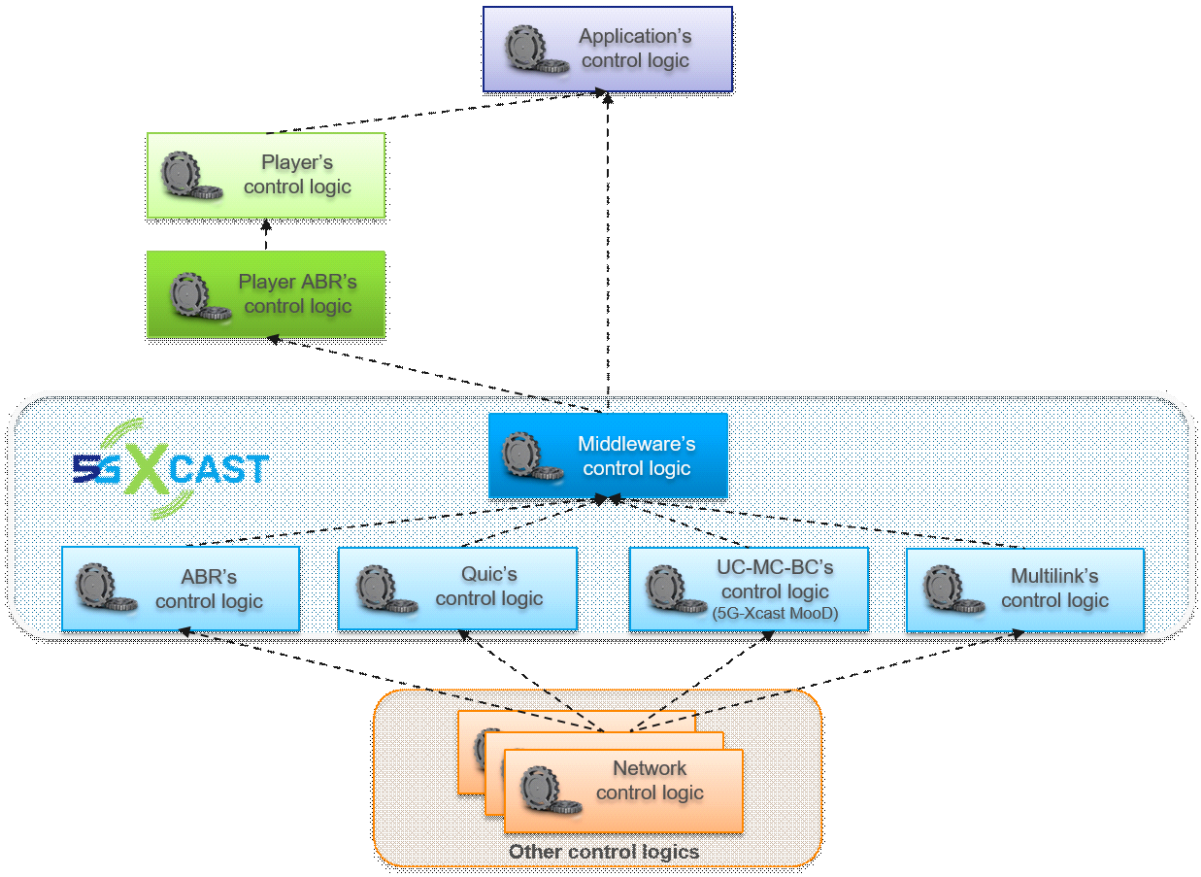

Figure 2 Control logics in the Application Intelligence Layer.

Note: Arrows indicate how actions can affect distinct components.

- Every generic user's application has a control logic managing all its functionalities (e.g. the Graphical User Interface, data storage, etc.).
- In the scenario of consumption of audio-visual content, usually applications resort to off-the-shelf players, embedded in web-pages or integrated within native applications. The player's controller manages all the functionalities to correctly play the content and to manage interactions with users. Applications traditionally interact with players through a set of APIs, providing the resources to play (e.g. their URIs) and receiving notifications to track the playout state. These interactions could make some application control logics dependant on player's control.
- The player adaptive bitrate controller implements the algorithm that fetches the content (at specific bitrates) and manages the internal buffer for incoming data packets. For instance, when considering DASH as an example of adaptive bitrate technique, several distinct logics can be instantiated depending on the real implemented algorithm (e.g. YouTube MPEG-DASH [15], HLS [16]).
- The 5G-Xcast middleware implements the main controller managing and aggregating the information conveyed through the optimisation techniques analysed and developed in the 5G-Xcast project. In a traditional case the player interacts with the operating system to fetch content from the network(s). In 5G-Xcast, the Player ABR's control logics can depend on the Middleware controller as Audio/Video (AV) data can be fetched through ABR techniques as well as

other existing technologies mentioned in this document. Similarly, when data flows are forced to pass through the Middleware, the generic application becomes exposed to the events triggered by that Middleware execution.

- Further control logics exist outside the boundaries of the Application Intelligence Layer. For example, the functions that manage QoS in a network slice can lead the Middleware to switch from a mobile access network to a fixed one to preserve the QoE in playout sessions.

2.3 Relationship between multicast and Quality of Experience

Quality of Experience deals with the level of satisfaction (or dissatisfaction) of users interacting with services. Consequently, it makes sense to consider the level of QoE in the case of services provided through multicast delivery.

In real delivery infrastructures such as IPTV networks, multicast involves a portion of the distribution chain usually composed of multicast gateways (e.g. physically deployed in access networks or implemented as functions on domestic set-top boxes) and multicast servers. Moreover, in addition to multicast functionalities, gateways can implement unicast mechanism as well as caching systems used to store pre-fetched content. The association of these elements and the usage of fast channel techniques [17] can guarantee that the global content acquisition time is not affected by multicast delivery even during channel switches [18]. Consequently, in such infrastructures the deployment of multicast mechanisms does not impact Quality of Experience. In the absence of such network elements and functionalities, multicast delivery could affect the viewing experience as explained in the following paragraphs. Internally, multicast configurations are realised by building and pruning multicast trees; normally this process entails a reconfiguration of routers, where routes are dynamically added and removed when nodes *join* and *leave* multicast groups (e.g. by sending IGMP messages).

- Depending on its latency, a *join* procedure could impact the initial delay during the establishment of a streaming session.
- In the context of ABR multicast, quality switches from lower to higher resolutions normally involves *leave* and *join* operations to multicast groups delivering data at higher bitrates. In this case, the latency of a *join* operation could increase the percentage of time spent consuming content at lower quality.
- Similarly, quality switches from higher to lower resolutions entail the execution of *leave* and *join* operations to multicast groups associated with lower bitrates; such transitions are performed when the quality representation is too high with respect to the available bandwidth. If the completion of a *leave* operation has a considerable latency it could increase the number of stalling events and their lengths because nodes keep receiving data at high bitrates until they are successfully removed from multicast trees. Moreover, in case the execution of a *join* succeeds before completing the *leave* operation, the overlap of incoming flows may completely overwhelm the receiver (and, in some cases, even the access network itself), producing a stronger impact on both stall metrics. Such problem is generally solved by multicast encapsulation protocols (e.g. [19]).
- Even though it is not directly measured by the model, throughput plays a key role in QoE. Perceived throughput is generally influenced by all activities carried out by the multicast encapsulation protocol, even though some components such as congestion control may have stronger impacts. In this context it is important to figure out the role of important variables such as round-trip time and probability of data loss in influencing throughput. For example, the throughput of TCP is approximated by the following equation:

$$T = \frac{s \cdot c}{RTT\sqrt{p}}$$

where RTT is the round-trip time, s is the packet size, c is a constant ranging in $[0.9, 1.5]$ and p is the packet loss rate. A multicast encapsulation protocol may present a different equation, i.e. different throughput for the same values of variables. For example, if data recovery is mainly realised in unicast HTTP, already a simple change from non-persistent to persistent connections has an impact on the throughput equation.

3 Key Features of the Content Delivery Framework

The Content Delivery Framework is set out in 5G-Xcast deliverable D5.2 [2]. To help make the current document self-contained, we will briefly review the key elements of this here.

Our overall vision to combined normal Internet content delivery, this means the use of Content Delivery Networks for global scaling, analytics etc., with point to multipoint distribution at the edge of the network for efficient delivery of media streams that are consumed simultaneously. This is represented in Figure 3.

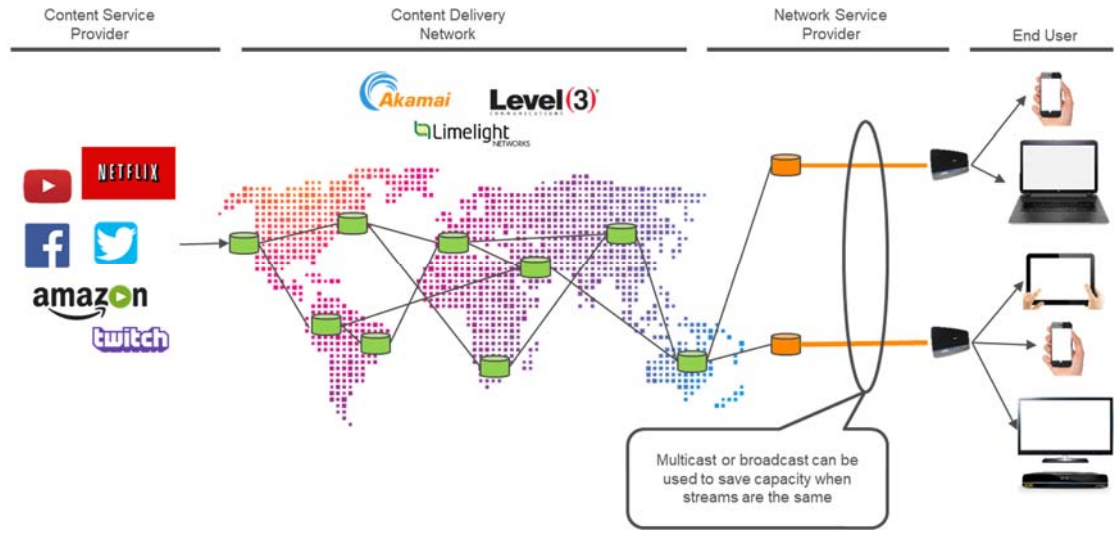

Figure 3 High Level Overview of CDN networks

To realise this vision in this specific exemplary architecture, two proxies (or generic Functions in our document) are introduced to tunnel and hide the complexity of using multicast from the client applications and to some extent, from the CDN. The placement of these proxies on the fixed and mobile networks is shown in Figure *4*. 5G-Xcast framework (D5.2 [2]) identifies the root of the multicast tree as “Function X” and the leaves of the multicast tree as “Function Y”. In DVB Multicast ABR (mABR) as just one exemplary architecture, these two Functions are named “Multicast Server” and “Multicast Gateway” respectively [20]. We find keeping Function X and Y abstract naming useful to avoid introducing pre-conceptions of the roles of these functions or specific architectures and implementations.

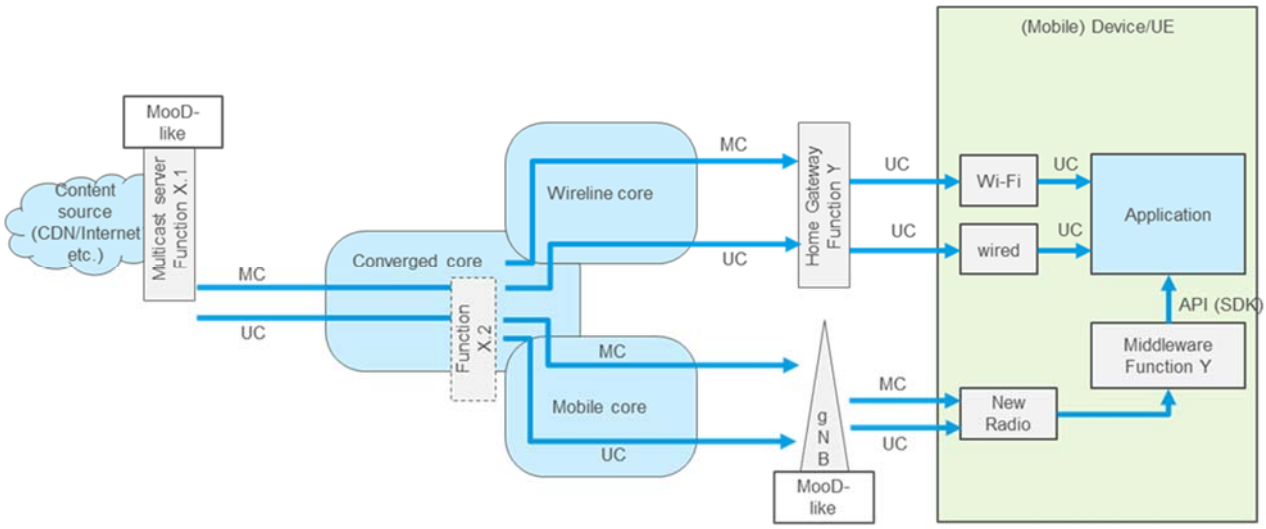

Figure 4 5G-Xcast WP5.2 Framework

Insertion of proxies X and Y at the root and leafs of the multicast tree respectively.

The Function X will ingest content from the Content Delivery network to convert it into multicast for delivery to the functions which have an interest in the content. The functions Y will present the content back to the client application as it if had come directly from the Content Delivery Network.

In the next sections, we will discuss the operation of various techniques which are, or could be, applied to improve the QoE of streamed content. For each of these techniques, we will discuss the impact of introducing multicast into the delivery path.

4 Impact of hybrid CDN-multicast delivery

Normally, CDN's deliver unicast streams. Inserting multicast in the delivery path will necessarily have an impact on some of the CDN behaviour. This section discusses this impact. We start with a basic review of normal CDN operation, then discuss the impact of inserting multicast in the delivery path.

4.1 CDN concepts

CDNs are composed of a system of servers allocated on different Points of Presence (POP) in an operator's network. POP may have multiple meanings. It refers here to the dedicated CDN terminology. This applies to both CDN Operator (where the network provider maintains its own CDN) and CDN "As a Service" approaches (where the CDN provider deploys its POPs within operator's network). The servers are either physical or virtual machines which storage capacity, processing, and input/output bit rate are optimised for caching operations. These servers make the video retrieved from the encoders available in the delivery networks and replicate the video for every end-user.

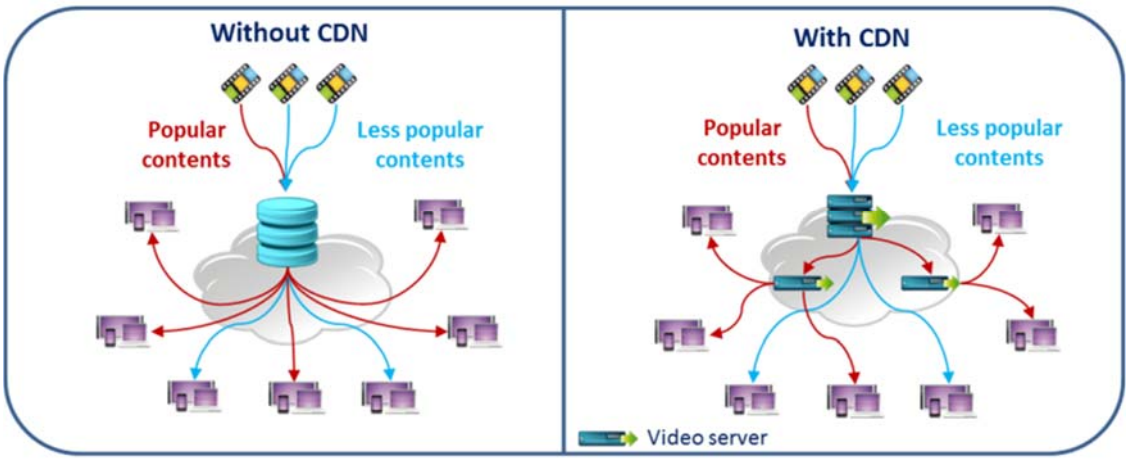

Figure 5 CDN Topology overview

A CDN is also defined by its topology (i.e., the allocation of points of presence across a certain territory) and the rules that determine the "best" server to distribute video content.

If all video content is stored in a centralised point, then all the requests will be served from this point with increased latency caused by a high computing load at the centralised point, a network congestion on routes to end-users. With this approach, there are multiple drawbacks. The longer content has to travel, the more significant is the cost of delivery for the service provider. In addition, congestion can cause delays resulting in service degradation.

On the contrary, an Internet Service Providers could decide to replicate all video content in regional points of presence to minimise transport costs. However, this approach increases storage costs.

CDN purpose is to find the right balance between these two extremes, relying on the popularity of content. Indeed, video consumption complies with a Pareto-type law: in a standard system, 20 percent of content represents 80 percent of viewings. These figures can vary according to video services offered and users, but the principle remains: not all content has the same popularity factor with viewers. The optimisation

consists of locally replicating the most popular content for a specific region only and centrally storing the other content. As the popularity of content varies over time, it leads to a complex arrangement orchestrated by the CDN. If the requested content is available at the edge servers, the session is streamed from there, if it is not, then the session goes back to the central location.

POPs are part of the CDN topology and are deployed in strategic location closer to the end user. End Users are redirected to those POPs (through HTTP redirect, BGP routing, DNS) based on their locations, profile. This allows the reduction of the bandwidth usage on the ISP backbone and the enhancement of the QoE.

A Content Provider can extend its POPs and grow up its topology based on user demand or marketing strategy.

4.2 Impact of Multicast on CDN Operations

This section discusses the potential impact of the our framework defined in D5.2 [2] on the CDN operations, and if our framework can help CDN operations

CDN possible operations include (non-exhaustive):

- Redirection to a POP (e.g. based on user geolocation, type of service, type of device)
- Analytics: session counting and player/network metrics (e.g. bandwidth usage, time per layer, buffering time etc.)
- User authentication

In order to avoid disrupting the CDN operations, unicast requests must be sent to the CDN. A solution where the Function Y intercepts all the traffic without interacting with the CDN would not be viable.

In D5.2 [2] a message sequence describing how the framework could work has been proposed (for convenience it has been copied in “Framework for playing video” (A)). This is just one option and others may be envisioned. It will be used as reference for the next sections. In this workflow we can see that the UE starts the session with a unicast request to the CDN Node. As a result, the CDN is still informed of the new sessions; it is the CDN that redirects the UE to the function Y (as part of its redirection procedure)

Our framework focuses on the delivery of the content in multicast, any proprietary messages used for analytics purpose (Session start, session stop, bytes rea, etc.), authentication or any CDN specific methods still remain in unicast.

Taking this into account we can consider that our framework is compliant and should be transparent to CDN operations. CDN operational logic might however need to be adapted as, for example, the initial redirect differs from the current redirection algorithms.

4.3 Content protection

Many content providers encrypt the media payload of streams in order to protect the digital rights embodied in the content. This encryption applies to the actual data bytes, and not to the underlying IP transport layers; the packets are readable, but their contents are unintelligible without decryption (it is of course possible to combine transport *and* payload encryption but in most application applying two levels of

encryption is considered unnecessary – if the first lock is impenetrable then there is no need for a second lock).

Even when the media payloads are encrypted, some components of the content are often left not encrypted. Examples of unencrypted content components are the manifests for ABR streams or the MPEG 2 headers of TS packets when Conditional Access is used.

In the Content Delivery Framework being explored in this project, some knowledge of the media is required to select the multicast bitrate rate. It may be possible to signal this explicitly from, for example, a CDN operator, or by inspecting the manifest of the content to be streamed.

4.4 HTTPS Session protection

All communications sent over regular HTTP connections are in 'plain text' and can be read by any hacker that manages to break into the connection. This presents a clear danger if the 'communication' is sensitive. With a HTTPS connection, all communications are securely encrypted. This means that even if somebody breaks into the connection between the receiver and the server, the data which passes there is still protected.

Hyper Text Transfer Protocol Secure (HTTPS) is the secure version of HTTP. When this protocol is used, all communications between the two entities at stake are encrypted.

HTTPS uses the Transport Layer Security (TLS) protocol to encrypt communications between a web client and a web server. This is based on an 'asymmetric' Public Key Infrastructure (PKI) system that uses two 'keys' to encrypt communications: a 'public' key and a 'private' key. Anything encrypted with the private key can only be decrypted by the public key and vice-versa.

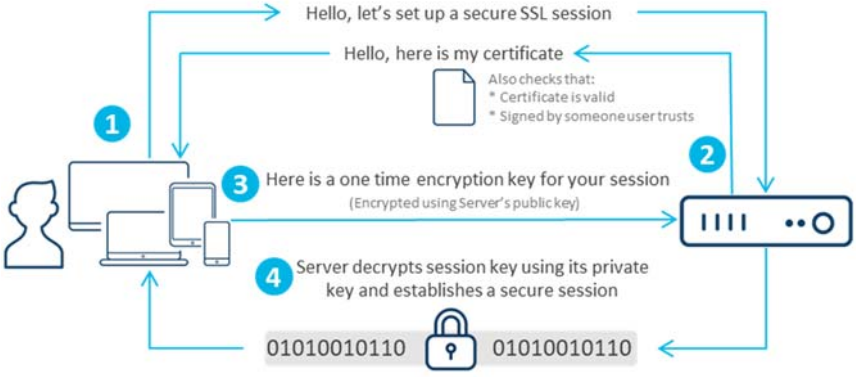

Figure 6 Encryption using HTTPS

As the names suggest, the 'private' key should be kept strictly protected and should only be accessible to the owner of the private key. In the case of a website, the private key remains securely ensconced on the web server. Conversely, the public key is intended to be distributed to anybody and everybody that needs to be able to decrypt information that was encrypted with the private key.

When an HTTPS session is initiated by a client, the server sends a certificate back to the client containing its public key. A negotiation between client and server (called the 'handshake') then generates shared secrets (called 'session keys') that establish a uniquely secure connection between the two parties.

4.4.1 Multicast and HTTPS Challenges

The following diagram depicts the connections that can be secured by using HTTPS transport (green arrows) in a generic CDN architecture:

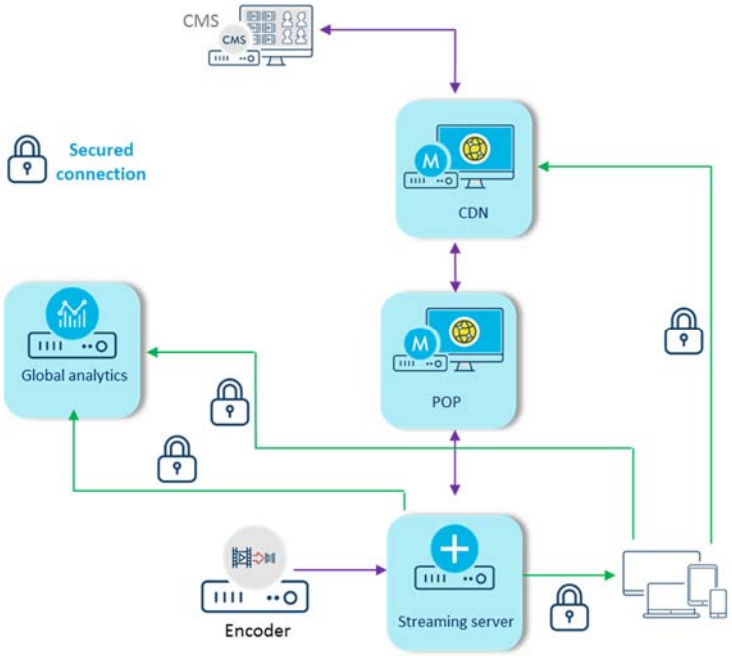

Figure 7 Reference CDN Architecture with HTTPS

In our case, we are interested in the content delivery. Two secured channels are therefore of interest:

- Between the media player and the CDN: This link is used for session initiation. The CDN based on its topology redirects the UE to an origin server.
- Between the origin server (piece of software in the POP which is in charge of delivering the session) and the video player: This is the effective link used during video playback. The UE directly retrieves chunks from the streaming server

The next figure now adds the Function Y and X defined by our Framework on top of this:

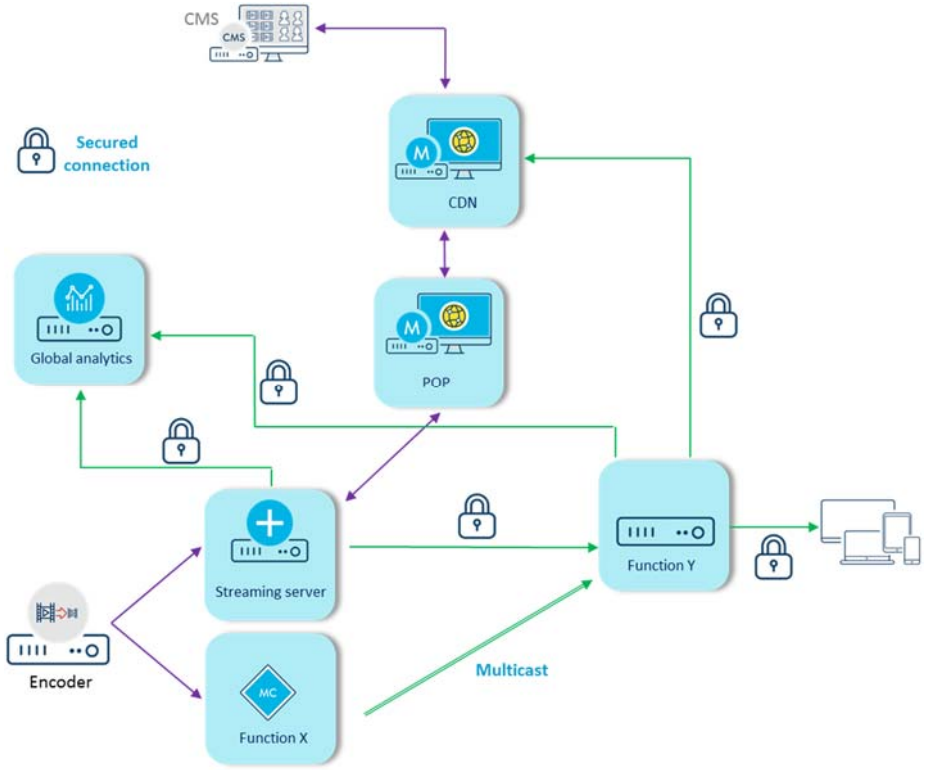

Figure 8 mABR CDN Architecture with HTTPS

On a very basic scenario, Function Y would be implemented as a proxy. However, it is not possible for a proxy to intercept any outgoing HTTPS packet due to the data encryption without a sufficient level of trust.

In our Framework a work-around can be realized by issuing multiple redirections. If we take a close look at the message exchange depicted in deliverable D5.2 [2],we can see that the CDN redirects the player to the NSP domain. The media player then resolves NSP domain through a DNS request to the Home Gateway, which in turn redirects to Function Y.

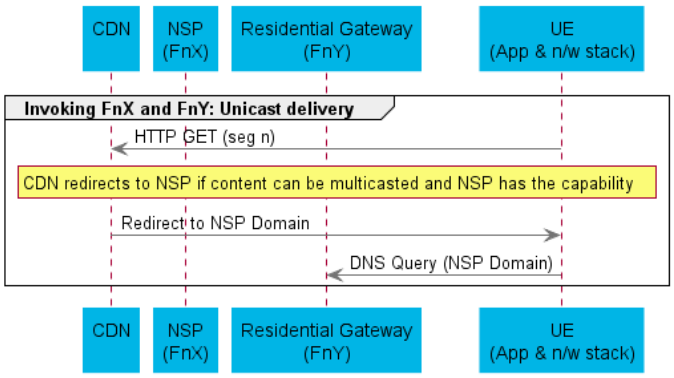

Figure 9 5GXCast framework redirection call flow

All following unicast requests will be done on the Residential Gateway. These unicast requests can be HTTPS encrypted, the certificate being exchanged between the Residential Gateway and the UE. The Residential Gateway acts a HTTPS proxy to the CDN and is in charge of encrypting the session to the CDN.

Once the content stream is available in Multicast, it is transparent for the media player as it continues to communicate with the Residential Gateway. The transport protocol used for the multicast may employ Datagram Transport Layer Encryption (DTLS).

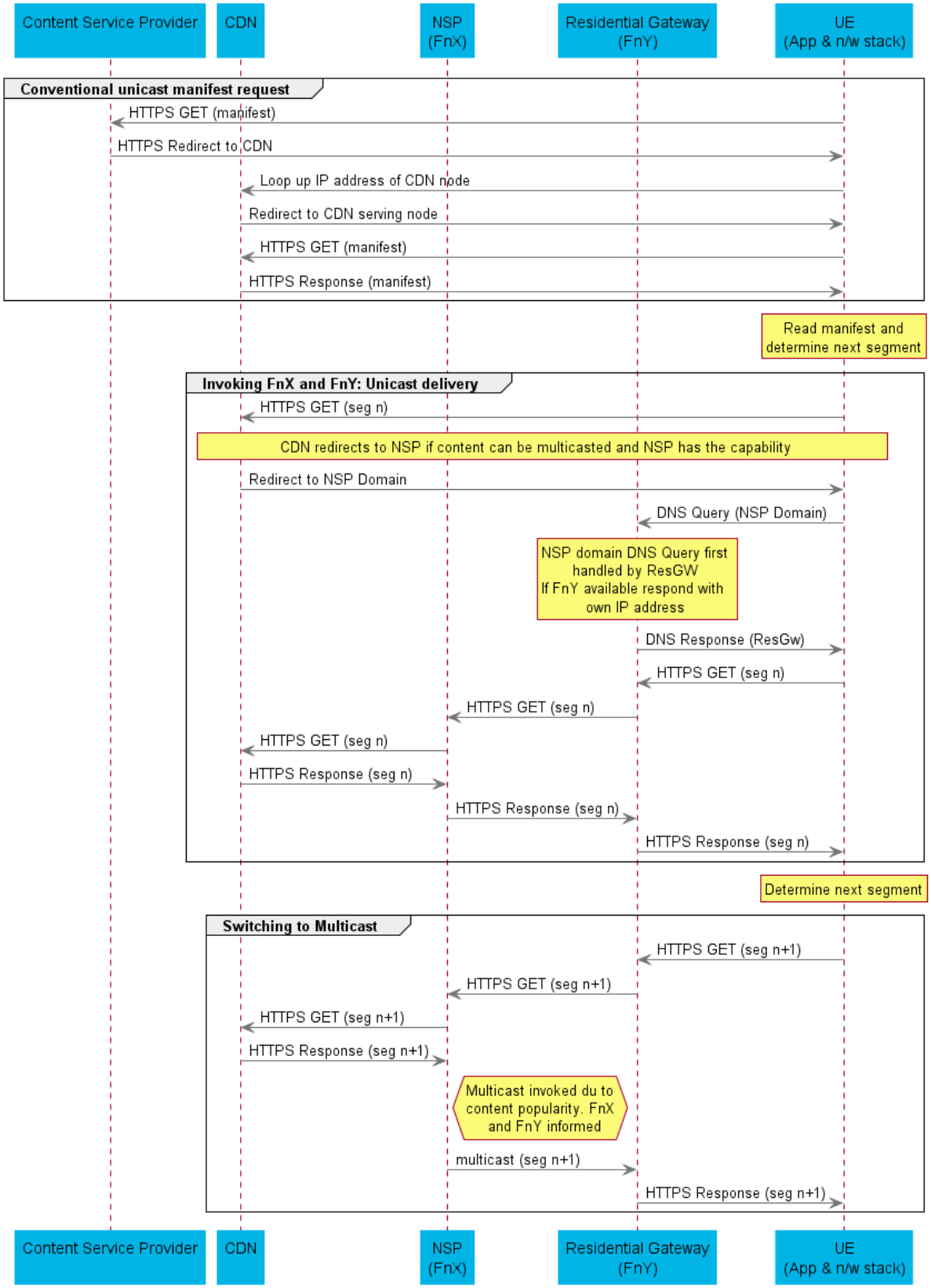

Figure 10 5GXcast framework call flow with HTTPS

It should be noted that this is one possible implementation. The processing of two HTTPS message flow on the Residential Gateway can have severe dimensioning impacts in terms of CPU and RAM usage, as it typically runs on “small” environments (mobile phones, Home Gateways, STBs…). If the CDN and the Function X is operated by the same entity and the Function Y is an HTTP handler located in the UE (e.g. middleware, SDK, library), the CDN can advertise a multicast QUIC session to the Function Y. This architecture requires that the Function X is located so that network infrastructure between the Function X and Function Y supports IP multicast transport, which most likely requires a collaboration between CDN and NSP. The Function Y may be provided by a third party that has no relationship with the CDN nor the NSP.

4.5 Multi-CDN

4.5.1 Using multiple CDN connectivity

In a typical CDN architecture when a user requests for a streaming session this request is redirected to a single CDN. This type of architecture has one major drawback; the service provider relies on the CDN behaviour. The CDN becomes a single point of failure in the delivery chain:

- If for any reason the CDN fails to deliver the media session, then the whole end to end chain is broken.
- If connectivity to the CDN is bad due for example to network routing issues or if it is overloaded then the users will suffer from bad QoE.

By using multiple CDNs it is possible to remove this single point of failure, thus improving QoE and system robustness

There are other use cases where a “multiple CDN” architecture brings benefits:

- Thanks to the optimisation of the network conditions, it allows reducing the end to end latency by minimizing the required buffer on the media player. How the minimization of the buffer size impacts end to end latency is further described in section 5.3
- The media session is resilient to any CDN failure. If a CDN failed, then the packet requests are naturally handled by the still up CDNs, thus offering a seamless failover.

While MultiCDN technology brings benefits, there are some drawbacks:

- MultiCDN middleware may need to be integrated with various CDNs.
- The media delivery chain becomes even more complex with the addition of a new software on top of the media player

4.5.2 Dealing with multiple CDNs

Nowadays, media players request content using a single URL and, as a result, only one CDN is used to perform the streaming of the session. In order to use multiple CDNs to overcome the problems described above of using a single CDN, we have to trick the player by intercepting requests and redistributing them to different destinations.

The following figure proposes a basic workflow to illustrate this behaviour.

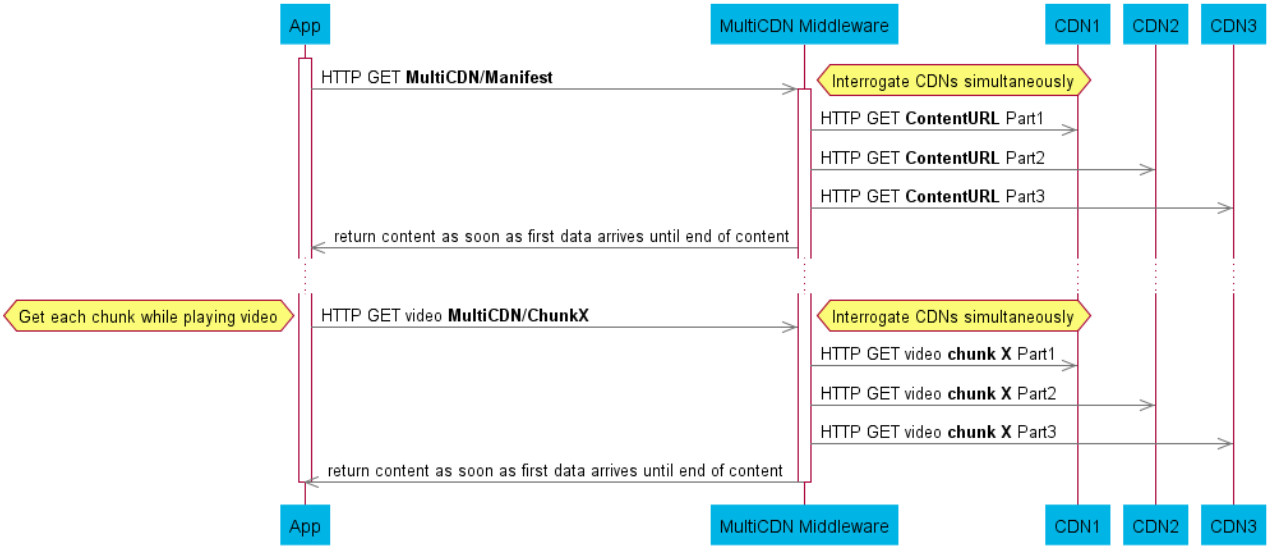

Figure 11 MultiCDN call flow

In this workflow, the media player application (App) tries to fetch the manifest or segments from a configured URL. Requests, using the standard manifest to this specific URL are intercepted by a MultiCDN middleware in the UE which is in charge of distributing the requests on various CDNs. Afterwards the responses are aggregated and the merged result is sent back to the application. The use of multiple CDNs does not necessarily imply that the content is stored in different locations; the Content Provider may have its Origin server and the CDN network is just used to deliver it. On the other hand, we can imagine that each CDN has its own local POPs containing the content. The use of MultiCDN technology is completely transparent in that regard.

4.5.3 Request distribution intelligence

The middleware added value resides in the request distribution intelligence. Several algorithms could be envisaged (non-exhaustive):

- **Round robin**: all CDNs are interrogated in a rotating sequential manner
- **Weighted round robin**: equivalent to the latter but each CDN has a different weight. This weight could be statically configured or obtained dynamically using network information (RTT, Bandwidth…)
- **Least connection**: the CDN with less session is asked first. This however implies that the middleware knows the number of session on each CDN
- **Chained Failover**: all requests goes to a single CDN, if this one fails then requests are sent to the second in the list and so on.

Distribution can be done at different level. In the context of ABR, we could just balance the chunks requests. By doing so, we remove the single point of failure. The increase in terms of QoE is however limited. Chunks are usually too large (around 6 seconds) to get a real benefit. Indeed, if one of the CDNs connections has limited bandwidth then the player may decide to switch to a lower layer.

4.5.4 HTTP Range requests for fine-grained load balancing

Let's take a step back and analyse where ABR protocols come from: they rely on **HTTP protocol** which is based on **TCP**. Those two protocols provide two interesting features:

- TCP inherently adapts its bitrate using the available bandwidth on the link thanks to its congestion algorithm. This means that we can use “intelligent” load balancing algorithms like Weighted Round Robin
- HTTP offers an interesting feature introduced for download pause/resume. If the server is compliant (and as enabled this feature), HTTP client may ask fragments of an asset instead of the full content. A content can therefore be divided in multiple fragments

Combining these two functionalities, we can adapt our algorithm: divide the chunks in fragments of various sizes depending on the CDN’s available bandwidth:

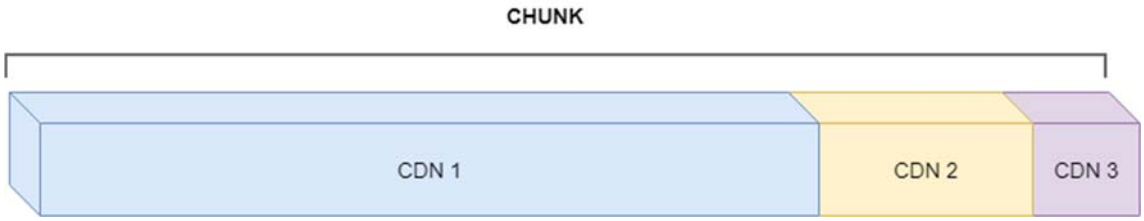

Figure 12 MultiCDN load balancing

A second option would be to divide the chunks in fixed fragment size and allocate more or less fragments per CDN depending on the available bandwidth:

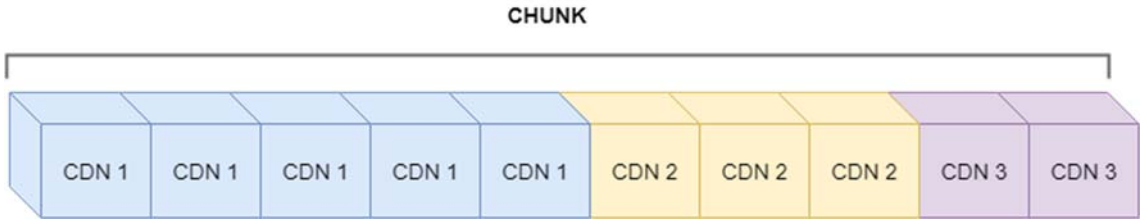

Figure 13 MultiCDN enhanced load balancing

Based on this concept, a lot more complex algorithms can be imagined but that is left to implementers’ discretion.

4.5.5 Impact of multicast on MultiCDN

The technology described here relies on the emission of unicast requests to multiple CDNs. One key element is the use of HTTP range request to load balance the traffic amongst these CDNs. On the other hand, our framework uses the multicast as a way to enhance the user QoE. We will here discuss if the use of this framework and the use of multicast prevents the MultiCDN technology from working and vice versa.

The message flow described in our framework focuses on the use of a single CDN. This CDN redirects to the residential gateway if a multicast can be used.

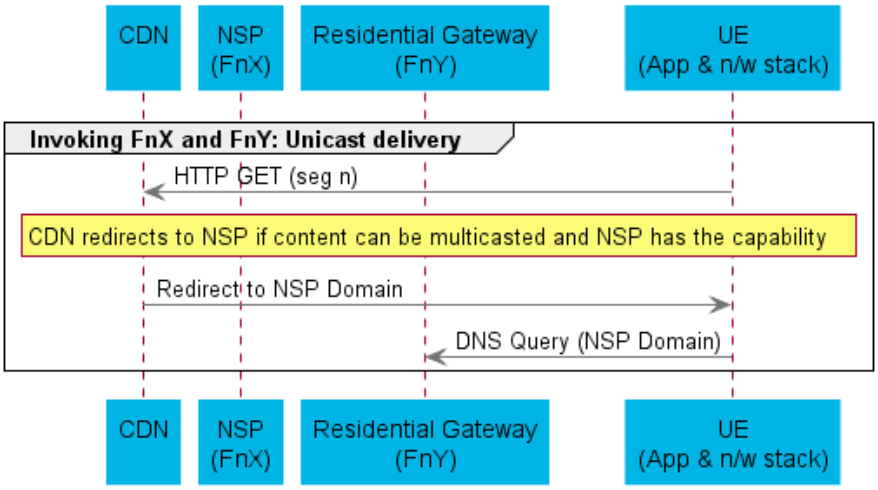

Figure 14 5GXcast framework redirection call flow

With MultiCDN technology the video player will request multiple CDNs and each of these will make their own redirection.

We can imagine a scenario where one of those CDNs implements the 5GxCast framework. The following sequence diagram describes this case and enlightens the fact that our Framework is transparent to the MultiCDN UE. Three GET requests are launched to three different CDNs with various byte ranges. CDN1 is 5GxCast Framework enabled while the other two are regular CDNs.

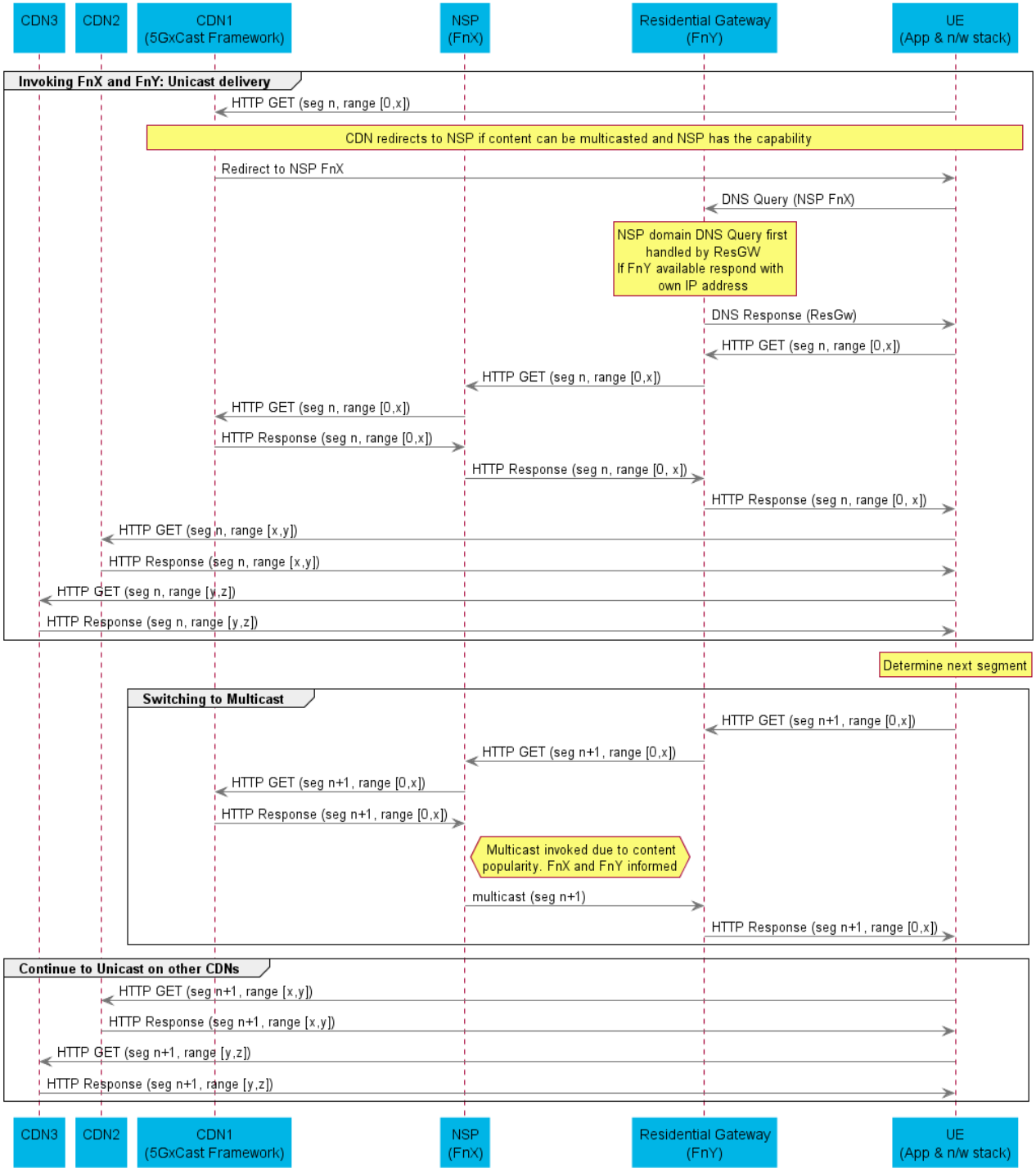

Figure 15 5GXCast framework call flow with MultiCDN

In that particular scenario, once this 5G-Xcast enabled CDN decides to switch to multicast, we can see that this will be completely transparent for the video player since Function Y continues to convert the multicast back to unicast. The only prerequisite is support for HTTP Range requests on Function Y (which is a basic feature of any modern HTTP server) if the fine-grained load balancing feature is desired.

Although this solution may work, the various technologies involved (unicast vs multicast) could make it difficult to integrate. Also, multicast is put in place to improve

the efficiency of the network and with only a portion of the CDNs using multicast this efficiency is considerably reduced.

Going one step further, MultiCDN and our Framework can work together resulting in a win-win situation. If the MultiCDN Middleware is informed of the multicast availability on one of the CDNs (thanks to a dedicated API), it can rearrange its load balancing algorithms by forwarding all requests to this particular CDN. By doing this, we reduce the traffic on the network and enhance the user QoE.

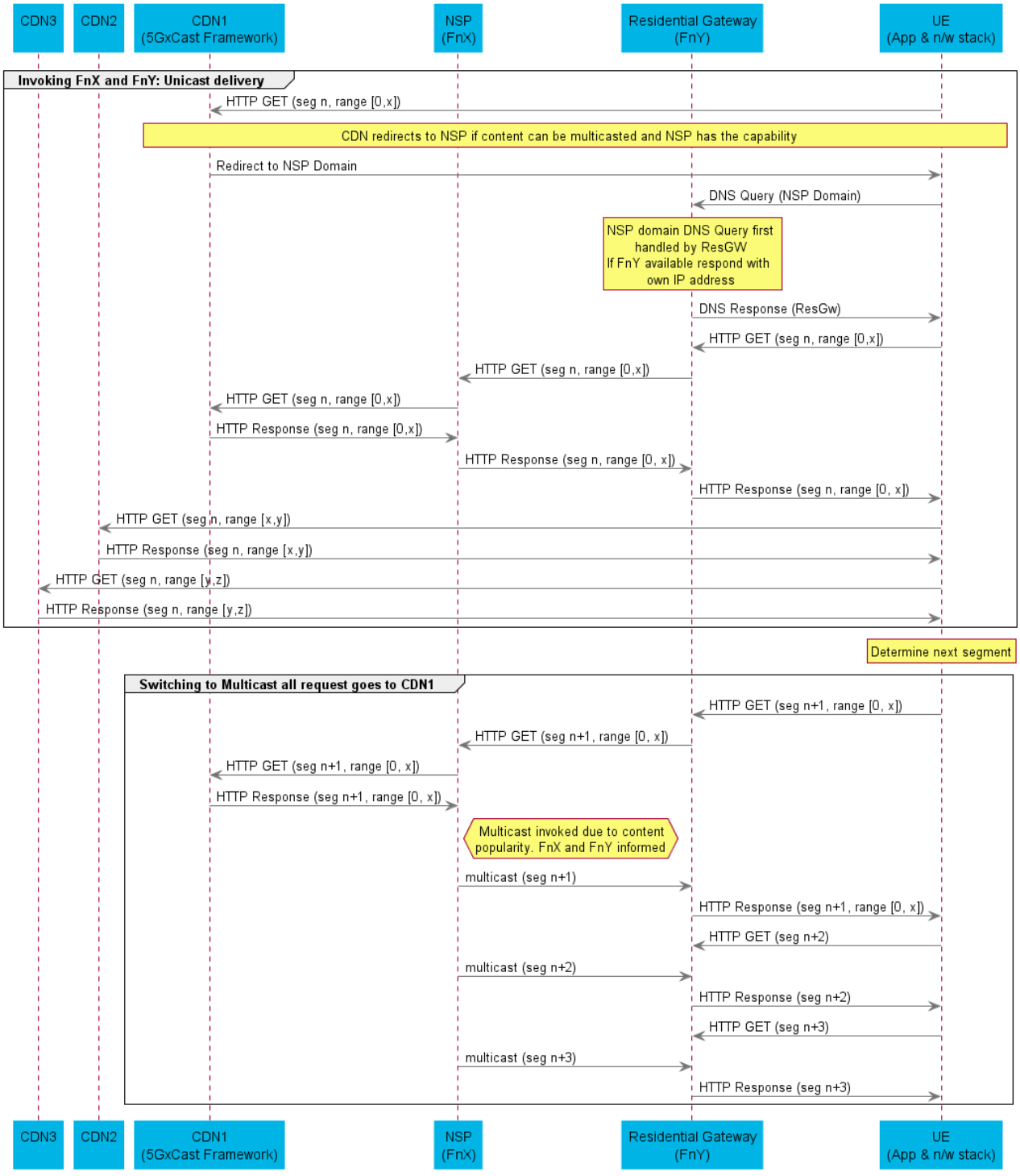

Figure 16 Intelligent 5GXCast framework call flow with MultiCDN

In the following we give some other possible use cases where MultiCDN technology can efficiently complement our framework.

4.5.5.1 Mobility Use Case

In the mobility Use Case, Multi-CDN provides a client with a choice of several CDNs when it moves out of multicast reception and needs to switch into unicast mode. This may enable the media player to maintain the end user's Quality of Experience. Along their journey, this user may get into/out of reach of a multicast channel. This is where the MultiCDN technology can come into play by allowing a user to keep a high QoE. In that case, the MultiCDN technology would be used when no multicast is available. Once the multicast is no longer available on the residential gateway, it could inform the middleware to switch back to MultiCDN mode.

Starting from this statement we can imagine other cases where multicast may not be available:

- The device is not compliant
- The channel is not popular enough to be multicast
- Multicast is not available in this region
- Video on Demand

Multicast cannot reach 100% of the devices or services; there will always be a need for unicast requests.

4.5.5.2 Packet loss

When using multicast, packets may be lost along the way due to the use of User Data Protocol (UDP). Reconstruction of those lost packets can be done by sending unicast retransmission requests to the origin. The use of Multi CDN combined with HTTP range requests is interesting here to reduce the latency for the availability of such erroneous packets.

5 Impact on Video Streaming technologies

Having considered the impact on a CDN of introducing multicast at the edge of the network, we now turn our attention to the impact on the point to point streaming technology, from the CDN edge node to the client application.

Our proposed approach for large-scale content distribution requires the insertion of a multicast path into an otherwise unicast media stream. Multicast streams will use UDP, in which the function X will dispatch packets, usually at a fixed rate (equivalent or greater than the [video] coding rate).

In this section, we will quickly review the normal operation of ABR streaming, then discuss the implications of introducing multicast into the delivery path.

5.1 ABR Concept

In unicast, ABR is implemented in the UE Middleware (MW) requesting the relevant bitrate piece/segment/part using HTTP GET to the Content Provider. When the throughput to that UE drops, the MW will request lower bitrate content piece, this decision is made by the video player control logics. The Content Provider responds with the next segment according to the different bitrate in that request.

A variety of ABR implementations exists, among those, the most used are:

- MPEG DASH
- Apple HLS
- Microsoft Smooth Streaming

These implementations rely on a - manifest file which contains information describing the various segments and their bitrate. The video player first retrieves this manifest and then chooses the bitrate of the next segments to be fetched from the source.

The main benefits of the ABR technology are the following:

- Works over HTTP/TCP protocol suite. This means that the packets have no difficulties traversing a Firewall or NAT.
- No “intelligence” is required on the streamer (stateless sessions) which allows the development of high-performance streamers (over 100GB/s). This greatly enhances the overall scalability.
- The client application manages the packets it wants to receive based on network conditions, user preferences, available computing resources as well as any client specific intelligence.

In ABR, the video player switches from different video encoding bitrates depending on the network conditions. In general, the switch relies on dedicated algorithms. One of the main triggers for those algorithms is the time to download segments from the origin.

The purpose of this section is to identify the possible impacts when replacing the unicast connection to the origin by a Multicast stream.

5.2 Using multicast with ABR

One of the main objectives of the 5G-Xcast content delivery framework is to limit the impact on the current CDN ecosystem (network, video streamers, video player, etc.). While using Multicast we have to ensure that the existing ABR algorithms in media players are not adversely impacted.

As stated in section 5.1, an ABR video content is encoded in multiple bitrates, each of which is expressed as a different representation. The lowest bitrates are used by the player when the network conditions are low, while the highest ones will be used when the conditions are high. This bitrate selection algorithm is the essence of the ABR technology. If the player selects a wrong layer then users will suffer from bad QoE: either bad video quality, or high number of rebuffering events.

The availability of multiple encoding profiles and their correct selection is therefore a prerequisite for any ABR video streaming solution. Adding a Multicast stream within the end to end delivery path needs to be done without interfering with the layer selection algorithms. Some possible options are described hereafter:

- Option 1: Have a multicast stream for each possible layers of every content
    - PROS: simplest implementation, transparent for the content and the player.
    - CONS: expensive in term of network resources. Moreover, Function X and Function Y CPU consumption can be high due to the number of multicast streams to send (on Function X side), join and cache (on Function Y side).

The following figure describes this option and will serve as a reference for the other possibilities. In this figure, we describe a case where two distinct users try to access the same live content. Those users are connected to a different Home Gateway (playing the role of Function Y) and use a different device. Layer selection algorithms chose layer 1 for User 1 and layer 3 for User 2. Function Y in this figure subscribes only to the multicast that are prone to be used by downstream clients.

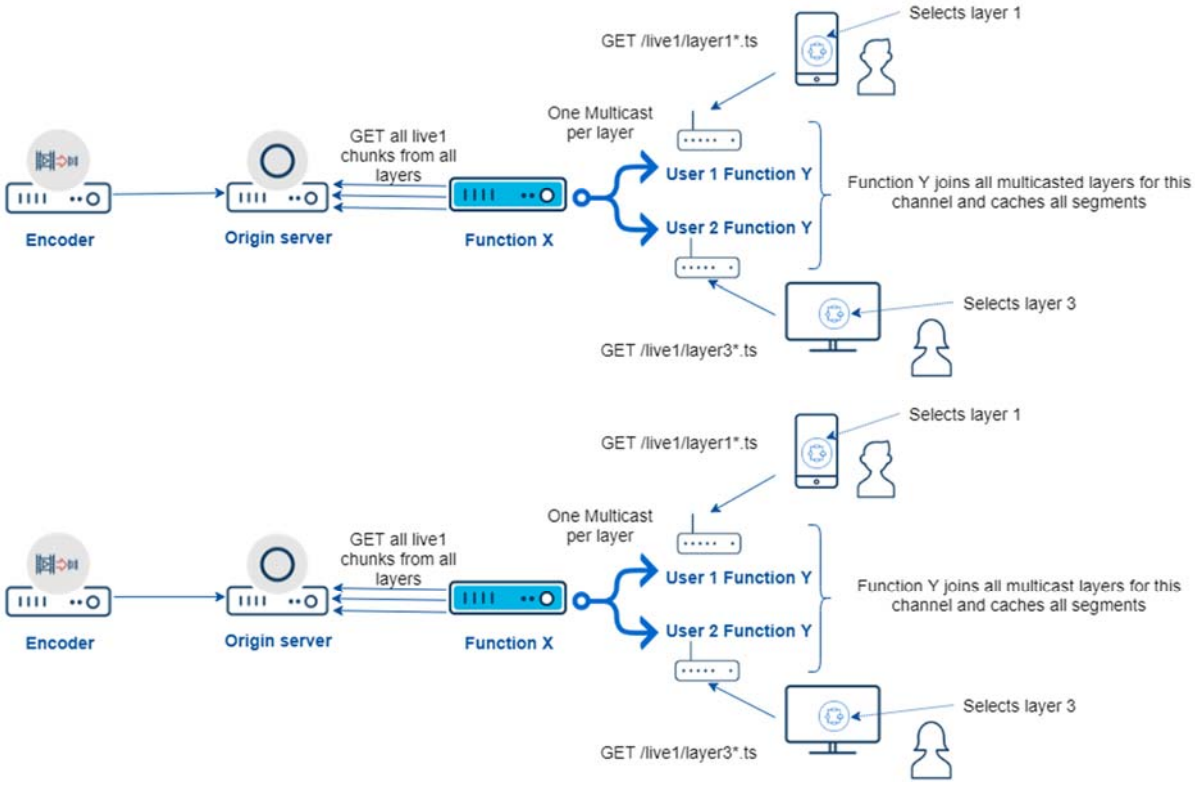

Figure 17 Multicast all layers

- Option 2: Multicast a subset of the available representation layers and provide a means to switch seamlessly from a unicast layer to a multicast one (and vice versa).
    - PROS: Transparent for the content and the player.
    - CONS: Additional intelligence required in the Function Y. CDN logic can be more complex relying on dedicated analytics algorithms to select the layers to be multicast.

This possibility is illustrated in the next figure. We can see that the media player of User 2 still selects layer 3. However, as this layer is not available in Multicast, the Function Y retrieves them directly from the Origin Server. Function Y still needs to receive requests from the player to ensure that in case of layer switching (to layer 2 for example), Function Y can use the segments cached for this layer. Otherwise, if we redirect directly the UE to the Origin Server, then the Function Y will never be informed of the layer switching. In the following figure, the Residential Gateway joined the multicast before the session occurred.

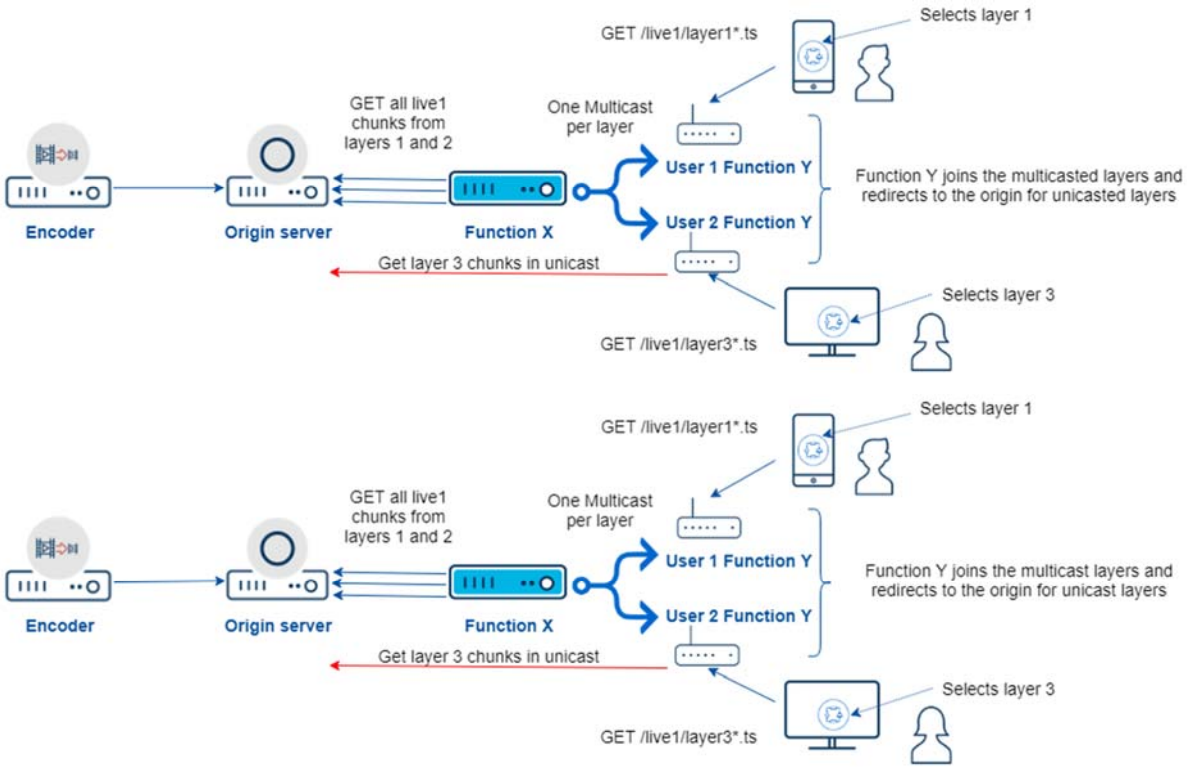

Figure 18 Multicast some layers

- Option 3: Modify manifest when it is requested so that media player is only aware of the multicast layers
    - PROS: less impact on the network, only multicast layers are used (the players do not know about the other layers).
    - CONS:

- Requires interception of the manifest request and modification of the response. This adds a burden on the Residential Gateway software which typically runs on low end piece of equipment.
- Only a small set of layers are used: if the available layers do not correspond to what the player would choose in unicast then the user will suffer from bad QoE. If the highest bitrates are not made available to the video player (although it could have been used due to network conditions), then the user will not obtain the QoE he would have expected.

Figure 19 hereafter depicts this last option, showing the manifest modification done by Function Y.. It also tricks the video player behaviour by forcing it to choose a different layer, since layer 3 is no longer known. In this case, User 2 will not get the highest bitrate due to the unavailability of layer 3 which would have been used otherwise.

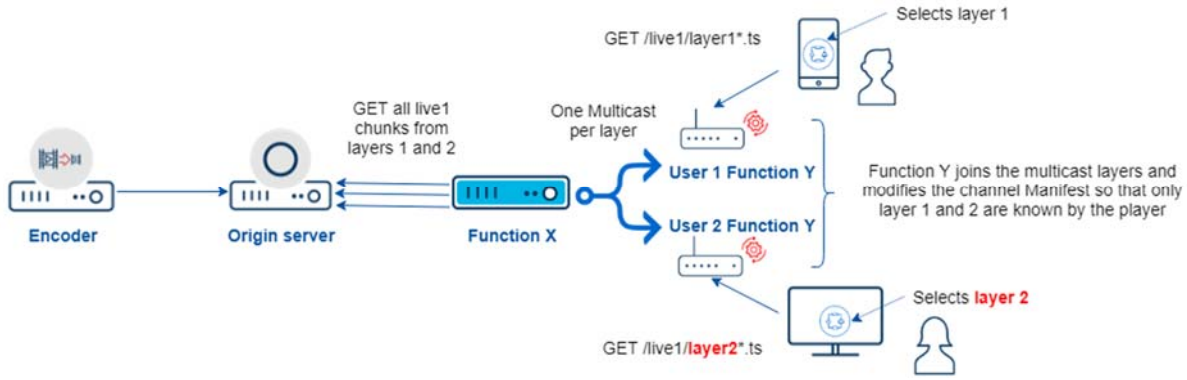

Figure 19 Multicast some layers and modify manifest

Choosing between those options is a matter of optimisation to be decided by the network operator. Other options may also be considered. For example, the Residential Gateway may deliver a bitrate different than the one requested by the video player.

Multicasting all layers from all channels might be tremendously expensive. A trade off accomplished by the second and third option, could be to multicast only some layers of the most popular channels.

5.3 Multicast impacts on latency

5.3.1 The origins of ABR Latency

New devices, including tablets and smartphones, enable viewers to enjoy live sports and news anytime, anywhere, but latency remains a real issue. It is not uncommon for consumers to be watching a soccer match and hear the neighbour next door shout “Goal!!!!!” before seeing it happen. Video streaming delays frequently happen and can be especially unsatisfying during live sports events.

Video services delivered over managed networks typically experience lower delays because there is guaranteed bandwidth and a resulting in limited buffering in the set-top box (STB). For ABR video streaming in Apple HLS, Microsoft Smooth Streaming or MPEG-DASH that is delivered as an OTT service, it’s a whole different scenario. Secondary screens such as connected TVs, smartphones, and tablets are accessible on various unmanaged networks (i.e., 3G, 4G and OTT) where the HTTP ABR streaming format is used. HTTP over TCP is a best-effort protocol, not initially designed to manage live video delivery that requires constant data sending rate. It is subject to

jitter as a consequence of dealing with packet loss and/or router's buffer bloating. It is therefore necessary to use buffering at the player level to guarantee a smooth playback without stalls, with the buffer absorbing the irregular aspects of the HTTP/TCP traffic. That buffer sometimes accounts for even 70 percent of end-to-end delay (sometimes up to 25 seconds). The buffer size is typically a function of the segment duration (typically 3 times a segment duration). The segment duration is selected depending on various criteria including the type of video, the delivery maximum throughput, the number of [quality] representations sub-streams. It is typically between 2 seconds and 6 seconds.

The following figure depicts the various delays in both IPTV and ABR assuming for the latter a 6 seconds segment duration and a player's buffer of 3 segments: Packaging consists in forming an entire segment as a file before sending it.

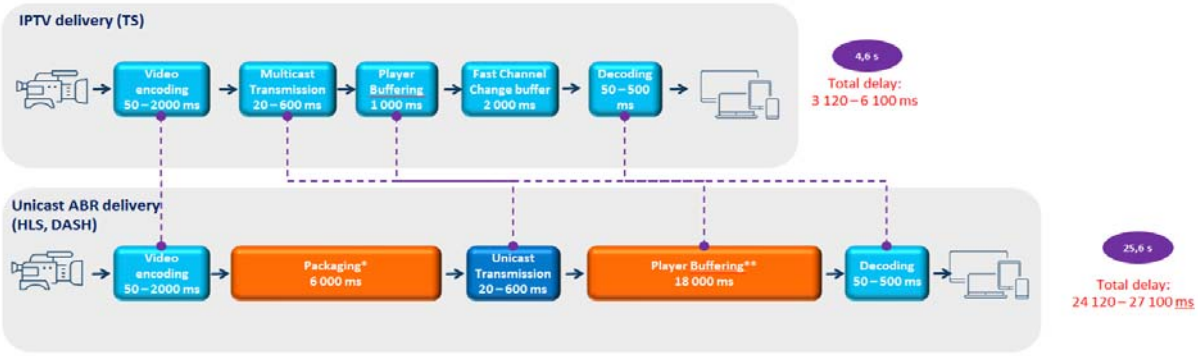

Figure 20 End to End latency when using regular ABR

5.3.2 Solving the jitter problem using Multicast

By using multicast ABR technology the buffer issue can be reduced significantly. Multicast ABR enables operators to successfully stream video without significant buffering on the player side to guarantee a good QoE. The solution involves deploying a unicast-to-multicast transcaster in the head-end and multicast-to-unicast agents at the other end of the multicast pipe (in the home gateways or set-top-box), allowing players to decrease their buffer size significantly assuming stable in home delivery.

Next figure illustrates this and shows the gain in terms of delay when using mABR technology. Function X is represented here by Multicast ABR Encapsulation; Function Y is not shown as it does not impact the delay.

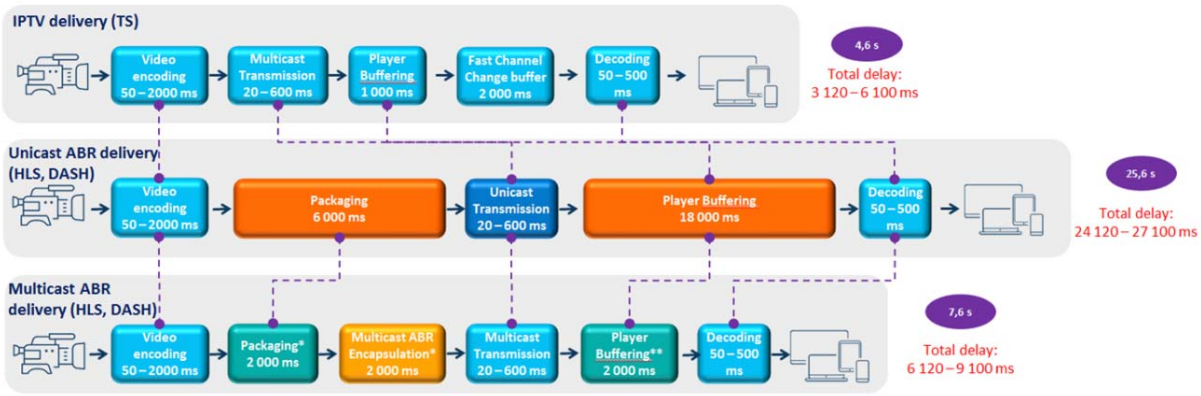

Figure 21 End to End latency when using mABR

But 7 to 8 seconds is still too high nowadays. CMAF standard [21] in addition to HTTP Chunked Transfer Encoding [22] technology allows to bring the latency for video streaming down even further — to 2 seconds and below.

5.3.3 Solving the latency issue with CMAF and CTE

CMAF includes a low latency mode that allows encoders/packagers to generate pieces/chunks of a segments on the fly, a CMAF chunk being (i) smaller (in duration) than a typical HLS of DASH segment and (ii) the CMAF chunk structure being aligned with the coding structure (e.g. a CMAF chunk carries an image) allowing a decoder to proceed CMAF chunk per CMAF chunk. HTTP Chunk Transfer Encoding (HTTP CTE) allows transferring a file in pieces called chunks. Instead of waiting for the segment to complete before transmitting it, operating the HTTP CTE allows sending the segment, chunk by chunk, nearly on the fly when processing live streaming segments. Inherently, this greatly reduces the latency. By combining low latency CMAF format with the HTTP chunked transfer encoding technology at the level of its unicast to multicast transcaster component, video chunks can be sent while they are being processed. Prior to chunked transfer encoding support, the transcaster would need to process a complete media segment before it could be sent to the next phase of the delivery chain, introducing a delay of a segment's length (2 to 6 seconds). The combination of both elements, smaller chunks and delivery of chunks before a whole segment being fully processed, lowers latency in the head-end.

The combination of these technologies: CMAF, Chunked Transfer Encoding and multicast ABR results in a streaming latency even lower than in IPTV with a total delay ranging from 1.6 seconds to 3 seconds.

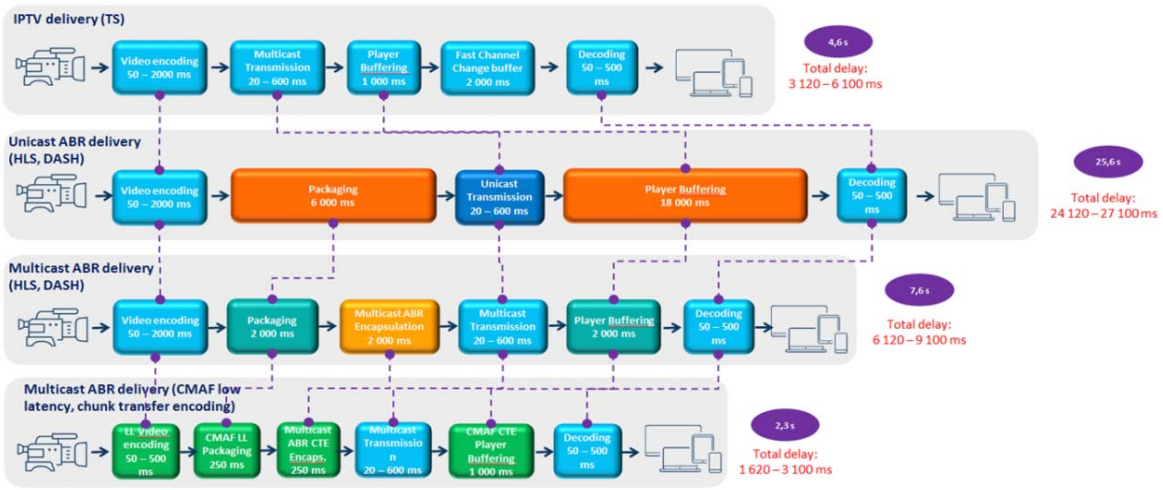

Figure 22 End to End latency with CMAF and Chunked Transfer Encoding plus multicast

5.3.4 HTTP/2

Streaming formats have various ways to avoid terminals from making requests for media segments before they are available. MPEG-DASH uses a timeline to signal the availability time of a segment to the player in the client. Servers and clients are typically not tightly synchronised which adds to the latency. In the 5G-Xcast Content Delivery Framework, HTTP is used between a CDN and the Function X, and between the Function Y and a device. The support of HTTP over multicast QUIC [23] will allow HTTP to be used also between the Function X and the Function Y, wherein the Function Y is located at a home gateway or a device.

The HTTP/2 [24] introduces new functionality that allows servers to push data to clients in an unsolicited manner. The push functionality may further help reduce the latency. A server can push a segment or a chunk to a client as soon as the segment or the chunk is available. A server can select a representation layer (i.e. media quality) which is also based on earlier request from a client. The unsolicited push may cause an issue when the connection between a server and a client deteriorates fast and a server is pushing a high-quality representation layer. Only the reception of request for a lower-quality layer can trigger the server to react to the connection changes.

If a client is connected to a network that supports MEC, a server could access up-to-date access network information of a client via MEC API, e.g. radio network information [25]. Each request response in HTTP/2 is associated with own stream. So, each segment is delivered over its own stream. The PUSH_PROMISE frame header block fragment contains request header fields so the client knows what resource it receives through the push stream. A server could then select a representation layer not only based on earlier requests from a client but also access network information. The MEC API could help a server to detect deteriorating connection quality before receiving a request for lower-quality representation, which should address the issue of a server pushing high-quality representation when connection deteriorates.

The push functionality is a foundation of HTTP over multicast QUIC thus also a multicast session can benefit from lower latency. A server may establish and advertise a multicast session for each representation layer. Here we require that each representation is associated with a unique domain name. This means that a server can advertise only a multicast session for the same representation layer that is requested by a client. Switching to a multicast session of another presentation layer requires a client to send a request to the server, which then may advertise another multicast session for the newly requested representation layer. QUIC multicast session is terminated either explicitly by means of the connection close header sent by a server or implicitly upon the expiry of idle timeout. The current draft of HTTP over multicast QUIC does not allow to explicitly signal a session duration. The consequence is that a client has to always request a representation layer over unicast before it can join a multicast session even if the multicast session has been active and session parameters did not change because the client cannot be aware of this.

5.4 Transport protocol optimisation (e.g. QUIC)

TCP has a very complex set of dynamics and control loops, determined by the interaction of a congestion algorithm with network queues and packet loss. The goal of a congestion algorithm is to adapt the sending rate of TCP segments to match the network throughput and be ‘fair’ [reference to IETF RFC on network fairness] to other TCP streams which pass through the same bottlenecks.

The traditional TCP congestion algorithms will be clocked by acknowledgements, such that their rate of adaptation to changing network conditions will increase with increasing RTT. These algorithms also typically treat all packet loss as a sign of network congestion, and so will reduce their sending rate.

Research over several decades has refined TCP’s congestion algorithm and proposed a number of variants which may have value in particular situations. CUBIC [26] for example, tries to track changes to throughput very quickly and not disadvantage streams on larger RTT connections. BBR [27] uses latency rather than packet loss as an indicator of congestion, and so is less sensitive to physical layer packet loss (which is not congestion-related).

Google also recently introduced a new transport protocol, QUIC, which is layered over UDP, putting the congestion control algorithm in user code, rather than kernel code. This makes it much easier to have plug-in congestion algorithms, either for experimentation, or for use in operational systems to allow the congestion algorithm to behave differently on different network types (such as fixed or mobile networks) or for different content types. QUIC is now in the process of being standardised by the IETF [23]. Google have stated that they will update their clients and servers to use the IETF version, once this is ratified. Others may follow.

As well as research on generic congestion algorithms, there has been work on maximising QoE for certain classes of application, such as video streaming. For example, the congestion algorithm could be adjusted according to the encoding complexity of the content, which results in multiple TCP sessions sharing bandwidth based on QoE fairness rather than bitrate fairness [28]. Alternatively, the congestion algorithm could be linked to the HTTP request/response cycle as a means to reduce variability in delivery times and therefore allow less client buffering to the used [29].

Another way that network operators particularly often try to improve TCP performance is to insert a middlebox into the TCP path to carry out TCP optimisation [30]. In its simplest form, TCP optimisation is the breaking of a single TCP session into two so that each of the two sessions will individually have lower RTT than the original single session. This can significantly increase the bitrate and thus the overall throughput of the end to end connection. This is especially the case when the end to end path comprises of a long RTT backhaul with “unlimited” bandwidth and short RTT (Radio) Access Networks with limited capacity.

The 5G-Xcast Content Delivery Framework proposes inserting multicast into the unicast path for the efficient delivery of HTTP responses to multiple endpoints. This of course will mean breaking the unicast path so that it cannot be a single TCP connection from CDN through to end device. This will have much the same advantages and disadvantages as a middlebox. That is, at least two TCP sessions will be required in the data path, each of which will operate over with a lower RTT than the original end to end TCP session (e.g. CDN to user device). In most cases, this would actually accelerate the TCP throughput, especially if the multicast is delivered over a rate-assured connection.

Techniques such as TCP congestion tuning by a CDN operator will no longer work with multicast in the return path, just as they would not with a middlebox in the path, since the underlying assumption of these techniques is a single TCP session runs between the CDN operator and the user device.

5.5 Multilink

5.5.1 Continuous streaming during mobility in converged network

Multi-connectivity means simultaneous connectivity using more than a single IP connection. Often this is done across different technologies such as combinations of several connections of 5G, LTE and unlicensed technologies such as IEEE 802.11 (Wi-Fi).

In heterogeneous networks, multi-connectivity can provide an improved user experience (e.g. higher bandwidth by the aggregation of the bandwidths from the multiplicity of connections, improved service coverage, reliable mobility etc.). One subcase of multi-connectivity is multilink (ML) is described in detail in Deliverables D.5.1 [1] and D.5.2 [2]. Being an application overlay (above the IP layers), ML is not standardised yet as a whole solution. ML does use a set of standardised protocols and technologies mainly at the IP and below layers.

Existing multilink aggregation is currently implemented for unicast streams, bringing together distinct unicast connections to support high quality of the stream. This is many times termed “bonding”. Currently, bonding does not apply to multicast because it acts by loading each of the links according to its specific momentary “goodput” according to each link own momentary performance parameters (latency, throughput, jitter etc.). Hence it utilises differently each of the links to each if the UEs, clearly not a multicast orientation. Usually, there’s also a feedback channel to report link performance to improve the transmitting side to utilise each of the available links at each point in time. This feedback channel from each UE is another caveat in multicast, where the purpose is to reduce network load and avoid bi-directional management communication with the UE applications.

Overall, within 5G-Xcast WP5 Application layer, the focus is mostly on the unicast multilink solution to provide better QoE and seamless transitions in the use cases where the UE is moving between Unicast (UC) and Multicast (MC) service areas, at coverage edge areas, when sending additional synchronised information for specific UEs and similar.

When the mobile device (user equipment (UE)) is on the edge of the broadcast/multicast area, ML may be used. Due to the 5G New Radio (NR) low received signal strength (for example, in buildings) and as additional other wireless links may be available (e.g. WiFi), this UE may make a decision to optimise its network resources and the overall QoE (seamless transition in this case) by using the multiplicity of available links. So, the user can watch a high quality and reliable video if he receives it via multiple channels using ML. For example, when using Scalable Video Schemes (SVC), then the main stream (base layer) may be transmitted and received via multicast 3GPP, whereas both a duplicated base layer and a higher quality layers stream (enhancement layers) may be transmitted and received via unicast non-3GPP access. While on their own each of the streams may be at non-optimal reception levels, suffering packet losses for example, yet combined together may achieve the desired QoE. It will allow the user to get the better quality video over UC and also allow seamless and quick transition between BC/MC and UC, including during mobility and in the house as it is “always connected” rather than forcing “break before make” in such transitions.

A possible solution may be the usage of ML within the 5G-Xcast Content Delivery Framework:

- For providing seamless user experience during mobility:
    - PROS: easy to implement, simplest implementation, reliability and availability of the service.
    - CONS: extra resources usage.

In this case, when Mobile Network Operator (MNO) sends data over the multicast channel, temporarily an additional data flow may be created which will be recieved via a unicast channel of specific viewers, when they want or need to switch the delivery mode to unicast or when they want to combine MC with the UC for better overall QoE when either or both are poorly received. It is not difficult to implement. During the transition from MC to UC the UE may consume slightly additional power in order to receive both traffic flows and the network spends resources to deliver the additional unicast stream. However, this is a relatively small price to be paid for the seamless user experience when a MC QoE service to specific users is low or during the temporary mobility between different service type coverages within the converged network.

5.5.2 Other multipath protocols

This section discusses two multipath / multilink protocols: multipath TCP (MPTCP) and Multi Path QUIC (MPQUIC). These two protocols are described in [2].

In normal conditions with MPTCP data exchange relating to the same IP connection becomes possible on different routes/paths. Within the 5G-Xcast, in addition to congestion control, MPTCP may be used to improve connection or service reliability. There are several papers [31], [32], where authors proposed to combine the multiplicity feature of MPTCP with the application-layer multicast (ALM). However, such researches were mostly devoted to multicast in fixed networks with more stable performance parameters of each of the connections. In mobile networks during a multicast streaming session, due to variations of the network parameters (packet loss, delay etc.), it is very difficult to predict the performance of each link even for the next packet transmission. Also, preliminary experiments showed that the throughput over an MPTCP connection is quite bursty. Namely, the burstiness increases with an increase in the number of subflows (or IP connections), packet loss and delay. As multiple paths are used, even a bandwidth variation on one of the paths results in a variation of total bandwidth. These variations become worse when all used paths are unstable, resulting in frequent variations. In turn, the adaptation experiences more switching events over time. That's why this solution will be impossible to use with multicast streaming. Of course, relying on the underlying TCP is generically impractical in widely fluctuating networks in terms of latency or RTT, such as wireless ones. In 5G the fluctuations remain high and even higher as the potential lower latency is low yet the maximal latency, end-to-end, remains quite high.

Under low delay conditions, the high packet loss rate causes frequent negative spikes. In these cases, packet losses cause the congestion window to be reset. With an increase in delay, the congestion window grows slowly. Also, frequent packet losses cause the congestion window to reset often. This causes the congestion window to remain at lower values. As a result, the throughput remains smaller than the actual available bandwidth most of the time. So parallel usage of few multicast or set of multicast/unicast channels doesn't give expected outcome (better QoE) or even can cause fading pictures. The situation worsens when the delay along the path doubles and the throughput is less than half of the available bandwidth.

As mentioned above, frequent packet losses in any of the paths cause a low MPTCP throughput and high throughput variations. This is prohibitive especially for live video streaming with high resolution.

To mitigate these effects in MPTCP multicast streaming a modification of adaptation logic is needed. To overcome the burstiness of MPTCP the adaptation logic for MPTCP uses an additional operational parameter called path-stability for multicast/unicast channel. Path-stability on each path represents the instability (variations) of the bandwidth and the total path-stability of all paths represents the instability of the whole MPTCP connection. Thus, path-stability approximates its optimal values only at stable, consistent conditions, which are clearly not the case in cellular networks. Under real conditions it is necessary to use other proprietary specific algorithms and schemes, which are out of scope of 5G-Xcast.

The other problem of MPTCP is that it still has the same head of line blocking if multiple streams are over the same TCP connection and does not handle other traffic types such as various IETF protocols with unreliable performance (such as video UDP over the fluctuating cellular connection). In addition, a lot of middleboxes can block ingress traffic. That will create additional problem for MPTCP implementation into 5G-Xcast framework.

Experience gained with Multipath TCP [2] can be used to propose simple extensions that enable Quick UDP Internet Connection (QUIC) to efficiently use multiple paths during the lifetime of a QUIC connection.

MPQUIC [33] is an under-development and standardization protocol. Potentially MPQUIC could be developed to work with multicast as well, yet on the current stage it has almost the same problems as MPTCP. QUIC encrypts much of the transport layer header, which has good and bad implications. The bad in this case may be that it prevents the possibility of performance-enhancing middleboxes that may help to mitigate performance and reliability issues during multicast streaming. Still MPQUIC is more suitable for multicast streaming than MPTCP. Without packet losses, while the performance of single-path TCP and single-path QUIC are similar, MPQUIC has to potential and goal to outperform MPTCP for both UDP and TCP based traffic. In lossy scenarios, QUIC and MPQUIC are more suited than TCP and MPTCP. QUIC intrinsically provides security by encrypting all content and headers, specifically to prevent observation or interference by middleboxes; hence no additional security is required. However, MPQUIC may degrade the performance of HTTP video delivery (i.e., MPEG-DASH), resulting in problematic impact on the WP5 application layer framework.

To summarise using MPTCP and MPQUIC multipath protocols within the 5G-Xcast framework:

- Multipath TCP:
    - PROS: overall bandwidth, reliability increasing, backward compatibility with plain TCP, seamless network handover.
    - CONS: poor interaction with short/small flows, a lack of infrastructural support for multipath policy (MPTCP extension often can be blocked by middleboxes), huge problems with implementations under real cellular conditions. Not designed for non TCP-based traffic.
- Multipath QUIC:
    - PROS: better aggregation with QUIC than TCP, less packet losses than in MPTCP, simple transition from single path QUIC to MPQUIC, seamless network handover
    - CONS: QUIC is not standardised yet. First version of QUIC protocol will focus on the basic scenarios. Potential problematic performance for non TCP and under lossy and fluctuating conditions. Multipath extensions will only be addressed later, once this first version is ready. Limited number of scenarios. There are some security issues.

5.6 Network handover

Within the 5G-Xcast project, it is now becoming possible to offer seamless video streaming and decreasing the influence of changes in content delivery conditions during cellular handovers. There are several handover schemes to provide smooth multimedia delivery across unicast and multicast networks (for example [34], [35]). These solutions in most cases are most suitable for fixed networks (i.e. Wi-Fi). A solution is still needed that allows very light-weight implementations on resource-limited client devices such as smartphones. Furthermore, desired solutions should allow payload data caching and retransmission in case of gaps of network coverage. Within

5G-Xcast several types of network handover are considered (Figure 23): NR multicast mode – NR unicast mode, NR – LTE, LTE – Wi-Fi, NR-Wi-Fi and others.

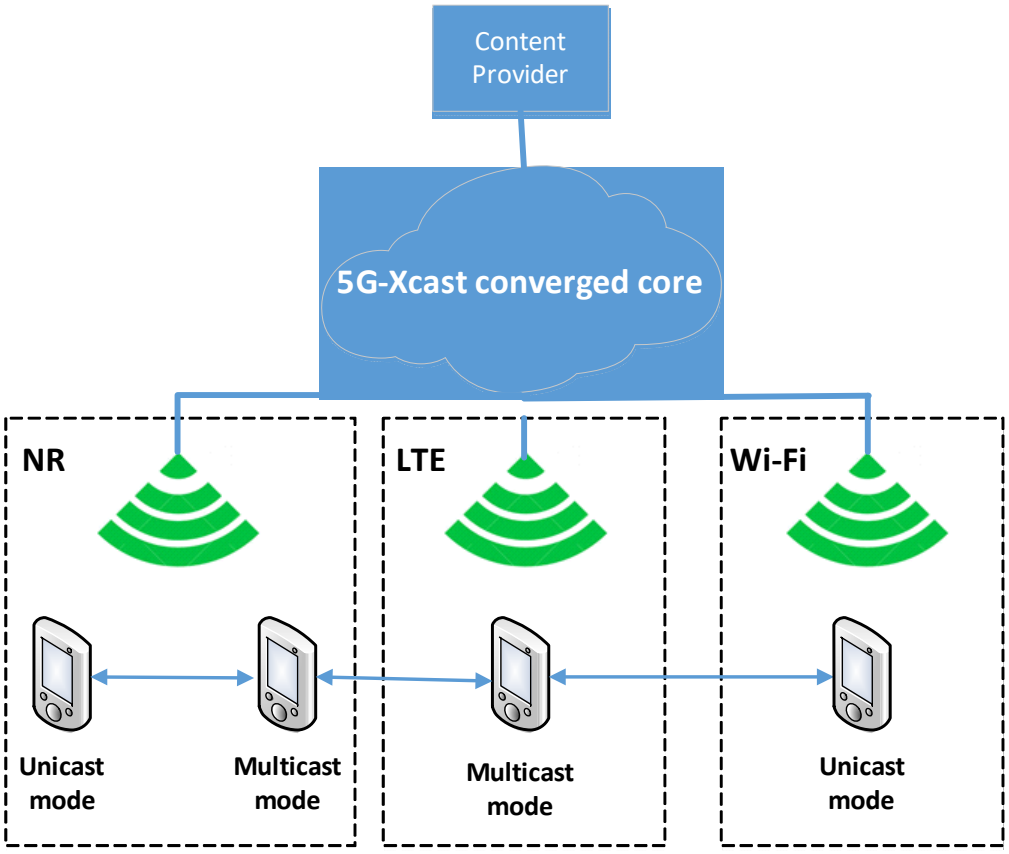

Figure 23 5G-Xcast handover types

If we consider the influence of network handover on continuous video streaming, it would be desirable that the handover process takes less than 50 ms [36]. Of course, in some cases a short handover process, or handover process in general, might cause high levels of packet losses and retransmissions. Another challenge during the network handover is to ensure existing available resources in the network to which the subscriber is moving. In general, due to the MC nature of the 5G-Xcast framework, occasional disruptions due to handover to specific users may be acceptable. However, mobile in many mobility-oriented use cases like communicating while on a train, using ML of 5G connection and the train WiFi (satellite-based or MW-based or other) may overcome these impacts of cellular handovers.

The other type of handover (between MC and UC and vice versa) might be more critical in its QoE expression. [37] shows that the delay caused by this type of handover in realised schemes may reach up to 1 second and even greater. Of course, it will impact the QoE significantly, probably even causing abandoning the viewing of that stream. For the seamless switching, transparent to the subscribers, it is needed to extend buffer size and/or to use specific buffer size changing adaptive algorithms.

Hence ML is considered to provide seamless handover in 5G converged networks within the proposed in the 5G-Xcast Framework [2]. The ML continuous connectivity allows switching between different types of networks, coverage areas and for seamless dynamic switching between different modes (multicast and unicast). The multilink solution does add a new buffering to the middleware within the UEs to accommodate

for the different latencies on the different links. For 5G-based connections such buffering is expected to be rather low (tens of miliseconds at most).

6 Conclusions

5G-Xcast is focussed on developing a point to multipoint 5G capability as part of a wider vision of content delivery which integrates traditional CDN's with broadcast or multicast at the edge of the network. WP5 is focused on introducing such capability at the network operator's level, de-coupled from the mobile or core network. Introducing a broadcast or multicast segment into the delivery path of an otherwise unicast system will certainly have an impact on many of the assumptions built-into these CDN-based unicast delivery systems.

In this document, we examine the impact of introducing multicast into the delivery path on existing distribution technologies at the content provider's and operator's level. Further, we explore whether the use of multicast will still be effective with these optimisation techniques.

Our conclusion is that, in most cases, this approach to Content Delivery has no negative impact on overall QoE of individual viewers and in total. Indeed, there are case where using multicast could reduce latency or improve media quality while reducing the delivery cost and overhead.

Careful design of the way that multicast is inserted into the delivery path is required in order not to confuse ABR adaptation algorithms, but this is not an insurmountable problem.

Other contemporary optimisation techniques, such as TCP optimisation through the use of tailored congestion algorithms that aim to enable CDN and multi CDN operations are disrupted by the insertion of multicast into the delivery path. The counter to this is that the insertion of multicast will in most cases result in more predictable and manageable throughput, reducing the need to TCP optimisation.

We therefore believe that the efficiency gains from using multicast will outweigh any minor loss of effectiveness in any specific optimisation technique.

A Framework for playing video

Following is a simple message flow reference from D5.2 [2] Framework. It describes how a UE can play a video session when using our Framework.

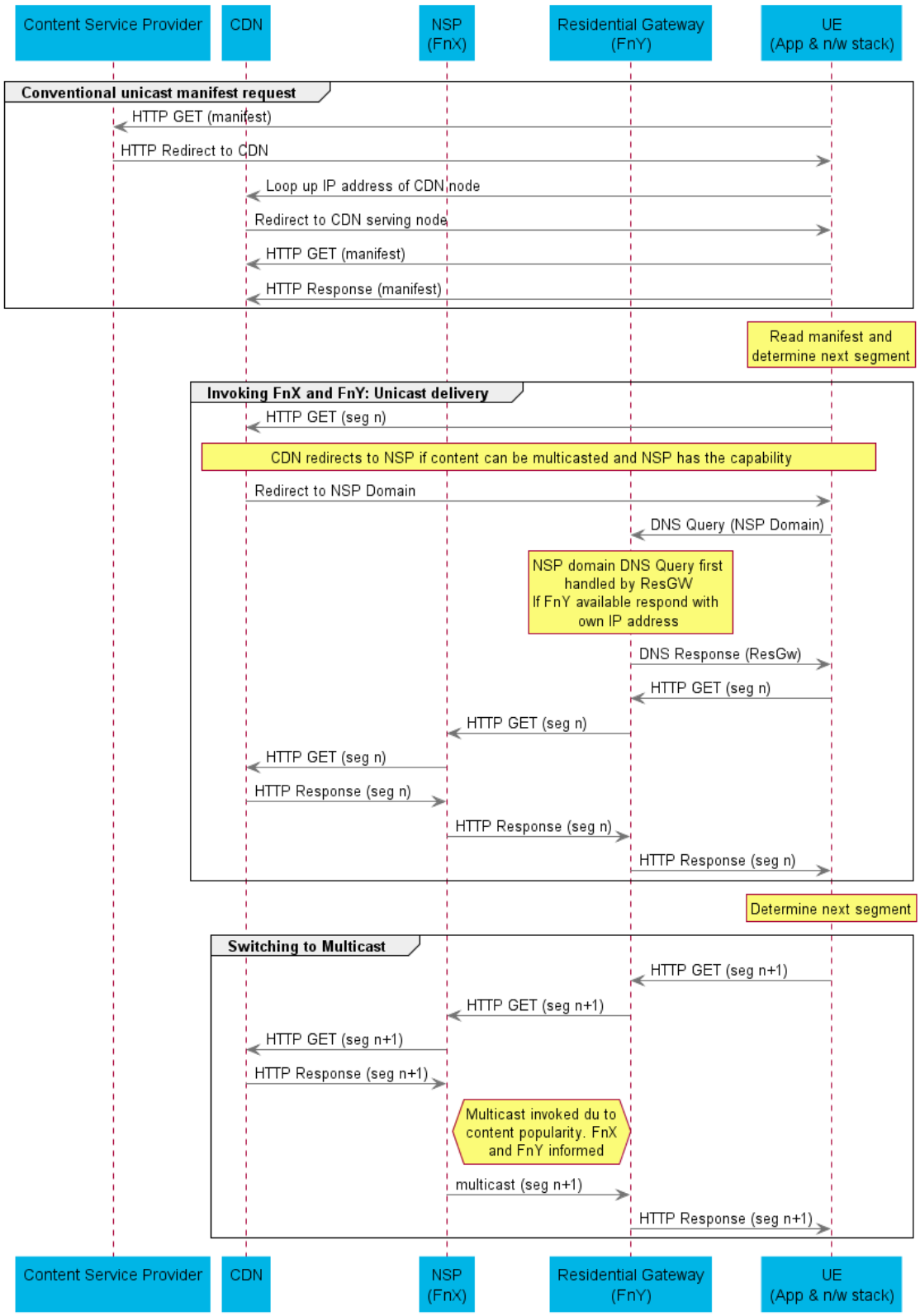

Figure 24 5GXcast framework call flow

B Requirements

Following are the main requirements that the sessions and resource management need to address. These requirements were analysed from the work of WP2 D2.1 “Definition of Use Cases, Requirements and KPIs” [38] and WP5 T5.2 [2].

Table 1 - Relevant WP2 requirements.

Requirement ID	Requirement description WP2	Requirements for 5.3
M&E1_R1	End users have seamless access to audio-visual content both at home and on the move including seamless mobility between access networks, and across different types of devices (stationary, portable/ mobile, mounted in a vehicle). • The user’s device is able to automatically connect to the best available network/s to give the highest QoE to the user, including simultaneous access to multiple networks. • It is desirable to allow using multiple network types together to increase QoS/QoE to any segment of the population that can support this.	* For many content services, it is possible to handle session handover at the application layer. There is often sufficient buffering on the end device to allow time to detect that one connection has been broken and to establish a new one, without interrupting the session from the user’s perspective. * If we wish to use a point to point network in combination with a point multipoint network, or combine different independent networks in a way that is hidden from the application, then a function will need to be introduced which aggregates these access networks into a single logical port that the application can use. On the mobile network, this aggregation function would need to reside on the UE. On the fixed network, it could reside in the Residential Gateway. Multipath TCP would be an example of such an aggregation technology
M&E1_R3	The network resources required to deliver the service to a given audience should grow much less than linearly with audience size, particularly for large audiences of very popular content. • An audience may be concentrated in a limited geographical area or distributed • Minimising the distribution costs for the content service provider	The network resources required to deliver the service to a given audience should grow much less than linearly with audience size 5G intelligence shall take care of that
M&E1_R4	It should be possible for different network types to carry different content elements that constitute the user experience. • It is desirable that networks operated by different operators can carry different content elements that constitute the user experience.	Related to M&E_R1. The content service may be constructed from multiple elementary streams which could be delivered over different, and even independent, networks. It falls to the application to assemble these elementary streams into a coherent user experience. If the application needs to control the selection of network (and implicitly network type), then it will need to override the default behaviour of most operating systems and explicitly select the network device associated with that network type. 5.3 Intelligence shall coordinate such delivery as needed.

M&E1_R5	If multiple networks are used it should be possible to offload the traffic between them. • e.g. fixed, mobile and/or broadcast networks	This puts the decision of which of the delivery mechanisms available to the network operator should be used for a particular stream of traffic. This is central to the Content Delivery Framework. It will allow the network operator to switch between point to point and point to multipoint networks according to the network operator's own policies.
M&E1_R7	It is desirable that the network supports dynamic optimisation of resource allocation based on individual operators' policies, e.g. automatically initiating the switching between unicast, multicast and broadcast. • Means should be given to allow implementation of various deployment and optimisation policies of network resources vs QoE of the population as a whole or segments of.	Same as M&E_R5 Here too the 5.3 Intelligence shall perform such optimisation and QoE management as needed
M&E1_R8	It is desirable that a user can easily discover an existing audio-visual service, including free-to-air.	The content delivery frame work is not directly involved in indicating the availability of services to the end user. However, there will need to be a mechanism to signal to the devices at the edge of the network what networks they will need to connect to in order to receive all the elements of a content service. 5.3 Intelligence shall ensure such a discovery mechanism is included in the architecture and interworks with the intelligence itself in deciding how to serve that discovered content
M&E1_R11	Parallel delivery of a given content at different QoS/QoE levels to different portions of the population in the same geographical area should be supported.	Partitioning the audience for a given piece of content according to QoS/QoE requirements will have an impact on the benefit that would be achieved by using point to multipoint networks. This requirement may encourage the use of scalable coding in order to avoid this. 5.3 Intelligence mechanisms such as mABR shall handle this.
M&E1_R12	Transition between unicast and broadcast and multicast should be allowed during service, without impact on viewers and other users, and within a minimised transition time (in the order of seconds).	This is a core requirement for the Content Delivery Framework. Particularly for live event-driven viewing (sports matches) audiences may grow very large very quickly, then at the end of the event drop to very low levels just as quickly.

M&E1_R13	Both, conventional and object-based delivery should be enabled	HTTP streaming delivers content as data files, to be concatenated by the client player. So, in this sense, the delivery framework would already have to deal with delivery of files. One of the differences is that there might not be an obvious sequence of files, as there is with a video stream
M&E1_R16	The system should be scalable to serve very large concurrent audiences while maintaining the required Quality of Service for each user irrespective of the size of the audience. The number of concurrent users can be very high, i.e. >106 for the most popular content. The system should support variations in the number of concurrent users (e.g. driven by the changing popularity of content).	5.3 Intelligence shall evaluate the WP5 application level mechanisms effectiveness in addressing these scales.
M&E1_R21	The 5G-Xcast solution should support authentication of the content origin	[Not sure what this means. Does 'content origin' mean 'origin server', or is it about authenticating and authorising the content itself – e.g. validating signatures etc. What is the content origin authorised to do, once it is authenticated? Need more info.
M&E1_R24	Latency: • End-to-end latency is allowed to be in the order of 50 ms or even higher o Delay from live should be no worse than other delivery methods • Difference in delay between different streams on the same device shall not be perceivable by the users • Channel change latency should be of the order of 1 second, not excepting additional contributions from latencies that may be outside the scope of the 5G-Xcast system such as communication with a decryption key server	This is a critical issue for the content delivery framework. The content delivery framework is likely to introduce extra request redirections, proxies and format conversion functions. It is essential that these are designed and implemented in such a way that latency is kept within tight bounds

M&E1_R25	Quasi error-free reception: • 1 uncorrected error event per hour	As part of its QoE approach, 5.3 Intelligence shall evaluate “error free reception” under its mechanisms.
M&E1_R26	The solution should not be restrictive to service / application related requirements, such as: • audio or video formats* • codec* • transport containers* • multiple languages* • subtitle formats* • access services* • ad insertion • EPG data • metadata transport • protection of content rights • location based features (e.g. local weather forecast, directions on the map, targeted ads) • combining content from different sources (e.g. multiple media types such as video, audio, text, and data) • time availability of content (e.g. live, time-limited access in a library, unlimited)	The infrastructure should be agnostic to the media formats used. The only relevant implication for the framework should be in terms of ensuring that elements of a service are delivered in a timely manner
M&E1_R27	Content should be delivered to the user device as designed by the content service provider, i.e. without undesired alterations (e.g. interruptions, overlays, distortions, reduced image quality).	The content delivery framework should not modify the content in any way. It should deliver exactly the media objects that we created by the content service provider
M&E1_R28	Geographical availability - the service provider should be able to define in which territory the content / service should be made available, i.e.: • globally • in one or more individual countries • regional • local • one or more specific venues	The two primary mechanisms used to identify location are the IP address of the end device, or residential gateway, and the location services available directly to the end device. The content delivery framework could obscure the IP address of the end device as a result of the use of proxies. However, most content requests will initially be directed to the content service provider for authentication and authorisation before the request is signed and re-directed. We anticipate that part of this authentication and authorisation step could be to validate the geographic region of the content

M&E1_R35	It is desirable that the networks support different business arrangements (e.g. free-to-air, subscription, pay-per-view, usage deducted from a subscriber's data allowance) including both OTT and managed services with guaranteed QoS. • Whilst 5G-Xcast would not develop a billing mechanism, the solution should provide sufficient information to feed a billing system.	Some of this relates to providing network metrics for billing, FUP etc. Business concerns relating to QoS could be more of a challenge. One of the goals of the content delivery framework is to avoid the need to expose direct network control to third parties by treating resource allocation as an internal optimisation problem, so we would consider the direct exposure of fixed bandwidth connections to be out of scope. We will however, consider a means to exert influence over the QoE of a service from the server and application, without need an explicit interface with the network operator

References

[1] N. Nouvel, (Ed.), “Content Delivery Vision,” Deliverable D5.1, 5G-PPP 5G-Xcast project, Nov. 2017.

[2] T. Stevens, (Ed), “Key Technologies for the Content Distribution Network,“ Deliverable D5.2, 5G-PPP 5G-Xcast project, August 2018

[3] Key Network Delivery Metrics, Quality Experience working group Streaming Video Alliance, May 2016.

[4] DASH-IF position, “Proposed QoE Media metrics standardisation for segment media playback”, *Dash Industry Forum*, version 1.0, October 2016.

[5] Bram Tullemans, “Technical Practices for a Multi-CDN Distribution Strategy,” *EBU, IBC,* October 2017.

[6] Benjamin Lembke, Clemens Kunert, “MOS in eMBMS”, *IRT Project Report*, May 2017.

[7] ITU-T P.800.1: “Mean Opinion Score (MOS) terminology,”

[8] ITU-T J.341 Recommendation, “Objective perceptual multimedia video quality measurement of HDTV for digital cable television in the presence of a full reference,” March 2016.

[9] ITU-T J.343 Recommendation, “Hybrid perceptual bitstream models for objective video quality measurements,” November 2014.

[10] M. Torres Vega, E. Giordano, D. C. Mocanu, D. Tjondronegoro, A. Liotta, “Cognitive No-Reference Video Quality Assessment for Mobile Streaming Services”, *7th International Workshop on Quality of Multimedia Experience (QoMex)*, At Costa Navarino (Messinia, Greece), May 2015.

[11] P. Goudarzi, “A no-reference low-complexity QoE measurement algorithm for H.264 video transmission systems,” *Scientia Iranica, Transactions D: Computer Science & Engineering and Electrical Engineering*, January 2013.

[12] D. Ghadiyaram, C. Chen, S. Inguva, A. Kokaram, “A no-reference video quality predictor for compression and scaling artifacts”, *IEEE International Conference on Image Processing*, 2017.

[13] D. Pal, V. Vanijja, “Effect of network QoS on user QoE for a mobile video streaming service using H.265/VP9 codec”, *8th International Conference on Advances in Information Technology* (IAIT), Macau, China, 19-22 December 2016.

[14] T. Tran, (Ed), “Content Distribution PoC Prototype,“ Deliverable D5.4, 5G-PPP 5G-Xcast project, February 2019.

[15] J. Añorga, S. Arrizabalaga, B. Sedano, J. Goya, M. Alonso-Arce, J. Mendizabal, “Analysis of YouTube's traffic adaptation to dynamic environments”, *Journal Multimedia Tools and Applications*, Vol 77, Issue 7, April 2018.

[16] R. Pantos, W. May: “HTTP Live Streaming”. *IETF draft*, -12, Oct 2013.

[17] M. Long, S. Radhakrishnan, S. Karabuk, J. Antonio, "On Zap Time Minimization in IPTV Networks", *International Conference on Computing, Networking and Communications, Internet Services and Applications Symposium*, Maui, HI, USA, January 2012.

[18] Cable Television Laboratories, Video IP Multicast, IP Multicast Adaptive Bit Rate Architecture Technical Report, OC-TR-IP-MULTI-ARCH-C01-161026.

[19] M. Luby, “Wave and Equation Based Rate Control (WEBRC) Building Block”, *RFC 3738*.

[20] DVB document: A176, “Adaptive Media Streaming over IP Multicast”, 8th March 2018.

[21] “Information technology -- Coded representation of immersive media -- Part 2: Omnidirectional media format”, ISO.IEC 23090-2

[22] IETF RFCs 7230-7237, “Hypertext Transfer Protocol -- HTTP/1.1”.

[23] “IETF working group on QUIC”

[24] RFC 7540, Hypertext Transfer Protocol Version 2 (HTTP/2)
[25] ETSI GS MEC 012 V1.1.1 (2017-07), “Mobile Edge Computing (MEC), Radio Network Information API”.
[26] I. Rhee, et al., “CUBIC for Fast Long-Distance Networks”, *IETF RFC 8312*, February 2016.
[27] Scholz D., Jaeger B., Schwaighofer L., Raumer D., Geyer F. and Carle G., “Towards a Deeper Understanding of TCP BBR Congestion Control,” IFIP 2018.
[28] P. Mulroy, et al., “The use of MulTCP for the delivery of equitable quality video”, *17th International Packet Video Workshop*, May 2009.
[29] Y. Humeida, et al., “TCP Congestion Response for Low Latency HTTP Live Streaming”: *19th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2018.
[30] IETF RFC 3234, “Middleboxes: Taxonomy and Issues”.
[31] A. Ali, *et al*., “MP-ALM: Exploring Reliable Multipath Multicast Streaming with Multipath TCP,” IEEE *Conference on Local Computer Networks (LCN), Dubai, United Arab Emirates,* 2016.
[32] Q.De Coninck, et al., “Multipath Extension ofr QUIC,” *QUIC Working Group*.
[33] B. Liu, et at., “An OpenFlow-based performance-oriented multipath forwarding scheme in datacenters,” *Frontiers of Information and Technology & Electronic Engineering,* 2016.
[34] Y. Xu, et al., “A handover scheme for video streaming over heterogeneous multicast/broadcasting and unicast networks,” *International Journal of Innocative Computeing, Information and Control*, v.13, n1, February 2017.
[35] M. Tham, et al., “Seamless handover between unicast and multicast multimedia streams,” *Journal of Zhejiang University SCIENCEC*, October 2014.
[36] “LTE Handover Latency Calculation (Access Node),” *LTE, Tech Fundas*, January 2018.
[37] J.W. Byers, G. Horn, M. Luby, M. Mitzenmacher, W. Shaver, "FLID-DL: Congestion Control for Layered Multicast", *IEEE Journal on Selected Areas in Communications*, vol.20, n. 8, October 2002.
[38] D. Ratkaj and A. Murphy, Eds., “Definition of Use Cases, Requirements and KPIs,” Deliverable D2.1, 5G-PPP 5G-Xcast project, Oct. 2017