ETSI TR 126 957 V15.0.0 (2018-07)

TECHNICAL REPORT

Universal Mobile Telecommunications System (UMTS);
LTE;
Study on Server And Network-assisted
Dynamic Adaptive Streaming over HTTP (DASH) (SAND)
for 3GPP multimedia services
(3GPP TR 26.957 version 15.0.0 Release 15)

Reference

RTR/TSGS-0426957vf00

Keywords

LTE,UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Copyright Notification

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Report (TR) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Contents

Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document evaluates the MPEG DASH SAND (ISO/IEC 23009-5) specification to:

- Ensure that 3GPP and MPEG remain aligned on their respective DASH specifications.
- Identify enhancements offered by MPEG DASH SAND (ISO/IEC 23009-5) in the 3GPP environment, and recommend necessary modifications to the 3GPP specifications including DASH to enable these enhancements.

In particular, the following elements of functionality may be of particular 3GPP relevance and are studied:

- Streaming enhancements via intelligent caching, processing and delivery optimizations on the server and/or network side, based on feedback from clients on anticipated DASH Segments, accepted alternative DASH Representations and Adaptation Sets, and requested bandwidth.
- Improved adaptation on the client side, based on network/server-side information such as cached Segments, alternative Segment availability, and network throughput/QoS.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] ISO/IEC 23009-5:2016: "Information Technology — Dynamic adaptive streaming over HTTP (DASH) — Part 5: Server and network assisted DASH (SAND)".

[3] ISO/IEC 23009-1:2014: " Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats".

[4] 3GPP TS 26.247: "3GPP TS 26.247: "Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)".

[5] 3GPP TS 26.233: "Transparent end-to-end packet switched streaming service (PSS); General description".

[6] 3GPP TR 26.938: "Packet-switched Streaming Service (PSS); Improved support for dynamic adaptive streaming over HTTP in 3GPP".

[7] 3GPP TS 23.203: "Policy and Charging Control Architecture".

[8] 3GPP TS 23.207: "End-to-End Quality of Service (QoS) Concept and Architecture".

[9] 3GPP TS 29.213: "Policy and charging control signalling flows and Quality of Service (QoS) parameter mapping".

[10] 3GPP TS 29.214: "Policy and charging control over Rx reference point".

[11] M. Andrews, Q. Lijun, and A. Stolyar, "Optimal utility based multi-user throughput allocation subject to throughput constraints," in *Proc. IEEE INFOCOM 2005*, vol. 4, pp. 2415-2424 vol. 4, 2005.

[12] V. Ramamurthi and O. Oyman, "Video-QoE Aware Radio Resource Allocation for HTTP Adaptive Streaming ", *in Proc. IEEE International Conference on Communications (ICC), 2014*, Sydney, Australia, June 2014.

[13] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *Proc. IEEE 51st Vehicular Technology Conference, Tokyo*, vol. 3, pp. 1854-1858, 2000.

[14] Z. Xiaoqing, T. Schierl, T. Wiegand, and B. Girod, "Distributed Media-Aware Rate Allocation for Video Multicast Over Wireless Networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, pp. 1181-1192, 2011.

[15] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, and J. G. Andrews, "Video capacity and QoE enhancements over LTE," in *Proc. IEEE International Conference on Communications (ICC), 2012* pp. 7071-7076, 2012.

[16] W3C Recommendation, Cross-Origin Resource Sharing, http://www.w3.org/TR/cors/.

[17] IETF RFC 6455: "The WebSocket Protocol".

[18] OMA-ERELD-DM-V1_2-20070209-A: "Enabler Release Definition for OMA Device Management, Approved Version 1.2".

[19] 3GPP TR 36.933: "Study on Context Aware Service Delivery in RAN for LTE".

3 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

CDN	Content Delivery Network
CORS	Cross-Origin Resource Sharing
DANE	DASH-Aware Network Element
DASH	Dynamic Adaptive Streaming over HTTP
MPD	Media Presentation Description
OTT	Over-The-Top
PED	Parameters Enhancing Delivery
PER	Parameters Enhancing Reception
RNE	Regular Network Element
SAND	Server and Network Assisted DASH
XML	Extensible Markup Language

4 Overview of MPEG Server And Network Assisted DASH (SAND) in ISO/IEC 23009-5

MPEG's Server and Network Assisted DASH (SAND) technology, i.e., specified in ISO/IEC 23009-5 [2] offers standardized interfaces for service providers and operators to enhance streaming experience. In order to enhance the delivery of DASH content, SAND introduces messages between DASH clients and network elements or between various network elements for the purpose to improve efficiency of streaming sessions by providing information about real-time operational characteristics of networks, servers, proxies, caches, CDNs as well as DASH client's performance and status. SAND addresses the following:

- Unidirectional/bidirectional, point-to-point/multipoint communication with and without session (management) between servers/CDNs and DASH clients,

- Mechanisms for providing content-awareness and service-awareness towards the underlying protocol stack including server and/or network assistance,

- Various impacts on elements of the existing Internet infrastructure such as servers, proxies, caches and CDNs,

- QoS and QoE support for DASH-based services,
- Scalability in general and specifically for logging interfaces, and
- Analytics and monitoring of DASH-based services.

The SAND reference architecture is based on four broad categories of elements:

i) DASH streaming clients.

ii) Regular network elements (RNE), which are DASH content unaware and treat DASH-related video delivery objects as any other object, but are present on the path between origin server and DASH clients, e.g. transparent caches.

iii) DASH-aware network elements (DANE), which have at least minimum intelligence about DASH; for instance they may be aware that the delivered objects are DASH-formatted objects such as the MPD or DASH segments, and may prioritize, parse or even modify such objects, and

iv) Metrics server, which are DASH aware and are in charge of gathering metrics from DASH clients.

Based on these elements, the SAND reference architecture is defined as shown in Figure 4.2. Within this architecture, the following four categories of messages, called SAND messages as shown in Figure 4.1, are exchanged:

- Parameters Enhancing Delivery (PED) messages that are exchanged between DANEs.
- Parameters Enhancing Reception (PER) messages that are sent from DANEs to DASH clients.
- Status messages that are sent from DASH clients to DANEs.
- Metrics messages that are sent from DASH clients to Metrics servers.

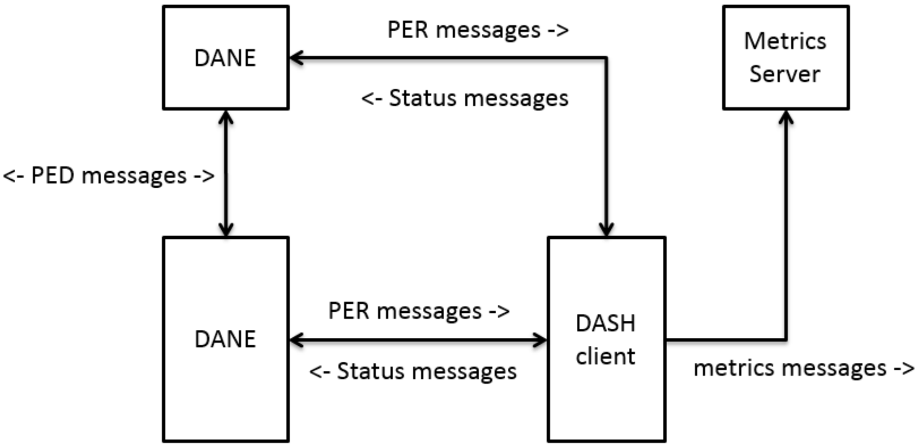


Figure 4.1: SAND messages (taken from ISO/IEC 23009-5 [2])

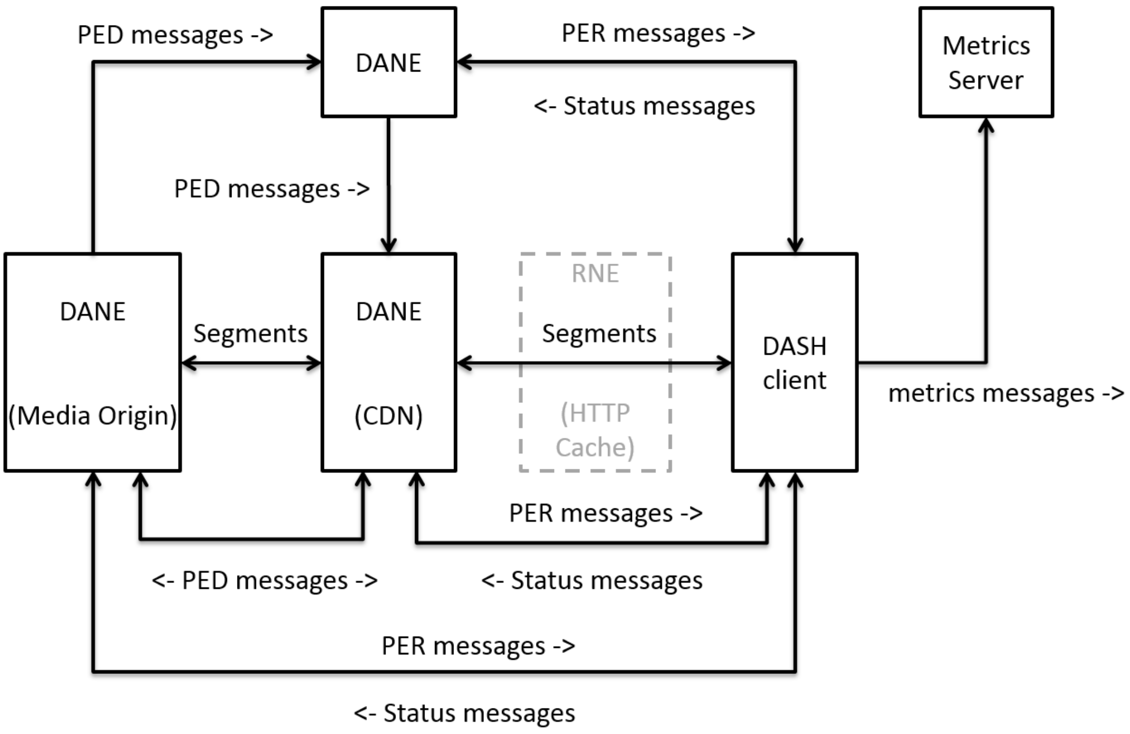


Figure 4.2: SAND reference architecture (taken from ISO/IEC 23009-5 [2])

Most of the SAND messages are delivered in Extensible Markup Language (XML) format using HTTP protocols with the detailed syntax of each message defined in the SAND specification [2]. In case of small metric messages, status messages or PED messages, the DASH client may attach the SAND message in a non-XML format to the uplink (HTTP GET or POST) message.

Using the metrics and status messages, the DASH clients can inform the network (i.e., DANE) about requested bandwidth / quality, anticipated DASH segments, acceptable alternative content, etc. This leads to intelligent caching and real-time media processing at the server or proxy. As defined in the SAND specification [2], Metrics and Status Messages are comprised of the following:

- *QoE metrics from DASH Part 1*, i.e., ISO IEC 23009-1 [3], including average throughput, buffer level, initial playout delay, HTTP request/response transactions, representation switch events, and playlist, as also described in 3GPP TS 26.247 [4]. QoE metrics are beneficial for detecting and debugging failures, managing streaming performance, and allowing for QoE-aware network adaptation and service provisioning useful for the network operator and content/service provider.

- *SharedResourceAllocation*, allows a DASH client to provide information on a set of operating points (such as desired bandwidth and quality) to one or several DANE(s) with an intent to share network resources.

- *AnticipatedRequests*, allows a DASH client to announce to a DANE which specific set of segments it is interested in. The intent is to signal the set of segments in representations that the DASH client is likely to select and request soon.

- *AcceptedAlternatives*, allows DASH clients to inform DANEs on the media delivery path (typically caching DANEs) when they request a given DASH segment that they are willing to accept other DASH segments alternatives.

- *AbsoluteDeadline*, allows DASH clients indicating the DANE the absolute deadline in wall-clock time by when the requested DASH Segment needs to be completely received. As such, further action can be taken by the network, e.g., the DANE can pre-fetch content to ensure the timely delivery to the client.

- *MaxRTT*, allows DASH clients indicating the DANE the maximum round trip time of the request from the time when the request was issued until the request needs to be completely available at the DASH client.

- *NextAlternatives,* allows DASH clients to inform a DANE about which alternatives they are willing to accept for the request of the next segment.

- *ClientCapabilities*, allows DASH clients to share their SAND capabilities, i.e., the set of SAND messages they support, with the DANE.

NOTE: See [2] for the detailed semantics of the SAND messages.

Using the PER Messages, the DANE can inform the client about cached segments, alternative segment availability, timing information for delivery, network throughput/QoS, etc., which leads to intelligent DASH client adaptation behavior. As defined in the SAND specification [2], the PER Messages are comprised of the following:

- *ResourceStatus*, allows for a DANE to inform a DASH client – typically in advance – about knowledge of segment availability including the caching status of the segment(s) in the DANE. The DASH client adaptation can take advantage of this information and potentially prefer accessing the content cached at the edge due to faster download times.

- *DaneResourceStatus*, allows DANEs to signal the available and possibly anticipated to be available data structures to the DASH client and also signal which data structures are unavailable. This method is complementary to the ResourceStatus message mentioned above as it allows to express the available segments at the time of the status message.

- *SharedResourceAssignment*, allows the DANE to send to DASH clients competing for bandwidth over the same network information about how much bandwidth they should use in order to stay in a fair sharing of the total bandwidth. This message is usually send to DASH clients as a response to a SharedResourceAllocation message and is usually sent by a DANE who acts as a resource allocation entity.

- *MPDValidityEndTime*, provides the ability to signal to the client that a given MPD, whose @type is set to 'dynamic' and @minimumUpdatePeriod is present, can only be used up to at a certain wall-clock time.

- *Throughput*, allows a DASH client to have – in advance – knowledge of the throughput characteristics and the guarantees along with this from the DANE to the DASH client.

- *AvailabilityTimeOffset*, allows a DASH client to have – in advance – knowledge of the availability time offset from the DANE to the DASH client. The status may be different for different baseURLs or different Representation IDs used, allowing to signal availability time offset dependent on the network delivering it.

- *QoSInformation*, signals to a DASH client about the available QoS information, including parameters such as guaranteed bitrate (GBR), maximum bitrate (MBR), delay and packet loss rate. A DASH client can take the available network QoS information into consideration when requesting media segments such that the consumed content bandwidth remains within the limits established by the signaled QoS information.

- *DeliveredAlternatives* serves as a response to an AcceptedAlternatives message sent by a DASH client, where a DANE may deliver an alternative segment rather than the requested segment. If so, the DANE also sends a DeliveredAlternatives message to the DASH client to inform him that the response contains a segment alternative and not the requested segment.

- *DANECapabilities*, allows DANEs to share their SAND capabilities, i.e., the set of SAND messages they support, with the DASH clients.

The complete set of SAND messages is shown in Table 4.1.

Table 4.1: messageType values for SAND messages (taken from ISO/IEC 23009-5 [2])

messageType	Message description
0	Reserved
1	TCPConnections, see clause 6.3.2 of [2] for the detailed semantics
2	HTTPRequestResponseTransactions, see clause 6.3.3 of [2] for the detailed semantics
3	RepresentationSwitchEvents, see clause 6.3.4 of [2] for the detailed semantics
4	BufferLevel, see clause 6.3.5 of [2] for the detailed semantics
5	PlayList, see clause 6.3.6 of [2] for the detailed semantics
6	AnticipatedRequests, see clause 6.4.1 of [2] for the detailed semantics
7	SharedResourceAllocation, see clause 6.4.2 of [2] for the detailed semantics
8	AcceptedAlternatives, see clause 6.4.3 of [2] for the detailed semantics
9	AbsoluteDeadline, see clause 6.4.4 of [2] for the detailed semantics
10	MaxRTT, see clause 6.4.5 of [2] for the detailed semantics
11	NextAlternatives, see clause 6.4.6 of [2] for the detailed semantics
12	ClientCapabilities, see clause 6.4.7 of [2] for the detailed semantics
13	ResourceStatus, see clause 6.5.1 of [2] for the detailed semantics
14	DaneResourceStatus, see clause 6.5.2 of [2] for the detailed semantics
15	SharedResourceAssignment, see clause 6.5.3 of [2] for the detailed semantics
16	MPDValidityEndTime, see clause 6.5.4 of [2] for the detailed semantics
17	Throughput, see clause 6.5.5 of [2] for the detailed semantics
18	AvailabilityTimeOffset, see clause 6.5.6 of [2] for the detailed semantics
19	QoSInformation, see clause 6.5.7 of [2] for the detailed semantics
20	DeliveredAlternative, see clause 6.5.8 of [2] for the detailed semantics
21	DaneCapabilities, see clause 6.5.9 of [2] for the detailed semantics
22..127	reserved for future ISO use
128..255	reserved for private use

SAND as defined in ISO/IEC 23009-5 [2] mandates HTTP as the minimum transport protocol to be supported by SAND-enabled elements. It does not preclude that other additional transport protocols could also be implemented.

The use of HTTP as a minimum transport protocol to implement is defined in [2] for:

a) Metrics messages (from DASH client to DANE)

b) Status messages (from DASH client to DANE)

c) PER messages (from DANE to DASH client)

Depending on the nature of SAND messages, the use of HTTP protocol by SAND network elements varies. Table 4.2 summarizes which HTTP usages is mandatory in [2] for a SAND element (in bold in the table) or may be optional depending on the nature of the SAND message.

Table 4.2: Mandatory usages of HTTP for carrying SAND messages (taken from [2])

Metrics messages	**HTTP POST** HTTP headers may be used for small metrics messages.
Status messages	**HTTP headers**
PER messages	**HTTP GET**

Further details on the transport protocol to carry SAND messages can be found in clause 8 of [2]. Moreover, the signalling of SAND communication channel is described in clause 9 of [2], optional transport protocols to carry SAND messages (e.g., WebSocket) are provided in clause 10 of [2] and metrics reporting via SAND protocols is specified in clause 11 of [2].

5 Architectural considerations for SAND

5.1 SAND in PSS architecture

In the PSS architecture for 3GP-DASH in TS 26.233 [5], SAND functionality can be supported by hosting the DANE capabilities described in [2] in the PSS server, and by hosting the SAND-capable DASH client capabilities described in

[2] in the PSS client. Moreover, the PSS server may also host the metrics reporting server described in [2]. This is illustrated in Figure 5.1. As such the relevant SAND messages, including PER and status messages, as well as the QoE metrics, can be exchanged between the PSS server and PSS client.

Despite being part of the PSS Server logically, the SAND Functionality of the PSS Server could be co-located with other functions that are separate from the MPD Delivery Function and Segment Delivery Function. One particular realization of such a split is when the SAND Functionality is located near to the network edge, in order to provide segment delivery assistance. Here the SAND messages and Metrics Reporting are out-of-band of the media flow, i.e. carried in signalling flows that are separate from the MPD and Media Segment flows.

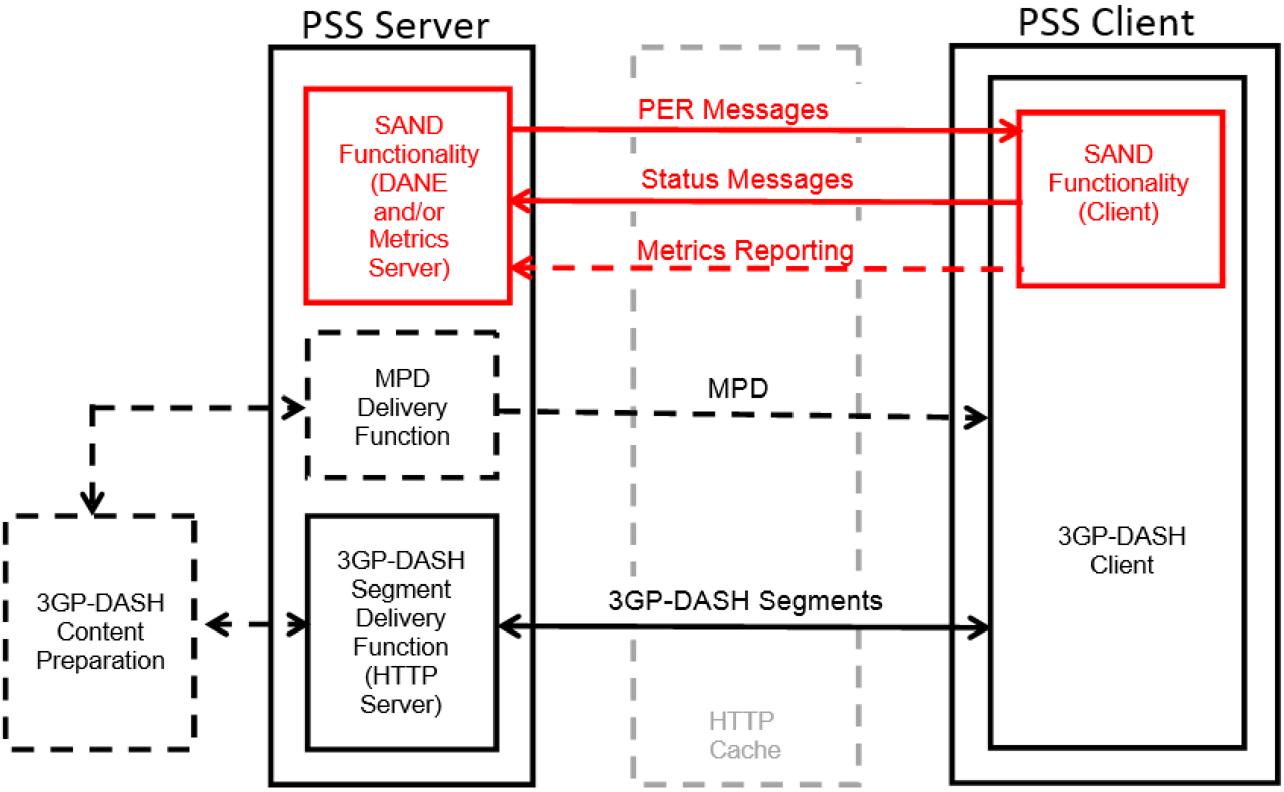

PSS Server
PSS Client
SAND Functionality (DANE and/or Metrics Server)
PER Messages
Status Messages
Metrics Reporting
SAND Functionality (Client)
MPD Delivery Function
MPD
3GP-DASH Client
3GP-DASH Content Preparation
3GP-DASH Segment Delivery Function (HTTP Server)
3GP-DASH Segments
HTTP Cache

Figure 5.1: System Architecture for SAND over PSS

5.2 SAND architecture for OTT streaming services

The described architecture model in Figure 5.1 assumes that the PSS server hosting SAND functionality (DANE and/or metrics server) delivers the streaming content too. This assumption is valid when the streaming service is hosted by the operator. This assumption cannot apply for managed streaming services that deliver OTT streaming content - it is of great importance for the operator to provide the desired SAND-based network assistance and streaming enhancements without the access of streaming content itself. The modified SAND system architecture applicable for OTT streaming services in depicted in Figure 5.2.

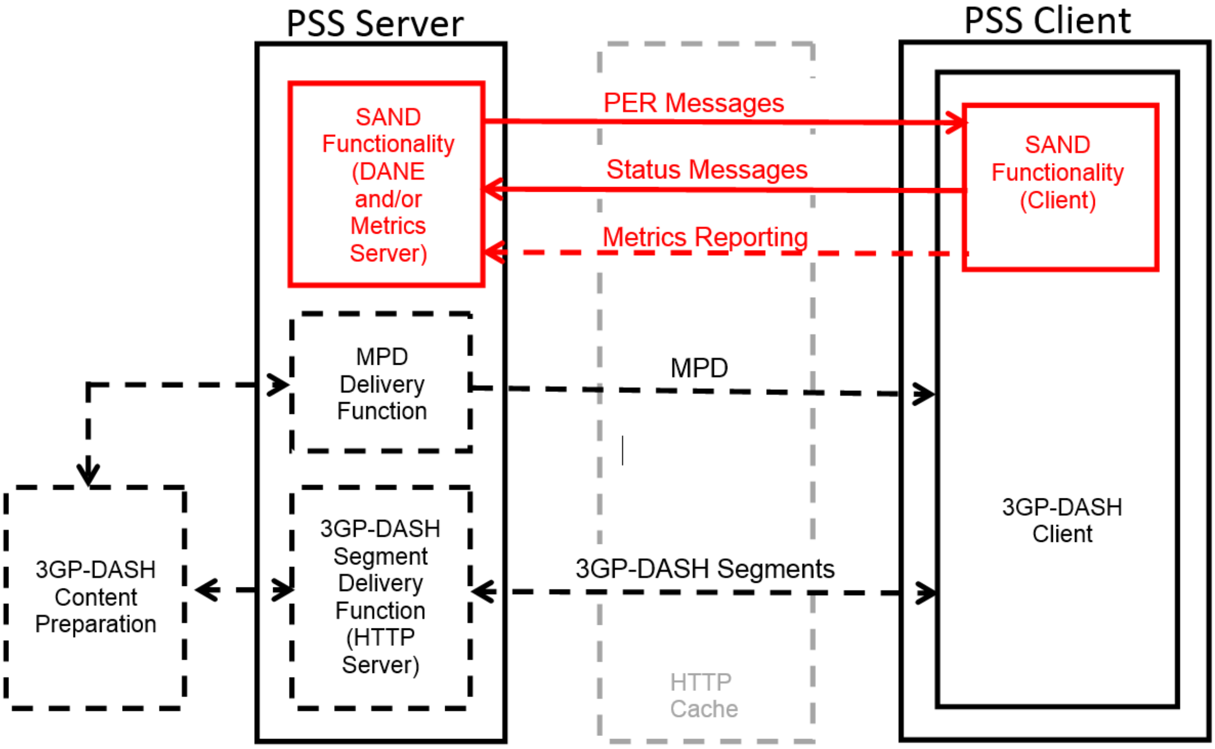

PSS Server
SAND Functionality (DANE and/or Metrics Server)
MPD Delivery Function
3GP-DASH Segment Delivery Function (HTTP Server)
3GP-DASH Content Preparation
PER Messages
Status Messages
Metrics Reporting
MPD
3GP-DASH Segments
HTTP Cache
PSS Client
SAND Functionality (Client)
3GP-DASH Client

Figure 5.2: System Architecture for SAND for OTT Streaming Services

In the SAND context, a managed streaming service refers to a service for which the MNO and the OTT service provider have some agreement in order to exchange information relevant for enabling SAND functionality. The DASH client is managed by the OTT servicer provider, while the operator hosts the entities required for realizing SAND functionality. In particular, the DANE capabilities described in [2] of SAND are hosted by the operator in the PSS server (including the DANE and/or metrics server), and SAND-capable DASH client capabilities described in [2] are hosted in the PSS client. Furthermore, the PSS client retrieves streaming content from the OTT streaming server - thus, the 3GP-DASH segment delivery function in the PSS server is shown in dashed lines and may not be present. The OTT streaming content is assumed to conform to the 3GPP file format and the content delivery over the OTT streaming service is assumed to comply with the requirements of 3GP-DASH. Figure 5.3 further depicts this architecture clearly distinguishing the entities in the third party domain and operator domain.

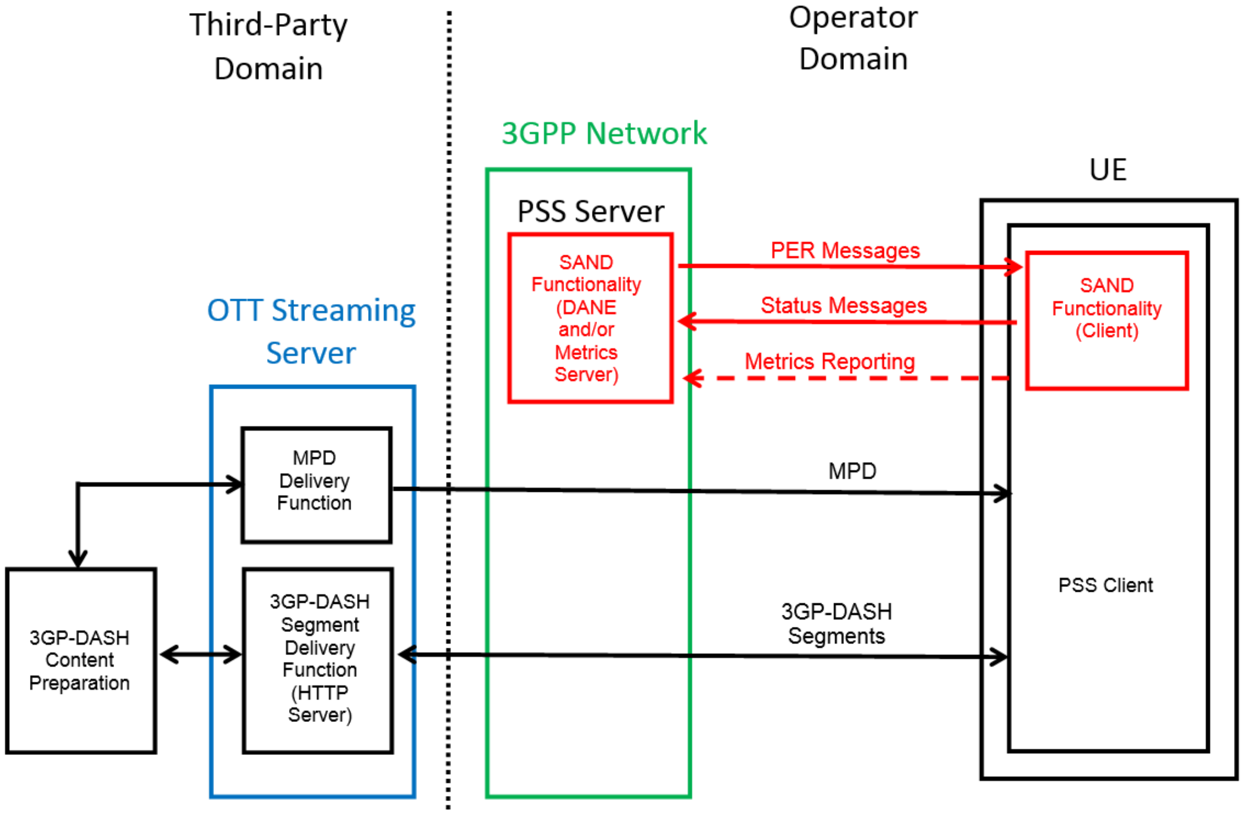


Figure 5.3: SAND architecture for OTT Streaming Services in relation to third-party domain and operator domain

Based on this architecture, the business-level agreement between the operator and OTT service provider involves exchange of the following information:

- streaming service provider allows metadata (e.g., MPD) of the streaming content can be collected by the operator, i.e., by the PSS client or PSS server;
- service provider provides certain SAND status messages and QoE metrics reporting of the streaming service to the MNO in real-time;
- MNO provides PER messages to the DASH client operated by the OTT service provider.

As such, the PSS client may interact with the DASH application running in the UE to collect relevant SAND status messages and QoE metrics, and convey PER messages.

Access to the SAND client functionality in the UE could be realized flexibly, i.e. the implementation of the UE-side SAND client functionality could be realized by any of the following approaches:

- The SAND client functionality could be embedded in the media player element or application.
- The software platform on the UE could include the SAND client functionality in a platform library amd offer an API that is used by third-party applications.

5.3 Alternative SAND architecture for OTT streaming services

This clause describes another SAND architecture for OTT streaming services depicted in Figure 5.4. The difference from the architecture in clause 5.2 is that the OTT streaming client in the UE is not part of the PSS client and does not have to comply with 3GPP DASH and 3GPP file format. Only SAND-capable DASH client capabilities described in [2] are hosted in the PSS client, which still interacts with the DANE hosted in the PSS server to exchange SAND messages. Furthermore, the PSS client communicates SAND messages to/from the OTT streaming client in the UE, i.e., it can collect relevant SAND status messages and QoE metrics, and convey PER messages. It is assumed that the OTT streaming client is able to generate and process SAND messages as part of its operation.

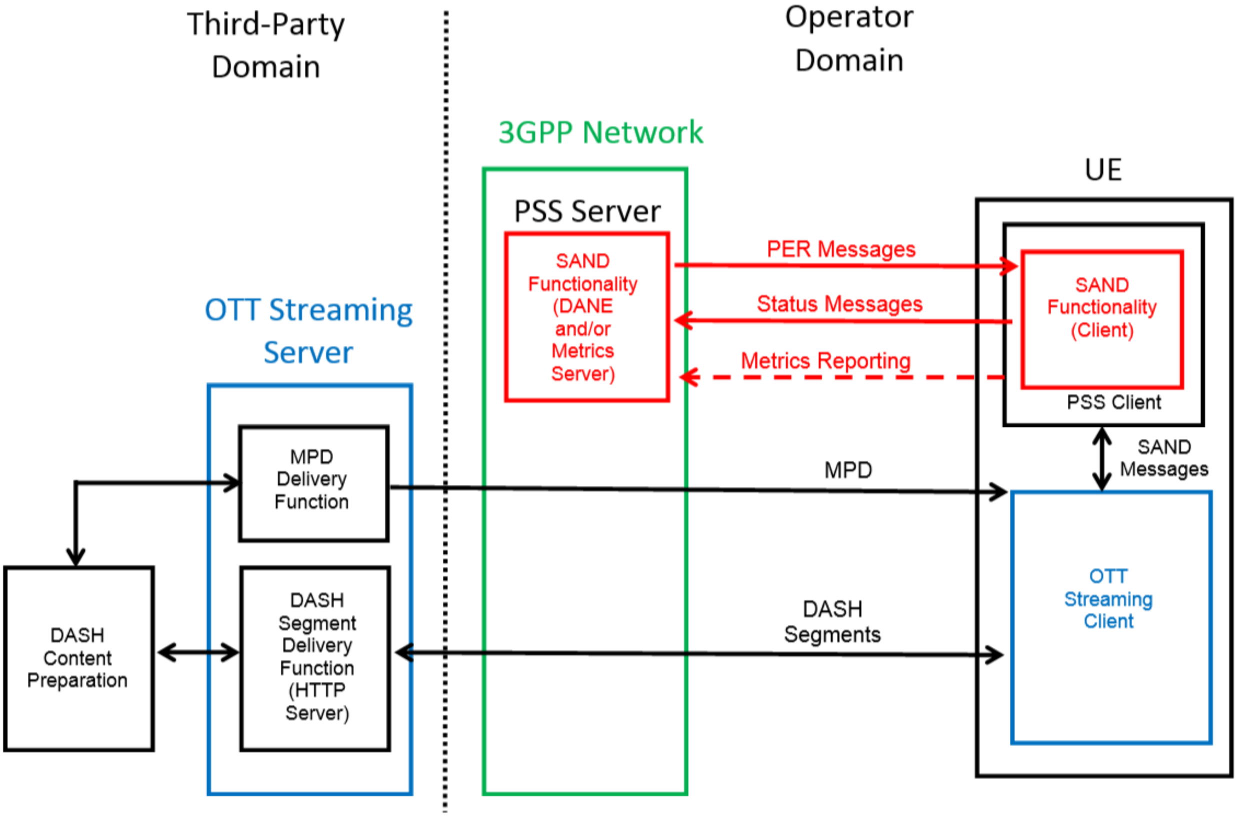


Figure 5.4: Alternative SAND architecture for OTT Streaming Services

5.4 SAND architecture for generic OTT streaming services

This clause describes another SAND architecture for generic OTT streaming services depicted in Figure 5.5. The difference from the architecture in clause 5.3 is that the OTT streaming client and OTT streaming server do not have to comply with the DASH format, and media delivery can be based on other adaptive streaming formats.

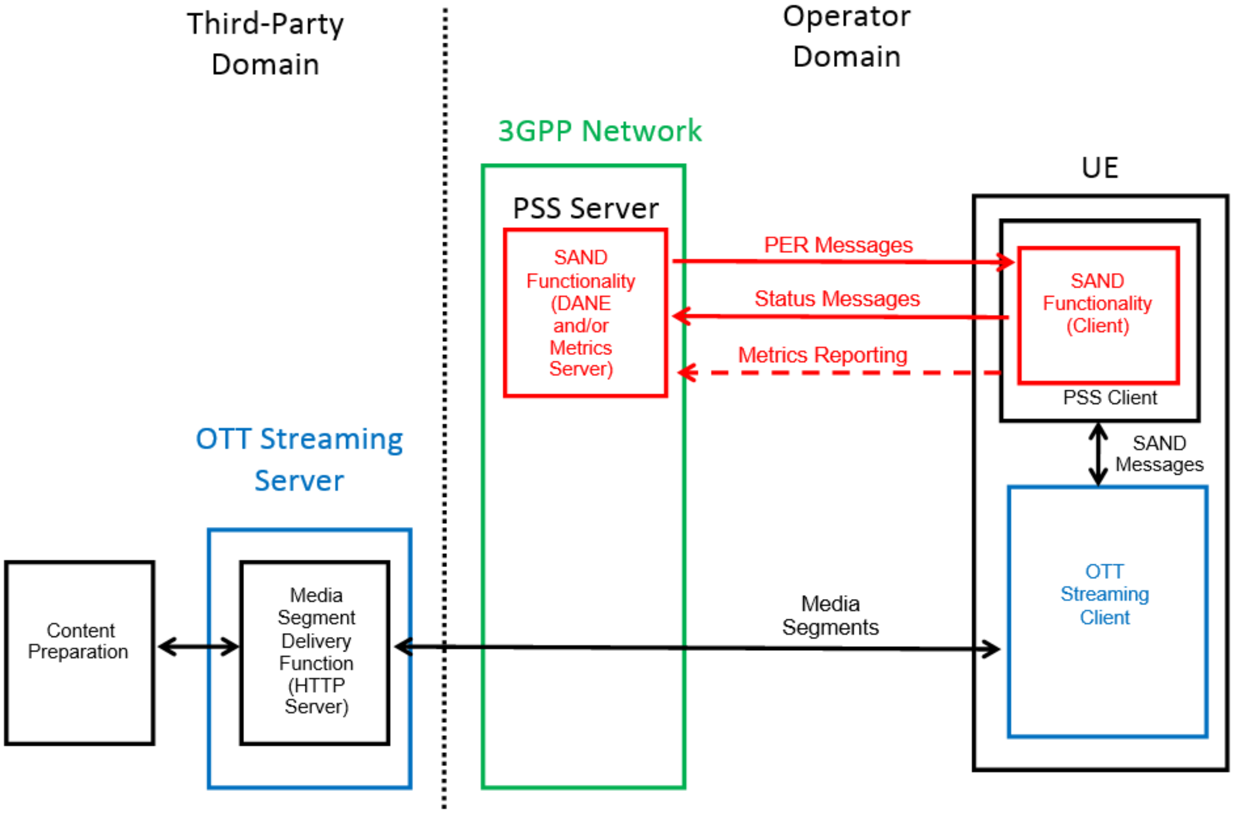


Figure 5.5: Alternative SAND architecture for Generic OTT Streaming Services

In principle, SAND functionality (i.e., SAND interfaces and associated messages) can be supported using any adaptive streaming technology, including e.g. 3GP-DASH, HLS, HDS etc. In particular, the applicability of SAND for the use cases described in clauses 6.2 and 6.4, namely Consistent QoE/QoS and Network Assistance, does not depend on the 3GP-DASH container format and hence may be applicable for a broader set of adaptive streaming formats.

6 SAND use cases and relevance to 3GPP environment

6.1 Use Case # 1: Content-Provider Optimized Zero-Rating

6.1.1 Use case description

A mobile network operator provides the ability that video offerings, under which customers on qualifying rate plans choose to receive standard quality video (typically 480p or better) receive zero-rated video (with associated audio) streams from qualifying content providers. Zero-rated means that the traffic is not counted against data caps of the user. The content provider optimizes the delivery in a way that the shaping of the video traffic is supported. When the user has accepted to participate in the rate plan, the bandwidth for the service is limited in typical cases by the MNO to a specific threshold, e.g. to 1.5 Mbit/s. Other optimizations may also be performed.

In order to support the use case, the following further conditions are taken into account:

1) The media streams are delivered over the MNOs network in a way that allows the MNO to identify the video traffic under the rate plan, for example by media types or specific tokens on HTTP or IP level.

2) The use case should be supported for both non-encrypted and encrypted traffic.

3) The category of content (i.e. video) that is eligible under the program is delivered by the content provider to MNO in such a way that is distinguishable from other categories that are not qualified as eligible under the rate plan, for example by media types or specific tokens on HTTP or IP level.

4) The content provider provides video over MNO's network using adaptive bitrate technology where delivery bitrate is expected to adapt based on the capabilities of the data connection or as otherwise indicated by the MNO network. The network typically limits the bandwidth available to detectable videos (i.e., video traffic that satisfies conditions 1 and 3) to a level to be set by the MNO (e.g., at 1.5 Mbps).

5) The content provider is aware of adjustments applied by the MNO. The DASH client may also be aware of these adjustments. Based on this the content provider makes technical adjustments to support a high quality end-user experience and to improve the utilization of network bandwidth (e.g., bit rate set at approximately the bandwidth level set by the MNO of the per connection averaged over one minute of video).

6.1.2 Recommended requirements and working assumptions

- It is recommended that the traffic includes information if it is a suitable traffic to be throttled to a certain bitrate.

- It is recommended that the operation of throttling does not depend on any messages exchange.

- It is recommended that the throttling bitrate is signalled to the DASH client in order to make use of the information in the rate adaptation. If reported, it is recommended that the throttling bitrate is exactly defined, for example by a leaky buffer model.

6.1.3 Gap analysis w.r.t. existing 3GPP technologies

3GPP services do not support signalling the throttling bitrate to the DASH client. This may help the DASH client to adapt the rate adaptation to the signalled maximum bitrate.

6.1.4 Potential solutions including relevant SAND functionality

The use case itself is already supported within existing 3GPP services. However, the operation may benefit from providing this information to the DASH client.

Two cases need to be differentiated, one for which the traffic is encrypted and non-accessible to the MNO and one for which the traffic is unencrypted.

- If not encrypted, then the MNO is able to operate a DANE, detect DASH traffic and communicate the information on bandwidth throttling with SAND functionalities to the client.

- If the traffic is encrypted, then detection and communication may be done differently. Either the operator needs to use IP-based signaling (i.e. outside the encrypted stream) or the content provider needs to add the information as SAND messages.

6.2 Use Case # 2: Consistent QoE/QoS for DASH Users

6.2.1 Use case description

A network operator deploying DASH services or a network operator supporting the delivery of DASH services of an OTT service provider has the ambition to provide consistent quality for users in its network. For this purpose, the network operator wants to provide sufficient QoE to all users that have been granted acquisition to the network and the service. It may also have the ambition to provide certain premium users to maintain a certain service quality when the user plane is congested. The operator may want to influence its QoS control and resource allocation to actively support such use cases, e.g., communicate with the UEs to decrease the bitrate for the video to a certain value that would allow the cell to accommodate the load. Here are some more specific examples around this use case:

1) Jari (regular user) and Jarison (premium user) enter a congested radio area. The mobile operator wants to restrict the required bitrate, but ensure that a basic video quality is maintained for its regular users (Jari) and some higher quality for premium users (Jarison). For this purpose, the operator assigns certain bitrate quality levels to different users on their HTTP connections carrying DASH-content.

2) Jari wishes to watch high-definition video content over his tablet, while Jarison would like to watch standard-definition video content over this smartphone. The operator is able to influence its QoS control and resource allocation to ensure that both Jari and Jarison are simultaneously able to watch their desired content with consistent quality of experience, e.g., with sufficient video quality and without any rebuffering or playback interruption.

6.2.2 Recommended requirements and working assumptions

- The network can signal assigned bandwidth allocation information based on MPD-level bandwidth attributes (such as `MPD@bandwidth`, `MPD@minBandwidth`, `MPD@maxBandwidth,` as described in TS 26.247 [4]) to DASH clients toward assisting the client adaptationwith the assumption that network has access to the MPD.

- The network can signal QoS information based on parameters described in Annex I of TS 26.247 [4] and clause 6.8.3 of TR 26.938 [6] to DASH clients toward improving the operation of the client adaptation logic.

- The operator can influence the QoS and resource allocation among the streaming users based on the consideration of the real-time QoE metrics reported by DASH clients and/or quality estimation performed by the QoE reporting server.

6.2.3 Gap analysis w.r.t. existing 3GPP technologies

Annex I in TS 26.247 [4] describes QoS handling in 3GPP DASH and derivation of QoS mapping guidelines from the DASH MPD to be used by the application function (AF) of 3GPP Policy Charging and Control (PCC) architecture [7]-[10]. Clause 6.8.3 and Annex B of TR 26.938 [6] further describe the utilization of QoS information as part of the DASH client adaptation logic. In the meantime, these solutions pertain to managed streaming services and do not apply for OTT streaming services where there is typically no QoS policy enforced by the operator and corresponding DASH-based video streams are treated as best-effort traffic.

6.2.4 Potential solutions including relevant SAND functionality

See clause 8 of the present document on streaming enhancements from QoE-aware resource allocation. In particular, relevant streaming enhancements are described in clauses 8.3 and 8.4, and corresponding SAND functionality is described in clause 8.5.

An example workflow realizing the consistent QoE/QoS use case is depicted in Figure 6.1, where the use of the WebSocket protocol [17] is considered and the DANE functionality is hosted at the PSS server and SAND-capable DASH client capabilities are hosted in the PSS client (consistent with the SAND architectural considerations in clause 5).

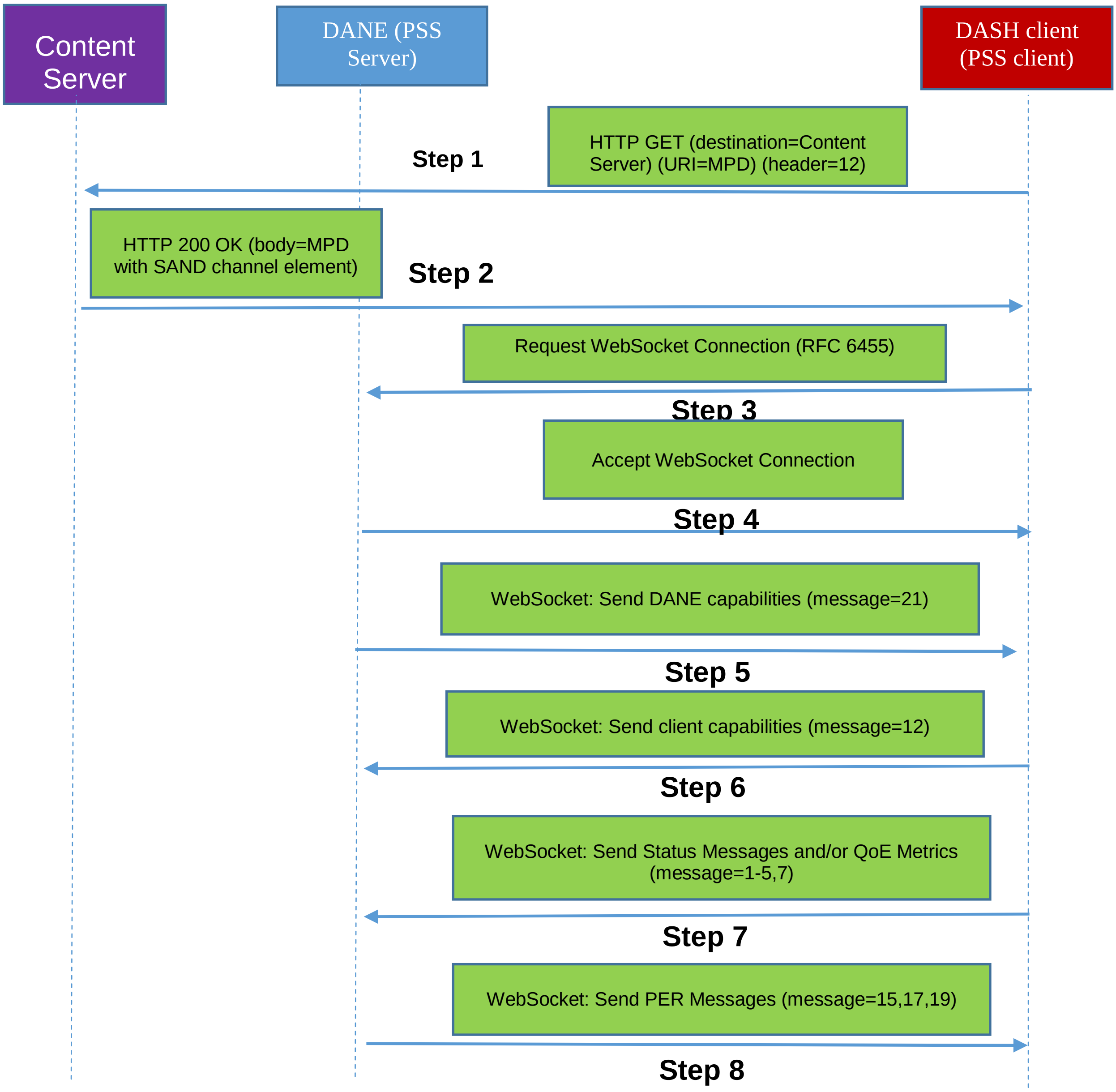

Content Server
DANE (PSS Server)
DASH client (PSS client)
Step 1
HTTP GET (destination=Content Server) (URI=MPD) (header=12)
HTTP 200 OK (body=MPD with SAND channel element)
Step 2
Request WebSocket Connection (RFC 6455)
Step 3
Accept WebSocket Connection
Step 4
WebSocket: Send DANE capabilities (message=21)
Step 5
WebSocket: Send client capabilities (message=12)
Step 6
WebSocket: Send Status Messages and/or QoE Metrics (message=1-5,7)
Step 7
WebSocket: Send PER Messages (message=15,17,19)
Step 8

Figure 6.1: Example SAND workflow for Consistent QoE/QoS use case

Step 1: Client issues an HTTP GET and sends request for MPD to the content server. In the header of the HTTP request, client includes the SAND header that contains the status messages on client capabilities (SAND message 12).

Step 2: Content server responds with HTTP 200 OK with body containing the MPD. As specified in ISO/IEC 23009-5 [2], the MPD contains a sand:Channel element whose @schemeIdUri is "urn:mpeg:dash:sand:channel:websocket:2016" and WebSocket URI in the @endpoint attribute.

Steps 3, 4: The DASH Client parses the MPD starts downloading the segments. In addition, the sand:Channel element is located in the MPD element. Using this information, the DASH client initiates the WebSocket connection with the out-of-band DANE (e.g., located in the PSS server) as specified in RFC 6455 [17].

It is noted here that WebSocket-based SAND channel announcement may also be accomplished by the use of OMA DM [18].

Steps 5, 6, 7, 8: Upon successful establishment of the WebSocket connection between a DASH Client and DANE, the DASH client starts listening for incoming PER messages and may send metrics and status messages when needed. Since the WebSocket Protocol establishes a full-duplex connection, the DANE and the DASH client may exchange SAND messages travelling simultaneously in opposite directions over the channel.

The DANE sends the DANE capabilities message (SAND message 21) to the DASH Client (Step 5). When received, the DASH Client replies with a client capabilities (SAND message 12) message (Step 6). As specified by ISO/IEC 23009-5 [2], the messages are WebSocket messages sent in the text format and formatted in XML. More specifically, based on the messageType values documented in Table 4-3, the DANE and DASH client negotiate the use of the following SAND messages for consistent QoE/QoS:

- PER: 15, 17, 19, 21
- Status Messages: 7, 12
- QoE Metrics: 1-5

When the capabilities messages have been exchanged, the WebSocket connection stays open and further SAND messages relevant for consistent QoE/QoS may be exchanged, namely PER messages on shared resource assignment, QoS information and throughput (SAND messages 15, 17, 19), QoE metrics (SAND messages 1-5) and status messages shared resource allocation (SAND message 7).

6.3 Use Case # 3 Proxy Caching

6.3.1 Use case description

John, who lives at Europe, has discovered that his DASH-enabled device suffers from frequent playback quality variation and some playback interruptions, both of which he finds annoying. For streaming quality, John prefers to view streaming in a stable playback quality and fewer or preferably no playback interruptions. In addition, higher presentation media quality is preferable. John has also noticed that such quality variations and playback interruptions typically occur when he views media presentations for which origin servers are located at outside of Europe such as Asia. For content providers that deploy HTTP proxy caches worldwide (potentially covered by multiple CDNs), John notices that streaming quality is generally better with more stable playback, since the assistance of proxy help to save bandwidth and reduce delay.

6.3.2 Recommended requirements and working assumptions

- It should be possible for DASH clients to discover or request which content is already cached at the HTTP proxy cache.
- It should be possible for a DASH client to send an indication or hint (e.g., on anticipated requests) to the network toward enabling intelligent proxy caching by the network.
- In a multi-CDN environment, it should be possible for the network to send hints to a DASH client to steer the DASH client to a certain CDN.

6.3.3 Gap analysis w.r.t. existing 3GPP technologies

6.3.3.1 Partial representation caching

John's DASH-enabled device sends HTTP GET segment requests by parsing a specific Media Presentation Description (MPD). Prior to serving John's segment requests, the proxy cache may have served other DASH clients with the same media presentation where they created HTTP GET segment requests by parsing the same MPD as John's DASH-enabled device. The proxy cache may cache segments which have been sent to other clients for serving future clients requests. As DASH clients request segments, but also switch Representations dynamically, the proxy cache may cache multiple Representations, each of which may be completely or only partially cached. A partially cached Representation is defined as a Representation having segment gaps, i.e. not all segments of the Representation are cached.

6.3.3.2 Next segment caching

CDNs can optimize the delivery of DASH resources by pre-caching segments and subsegments into the cache.

This is issue is specifically relevant in the case of using segmented Representations in an On-Demand case. In case a single Representation is used, the use of byte ranges provides sufficient indication for the CDN to prefect additional data.

One way to accelerate delivery of segmented content over a CDN is to have the proxy server pre-fetch the next segment from origin at the same time as it retrieves the current segment. This means that the segment is ready and waiting when the next request arrives from the client. Since this proxy server serving the media segment is not necessarily the same server which served the MPD, it has no visibility in to what the next segment might be. Additionally, it is stateless, and retains no knowledge of prior requests or related MPD requests.

6.3.3.3 Multi-CDN offering

A content provider may want to utilize multiple CDNs for content delivery, e.g. because some CDNs offer better coverage in certain regions. The content offering can include all available delivery choices (e.g. multiple baseUrls). The content provider can use suitable signalling as defined by SAND to steer the DASH client to a certain CDN.

6.3.4 Potential solutions including relevant SAND functionality

To realize partial representation caching, SAND can be used to inform DASH clients about partially cached representations, e.g., via use of the PER messages *ResourceStatus* and *DaneResourceStatus*. Moreover, toward realizing next segment caching, SAND can be used by DASH clients to inform the network (i.e., DANE) anticipated DASH segments, acceptable alternative content, etc. leading to next segment caching, e.g., via use of the status messages *AnticipatedRequests*, *AcceptedAlternatives*, and *NextAlternatives.*

An example workflow realizing next segment caching is depicted in Figure 6.2, where DANE (PSS Server) caches content based on SAND-based status messages received from the DASH client (PSS client).

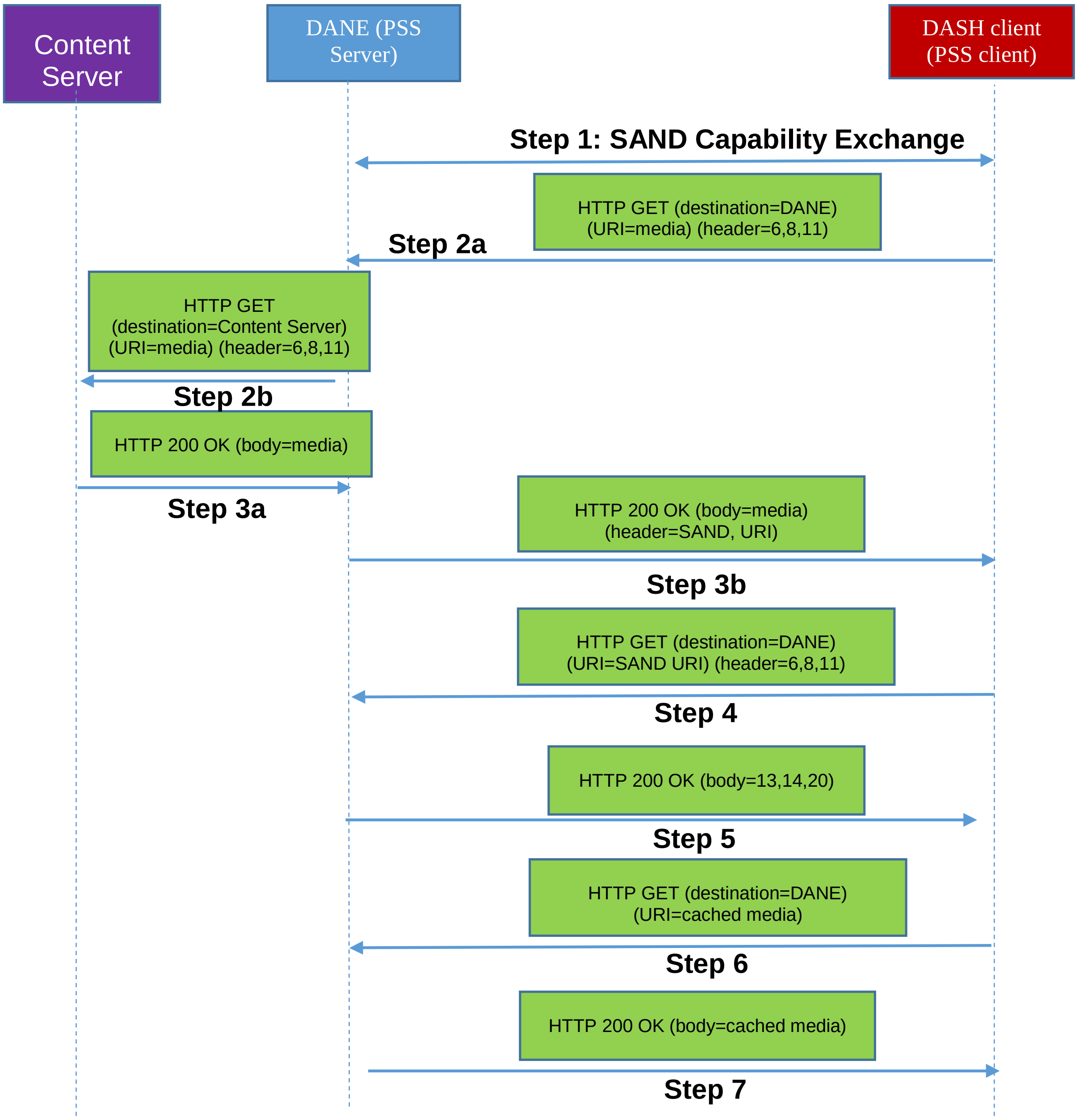

Content Server
DANE (PSS Server)
DASH client (PSS client)
Step 1: SAND Capability Exchange
HTTP GET (destination=DANE) (URI=media) (header=6,8,11)
Step 2a
HTTP GET (destination=Content Server) (URI=media) (header=6,8,11)
Step 2b
HTTP 200 OK (body=media)
Step 3a
HTTP 200 OK (body=media) (header=SAND, URI)
Step 3b
HTTP GET (destination=DANE) (URI=SAND URI) (header=6,8,11)
Step 4
HTTP 200 OK (body=13,14,20)
Step 5
HTTP GET (destination=DANE) (URI=cached media)
Step 6
HTTP 200 OK (body=cached media)
Step 7

Figure 6.2: Example SAND workflow for Proxy Caching use case

Step 1: The SAND capability exchange between the DANE and client will negotiate the use of the related SAND messages for proxy caching (using the SAND messages ClientCapabilities and DaneCapabilities as described in Clause 4). More specifically, based on the messageType values documented in Table 4-3, the DANE and DASH client negotiate the use of the following SAND messages:

- PER: 13, 14, 20, 21
- Status Messages: 6, 8, 11, 12

Step 2a: Client issues an HTTP GET and sends request for media to the DANE. In the header of the HTTP request (per the standardized formats in [2], as described in Table 4-4 in Clause 4), client includes the SAND header that contains the status messages on proxy caching, namely on anticipated requests, accepted alternatives and/or next alternatives (SAND messages 6,8,11). DANE receives these status messages, processes them and then forwards the SAND header that contains the status messages.

Step 2b: The DANE forwards the HTTP request for the desired media to the content server, since the DANE does not have a cached version of the media. DANE forwards the HTTP headers carrying SAND messages to without any modification.

Step 3a: Content server responds with HTTP 200 OK with body containing media.

Step 3b: In the HTTP response, DANE includes SAND header to advertise availability of PER messages on proxy caching with the URI hosted at the DANE for the corresponding PER messages, namely on resource status, DANE resource status and/or delivered alternatives (SAND messages 13, 14, 20).

Step 4: Client issues an HTTP GET request targeting the URI hosted at the DANE to fetch the PER messages on proxy caching, namely on resource status, DANE resource status and/or delivered alternatives (SAND messages 13, 14, 20). In the header of the HTTP request, client may include the SAND header that contains further status messages on proxy caching, namely on anticipated requests, accepted alternatives and/or next alternatives (SAND messages 6,8,11).

Step 5: DANE responds with the HTTP OK with body containing the PER message on proxy caching, namely on resource status, DANE resource status and/or delivered alternatives (SAND messages 13, 14, 20).

Steps 6,7: Client requests and downloads cached media from DANE.

6.4 Use Case # 4 Network Assistance for DASH

6.4.1 Use case description

6.4.1.0 Introduction

DASH clients typically perform rate adaptation based on their buffer fullness level, available representation rates and estimates of short-term future throughput. In a wireless network the throughput typically varies quite fast with time, while the client adaptation adjustment is relatively slow, leading to an estimation by the client that carries an error. Accumulated errors, and/or significant individual estimation errors can lead to buffer underrun and stalling of audio/video content playback during re-buffering.

The use case for network assistance consists of providing the client with better estimates of the short term throughput so it can better adapt to the throughput and avoid stalling of audio/video playback. This would be a beneficial functionality when introducing MPEG SAND into the 3GPP wireless mobile network context.

Based on the available media rates the network may assist the client with a recommendation of the highest suitable media rate for the next coming media download, i.e. a Recommended rate.

Further, to avoid buffer underrun, the network may also assist the client in situations with speeding up buffer filling where the buffer level is very low.

6.4.1.1 Network Assistance

The concept of network assistance is based on a general query/response approach. A video client may send a query for assistance, typically prior to a buffer filling occasion. Within the query message assistance information is provided, such as the available media rates and current media buffer status at the client. The network responses to the query with relevant information to the video client and the network may take actions to assist the video client to better QoE. The query/response procedure is repeated when the video client needs assistance, e.g. prior to a media segment download. The query/response procedure may be preceded with an initiation/authorization phase, e.g. at streaming session initiation in order to exchange initial parameters related to the client and/or media.

In the network response the network may assist the client with a recommended video payload rate. The recommended rate is only additional input to the rate selection algorithm in the client, and is the highest media rate the network recommends at this moment for good user experience. The recommended rate may take into account estimates of both radio link and transport link rates, radio quality and number of active users. It may consist of the estimated rate for the video application, if the bearer is shared with other services. It may also include network/operator policies aiming to restrict the rate. The recommended rate is neither enforced by the network, nor does the network make any commitment that the recommended rate will be honoured. How the information is gathered and relayed to the Network Assistance function is a subject for further study.

The information needed to make this type of rate estimation of the available network throughput for next coming time period is only available in the network. The client or UE only has historic rate information from previous downloads, and this historic information may be many seconds old depending on when last download was made.

Since buffer level and start of video session are reported by video client the network may at certain instants consider temporarily increasing the priority of data transfer to the specific client. This is denoted a throughput boost, and is a short temporary increase of throughput in the network, e.g. in the beginning of a video session for improved time-to-play latency and/or when the buffer level is low and there is risk for video playout stalling. A throughput boost may be realized in the Access Network or in the Core Network. The network informs the client when throughput boost is performed, such that the client is not misled in available throughput, and the maximum media rate the client may use.

Control mechanisms may be implemented in the network and the video client to analyse the accuracy of provided network assistance information. The operator may choose to provide the boosting function in Network Assistance to trusted clients only.

6.4.1.2 Relationship between Network Assistance and media server communication

From server and network perspectives, the Network Assistance communication is independent from the media server communication; the MPD and the media content are routed in a path that is separated from the Network Assistance communication. The media client will provide the necessary information, such as available media versions with the required bit-rates, and buffer level, to the Network Assistance function. The media server does not need to be aware of the Network Assistance function. The client will however utilize the Network Assistance function to make better choices for the requests directed to the media server.

The Network Assistance functionality is based on a general query/response approach, whereby the client may request to receive assistance information in the form of a recommended media rate from the network perspective. Hence, after an initiation process that may include client authentication and authorization, the client may at any time send a Network Assistance query. This process is independent of the client communication with the media server.

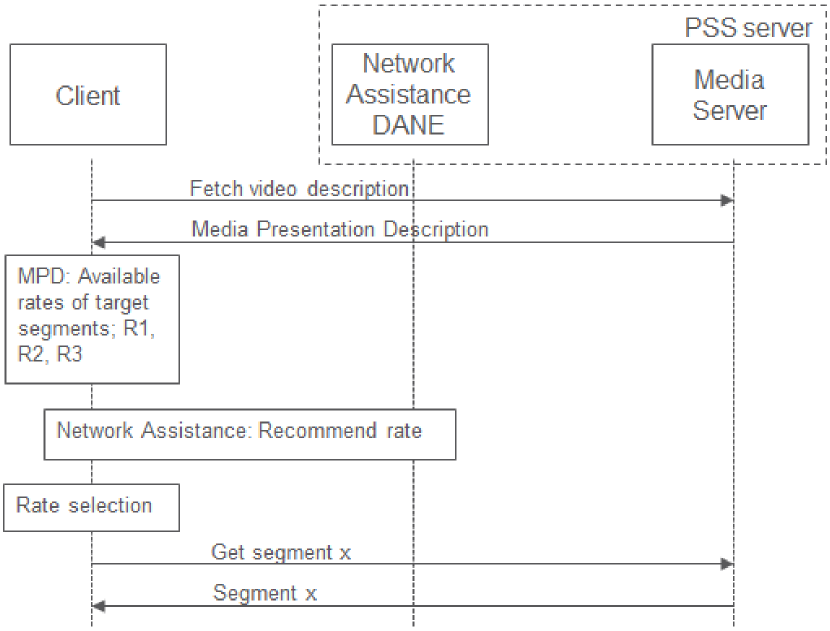

PSS server
Client
Network Assistance DANE
Media Server
Fetch video description
Media Presentation Description
MPD: Available rates of target segments; R1, R2, R3
Network Assistance: Recommend rate
Rate selection
Get segment x
Segment x

Figure 6.3: Generic sequence diagram for streaming session launch including Network Assistance

Figure 6.3 depicts the generic sequence of streaming client interaction with a media server and network entity when launching a streaming session including Network Assistance. Here it is assumed that the client has already been authenticated and authorized (if necessary) to use the Network Assistance function.

Typically the available data throughput in a cellular network will vary over time, and the client may have only historical throughput information. However, the Network Assistance function may provide information about the suitable media rate for a coming time period.

Regarding the time between Network Assistance queries it can be noted that a client algorithm for media rate selection is implementation-specific. A client algorithm implementation could include sending a Network Assistance query prior to a buffer filling operation. Since the client will have the opportunity to select media rate on a segment-per-segment basis, the segment length could give an indication with respect to the shortest expected time between two network assistance query messages from the same client. Client implementations may in certain use cases and/or client implementations conduct multi-segment downloads, where the client media buffer is filled with multiple segments in one download operation. In such implementations it is reasonable that a Network Assistance query is sent only upon each multiple segment download operation, significantly extending the time between each query.

Further, regarding the network load impact by the Network Assistance messages it can be noted that the amount of data traffic transmitted as Network Assistance query/response messages will be insignificant in relation to the amount of data traffic transmitted as video content.

6.4.1.3 UE considerations

6.4.1.3.1 Mitigation of unwarranted usage of the Network Assistance function

In theory a UE could try to take unfair advantage of the Network Assistance function by continually requesting buffer-refill boosts to increase the delivery throughput of an ongoing session. This is certainly a valid concern if the Network Assistance function were to be made available for download-type services, but this scenario should not occur for streaming-type services, as foreseen in the present use case, since the content is being stored only in a streaming playback buffer, i.e. a transient cache whose size is commonly kept to a minimum size that enables sufficient contingency for temporary delays in fetching content segments. Hence once this buffer is full there is no advantage to accelerate the fetching of the next segment. Further, network infrastructure could recognize such repeated needless requests for boosts and mitigate them quite easily. Furthermore, the UE could be subject to certification against a compliance regime that includes testing of its Network Assistance client functionality, before being authorized to access this feature in a real service.

6.4.1.3.2 UE access to the Network Assistance function

As is generally the case with SAND-related UE functionality, as described in clause 5, access to the Network Assistance function in the UE could be realized flexibly, and via any of the approaches listed in clause 5.

6.4.2 Recommended requirements and working assumptions

Void

6.4.3 Gap analysis w.r.t. existing 3GPP technologies

The current 3GP-DASH specifications are missing the following functionalities that are needed to assist the 3GP-DASH client in rate adaption:

- How the 3GP-DASH client provides information to the 3GPP network (i.e. the PSS server) about the available media rates for a particular DASH content item.

- How the 3GP-DASH client requests rate adaptation assistance from the 3GPP network (i.e. the PSS server).

- How the 3GPP network (i.e. the PSS server) provides information about the highest recommended media rate for a particular DASH content item to the 3GP-DASH client.

- The current 3GP-DASH specifications are missing the following functionalities that are needed to assist the 3GP-DASH client with buffer filling:

- How the 3GP-DASH client provides information on the current client buffer level to the 3GPP network (i.e. the PSS server).

- How the 3GPP network (i.e. the PSS server) requests the 3GP-DASH client to limit the media rate of the requested segments.

Also, the 3GP-DASH client needs to be able to obtain information in order to locate the assistance function in the 3GPP network (i.e. the PSS server).

6.4.4 Potential solutions including relevant SAND functionality

Network Assistance may be realized using SAND with a DANE function for network assistance. The network assistance function may be provided by an out of band DANE, and can be applied also together with encrypted video traffic. The Network Assistance function does not depend on detecting or reading media segments or the MPD, see Figure 6.4.

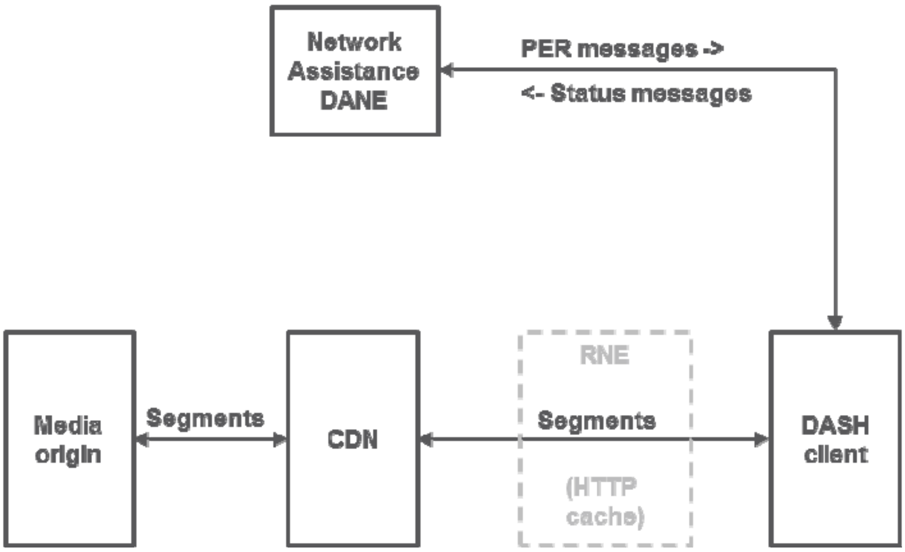


Figure 6.4: Network Assistance provided as an out-of-band DANE

The Network Assistance DANE may also be provided in combination with DANE's handling the media, see figure 6.5.

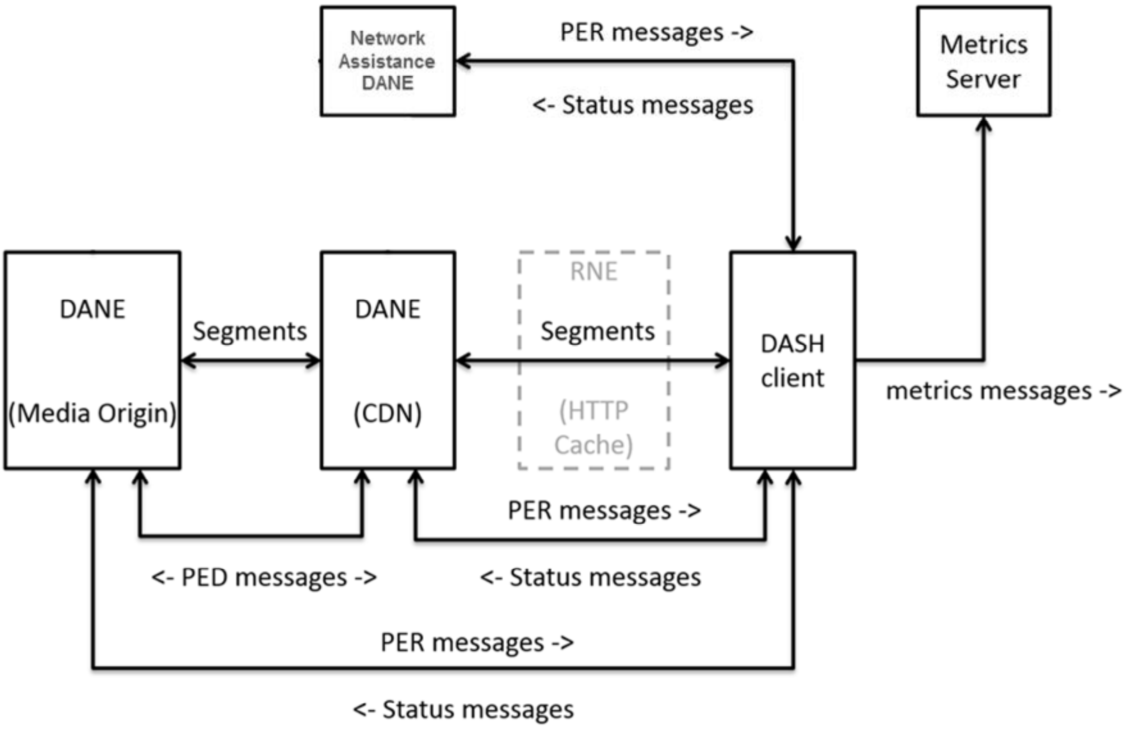


Figure 6.5: Network Assistance provided as an out-of-band DANE in an architecture with DANE's handling media

The following messages may be used to assist the 3GP-DASH client with rate adaption:

- The 3GP-DASH client may provide information of the available media rates for a particular DASH content item to the DANE using the SAND message *SharedResourceAllocation* with the parameter *bandwidth* for each operation point (see clause 6.4.2 of [3] for the detailed semantics). It is assumed that the parameter bandwidth for all operation points represents all available media rates for a particular DASH content.

- The DANE may provide information of the highest recommended media rate for a particular DASH content item to the 3GP-DASH client using the SAND message *SharedResourceAssignment* with the parameter *bandwidth* (see clause 6.5.3 of [3] for the detailed semantics). It is proposed that the parameter bandwidth represents the highest recommended media rate, based on a network estimation/prediction of the resource assignment for this 3GP-DASH client for the next coming period. The period is defined using the attribute *validityTime* in the SAND message envelope.

The following messages may be used to assist the 3GP-DASH client with buffer filling:

- The 3GP-DASH client may provide information about the current client buffer level to the network using the SAND message *BufferLevel* (see clause 6.3.5 of [3] for the detailed semantics).

- The DANE may request the 3GP-DASH client to limit the media rate of the requested segments using the SAND message *QoSInformation* with parameter *mbr* (see clause 6.5.7 of [3] for the detailed semantics) when the network applies quicker buffer filling strategies. The period of validity of *mbr* is defined using the attribute *validityTime* in the SAND message envelope.

Figure 6.6 shows a potential technical solution describing how the client is authorized to use Network Assistance, if necessary to do so, and how the client derives the address to the Network Assistance function. Once the 3GP-DASH client has information about the address to the Network Assistance function the communication between 3GP-DASH client and Network Assistance function can be transferred as user plane data. When the client is authorized (if necessary), the client queries the network for assistance. This may be repeated for every segment download.

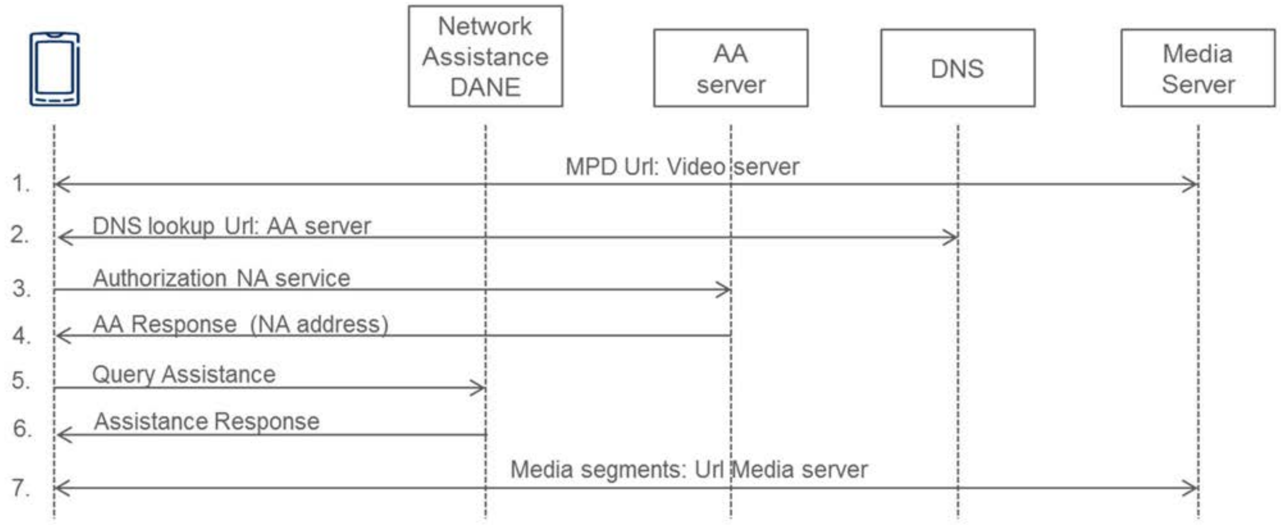


Figure 6.6. Sequence for Network Assistance between the media client and the network

1) Client requests download of MPD from media server

2) Client makes a DNS lookup of a default Authentication/Authorization server address

3) Client requests to be authorized and authenticated for use of Network Assistance service

4) AA server responds with a Network Assistance function address

5) Client queries assistance from the Network Assistance function

6) Network responds with assistance information

7) Client downloads media segments from the media server

NOTE 1: Steps 2-4 may also be performed before step 1.

NOTE 2: Steps 5-6 are repeated when the media client needs assistance, e.g. before every segment download.

The sequence depicted above is meant to be generic and rather hypothetical. It is not intended that any new or special authentication or authorization process is deployed just for network assistance, rather authentication and/or authorization may be performed according to the requirements of the service being delivered. Authentication/authorization for network assistance may also be granted by the DANE directly. It is also possible to make the UE aware of the location of the AA server, if required, and NA server, well in advance of the UE actually needing to request network assistance. For example, this information could be pre-configured or made known to the UE when the streaming service is launched on the UE, e.g. via OMA Device Management [18].

6.4.5 Simulation and test results

To verify the need for network assistance, simulations and lab tests are performed with and without the network assisting the client in rate adaptation. The simulations are made comparing two different client rate adaptation algorithms with a Network Assistance Rate Adaptation algorithm, NARA. The first client algorithm is based on client throughput measurements, where the media rate for the next segment download is selected based on the average throughput of the three last downloads. The second client algorithm is based on client buffer fill level, where the media rate is gradually increased with the increase of buffer fullness. The lowest media quality is selected when the buffer fullness is 30% or lower, while the highest media rate is selected when the buffer fullness is 80% or higher. The NARA algorithm is based on network throughput estimations of the next-coming period.

The simulations were made streaming a video session of 120 seconds long and with media segment length of 10 seconds, while the lab tests were made streaming a video session of 60 seconds long and with media segment length of 2 seconds.

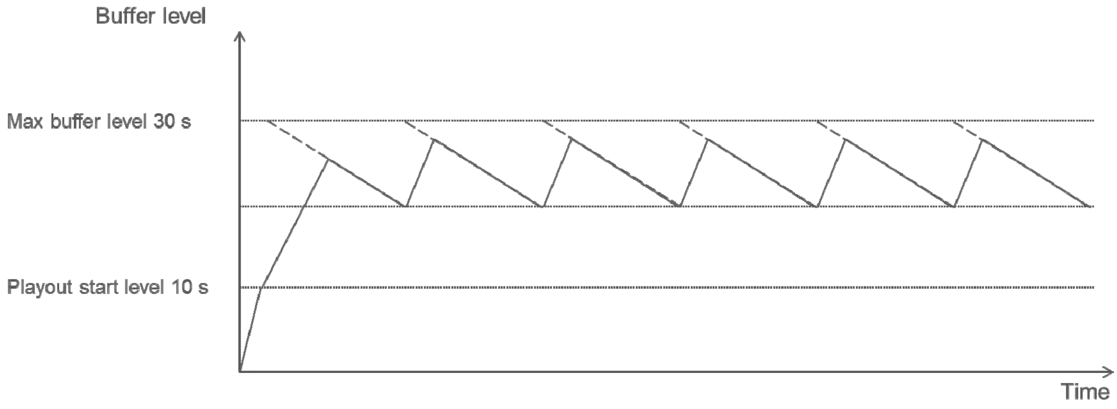


Figure 6.7: Buffer filling strategy used for all rate adaptation algorithms

Figure 6.7 shows the buffer filling strategy in the video client in the simulation case, but the principle is the same in the lab tests. The maximum buffer level is 3 media segments, i.e. 30 seconds, and the video playout starts when the buffer is filled with one media segment, i.e. 10 seconds of video. While playing the video, the client continues to fill the buffer up to 3 media segments. The downloading of media segments is paused until 2 media segments remain in the buffer and a new segment is downloaded. The maximum buffer level is not reached due to one media segment is always playing.

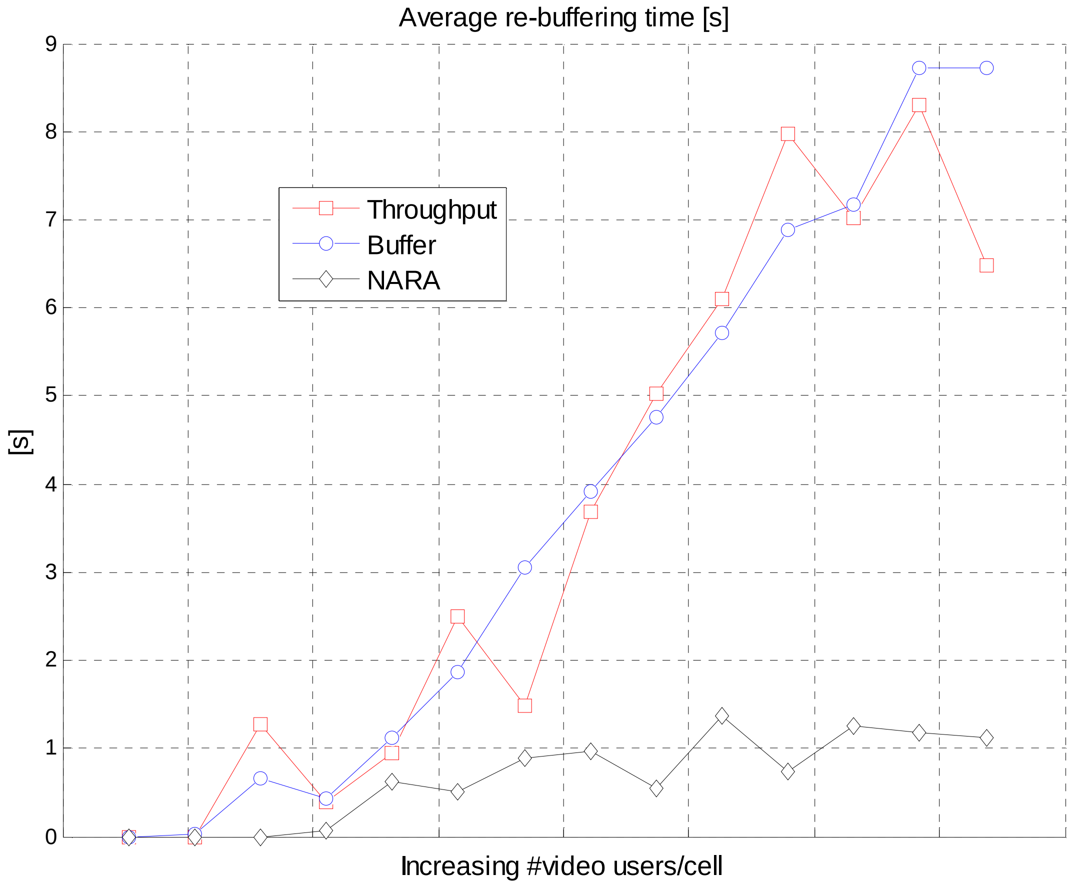


Figure 6.8: Simulations of average re-buffering time (30s buffer, 10s segments)

Figure 6.8 shows the average re-buffering time, i.e. the audio/video stalling time, over an increase of number of video users in a cell. These LTE system simulations were made in a 3GPP case 1 scenario, with a UE speed of 3 km/h and an inter-site distance of 500 m with 9 cells in total. There is also web-traffic in the background to create a mixed traffic scenario.

With increased load (number of video users /cell) and non-assisted rate adaptation, the average re-buffering time increases linearly, while with use of NARA - the average re-buffering time increases much more slow.

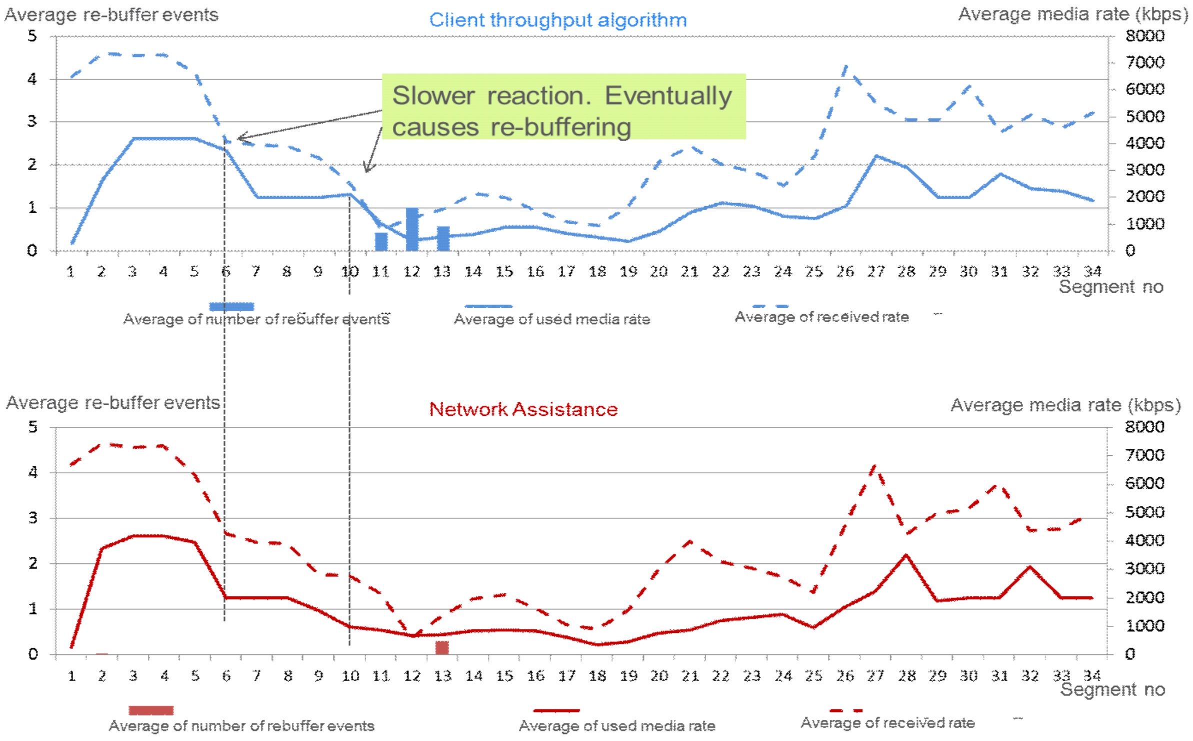


Figure 6.9: Lab test with video clients using a client throughput based algorithm and a client using network assistance

Figure 6.9 shows results from tests made in a LTE lab environment comparing a client throughput based algorithm and a client using network assistance, taking the average of 10 repetitive tests. Web traffic in the background and varying link quality corresponding to a UE speed of 3 km/h was used. The tests are made to evaluate if a throughput based client rate adaptation algorithms can make quicker decisions if it selects the media rate for the next segment download based on the average throughput of only the last media segment download. It may be seen in the top graph that the client still does not have the latest information of link throughput, and more re-buffering will occur when the link throughput decreases more rapidly compared to using the rate information with network assistance.

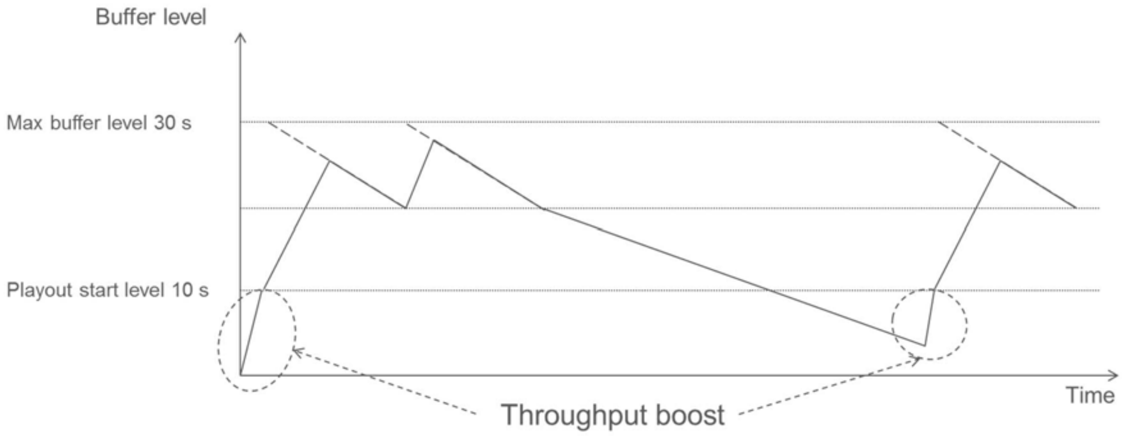


Figure 6.10: Throughput boosting to fill buffer at low buffer levels

As shown in Figure 6.10, throughput boosting (temporary increase of network throughput for this client) may be used in certain occasions to speed up the filling of the client buffer. In the beginning of the session throughput boosting may be used to shorten the time to playout, giving a better experience for the user. During the video session it may happen that the buffer level is very low due to large changes in the link throughput, and to avoid stalling of the video playout throughput boosting may be used to quickly re-fill the buffer to a certain level. When the network applies the throughput boosting the client should be informed, in order to not cause the client to be misled in available link throughput which may lead to that the client makes an erroneous media rate selection when the throughput is back to normal again, without boost, and selects a higher media rate than suitable for the next segment download. It should also be combined with a limit in allowed media rate not to cause unnecessary load.

7 MOOD for handling TLS protected unicast traffic

7.1 Description of the Release 12 solution for MOOD

The current MBMS MOOD and service continuity architecture assumes an HTTP proxy inside of the MBMS client (i.e. device proxy) in order to steer the DASH player towards unicast representations or broadcast representations. Note, a network HTTP proxy may be required for some MOOD operation modes. The term "MOOD" refers to unicast <-> broadcast switching due to the result of counting. The term "service continuity" refers to unicast<->broadcast switching due to mobility related events (i.e. no counting required), like the UE enters the broadcast area. Note, MOOD related counting may not be activated for a simple "service continuity" case.

The task of the MOOD device proxy (see figure below) in the present MOOD architecture is to direct HTTP requests from the DASH Player to either the remote HTTP server (unicast) or serve the segment from a local MBMS cache. Alternatively, the proxy may request segments from a local server.

The architecture below is derived from the MOOD TR 26.849 (Figure 7: Example Call Flow for UE-Elected Offloading Using Option 1A). A CDN Edge is implementing the Content Server / PSS Server. The unicast traffic may be routed through an HTTP Proxy of the BM-SC, when MOOD headers are added to the HTTP traffic. The BMSC HTTP Proxy is not depicted here. Any unicast HTTP traffic is routed through the P-GW and other 3GPP nodes (not depicted) to the client.

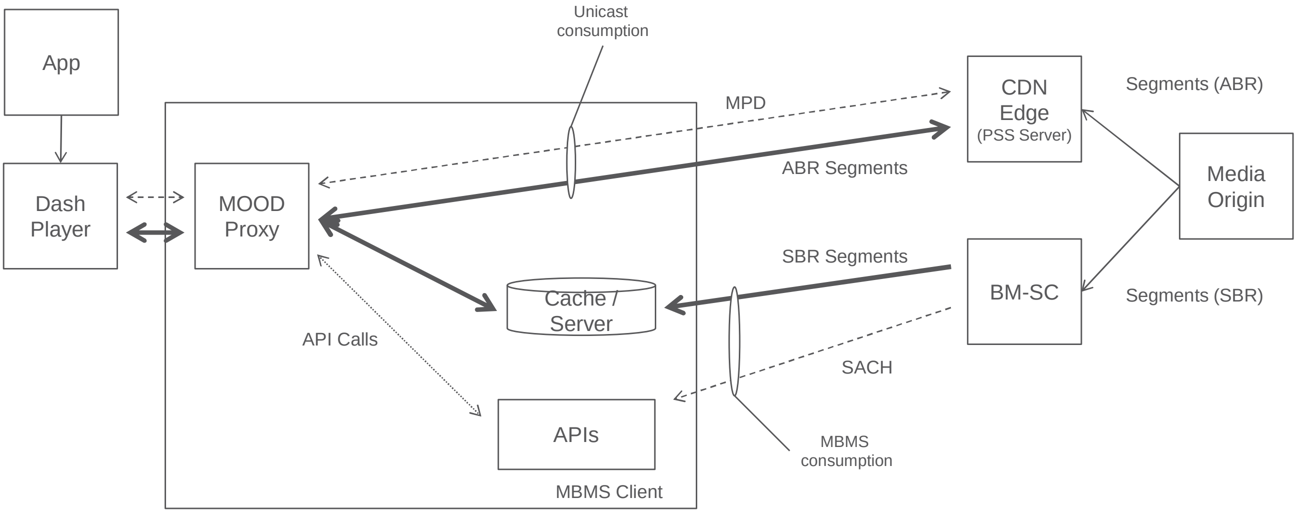


Figure 7.1

In some cases, the MBMS Client sends consumption reports to the BM-SC. The intention is that the BM-SC can monitor the popularity of the service and activate or de-active broadcast in some areas. MOOD headers can be used to monitor the popularity of unicast consumption, but not for broadcast consumption.

For unicast <-> Broadcast switching, the present MOOD proxy uses HTTP redirect, thus, the MOOD proxy will have to see the HTTP requests in clear-text.

The usage of transport protection (HTTPS) is more often used for unicast IP communication in order to secure privacy and also to secure control of the delivery pipe. The intention of using HTTPS is manifold:

- In case the content provider has no relation with the network provider (aka no business agreement in place), TLS is used to secure end-user privacy and to ensure authorized-only entities on the path. Here, neither a MOOD device nor MOOD network proxy may add MOOD headers or inspect the HTTP traffic due to counting.
- In case the content provider has a relation with the network provider (aka business agreement in place), TLS may still be required due to the App environment. For example, browsers start prohibiting the combined usage of secured and unsecured connections (also called mixed content). Some browser APIs are only accessible when secured connections are used. Here, the MOOD network proxy may have the TLS certificate of the content provider for that session (due to business agreement) and is therefore able to "see" the HTTP transactions and process the MOOD headers. However, it is unlikely that the MOOD device proxy has the TLS certificate of the content provider in order to add MOOD headers or to steer the DASH player (uc <-> bc switching).

In case of TLS protected HTTP Unicast traffic (it is assumed that the MOOD device proxy has NOT TLS domain certificates to intercept the HTTP messages), then:

- Any MOOD Proxy cannot steer the HTTP requests between unicast & broadcast anymore, since the MOOD proxy only forwards encrypted unicast traffic.
- The MOOD Proxy cannot add MOOD headers for counting since the MOOD proxy cannot add or modify the HTTP headers anymore. However, unicast consumption reporting can be used to convey consumption information, when needed, since the information is send separately from the payload.
- Two cases are foreseen for the study:

  1: The usage of TLS is required on all connections, independently whether unicast or broadcast is used.

  2: The usage of TLS is only required per access, i.e. all connections on unicast use HTTPS, while all broadcast received content use regular HTTP.

MPEG SAND can provide some solution to avoid the need for a local MOOD proxy on the device. Instead, a separate communication channel between the MBMS Client and the DASH Player could be established. The SAND WebSocket communication channel can be secured using TLS (although belonging to a different domain), so that all connections are secured. Note, the usage of the SAND WebSocket Channel is independent from the usage of HTTPS. There may be other reasons that a solution like the WebSocket Channel is preferred.

7.2 Overview of a SAND WebSocket solution for MOOD

7.2.1 Architecture

MPEG SAND enables a new procedure for Unicast <-> Broadcast switching using a MOOD DANEs. MPEG SAND defines two communication channels for PER messages to the client:

- Through addition of HTTP Headers (like DaneResourceSatus) in Segment or MPD responses, i.e. inband with the MPD or segment stream.
- Through a WebSocket communication channel, i.e. out-of-band of the MPD or segment stream using a separate channel.

The WebSocket communication channel is established through an HTTP request to the server, which contains a specific HTTP header in the request (i.e. 'Upgrade: websocket' header). The server grants the request and can use the established TCP connection in both directions. MPEG SAND has defined a message framing format for the WebSocket communication channel.

Through SAND, the DANE can steer the DASH Player into one way or another. It can make certain resources of the manifest un-available so that the client only uses a subset. It can also steer the DASH Player to a different MPD location. It should be studied, whether the DASH Player can handle MPDs with slight variations.

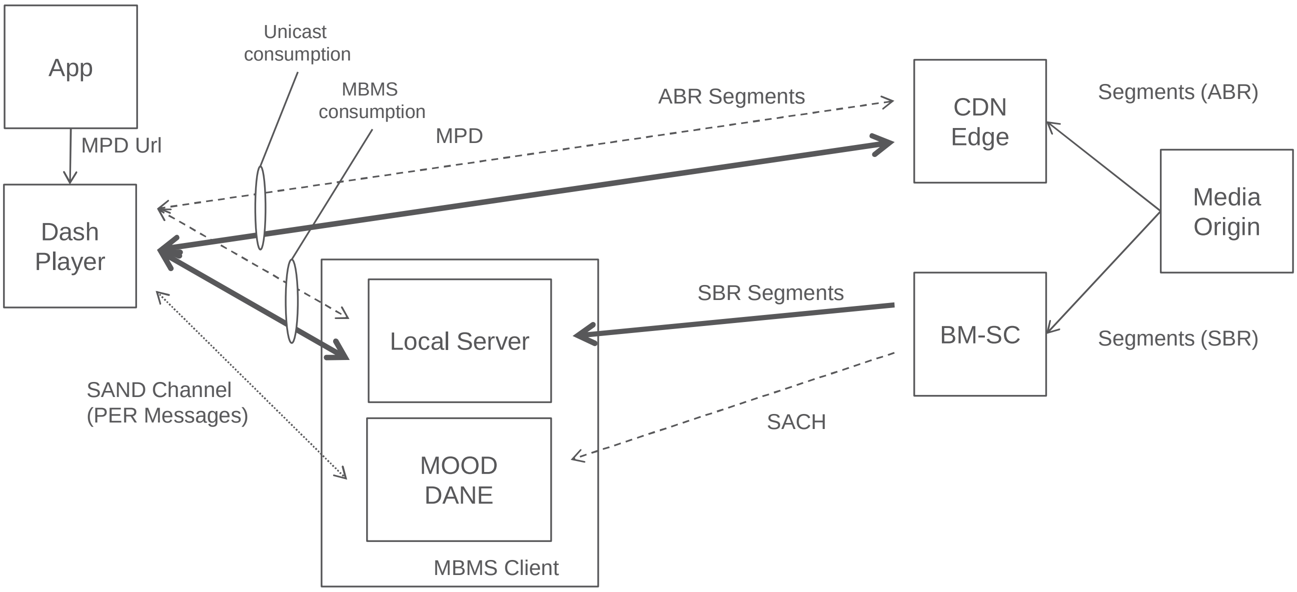


Figure 7.2

SAND allows signaling of MPD expiration (i.e. using the MPDValidityEndTime message type (number 14)) in a PER message and suggesting the DASH player to fetch the MPD from a different MPD location (i.e. through the element mpdUrl). When the MPD does not contain a baseUrl element, thus, when all segments Urls are relative to the MPD Url, then the DASH-Player starts fetching segments from a new location.

Consequently, the MBMS Client does not need to implement a proxy to redirect the different HTTP requests from unicast to broadcast or vice versa. This again allows usage of HTTPS for unicast traffic.

Consumption reporting information can be reported separately from the HTTP payload to the infrastructure using Consumption Reporting.

7.2.2 Procedures

A simplified procedure is depicted in the follow-up. Details depend on the progress of the TRAPI Work-Item.

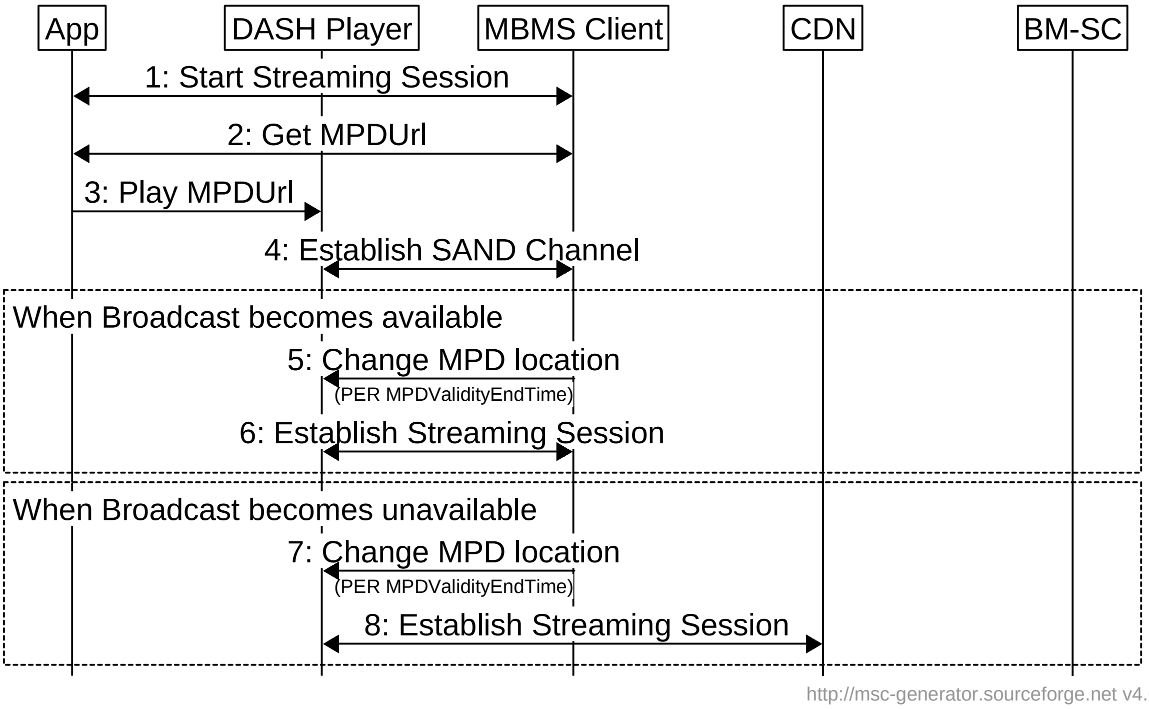

App
DASH Player
MBMS Client
CDN
BM-SC
1: Start Streaming Session
2: Get MPDUrl
3: Play MPDUrl
4: Establish SAND Channel
When Broadcast becomes available
5: Change MPD location
(PER MPDValidityEndTime)
6: Establish Streaming Session
When Broadcast becomes unavailable
7: Change MPD location
(PER MPDValidityEndTime)
8: Establish Streaming Session
http://msc-generator.sourceforge.net v4.6.2

Figure 7.3

The steps below focus on the control transactions and does not show the actual segment flow. It should be notes, that segments from the same DASH segmenter (aka PSS server) are fed into the CDN and also made available to the BM-SC.

1) The App triggers the start of a streaming sessions using the MBMS APIs as defined in TRAPI TR [1].
2) When the stream is started, the App gets the MPD Url from the MBMS Client (cf. [1]).
3) The App launches the Dash Player and passes the MPD Url to the DASH Player.
4) The DASH Player establishes the SAND WebSocket communication channel to the MBMS Client.

When Broadcast becomes available:

1) The MBMS Client triggers the change of the MPD location using the SAND MPDValidityEndTime in combination with the MPDUrl parameter. The MPDUrl contains a localhost URL. The MBMS Client may pass additional SAND messages to the client, e.g. DaneResourceStatus.
2) The DASH Player starts fetching the MPD Url from the MBMS Client (local server). The MPD does not contain any baseUrl statement so also the segments are fetched from the local server.

When Broadcast becomes unavailable:

3) The MBMS Client triggers the change of the MPD location using the SAND MPDValidityEndTime in combination with the MPDUrl parameter. The MPDUrl contains a CDN URL as conveyed in Service Announcement. The MBMS Client may pass additional SAND messages to the client, e.g. DaneResourceStatus.
4) The DASH Player starts fetching the MPD Url from the CDN Location.

8 Streaming enhancements from QoE-Aware resource allocation

8.1 Introduction

This clause describes simulation methodology and results on the evaluation of streaming enhancements from QoE-aware resource allocation over an LTE-based system-level simulation platform with DASH clients. In particular, an analysis is presented quantifying the performance benefits of intelligent QoS control and resource allocation mechanisms that take advantage of the awareness of client / device characteristics and real-time QoE measurements reported by the clients.

8.2 Simulation methodology and setup

Re-buffering has been identified as one of the most critical QoE metrics for streaming video. In a 3GPP DASH-based implementation of QoE metrics in the client device, this metric can be computed via monitoring the buffer status and/or play list metrics. Given the key importance of re-buffering in dictating the QoE delivered to the user, the service capacity of an LTE system is defined based on an outage criterion that is centered around the re-buffering percentage, i.e. the percentage of the total presentation time in which the user experiences re-buffering due to buffer starvation. In particular, a user is designated to be satisfactorily supported if its re-buffering percentage is smaller than a re-buffering outage threshold A^{out}. The service capacity is then defined as the maximum number of users that can be supported in the network such that the percentage of satisfied users is greater than the network coverage threshold A^{cov}. i.e.

$$C_{out}^{rebuf} = \mathbf{E}\left[\arg\max_{K} \left\{ \frac{\sum_{i=1}^{K} \mathbf{1}\left(p_{rebuf,i} \le A^{out}\right)}{K} \ge A^{cov} \right\} \right]$$

where **E[.]** is denotes the expectation over multiple user geometry realizations and **1(.)** denotes the indicator function.

Five VBR-encoded video clips (Sony, Citizen Kane, Die Hard, NBC News, Matrix Part1) are considered with different bitrate requirements hosted at the HTTP server with multiple versions of each video clip available at different quality levels in the PSNR range of 26-39 dB, as shown in Figure 8.1 and Table 8.1. Two video traces for each video representation level contain content information with regards to – i) size and quality information for each video frame and ii) offset traces which give information of the video quality obtained by concealing lost video frames with previous frames. PSNR was used to model video quality as a representative although other advanced metrics could also be used.

A cellular deployment is assumed based on an IMT-Advanced urban macro-cell (UMa) test environment with an inter-site distance (ISD) of 500 m, where each user in the LTE network randomly requests one of the five available video clips. A 19-cell scenario is considered, where the center cell generating video traffic is surrounded by two layers of interfering cells generating full buffer traffic. Users are randomly dropped in the center cell. The simulation parameter settings and assumptions on the LTE air interface are provided in Table 8.2 below. The additional assumptions include the following:

1) For the link to system mapping, Mutual Information Effective SINR Metric (MIESM) is used.

2) AWGN PER versus SINR curve corresponding to that modulation, code rate are used to determine the probability of error.

3) Channel Quality Indicator (CQI) are delayed by 5 ms.

4) HARQ retransmissions are delayed by 8 ms with a maximum of 4 retransmissions.

5) The base stations in all other cells generate interference patterns corresponding to a full buffer mode of operation.

6) 100,000 sub-frames were simulated to generate LTE link statistics.

7) Users were picked randomly from a user population of 684 dropped uniformly in the sector.

8) For each configuration, statistics were collected from thirty different random drops of users in the network.

9) Packet fragmentation based on the maximum MTU size of 1500 bytes is considered, and HTTP/TCP/IP layer protocol behaviour and overheads are also incorporated in the analysis - 40 bytes of header was included in each TCP segment (10 bytes for NALU prefix + 12 bytes for HTTP header + 8 bytes for TCP header).

10) All the main features of TCP Reno flavour were implemented in the simulator including flow control, slow start, congestion avoidance, RTT estimation, timeout, re-transmission, fast re-transmit and fast-recovery to account for the presence of TCP.

11) The Backhaul Network (BN) between the eNodeB (eNB) and S-GW is modelled with a fixed bandwidth of 1 Gbps.

12) Core Network (CN) from video servers to the S-GW was modelled using a fixed delay of 50 ms.

13) Core and back-haul networks are assumed to lossless and radio access network is considered as the main bottleneck.

14) Uplink transmissions are assumed to be errorless.

15) The delay involved in establishment of the dedicated bearer (e.g. GBR bearers) was not included in the assumed system model.

Multiuser resource allocation over the OFDMA-based downlink LTE air interface is performed based on the well-known proportional fair scheduling principles. Only half of the available bandwidth of the 10 MHz LTE system is assumed to be reserved for the DASH-based video streaming service while the remaining half is assumed to be dedicated for other services, e.g. voice and data services.

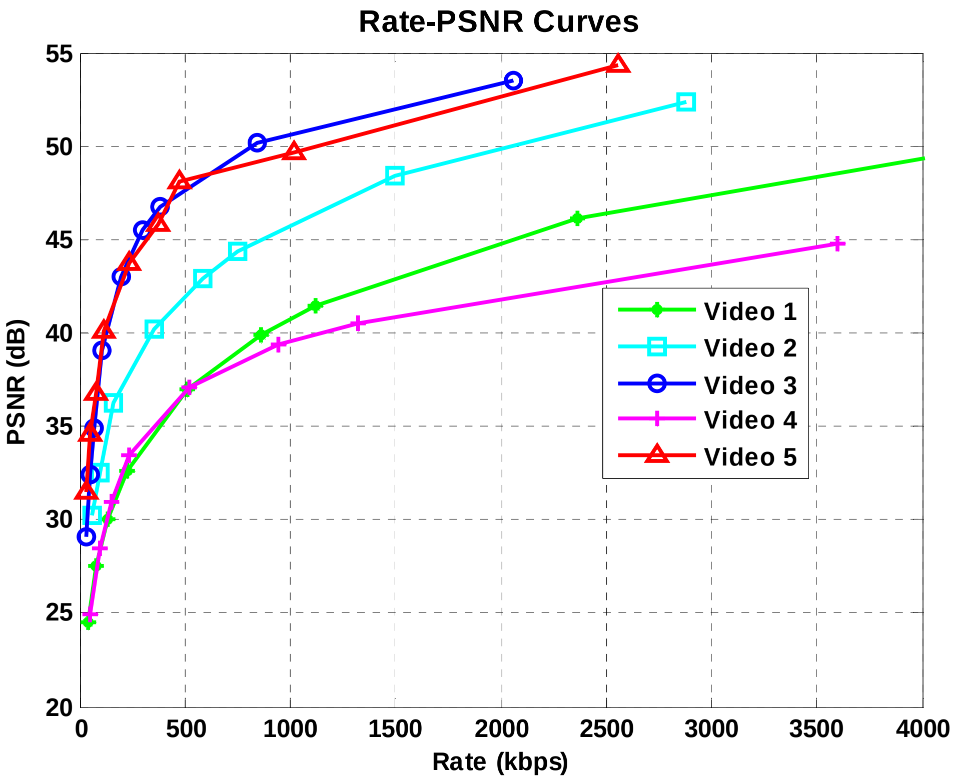
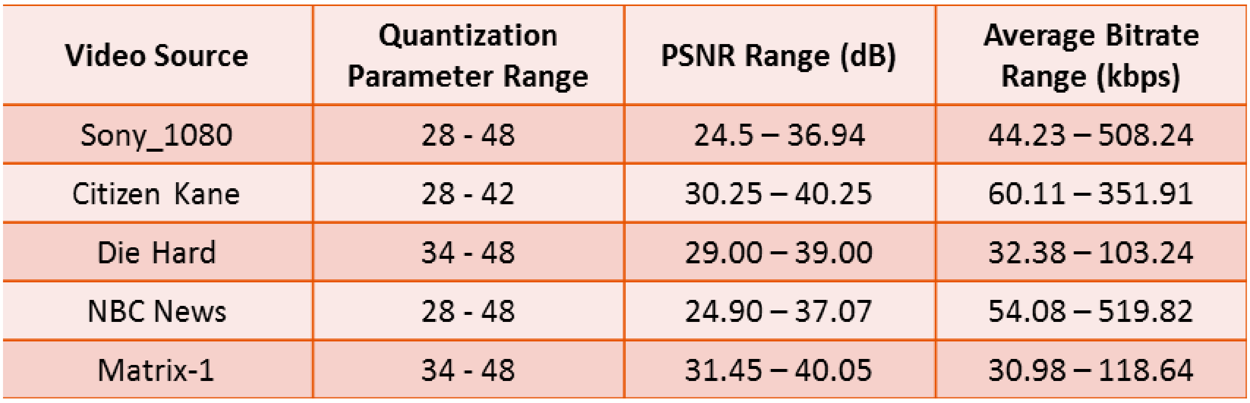


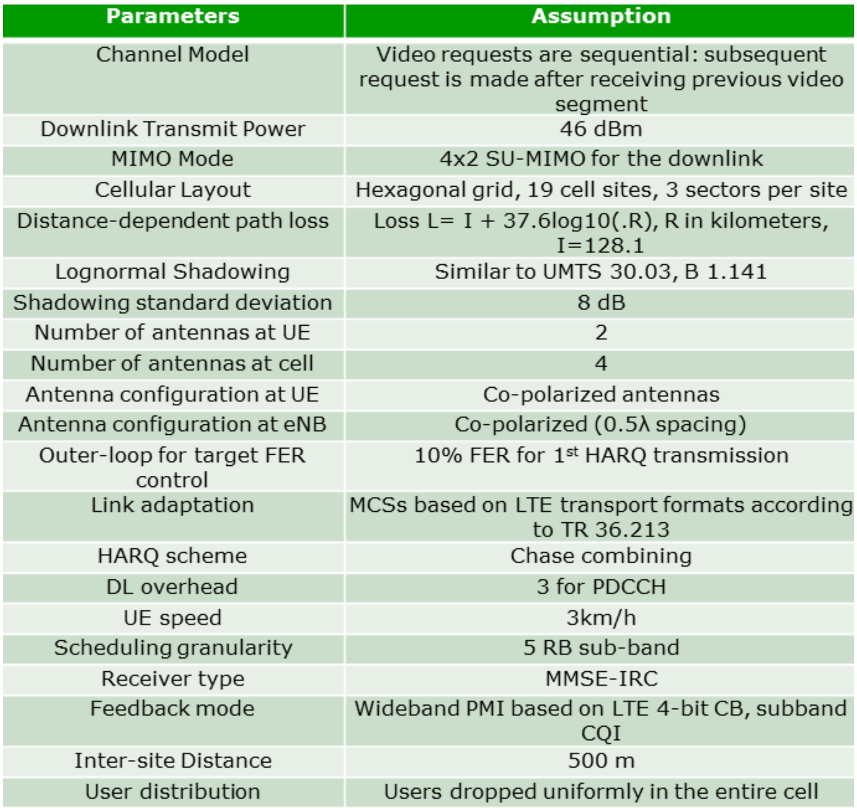
Figure 8.1: Rate-PSNR Curves of Sample Videos

Table 8.1: Details on the video content used in the evaluation



Video Source	Quantization Parameter Range	PSNR Range (dB)	Average Bitrate Range (kbps)
Sony_1080	28 - 48	24.5 – 36.94	44.23 – 508.24
Citizen Kane	28 - 42	30.25 – 40.25	60.11 – 351.91
Die Hard	34 - 48	29.00 – 39.00	32.38 – 103.24
NBC News	28 - 48	24.90 – 37.07	54.08 – 519.82
Matrix-1	34 - 48	31.45 – 40.05	30.98 – 118.64

Table 8.2: LTE Air Interface configuration



Parameters	Assumption
Channel Model	Video requests are sequential: subsequent request is made after receiving previous video segment
Downlink Transmit Power	46 dBm
MIMO Mode	4x2 SU-MIMO for the downlink
Cellular Layout	Hexagonal grid, 19 cell sites, 3 sectors per site
Distance-dependent path loss	Loss L= I + 37.6log10(.R), R in kilometers, I=128.1
Lognormal Shadowing	Similar to UMTS 30.03, B 1.141
Shadowing standard deviation	8 dB
Number of antennas at UE	2
Number of antennas at cell	4
Antenna configuration at UE	Co-polarized antennas
Antenna configuration at eNB	Co-polarized (0.5λ spacing)
Outer-loop for target FER control	10% FER for 1st HARQ transmission
Link adaptation	MCSs based on LTE transport formats according to TR 36.213
HARQ scheme	Chase combining
DL overhead	3 for PDCCH
UE speed	3km/h
Scheduling granularity	5 RB sub-band
Receiver type	MMSE-IRC
Feedback mode	Wideband PMI based on LTE 4-bit CB, subband CQI
Inter-site Distance	500 m
User distribution	Users dropped uniformly in the entire cell

According to the DASH-based adaptive streaming framework, users may consume varying qualities of video based on the working of the assumed adaptation algorithm, which selects the optimal quality/bitrate representation among the available video clips based on monitoring of user experience via 3GPP-based QoE metrics, i.e. particularly the playback buffer level. The different representations of the video requested by a representative client are indexed using letter k. In particular, k=1 represents the lowest bitrate representation level and k = N represents the highest representation level and b_k represents the bitrate of encoded video of representation level k, $b_1 \le b_2 \le b_3 \le \dots \le b_N$. Rate adaptation is client-driven and is done at segment level where each video segment might contain one or more GOPs (Group of Pictures).

The DASH-based adaptive streaming framework monitors the LTE link throughput and client buffer state and requests the video representations accordingly to realize the highest possible quality but also making sure to avoid playback buffer starvation. The DASH client starts playback with initial start-up delay of one second. It requests the video at a higher fetch rate during the buffering mode (playback buffer under a specified threshold) while the fetch rate is lower during the streaming mode (playback buffer above the specified threshold). Encountering playback buffer starvation, the client enters re-buffering mode while stalling the playback. The playback resumes after a certain targeted amount of media (i.e. 1 second) is aggregated in the media buffer.

A typical DASH-level throughput estimate is the average segment Throughput which is defined as the average ratio of segment size to the download time of the segment.

$$R_i^{seg} = \frac{1}{F} \sum_{s=S_i-F+1}^{S_i} \frac{S_{seg}(s)}{T_{dwnd}^{seg}(s) - T_{fetch}^{seg}(s)}$$

where $S_{seg}(s)$, $T_{fetch}^{seg}(s)$, $T_{dwnd}^{seg}(s)$ are the size, fetch time, and download time of the j^{th} video segment, S_i the number of segments downloaded until frameslot i, and F is the number of video segments over which the average is computed.

8.3 Video-Aware QoS optimizations at the Core

The architecture in Figure 8.2 is considered for video-aware network resource management. It consists of content servers, Video Aware Controller (VAC) in the core network, and DASH clients. An LTE network is considered consisting of an eNodeB and LTE Evolved Packet Core (EPC). A number of DASH clients establish streaming sessions to DASH servers through the eNB, EPC and the core network.

The "Video Aware Controller" (VAC) placed at a service provider central office at the network core is the central intelligence for video-aware resource management. The VAC is connected to the EPC. The VAC also contains a QoE reporting server to receive periodic feedback of media buffer levels from the DASH clients. This is done by establishing an application-level auxiliary control connection between the DASH client and the VAC for each streaming flow, e.g., an HTTP POST connection. The VAC has a synthetic view of the buffer levels of all the connected streaming clients. It computes a maximum bit rate MBR_j for each flow j and communicates the MBR_j of each flow j to the respective DASH client through the auxiliary control connection.

The key to this architecture is application-level feedback from DASH servers/clients to the VAC and QoS-signaling from the VAC. For example, the VAC can receive feedback of client media buffer levels from the DASH clients and video segment quality information from the content servers. Then the VAC processes the feedback information to determine QoS parameter settings for each flow in the network. Specifically VAC determines the Maximum Bit Rate (MBR) for each flow in the network. These QoS-parameters are then communicated to the DASH clients to which use this information in their video rate adaptation algorithm.

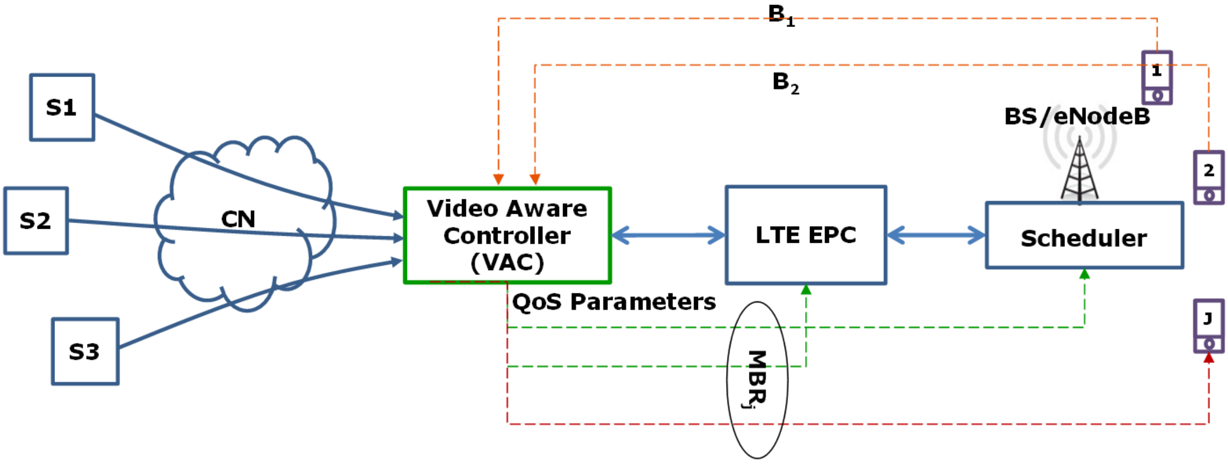


Figure 8.2: Video Aware QoS Signaling Architecture

Thus the VAC indirectly manages the network resources at various parts of the network by setting the QoS parameters that influence the DASH client adaptation behavior. The VAC controls the rate adaptation of the video clients by communicating the respective QoS-parameters to each client. At DASH client j, the computation of the best video representation level takes into account QoS parameters MBR_j and GBR_j of flow j as follows:

$$Q_i^{sup} = \arg\max_{k} b_k \, ;$$

$$\text{s.t.} \quad b_k \le \max\left(GBR_j,\ \min\left(R_j^t, MBR_j\right)\right);\ \ k = 1,\ldots,N$$

By setting MBRs for flows intelligently in a video-aware fashion, QoE-based outage capacity of the network is enhanced and by setting GBR for flows it is possible to guarantee certain minimum video quality for premium users. Moreover, this architecture has the ability to take into account both DASH client media buffer dynamics and variability in video content based on feedback from DASH clients and servers respectively. Through QoS-signaling to the DASH clients, it also establishes a close interaction between resource allocation to each flow and video rate adaptation by DASH clients.

NOTE: The clients for which the VAC sets MBR compete for the same bandwidth resources. This is regardless of where there is a single eNodeB or multiple eNodeBs reporting to the VAC. The latter depends on operator's frequency reuse and spectrum allocation policy. For instance, many deployments rely on full reuse, a.k.a. reuse-1, which means all cells and eNodeBs use the same spectrum to allocate resources, i.e., in this case clients in all cells would compete for the same spectrum. In that sense, it is up to the operator to decide how to use the VAC capabilities based on its spectrum allocation policy.

The forthcoming discussion focuses on the problem of setting MBR dynamically for DASH flows to increase QoE-outage based capacity. The buffer evolution at each client is determined by the playback and download processes. The playback process removes one video frame from the client media buffer each frame duration except in the startup/re-buffering states where it waits till the buffer level reaches a certain threshold. The download process determines the rate at which frames are entering the buffer. This in turn determines on the video segment representation level chosen. By setting a MBR for each DASH flow, the download process can be controlled. The VAC periodically tracks the media buffer evolution of each client. The difference between buffer levels for client j from feedback cycle (t-1) to feedback cycle t is given by:

$$B_j^{t,\text{diff}} = B_j^t - B_j^{t-1}$$

B determines the evolution of media buffer at DASH client j over the duration of a feedback cycle considering the playback and download processes. To avoid re-buffering, it is required that the download process is faster than the playback process. To minimize chances of re-buffering for each DASH client j, a requirement is set that the rate of change in the buffer level to be greater than a certain positive threshold i.e.,

$$\frac{B_j^t - B_j^{t-1}}{T} = \frac{B_j^{t,\text{diff}}}{T} > \left(X_j^t + \delta\right),\ \delta > 0$$

Where X_j^t is an indicator variable that indicates whether user j is in startup or re-buffering during the feedback cycle t. This can be easily determined by comparing the user buffer level relative to the buffering threshold. This parameter is used to ensure that even when there is no playback by the client, the rate of buffering should exceed the nominal playback back rate by the threshold δ. Since wireless conditions, video content etc. are dynamic, it is only required that the condition above on the rate of change of the buffer level holds in an average sense. A buffer aware user token parameter W_j^t is used to monitor the overall cumulative performance of the client j until feedback cycle t in terms of buffering rate exceeding the threshold. In every feedback cycle, W_j^t is updated based on B as follows:

$$W_j^t = \max\left(W_j^{t-1} + \left((X_j^t + \delta)\tau - B_j^{t,\text{diff}}\right), 0\right)$$

When B≤ (X_j^t + δ)τ in feedback cycle t, it means it means that the media buffer of user j is growing at a rate smaller than the target threshold. In this case W_j^t is then incremented by (X_j^t + δ)τ − B to reflect the penalty for having a low buffer change rate in feedback cycle t. Similarly when the buffer change rate for user j in feedback cycle t exceeds the target threshold, then W_j^t is then decremented to reward user j. Thus W_j^t represents the accumulated penalties and rewards for user j until feedback cycle t. A higher value of user token parameter W_j^t reflects the fact that the download rate for the user is not sufficient to sustain continuous playback.

A user is defined to be dis-satisfied when the user token parameter for the user exceeds a certain pre-defined threshold. User j is considered to be dis-satisfied at time t if the following condition is satisfied:

$$W_j^t > \gamma$$

The percentage of dis-satisfied users at time t, denoted by D(t), is then given by:

$$D(t) = \frac{\sum_{j=1}^{J} I(W_j^t > \gamma)}{J}$$

where I(.) is the identity function and J is the total number of users in the system. The VAC tracks the percentage of dis-satisfied users over time based on media buffer feedback from DASH clients. A window of observations consisting of the past L feedback cycles is taken into account to compute the trend of percentage of dis-satisfied users over time. Specifically at time t, observations D(t), D(t-1),…, D(t-L) are used to compute a linear fit as follows:

$$D_e(x) = m(t) * x + c(t), \qquad x = t, t-1, \ldots, t-L$$

m(t) and c(t) are the slope and the y-intercept for the linear equation depicting the trend of dis-satisfied users in feedback cycle t. A positive slope m(t) > 0 indicates that D(t) is increasing and a negative slope m(t) < 0 indicates that D(t) is decreasing.

MBR is used for each DASH session for controlling the percentage of dis-satisfied users. Decreasing the MBR has the effect of decreasing percentage of dis-satisfied users at the cost of video quality. Our basic idea is to set the MBR value just large enough keep the percentage of dis-satisfied users below a threshold ψ. At the beginning of every DASH session, the VAC initiates the MBR for the user to a nominal value and then updates it every feedback cycle depending on i) the percentage of dis-satisfied users D(t) and ii) the trend of percentage of dis-satisfied users m(t). The flowchart for our MBR update algorithm is depicted in Figure 8.3.

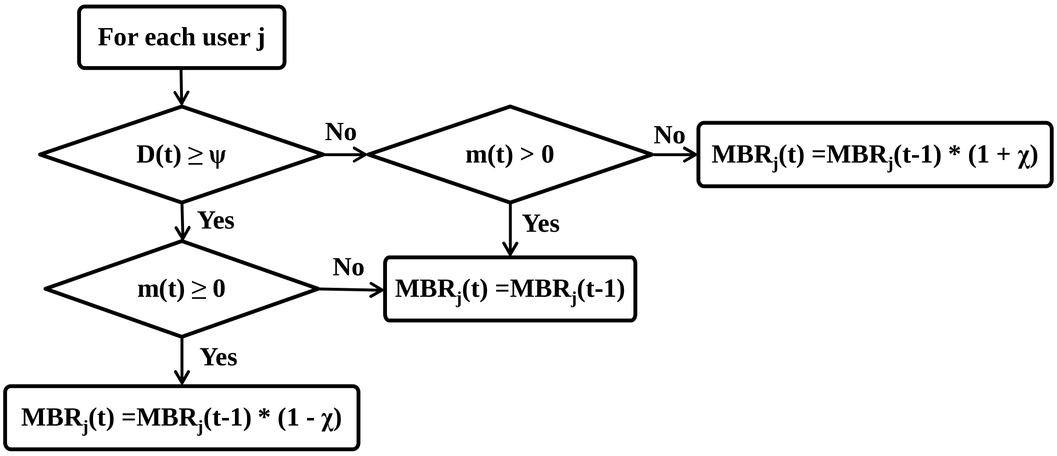


Figure 8.3: Flowchart for Dynamic MBR Update

There are 4 cases depending on the values of D(t) and m(t).

Case 1) D(t) ≥ ψ and m(t) ≥ 0: When D(t) exceeds the threshold ψ and is on a non-decreasing trend exemplified by m(t) ≥ 0, $MBR_j(t)$ is reduced by a fraction χ with the aim of reversing the trend of D(t) i.e.,

$$\mathrm{MBR}_j(t) = \mathrm{MBR}_j(t-1) * (1-\chi)$$

Case 2) D(t) ≥ ψ and m(t) < 0: In this case, D(t) is above the acceptable threshold ψ but has a decreasing trend as exemplified by m(t) < 0. Therefore the MBR is unchanged from its previous value i.e.,

$$\mathrm{MBR}_j(t) = \mathrm{MBR}_j(t-1)$$

Case 3) D(t) < ψ and m(t) > 0: In this case, D(t) is below the acceptable threshold ψ, but has an increasing trend. MBR is unchanged from its previous value as in Case 2.

Case 4) D(t) < ψ and m(t) ≤ 0: When D(t) is below the acceptable threshold ψ and has a non-increasing trend exemplified by m(t) ≤ 0, MBR is increased by a fraction χ with the aim of improving video quality i.e.,

$$\mathrm{MBR}_j(t) = \mathrm{MBR}_j(t-1) * (1+\chi)$$

8.4 Video QoE-Aware scheduling at the RAN

A media-buffer aware optimization framework is considered for multi-user resource allocation that constrains re-buffering probability for adaptive streaming users. A gradient based algorithm, called Re-buffering Aware Gradient Algorithm (RAGA), solves this optimization problem based only on periodic feedback of media buffer levels from streaming video clients. The scheduling priorities to users are continuously adjusted based not only on the absolute values of client media buffer levels but also on the rate of change of these buffer levels. Also since the optimization objective is not changed, this approach allows for flexibility in choosing custom optimization criterion including that of proportional fair and those based on video-quality metrics. Thus this approach is also friendly to non-video users that are served by the same base station.

In most cellular wireless networks, the UEs send to the BS periodic feedback regarding the quality of wireless link that they are experiencing in the form of Channel Quality Information (CQI). The CQI sent by the UEs is discretized, thus making the overall channel state "m" discrete. The eNodeB translates the CQI information into a peak rate vector $\boldsymbol{\mu}^{\mathbf{m}} = (\mu_1^m, \mu_2^m, ..., \mu_J^m)$, with μ_j^m representing the peak achievable rate by user j in channel state m. For every scheduling resource, the eNodeB has to make a decision as to which user to schedule in that resource. Scheduling the best user always would result in maximum cell throughput but may result in poor fairness. Scheduling resources in a round robin fashion might result in inability to take advantage of the wireless link quality information that is available. So, typical resource allocation algorithms in wireless networks seeks to optimize the average service rates **R** = (R_1, R_2, R_3, …R_J) to users such that a concave utility function H(**R**) is maximized subject to the capacity (resource) limits in the wireless scenario under consideration i.e.,

$$\begin{aligned} \text{Basic RA} : & \max \mathrm{H}(\mathbf{R}) \\ & \text{s.t. } \mathbf{R} \in \mathbf{V} \end{aligned}$$

where **V** represents the capacity region of the system. Utility functions of the sum form have attracted the most interest:

$$\mathrm{H}(\mathbf{R}) = \sum_j \mathrm{H}_j\left(\mathrm{R}_j\right)$$

where each $H_j(R_j)$ is a strictly concave continuously differentiable function defined for $R_j > 0$. The Proportional fair (PF) and Maximum Throughput (MT) scheduling algorithms are special cases of objective functions of this form with $H_j(R_j) = \log(R_j)$ and $H(R_j) = R_j$ respectively.

A video-aware optimization framework is considered for multi-user resource allocation in which client re-buffering is constrained. In order to avoid re-buffering at a video client, video segments need to be downloaded at a rate that is faster than the playback rate of the video segments. Let $T_j(s)$ be the duration of time taken by user j to download a video segment s and $\tau_j(s)$ be the media duration of the segment. Then the condition required for avoiding re-buffering is:

$$\mathrm{T}_j(\mathrm{s}) \leq \tau_j(\mathrm{s}) / (1+\delta) \quad \forall \ \mathrm{j, s}$$

where $\delta > 0$ is a small design parameter to account for variability in wireless network conditions. Segment download time $T_j(s)$ depends on the size of the video segment $S_j(s)$ and the data rates experienced by user j. $S_j(s)$ in turn depends on the video content and representation (adaptation) level that is chosen by the DASH client. DASH client choses the representation level for each video segment based on its state and its estimate of available link bandwidth. Based on all this, a Re-buffering Constrained Resource Allocation (RCRA) framework is devised as follows:

$$\begin{aligned} \text{RCRA} : \ & \max \mathrm{H}(\mathbf{R}) \\ & \text{s.t. } \mathbf{R} \in \mathbf{V} \\ & \mathrm{T}_j(\mathrm{s}) \leq \tau_j(\mathrm{s}) / (1+\delta) \quad \forall \ \mathrm{j, s} \end{aligned}$$

The key difference is the additional constraints related to re-buffering. Unlike prior approaches, this framework closely relates the buffer evolution at DASH clients to resource allocation at the eNodeB since $T_j(s)$ is related to the average service rate R_j obtained by user j. By intelligent resource allocation, the eNodeB can help reduce re-buffering in video clients. This approach requires some feedback from DASH clients in order to enforce the re-buffering constraints.

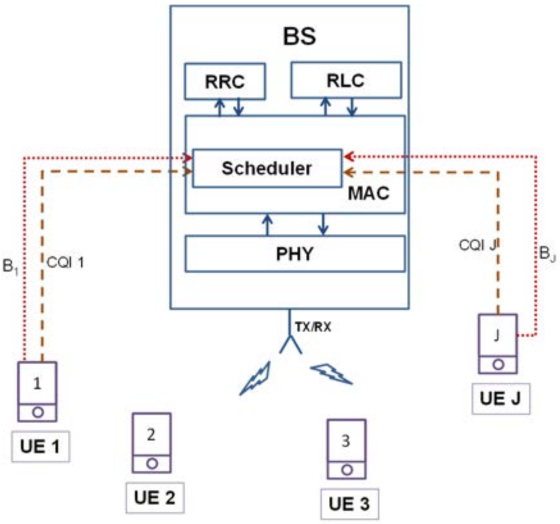


Figure 8.4: Buffer Level Feedback based Scheduling

The buffer-aware resource allocation framework is shown in Figure 8.4. In addition to CQI feedback as is standard in 3GPP cellular networks, each adaptive streaming user also feeds back its media playback buffer level periodically to the BS scheduler. This can be directly done over the radio access network or indirectly through the video server.

Scheduling algorithms for multi-user wireless networks need to make decisions during every scheduling time slot (resource) t in such a way it leads to the long-term optimal solution. Note that the scheduling time slot is typically at much finer granularity than a (video) frameslot in the LTE environment. The Re-buffering Aware Gradient Algorithm (RAGA) uses a token-based mechanism to enforce the re-buffering constraints. The scheduling decision of RAGA in scheduling time slot t when the channel state is m(t) can be summarized as follows:

$$\text{RAGA}: \ j = \arg\max_{j \in N} \left[e^{a_j(t) W_j(t)} \nabla H\left(R_j(t)\right).\mu_j^{m(t)} \right]$$

where $R_j(t)$ is the current moving average service rate estimate for user j. It is updated every scheduling time slot as in the PF scheduling algorithm i.e.,

$$R_j(t+1) = (1-\beta) R_j(t) + \beta \mu_j(t)$$

where $\beta > 0$ is a small parameter that determines the time scale of averaging and $\mu_j(t)$ is the service rate of user j in time slot t. $\mu_j(t) = \mu_j^{m(t)}$ if user j was scheduled in time slot t and $\mu_j(t) = 0$ otherwise. $W_j(t)$ is a video-aware user token parameter and $a_j(t)$ is a video-aware user time-scale parameter, both of which are updated based on periodic media buffer level feedback by the clients to the eNodeB. These parameters hold the key to enforcing re-buffering constraints at the eNodeB. For simplicity, it is assumed that such client media buffer level feedback is available only at the granularity of a frameslot. Therefore the user-token parameter and user-time scale parameter are constant within a frameslot i.e.,

$$W_j(t) = W_j^i \quad \text{for} \quad i\tau \le t < (i+1)\tau$$
$$a_j(t) = a_j^i \quad \text{for} \quad i\tau \le t < (i+1)\tau$$

Let B_j^i represent the buffer status feedback in the frameslot i in units of media time duration. The difference between buffer levels from frameslot (i-1) to frameslot i is given by:

$$B_j^{i,diff} = B_j^i - B_j^{i-1}$$

A positive value for $B_j^{i,diff}$ indicates an effective increase in the media buffer size in the previous reporting duration and a negative value indicates a decrease in media buffer size. Note that this difference depend on frame playback and download processes at the DASH client. To avoid re-buffering, the rate of change in the client media buffer level needs to be greater than a certain positive threshold i.e.,

$$(B_j^{i,diff} / \tau) > \delta\ ,\ \delta > 0$$

The media buffer aware user token parameter is updated every frameslot as follows:

$$W_j^i = \mathrm{Max}(W_j^{i-1} + (\delta\tau - B_j^{i,diff}), 0)$$

The intuitive interpretation of this result is that if rate of media buffer change for a certain user is below the threshold, the token parameter is incremented by an amount $(\delta\tau - B_j^{diff})$ that reflects the relative penalty for having a buffer change rate below threshold. This increases its relative scheduling priority compared to other users whose media buffer change rate is higher. Similarly, when the rate of buffer change is above the threshold, the user-token parameter is decreased to offset any previous increase in scheduling priority. W_j^i is not reduced below zero, reflecting the fact that all users that have a consistent buffer rate change greater than the threshold have scheduling priorities as per standard proportional fair scheduler.

The video-aware parameter a_j^i determines the time-scale over which re-buffering constraints are enforced for adaptive streaming users. A larger vale of a_j^i implies greater urgency in enforcing the re-buffering constraints for user j. The values of a_j^i can be set to reflect this relative urgency for different users. Therefore a_j^i is set based on the media buffer level of user j in frameslot i as follows:

$$a_j^i = 1 + \phi * \max\left(\frac{B_{thresh}^{Steady} - B_j^i}{B_{thresh}^{Steady}}, 0\right)$$

where ϕ is a scaling constant, B_j^i is the current buffer level in seconds for user j, and B_{thresh}^{Steady} is the threshold for the steady state operation of the DASH video client. If the buffer level B_j^i for user j is above the threshold, then a_j^i = 1 and if it is below the threshold, a_j^i scales to give relative higher priorities to users with lower buffer levels. This scaling of priorities based on absolute user buffer levels improves the convergence of the algorithm. Note that parameter $W_j(t)$ is updated based on rate of media buffer level change while parameter $a_j(t)$ is updated based on buffer levels themselves. Such an approach provides a continuous adaptation of user scheduling priorities based on media buffer level feedback (unlike an emergency response type response) and reduces the re-buffering percentage of users without significantly impacting video quality.

From a RAN efficiency perspective, the following observations can be noted on the use of the buffer-level feedback:

- In active state (e.g. at segment download activities) the UE can report buffer status, send CQI reports etc. without specific additional burden.
- In DRX and especially in idle state (e.g. after inactivity time expiry in-between segment downloads), the UE really wants to avoid any modem activity. Even sending a tiny status report will wake up modem subsystem and initiate new transition from idle to active, and then waiting again to move back. This will cause both signaling overhead and large power consumption impact in client.

So when a download activity of segments is finalized the client should not need to send anything to the scheduler, until it is time for another download activity.

8.5 Relationship to SAND

SAND provides the application layer framework to realize the streaming enhancements described in clauses 8.3 and 8.4. In particular, the VAC described in clause 8.3 can be realized by the DANE functionality defined by SAND, where DASH clients can indicate their desired bandwidth levels through the use of the SAND status message *SharedResourceAllocation* and also can report QoE metrics. In addition, when VAC determines the resource allocation across the DASH clients, it can rely on its DANE functionality to inform the DASH clients about their resource assignments and throughput / QoS, which can be achieved by the SAND PER message *SharedResourceAssignment*, *Throughput* and *QoSInformation*. Furthermore, provided that the QoE metrics reports enabled by SAND are also made accessible to the scheduler at the eNodeB, SAND can be an enabler for the QoE-aware multiuser scheduling algorithms described in clause 8.4.

8.6 Performance evaluation results

Dynamic MBR signaling (MBR-Sig) algorithm with RAGA [12], [15], Proportional Fair (PF) [13], Proportional fair With Barrier for Frames (PFBF) [12], [15], GMR (Gradient with Min rate) [11], and Congestion Aware (CA) [14] algorithms. For GMR, the minimum rate is set for each video user to the rate of the lowest representation level of the user's video. CA algorithm is an end-to-end distributed resource management algorithm based on congestion-signaling.

Re-buffering percentage is computed for each user as the fraction of total time the user spends in re-buffering state. Figure 8.5 plots the CDF of the % of users with re-buffering less than 2% as the load is varied. For a given load, PF and GMR have lowest % of users with re-buffering percentage less than 2%. PFBF performs better than PF and GMR. RAGA performs still better. CA algorithm has the highest number of users with low re-buffering because of its very conservative approach (very low video quality). Our proposed MBR-Sig algorithm comes close to CA in terms of number of users with low re-buffering. Video quality is measured in terms of PSNR. Figure 8.6. compares the average video quality in terms of PSNR for the various schemes. PF has the best quality at expense of huge re-buffering and CA has lowest video quality although it has low re-buffering. RAGA performs in between PF and CA in terms of quality as well as re-buffering .MBR-Sig obtains a video quality close to RAGA, but it obtains a re-buffering percent close to CA.

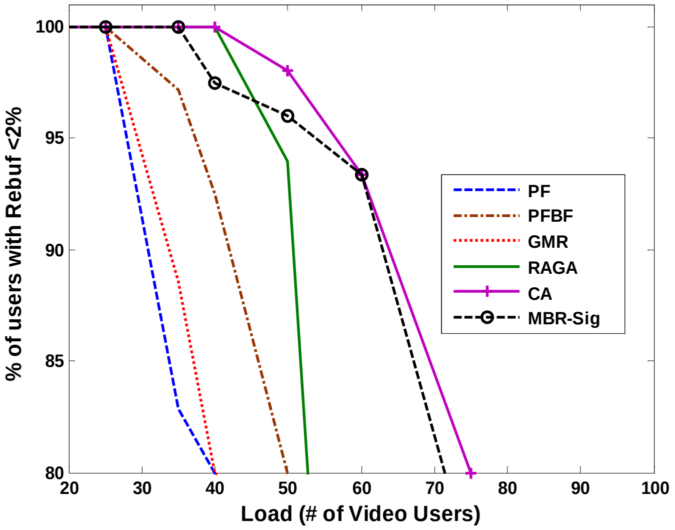


Figure 8.5: CDF of Re-buffering Outage

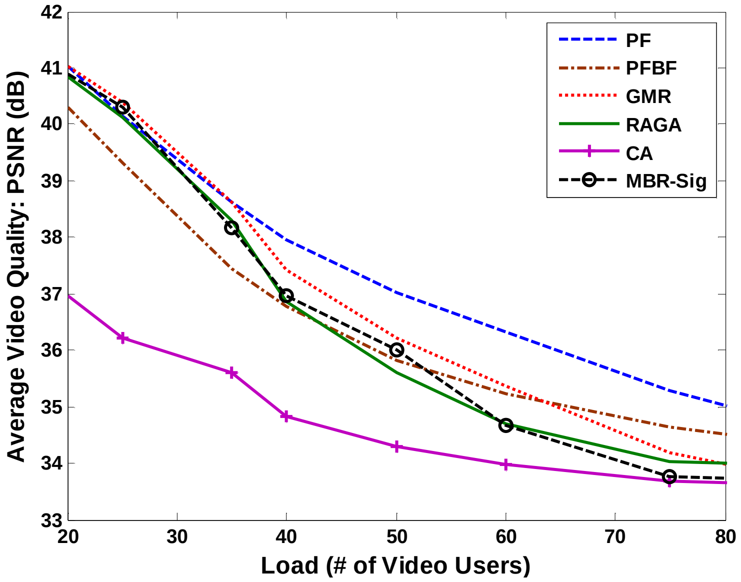


Figure 8.6: Average Video Quality Comparison

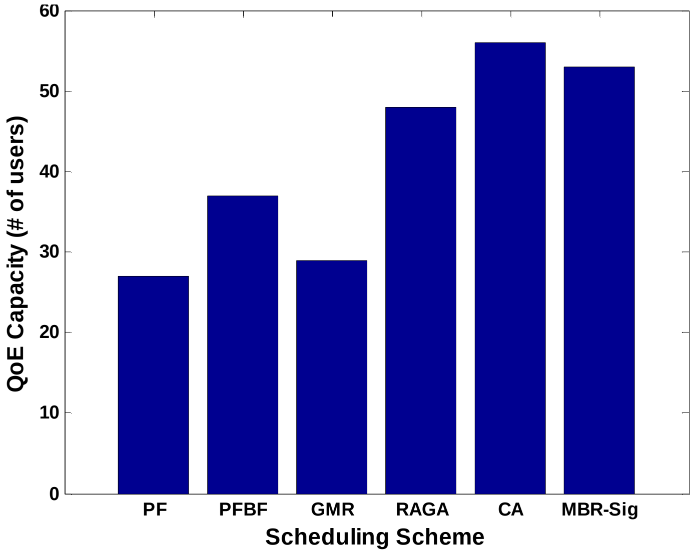


Figure 8.7: QoE-outage based capacity

Figure 8.7 compares the QoE-outage based capacity of various schemes. Observe that PF has the lowest capacity, and CA obtains the highest capacity. MBR-Sig does better than RAGA in terms of capacity and comes close to CA. Thus MBR-Sig approach is better than other schemes because it achieves the best balance of quality and capacity, with video-aware intelligence at network core.

The following evaluation results compare the performance of the resource allocation algorithm RAGA with standard Proportional Fair (PF), Proportional fair With Barrier for Frames (PFBF), and GMR (Gradient with Min rate) algorithms. For GMR, the minimum rate is set for each video user to the rate of the lowest representation level of the user's video.

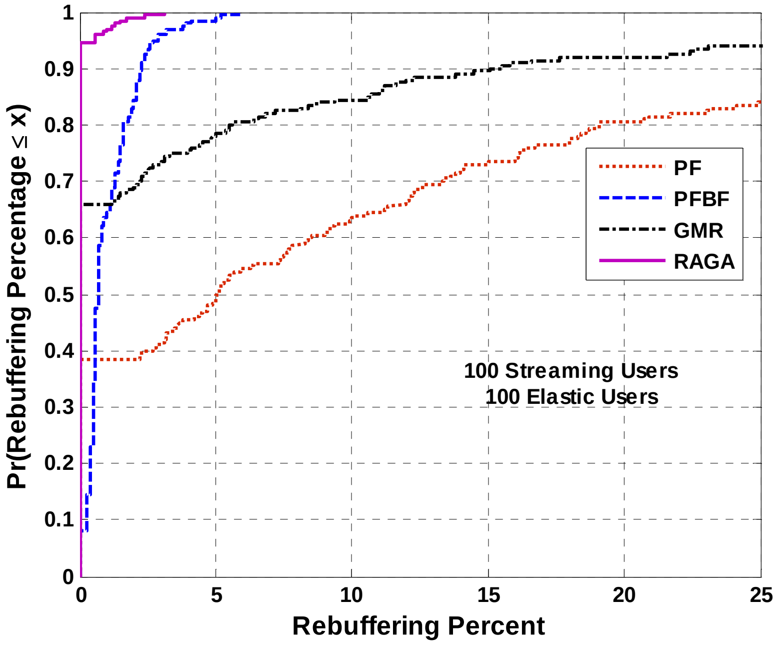


Figure 8.8: CDF of Re-buffering Percentage

Figure 8.8 plots the CDF of re-buffering percentage when there are 100 streaming and 100 elastic users in the system. It is observed that RAGA has the lowest re-buffering percentage among all the schemes across all the users. It has reduced number of users experiencing and smaller amount of re-buffering experienced by the users. PF has the worst re-buffering performance. GMR is better than PF, but it still lags behind due to lack of dynamic cooperation with the video clients. PFBF performs better than GMR in terms of peak re-buffering percentage but lags behind both PF and GMR in terms of the number of users experiencing re-buffering. This is because PFBF reacts to low-buffer in an emergency fashion and inadvertently penalizes good users to satisfy users with low buffer levels. On the other hand RAGA continually adjust the scheduling priorities of the users based on the rate of change of media buffer levels, thus avoiding emergency situations in the first place.

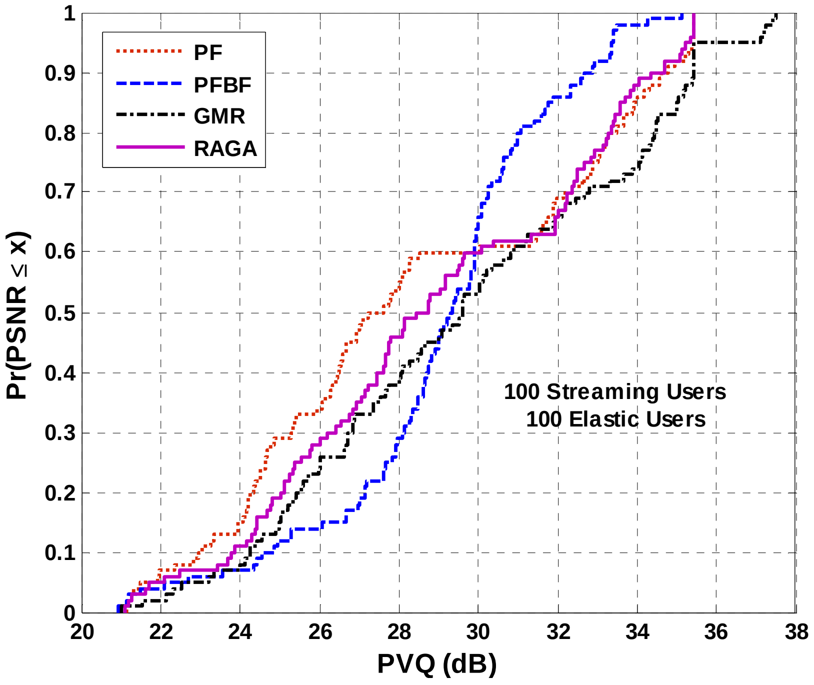


Figure 8.9: CDF of Perceived Video Quality (PVQ)

Figure 8.19 compares CDFs of Perceived Video Quality (PVQ) of video clients with the same loading. PVQ is computed as the difference between the mean and standard deviation of PSNR. Only played out video frames are considered in the computation of PVQ. The PVQ using RAGA is better than PF scheduling for all users. GMR appears to have only marginally better PVQ than RAGA but this is at a huge cost in terms re-buffering percentages. PFBF has better PVQ than all schemes for some users and worse than all schemes for others because of emergency response to low buffer levels and cyclic effect thereof. RAGA has the most balanced PVQ among all the schemes and also the lowest re-buffering percentages.

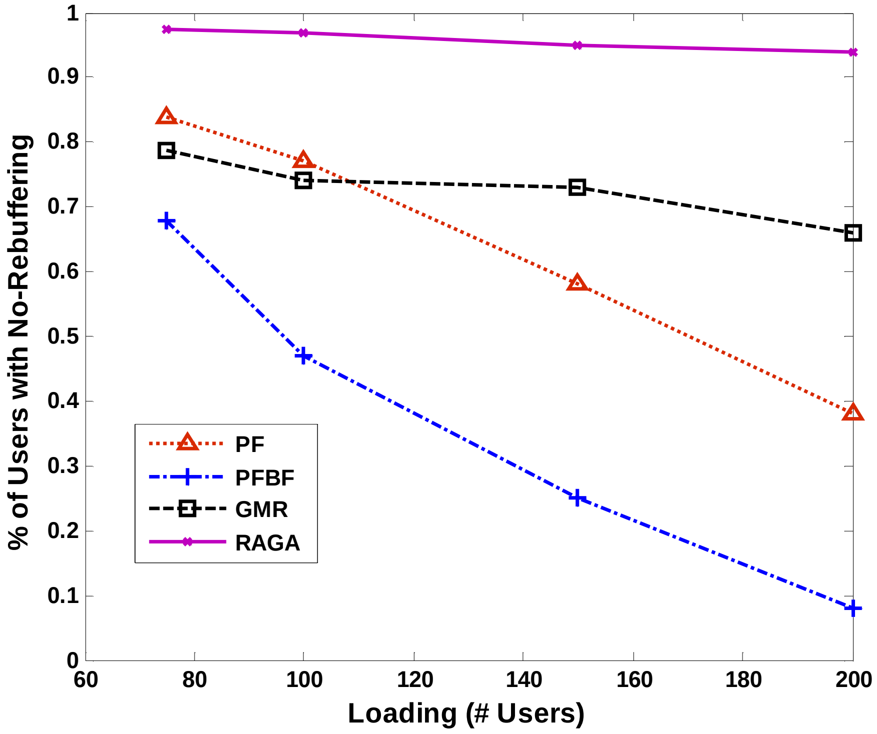


Figure 8.10: % of Users with No Re-buffering vs. Loading

Figures 8.10 and 8.11 plot respectively the percentage of users with no re-buffering and the peak re-buffering percentage as the user loading varied. RAGA performs the best compared to other algorithms on both these accounts. PFBF performs well in terms of peak re-buffering percentage compared to PF and GMR but loses out in terms of number of users experiencing re-buffering at high loads. At high loads, GMR performs better than PF, but the % of user experiencing re-buffering and peak re-buffering are significantly higher than RAGA. Also RAGA has the lowest slope in both cases indicating the stability it provides to DASH rate adaptation by constraining the re-buffering percentage.

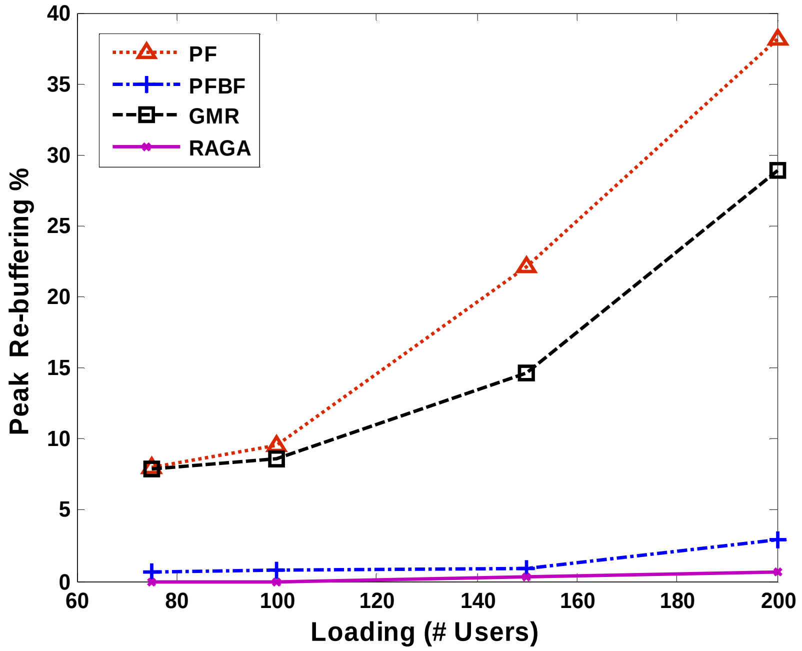


Figure 8.11: Peak Re-buffering percent

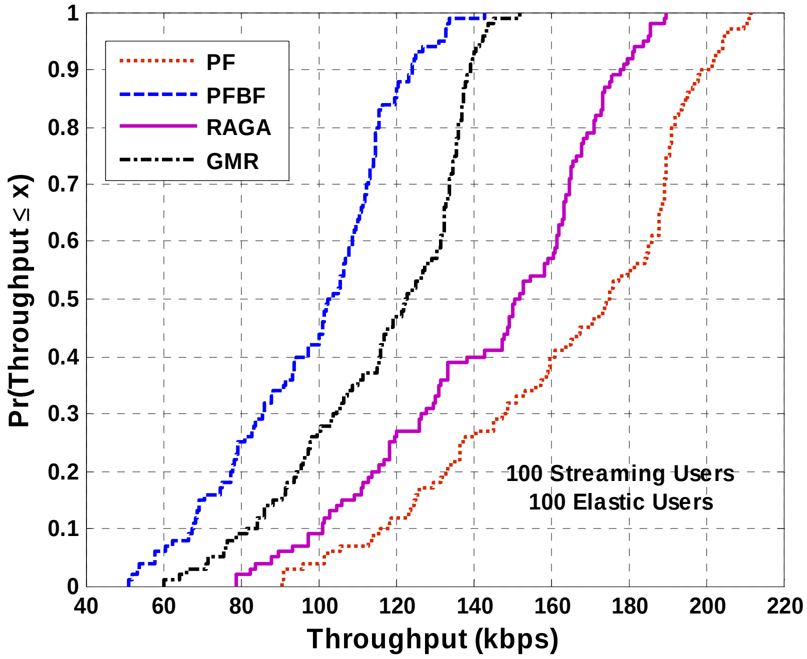


Figure 8.12: Elastic User Throughput

Figure 8.12 compares the CDFs of throughputs of elastic users when using the different scheduling algorithms. PF provides the highest throughput to elastic users as it treats both elastic and video streaming users alike. Among QoE-aware scheduling algorithms, RAGA is the closest to PF, indicating that its friendliness to elastic users.

9 Cross-Origin Resource Sharing (CORS)

9.1 Introduction

HTTP requests using Javascript are traditionally restricted by the Same Origin Policy, which implies that these requests will have the same domain name and port. Developers have tried to bypass this limitation by using different tricks such as proxying. However, these are usually not efficient and pose a security threat.

Cross-Origin Resource Sharing (CORS) is a W3C specification [16] that was developed to allow client-side cross-domain requests from the browser. In CORS, a response can include an "Access-Control-Allow-Origin" header with the origin of where the request originated from. The user agent verifies that the value of that header and the origin of where the response originated match. A preflight request, usually an OPTIONS request, may need to be sent in advance of the actual request to verify that the server is prepared to accept a cross-origin request. The server discovers that the HTTP request is actually a cross-origin request by checking the Origin header field that was inserted automatically by the browser agent. Additionally, authentication of the request may be requested by setting the value of Access-Control-Allow-Credentials to true. In such case, an identifier of the application, e.g. in form of a cookie is required to allow access to the cross-origin resource.

The procedure for gaining access to content from a different origin is described in the following message passing diagram:

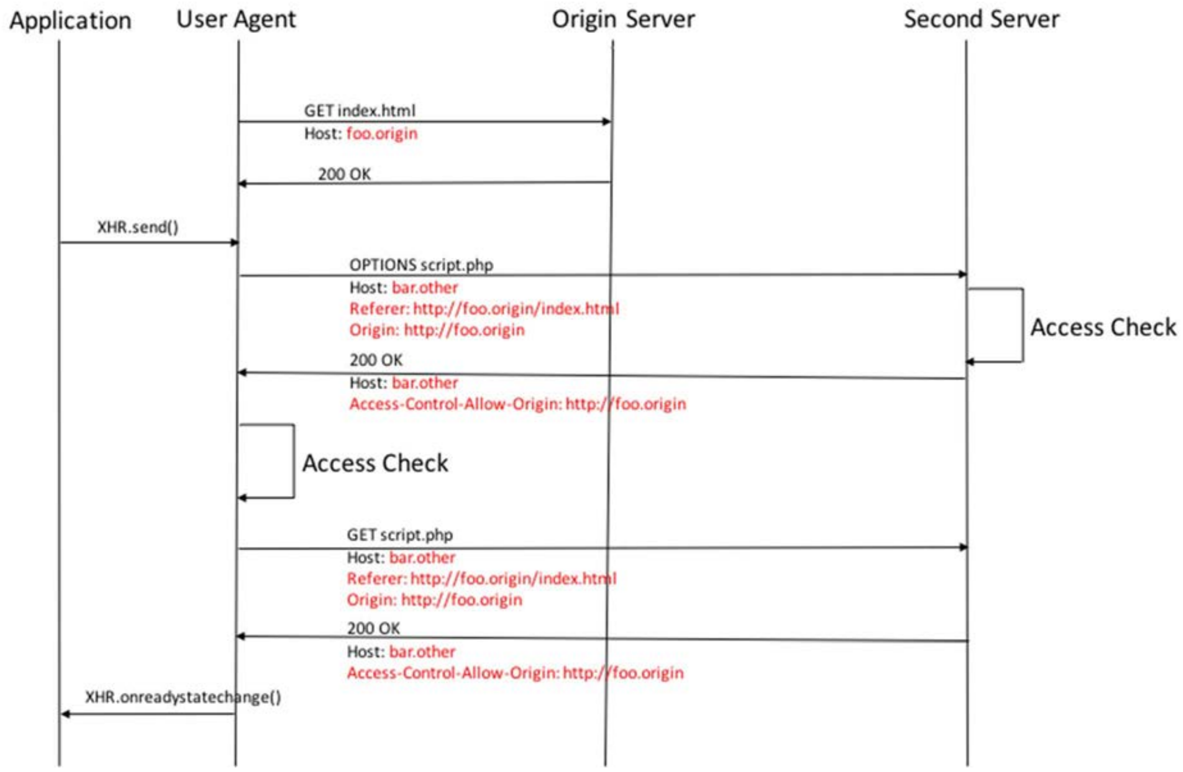


Figure 9.1

In addition, a restriction on mixed content is applied by user agents. Mixed content denotes the use case where secure requests are made from an insecure source or vice versa.

9.2 CORS Considerations for SAND

SAND may be used in the context of 3GPP for different purposes. A DANE may sit in the operator's network and assist the client to achieve an optimal streaming experience for both the client and also in terms of network resource efficiency. It can also be used in the context of MBMS client to DASH application communication.

SAND defines a main communication channel using connection triggers, through insertion of an HTTP header field in the response. Upon detecting this header field, the client will connect to the provided URL to fetch the SAND message. Alternatively, a WebSocket channel may be indicated by the MPD, in which case the client connects to a global DANE server to receive SAND messages. Both methods have their pros and cons. However, both methods are subject to the CORS and mixed content restrictions.

In particular, the URL of the DANE's SAND message that is provided in the "MPEG-DASH-SAND" message or as part of the WebSocket URL will allow the origin of this request. It also has to match the same security environment (i.e. usage or not usage of TLS consistently).

It is therefore recommended that the DANE servers will support CORS and allow the origin of the MPD in all cases. DANE's will also support the same security configuration to avoid the mixed content problem.

10 Connections to RAN Study on Context-Aware service delivery

A study on context aware service delivery in RAN for LTE is presented in TR 36.933 [19] . This study addresses RAN aspects of video streaming enhancements such as network-assisted DASH and video-aware scheduling. As such, the content presented in TR 36.933 is closely related to SAND, and particularly to the use cases on consistent QoE/QoS (presented in clauses 6.2 and 8), proxy caching (presented in clause 6.3) and network assistance (presented in clause 6.4). The RAN study presented in [19] analyses the potential impact to architecture, protocol, and signalling to support RAN based local cached delivery, local breakout; and support RAN optimizations based on context awareness.

In clause 4 of [19], the following issues in regards to video streaming are observed:

- Backhaul long latency, as described in clause 4.1 of [19].
- TCP E2E delay with throughput decreasing, as described in clause 4.2 of [19].
- Video transmission issues, including empty buffer, inaccurate throughput prediction for DASH and long video delay, as described in clause 4.3 of [19].

SAND can be a potential solution to address some of these issues (including potential solutions in clauses 6.2.4, 6.3.4, and 6.4.4 of TR 26.957), particularly the video transmission issue described in clause 4.3 of [19]. Such a potential solution where SAND is relevant is also presented in clause 5.4.2.2 of [19]. Moreover, Annex A of [19] documents performance enhancements from video playout buffer aware scheduling, as aligned with the SAND-based streaming enhancements via QoE-aware resource allocation, as described in clause 8 of TR 26.957.

11 Conclusions and Recommendations

Results of the studies for the SAND use cases on Content Provider Optimized Zero Rating, Consistent QoE/QoS, Proxy Caching and Network Assistance for DASH are presented respectively in clauses 6.1, 6.2, 6.3 and 6.4, each with a corresponding documentation of gap analysis with respect to existing 3GPP technologies and potential solutions including relevant SAND functionality.

Based on the gap analysis documented in the present document, the following conclusions can be drawn toward normative specification work on MPEG SAND:

- From an architectural standpoint, certain features of SAND can be enabled in the 3GPP environment for both managed and OTT streaming services based on the architectural options presented in Clause 5.
- Based on the use cases considered in Clause 6, the following new SAND functionality is recommended for 3GPP DASH in TS 26.247:
    - To realize partial representation caching described in clause 6.3, SAND can be used to inform 3GPP DASH clients about partially cached representations, e.g., via use of the PER messages *ResourceStatus* and *DaneResourceStatus*. Moreover, toward realizing next segment caching described in clause 6., SAND can be used by 3GPP DASH clients to inform the network (i.e., DANE) anticipated DASH segments, acceptable alternative content, etc. leading to next segment caching, e.g., via use of the status messages *AnticipatedRequests*, *AcceptedAlternatives*, and *NextAlternatives.*
    - To realize the streaming QoE enhancements described by the use case in clause 6.2, the following SAND functionality is recommended: 3GPP DASH clients can indicate the available media rates for a particular DASH content item through the use of the SAND status message *SharedResourceAllocation* and also can report QoE metrics. In addition, DANE functionality can inform the 3GPP DASH clients about their resource assignments and throughput / QoS of media layer, which can be achieved by the SAND PER messages *SharedResourceAssignment* and *QoSInformation*.
    - To realize the Network Assistance functionality described by the use case in clause 6.4, the following SAND functionality is recommended: 3GPP DASH clients can indicate the available media rates for a particular DASH content item through the use of the SAND status message *SharedResourceAllocation* (same as for QoE enhancements); DANE functionality can inform the 3GPP DASH clients of the highest recommended media rate for a particular DASH content item using the SAND message *SharedResourceAssignment;* 3GP-DASH clients may provide information about the current client buffer level to the network using the SAND

message *BufferLevel*; and DANE functionality may request the 3GP-DASH client to limit the media rate of the requested segments using the SAND message *QoSInformation* when the network applies quicker buffer filling strategies.

Based on these findings and evaluations, it is recommended to start normative work to specify the relevant solutions within the PSS architecture.

Annex A:
Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2016-09		SP-160603			Presented to TSG SA#73 (for information)		1.0.0
2016-12		SP-160775			Presented to TSG SA#74 (for approval)	1.0.0	2.0.0
2016-12					Approved at TSG SA#74 for Release 14	2.0.0	14.0.0
2017-03	75	SP-170035	0001	1	Corrections to SAND TR	14.0.0	14.1.0
2018-06	80				Vesrion for Release 15		15.0.0

History

Document history		
V15.0.0	July 2018	Publication