Adaptive media streaming over IP multicast

Implementation guidelines and worked examples

DVB Document A181 Rev.2

June 2023

DVB®

Contents

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 Technical Committee|ETSI Project|<other>} <long techbody> (<short techbody>).

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.

Introduction

DVB BlueBook A176 [i.1] defines a functional architecture for delivering live and on-demand adaptive streaming media to a large audience in a highly efficient and scalable way by leveraging IP multicast for media object delivery.

The specification itself does not prescribe a single system design, giving a large degree of flexibility in implementation options, which this document explores with some informative worked examples to aid the development of individual implementations.

1 Scope

The present document provides implementation guidelines for the system specified in DVB BlueBook A176 [i.1] and describes some example implementations of it. These examples are non-exhaustive, and as such do not detail every possible combination of configurations that are possible with the technical specification.

The baseline procedures and illustrative call flows for such a system are documented in annex A.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] DVB BlueBook A176: "Digital Video Broadcasting (DVB); Adaptive media streaming over IP multicast", third edition.

[i.2] ETSI TS 103 285: "Digital Video Broadcasting (DVB); MPEG-DASH Profile for Transport of ISO BMFF Based DVB Services over IP Based Networks", v1.3.1 February 2020.

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

[i.4] Broadband Forum TR-069: "CPE WAN Management Protocol".

[i.5] Broadband Forum TR-369: "User Services Platform (USP)".

[i.6] DASH Industry Forum: "DASH-IF Live Media Ingest Protocol" technical specification. https://dashif-documents.azurewebsites.net/Ingest/master/DASH-IF-Ingest.html.

[i.7] IETF RFC 950: “Internet Standard Subnetting Procedure”, August 1985.

[i.8] IETF RFC 4291: “IP Version 6 Addressing Architecture”, February 2006.

[i.9] Fielding, Roy Thomas: "Chapter 5: Representational State Transfer (REST)". from "Architectural Styles and the Design of Network-based Software Architectures" (Ph.D. dissertation), University of California, Irvine, 2000.

[i.10] IETF RFC 6266: "Use of the Content-Disposition Header Field in the Hypertext Transfer Protocol (HTTP)", June 2011.

[i.11] ETSI TS 103 770: "Digital Video Broadcasting (DVB); Service Discovery and Programme Metadata for DVB-I".

[i.12] DVB BlueBook A180: "Digital Video Broadcasting (DVB); Native IP Broadcasting".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms defined in DVB BlueBook A176 [i.1] apply.

3.2 Symbols

For the purposes of the present document, the symbols defined in DVB BlueBook A176 [i.1] apply.

3.3 Abbreviations

For the purposes of the present document, the abbreviations defined in DVB BlueBook A176 [i.1] apply.

3.4 Notational convention for diagrams

The diagrams in the present document conform to the following conventions adopted from DVB BlueBook A176 [i.1]:

- Interactions with a single solid line are in-scope interactions fully specified in [i.1], and any associated clauses that define that interaction are indicated in the text that follows the diagram.

- Interactions with a dashed line are defined by the reference architecture in clause 5 of [i.1] but are declared out of scope by it, and are therefore implementation-specific.

- Interactions with a double solid line are not defined by [i.1] and are entirely implementation-specific.

4 Implementation guidelines

4.1 Multicast server implementation guidelines

4.1.1 Using macro expansion in the multicast server configuration to generate multiple in-band multicast gateway configurations

The macro expansion mechanism specified in clause 10.2.5.2 of [i.1] allows a *Multicast server* to generate multiple distinct multicast gateway configuration instance documents from a single multicast server configuration instance document.

In one example usage of this feature a *Multicast server* under the control of a Content Provider distributes a common set of multicast transport sessions to multiple different Service Provider networks. Each Service Provider network has a distinct sub-population of deployed *Multicast gateway* instances all configured using the in-band configuration method described in clause 10.1.2 of [i.1]. Each Service Provider network has deployed its own *Content hosting* function as part of a tiered Content Delivery Network, to provide a better quality of service by hosting repair objects closer to the consumer.

Using the macro expansion mechanism allows the **`UnicastRepairParameters/BaseURL`** element(s) carried in the different multicast gateway configurations provided over reference point **M** to refer to the local *Content hosting* function appropriate for each sub-population of *Multicast gateway* instances.

To achieve this outcome, the multicast server configuration instance document declares one multicast gateway configuration transport session for each of the target Service Provider networks, including appropriate macro expansions for each one. As a result, the *Multicast server* creates a set of multicast gateway configuration instance documents by expanding the macros, and includes the correct resulting multicast transport object in each multicast gateway configuration transport session that it generates.

4.1.2 Analysing generated multicast transport objects

The in-band configuration method described in clause 10.1.2 of [i.1] specifies that the *Multicast server* has the capability to generate multicast gateway configuration transport objects for transmission at reference point **M** rather than fetching or receiving them directly via reference point **C_{MS}**. Using the macro expansion mechanism specified in clause 10.2.5.2 of [i.1], the *Multicast server* can potentially create multiple different multicast gateway configuration instance documents from a single multicast server configuration.

To ensure that these generated multicast gateway configuration documents conform to the system operator's desired configuration, the *Multicast server* may allow the multicast transport objects it creates to be analysed and debugged before or after the updated multicast server configuration is published to the live *Multicast server* instance. This could take multiple forms. For example:

1. An (undefined) logical function could subscribe to the output of the *Multicast server* at reference point **M** and observe the output of the *Multicast server*. This could then allow the system operator to view the output of both the live output and potentially any testing multicast interfaces exposed by the *Multicast server* as it runs.

2. The *Multicast server* could expose a RESTful resource [i.9] that allows the system operator to interrogate the set of multicast gateway configuration instance document(s) currently being transmitted at reference point **M**.

   In the case where one multicast server configuration generates multiple multicast gateway configuration instance documents, these could be returned as a single HTTP multipart response using the *Content-Disposition* header specified in [i.10].

3. The *Multicast server* could expose a RESTful resource [i.9] that allows the system operator to upload a test multicast server configuration, returning the set of multicast gateway configuration instance document(s) that the *Multicast server* would create from that test multicast server configuration.

   An HTTP multipart response could be used in cases where multiple multicast gateway configuration instance documents are generated, as above.

4. The *Multicast server* could expose a full web application that allows an operator to upload a test multicast server configuration and inspect the resulting multicast gateway configuration instance document(s) in a web browser. This web application could also be extended to have some form of authoring and live debugging interface to aid analysis by the operator.

4.1.3 Dynamic service component transmission

Transmission of multicast packets by the *Multicast server* at reference point **M** may be (de)activated by the *Provisioning* function for individual multicast transport sessions or for all multicast transport sessions in the scope of a parent multicast session using the reference point **C_{MS}** procedures specified in clause 10.4.3 of [i.1].

The decision to (de)activate one or more multicast transport sessions is taken by the *Provisioning* function based, for example, on:

1. Current audience size, using reporting information provided to the *Service reporting capture* subfunction of *Provisioning* function at reference point **R_S** by a sample of deployed *Multicast gateway* instances, as specified in clause 11 of [i.1].

2. Current available transmission bit rate budget, using feedback collated by the *Provisioning* function from the access *Network* by means beyond the scope of [i.1].

The implications on the operation of the *Multicast gateway* are described in clause 4.3.1.

4.2 Multicast rendezvous service implementation guidelines

4.2.1 Operation in absence of multicast transport session

When the *Content playback* request for a presentation manifest to the *Multicast rendezvous service* at reference point **B** includes the Ori query parameter (see clause 7.5.1 of [i.1]) but the *Multicast rendezvous service* is aware (through private means) that the requested content is not available from a multicast transport session, the *Multicast rendezvous service* may redirect the *Content playback* function's presentation manifest request directly to the *Content hosting* function.

In this case, the redirection URL is built by replacing the authority in the request URL with the authority signalled in the Ori query parameter.

The *Content playback* function then consumes a regular unicast MPEG-DASH presentation with no further interaction with the *Multicast gateway*.

4.3 Multicast gateway implementation guidelines

4.3.1 Operation in absence of multicast transport session

The *Multicast gateway* is expected to switch to unicast retrieval of media objects if a multicast transport session that is currently being consumed is deactivated as described in clause 4.1.3. In the case of unidirectional operation, where no unicast path exists, the *Multicast gateway* may use an implementation-dependent mechanism, for example to maintain continuous playback.

5 Fully worked examples

5.0 Introduction

DVB BlueBook A176 [i.1] defines the functional architecture of a system for delivering linear content in a scalable manner over multicast IP. It defines several logical functions and the reference points which carry interactions between them. These logical functions may have different operating modes which affects how they interoperate with other logical functions over the named reference points.

The system specified in [i.1] is designed to be modular and flexible: not all logical functions described are required to build a compliant deployment. For example, in a unidirectional deployment where unicast repair is not possible, the *Multicast server* may be configured in push-based ingest mode, in which case the *Content hosting* function is omitted from the system since it serves no purpose. It is useful when describing deployments to group the logical function into logical domains. This document defines the following logical domains:

- Functions deployed in the **content provider** domain are under the control of the owner of the linear service that is being delivered by a multicast session. This may be a broadcaster or an over-the-top media streaming service.

- The **third-party** domain is that of a content delivery network. The *Content hosting* function may be embedded within a service provider's domain, but the function is still under the logical ownership of the third-party content delivery network.

- Functions deployed in the **service provider** domain are under the control of an Internet Service Provider (or similar entity) and provide the connectivity for a consumer to receive the multicast session. The service provider may receive a raw stream from a content provider and then encode, package and distribute the media, in which case it would own functions such as the *Content preparation* function and not the content provider.

- The **customer premises** domain encompasses any equipment under the control of the consumer of the content. This may include the terminal device such as a set-top box, mobile phone or connected television set, and the home gateway device referred to in clause 6 of [i.1]. The terminal device or home gateway may be provided by a content provider or service provider, or they may be purchased off the shelf by the consumer.

Table 5.0-1 below shows a matrix of all the logical functions defined in clause 5.3 of [i.1] against the applicable deployment domains described in the present document:

Table 5.0-1: Matrix of logical functions mapped to applicable domains

Logical function	Content Provider domain	Third-party domain	Service Provider domain	Customer premises domain
Content preparation	✓		✓	
Content hosting	✓	✓	✓	
Multicast server	✓		✓	
Unicast repair service	✓		✓	✓
Multicast gateway			✓	✓
Provisioning	✓		✓	
Content provider control	✓		✓	
Content playback				✓
Multicast rendezvous service	✓		✓	✓
DRM licence management	✓		✓	
Application				✓
Service directory	✓	✓	✓	✓

The present document describes four possible implementation options:

- Clause 5.1 describes a bidirectional deployment incorporating unicast repair, where a *Multicast server* deployed in the service provider's network pulls content from a *Content hosting* function also deployed in the service provider's network, and a *Multicast gateway* is deployed in the home gateway device within the consumer premises.

- Clause 5.2 describes a bidirectional deployment incorporating unicast repair, where a *Multicast server* deployed in the service provider's network pulls content from a *Content hosting* function hosted by a third-party commercial content delivery service provider, and a *Multicast gateway* is deployed in the service provider's network edge.

- Clause 5.3 describes a bidirectional deployment incorporating unicast repair, where a content owner's *Content preparation* function pushes content into both a *Multicast server* and *Content hosting* function hosted in the service provider's network, and a *Multicast gateway* deployed in the service provider's network edge is co-located with the *Multicast rendezvous service*.

- Clause 5.4 describes a deployment with two separate operator networks: one fixed line bidirectional deployment incorporating unicast repair, and another unidirectional satellite-based deployment with no unicast repair capability. In both delivery networks, the content provider owns the *Content preparation* and *Multicast server* functions, as well as an origin *Content hosting* function. A third-party commercial operator provides a content delivery network to the fixed-line operator to support unicast repair with network edge caching, fed by the content provider's *Content hosting* (origin) function. In the fixed line delivery path, the *Multicast gateway* is deployed in the home gateway device within the consumer premises. In the satellite delivery network path, the *Multicast gateway* is deployed within the satellite set-top box (STB) alongside the *Content playback* and *Application* functions.

Table 5.0-2 below shows a matrix of how the four clauses described in this document map to the identified possible configuration options, including the availability of reference point **A** for unicast repair, whether the multicast gateway configuration is delivered over the control plane reference point $\mathbf{C}_{\mathbf{MR}}$ or carried in a multicast gateway configuration transport session over reference point **M** as described in clause 8.3.5 of [i.1], the use of push or pull ingest mode as described in clauses 8.3.1 and 8.3.2 of [i.1] respectively, and finally if only media objects or both media objects and presentation manifests are sent over reference point **M**.

Table 5.0-2: Matrix of object delivery options

Availability of unicast repair (bidirectional / unidirectional)	Multicast gateway configuration delivery method	Multicast server ingest mode	Delivery over IP multicast		Clause 5.1	Clause 5.2	Clause 5.3	Clause 5.4
			Manifest	Media objects				
Bidirectional	$\mathbf{C}_{\mathbf{MR}}$	$\mathbf{O}_{\mathbf{in}}$ (pull)	No	Yes	✓			
			Yes	Yes				
		$\mathbf{P}_{\mathbf{in}}'$ (push)	No	Yes				✓
			Yes	Yes				
	M	$\mathbf{O}_{\mathbf{in}}$ (pull)	No	Yes				
			Yes	Yes		✓		
		$\mathbf{P}_{\mathbf{in}}'$ (push)	No	Yes				
			Yes	Yes			✓	
Unidirectional	**M**	$\mathbf{O}_{\mathbf{in}}$ (pull)	Yes	Yes				
		$\mathbf{P}_{\mathbf{in}}'$ (push)	Yes	Yes				✓

5.1 Pull ingest with Multicast gateway in the consumer premises equipment

5.1.0 Introduction

Figure 5.1.0-1 below illustrates a vertically integrated deployment in which the system is configured entirely within the service provider's domain.

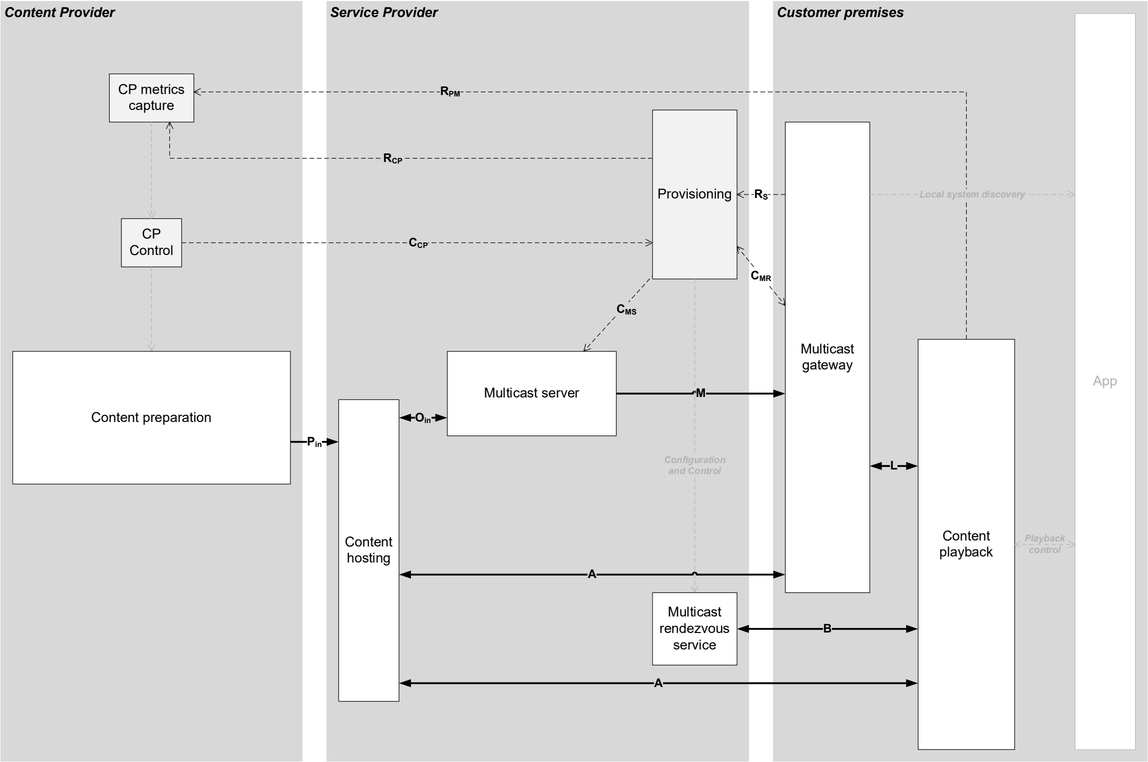


Figure 5.1.0-1: Deployment architecture diagram for vertically integrated deployment

Table 5.1.0-1 below shows example fully-qualified domain names for the key functions:

Table 5.1.0-1: Logical function owners and FQDNs for vertically integrated deployment

Logical function	Domain	FQDN
Provisioning	Service provider	dvb-provisioning.isp.net
Multicast rendezvous service		dvb-rv.isp.net
Content hosting		content-provider.cdn.isp.net
Multicast server		mcast-srv.isp.net
Multicast gateway		mg.subscriber9876.isp.net
Content playback	Customer premises	N/A

In this example, the following assumptions are made:

- The multicast service carries an MPEG-DASH media presentation, with media segment URIs generated using the substitution rules applied to a `SegmentTemplate` element in the MPEG-DASH MPD as specified in clause 5.3.9.4 of ISO/IEC 23009-1 [i.3].
- The *Multicast server* operates the pull-based content ingest method specified in clause 8.3.2 of [i.1].
- The presentation manifest and initialisation segments are delivered over unicast reference point **A**, with only media segments delivered via reference point **M**.
- The service provider allows consumers to subscribe to sufficient numbers of multicast groups that each service component is carried on its own multicast group.

- The multicast gateway is configured according to the out-of-band pulled configuration method as specified in clause 10.1.2 of [i.1]. The multicast gateway configuration transport session described in clause 10.2.5 of [i.1] is therefore not used.

- The *Multicast gateway* is deployed in the customer premises as a service on the home gateway device as described in clause 6.2 of [i.1]. It behaves as an HTTP(S) reverse caching proxy, and the *Asset storage* subfunction of the *Multicast gateway* acts as a HTTP object cache.

- It is assumed that the public-facing address of the *Multicast gateway* does not change during the playback sequence.

5.1.1 Provisioning sequence

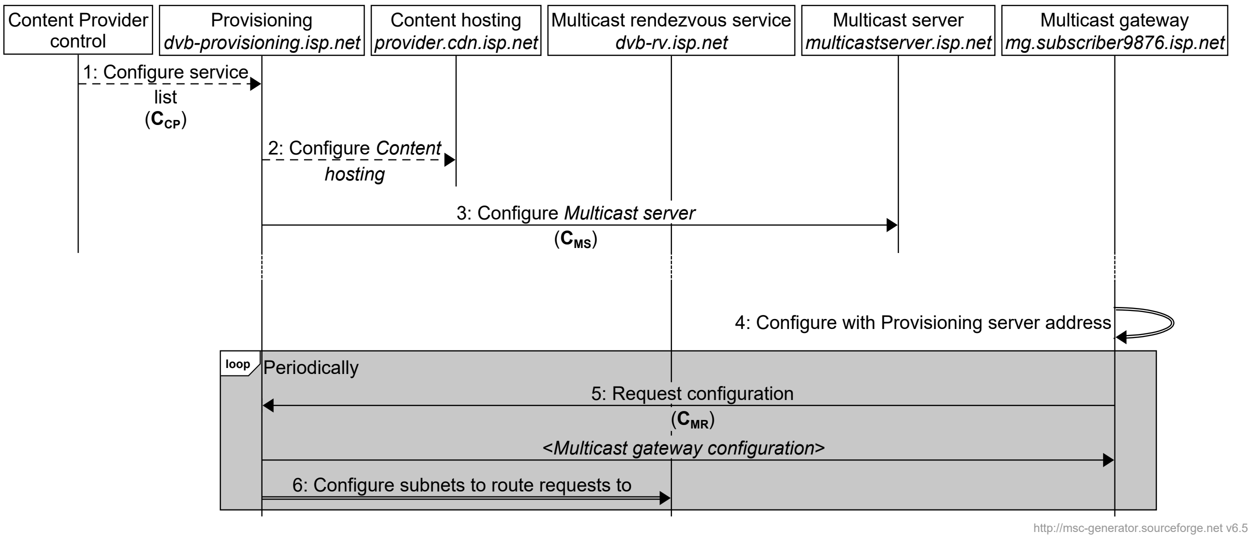

Content Provider control
Provisioning
dvb-provisioning.isp.net
Content hosting
provider.cdn.isp.net
Multicast rendezvous service
dvb-rv.isp.net
Multicast server
multicastserver.isp.net
Multicast gateway
mg.subscriber9876.isp.net
1: Configure service list
($\mathbf{C}_{\mathbf{CP}}$)
2: Configure Content hosting
3: Configure Multicast server
($\mathbf{C}_{\mathbf{MS}}$)
4: Configure with Provisioning server address
loop
Periodically
5: Request configuration
($\mathbf{C}_{\mathbf{MR}}$)
<Multicast gateway configuration>
6: Configure subnets to route requests to
http://msc-generator.sourceforge.net v6.5

Figure 5.1.1-1: Provisioning sequence

The steps in the provisioning sequence are as follows:

1. The *Content Provider control* function provides a list of available linear services to be delivered over multicast to the service provider's *Provisioning* function via reference point $\mathbf{C}_{\mathbf{CP}}$.

2. The *Provisioning* function configures the *Content hosting* function to make the media presentation available.

3. The *Provisioning* function uses the out-of-band pushed multicast server configuration method defined in clause 10.4.2.2 of [i.1] over reference point $\mathbf{C}_{\mathbf{MS}}$ to supply a multicast server configuration instance document to the *Multicast server* that describes the set of multicast sessions and multicast transport sessions that it is to transmit.

   This configuration references the presentation manifests needed to fetch ingest media objects and start transmitting multicast transport objects over reference point **M**.

Time passes.

4. The *Multicast gateway* is statically configured with the address of the *Provisioning* function, for example using TR-069 [i.4] or TR-369 [i.5].

5. The *Multicast gateway* instance periodically requests an updated multicast gateway configuration using the out-of-band pulled multicast gateway configuration method described in clause 10.4.4.2 of [i.1]. The *Provisioning* function returns a multicast gateway configuration describing the set of sessions that the *Multicast gateway* is expected to serve.

NOTE: The periodicity of this request is determined by the **`MulticastGatewayConfiguration`**`@validityPeriod` attribute in the received multicast gateway configuration.

The *Multicast gateway*'s request at reference point $\mathbf{C}_{\mathbf{MR}}$ includes an implementation-specific query string that carries extra information to help facilitate the *Multicast rendezvous service* redirecting *Content playback* functions to the appropriate *Multicast gateway*. The two field-value pairs are as follows:

- The *redirect-baseurl* query field carries the base URL to be used in the redirection URL to be supplied at reference point **B**.

- The *redirect-subnet* query field carries either an IPv4 subnet as described by [i.7] or an IPv6 subnet as described by [i.8]. If the *Multicast gateway* can support *Content playback* access from multiple different subnets, or is “dual stack”, then the query string may include multiple instances of this field.

An example query string is presented in table 5.1-1 below.

6. When the *Provisioning* function receives a request for a multicast gateway configuration from a *Multicast gateway* function, it updates the *Multicast rendezvous service* with a configuration that includes for each *Multicast gateway* that has requested a configuration in step 5 above:

    a. The set of client subnets for which new streaming sessions should be redirected to the *Multicast gateway* instance in question, based on the subnets it provided in step 5.

    b. The manifest redirect base URL nominated by the *Multicast gateway* in step 5.

    c. The manifest redirect path pattern nominated by the *Provisioning* function.

Table 5.1-1: Construction of the Multicast rendezvous service HTTP redirect URL

Multicast rendezvous service configuration	
Manifest redirect base URL	**https://mg.subscriber9876.isp.net**
Manifest redirect path pattern	**/content-provider/s1/client_manifest.mpd**
Example outcome	

Out-of-band pulled multicast gateway configuration request at reference point $\mathbf{C}_{\mathbf{MR}}$	https://dvb-provisioning.isp.net
	/dvb-multicast-gateway/v1/configuration?
	redirect-baseurl=**https%3A%2F%2Fmg.subscriber7986.isp.net**?
	redirect-subnet=**198.51.100.73%2F32?**
	redirect-subnet=**2001%3ADB84%3AE773%3A%3A%2F64**

Multicast rendezvous service HTTP redirect URL (step 13)	https://mg.subscriber9876.isp.net **/content-provider/s1/client_manifest.mpd**

5.1.2 Multicast server transmission sequence

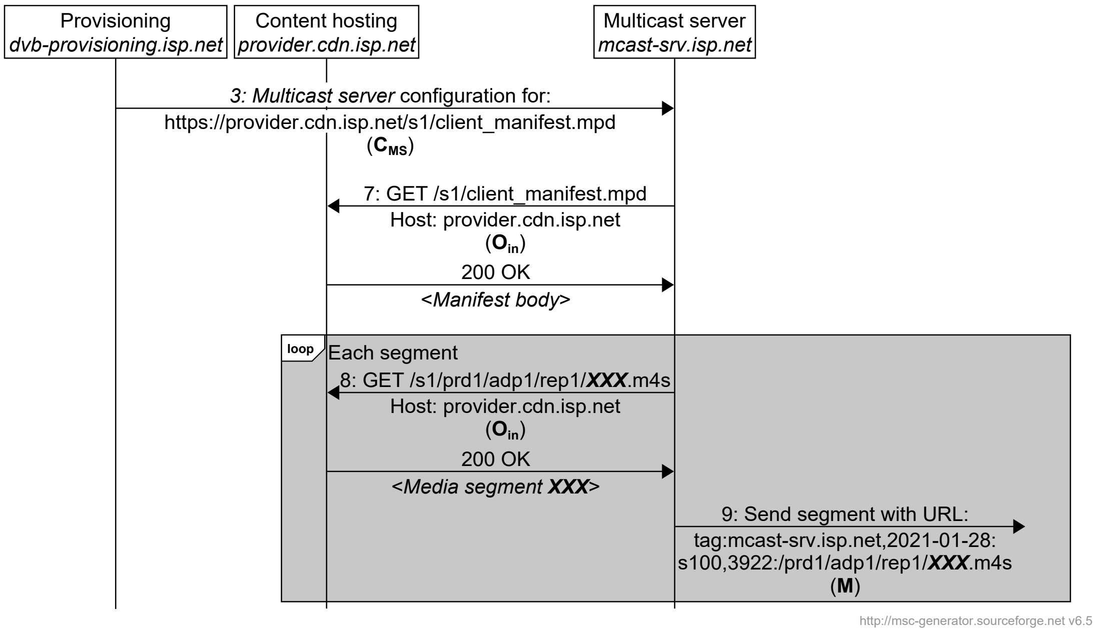

Provisioning
dvb-provisioning.isp.net
Content hosting
provider.cdn.isp.net
Multicast server
mcast-srv.isp.net
3: Multicast server configuration for:
https://provider.cdn.isp.net/s1/client_manifest.mpd
(CMS)
7: GET /s1/client_manifest.mpd
Host: provider.cdn.isp.net
(Oin)
200 OK
<Manifest body>
loop
Each segment
8: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: provider.cdn.isp.net
(Oin)
200 OK
<Media segment XXX>
9: Send segment with URL:
tag:mcast-srv.isp.net,2021-01-28:
s100,3922:/prd1/adp1/rep1/XXX.m4s
(M)
http://msc-generator.sourceforge.net v6.5

Figure 5.1.2-1: Multicast server transmission sequence

The steps in the *Multicast server* transmission sequence are as follows:

7. Using the presentation manifest URL from the multicast server configuration acquired in step 3 (see clause 5.1 above), the *Multicast server* requests the presentation manifest via reference point $\mathbf{O_{in}}$ as specified in clause 8.3.2 of [i.1].

NOTE: For brevity, the presentation manifest is requested only once by the *Multicast server*. This would be the case, for example, if the **`MPD`**`@minimumUpdatePeriod` attribute is omitted from the presentation manifest, and the media segment URLs are constructed using a Segment Template as specified in [i.3].

8. The *Multicast server* requests media objects using the pull-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point $\mathbf{O_{in}}$.

9. The *Multicast server* payloads ingest media objects as multicast transport objects and transmits them over reference point **M**, as specified in clause 8.3.4 of [i.1].

The *Multicast server* uses the **`UnicastRepairParameters/BaseURL`** and **`UnicastRepairParameters`** `@transportObjectBaseURI` carried in the multicast server configuration to convert the URL used to request media objects over reference point $\mathbf{O_{in}}$ into a multicast transport object URI at reference point **M**. (This is the inverse of the mapping performed by the *Multicast gateway* in step 26 in clause 5.4.) Per clause 8.3.3 of [i.1], this mapping is implementation-specific; table 5.1.2-1 below shows one possible example.

Table 5.1.2-1: Mapping of media object URL to multicast transport object URI in the Multicast server

Multicast session configuration	
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.isp.net/s1**
`UnicastRepairParameters` `@transportObjectBaseURI`	**tag:mcast-srv.isp.net,2021-01-28:mts100,3922:**
Example outcome	
Content ingest URL at reference point $\mathbf{O_{in}}$	**https://content-provider.cdn.isp.net/s1** /prd1/adp1/rep1/1234.m4s
Multicast transport object URI at reference point **M**	**tag:mcast-srver.isp.net,2021-01-28:mts100,3922:** /prd1/adp1/rep1/1234.m4s

5.1.3 Multicast gateway playback sequence

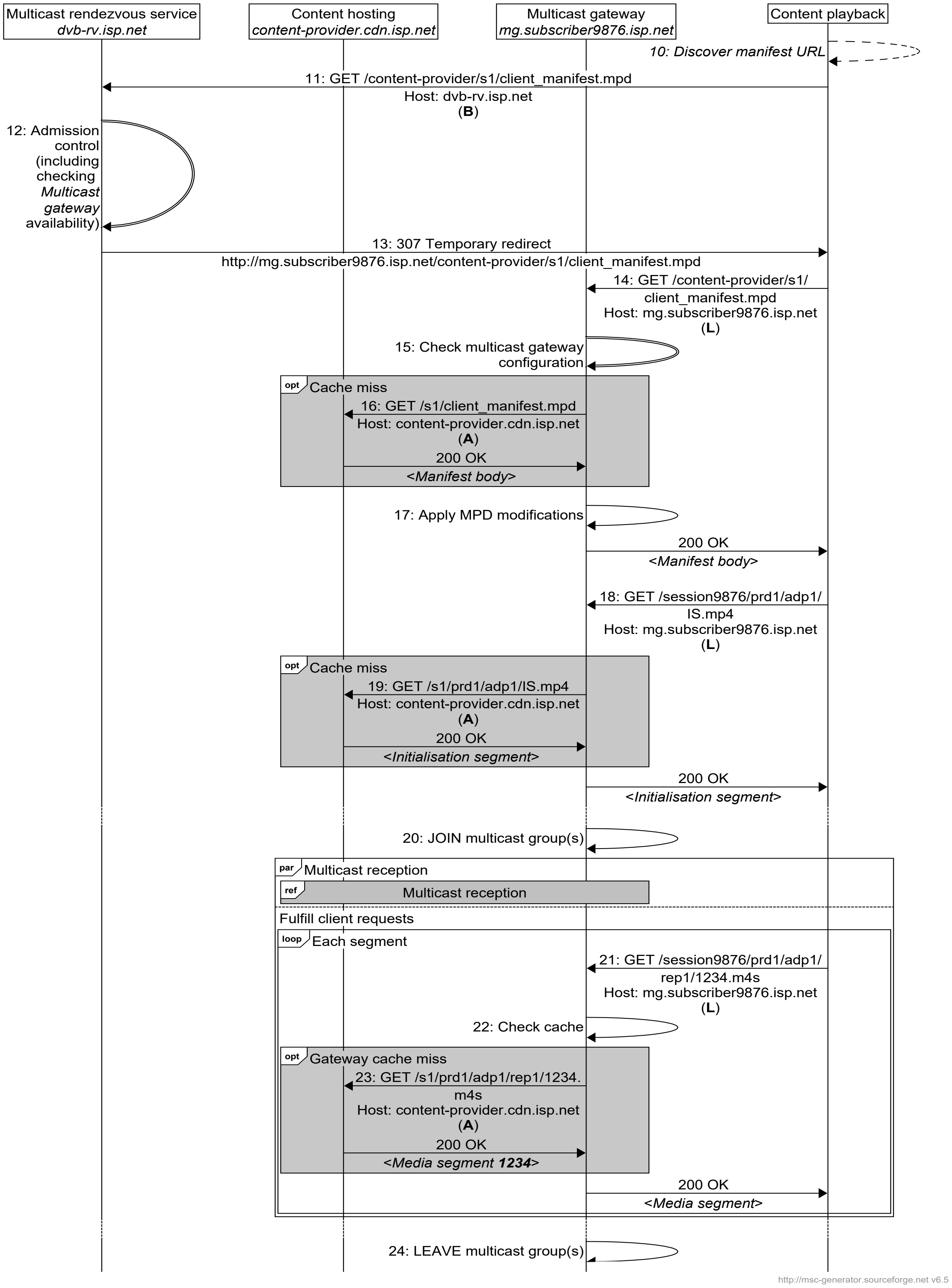

Multicast rendezvous service
dvb-rv.isp.net
Content hosting
content-provider.cdn.isp.net
Multicast gateway
mg.subscriber9876.isp.net
Content playback
10: Discover manifest URL
11: GET /content-provider/s1/client_manifest.mpd
Host: dvb-rv.isp.net
(B)
12: Admission control (including checking Multicast gateway availability)
13: 307 Temporary redirect
http://mg.subscriber9876.isp.net/content-provider/s1/client_manifest.mpd
14: GET /content-provider/s1/client_manifest.mpd
Host: mg.subscriber9876.isp.net
(L)
15: Check multicast gateway configuration
opt Cache miss
16: GET /s1/client_manifest.mpd
Host: content-provider.cdn.isp.net
(A)
200 OK
<Manifest body>
17: Apply MPD modifications
200 OK
<Manifest body>
18: GET /session9876/prd1/adp1/IS.mp4
Host: mg.subscriber9876.isp.net
(L)
opt Cache miss
19: GET /s1/prd1/adp1/IS.mp4
Host: content-provider.cdn.isp.net
(A)
200 OK
<Initialisation segment>
200 OK
<Initialisation segment>
20: JOIN multicast group(s)
par Multicast reception
ref Multicast reception
Fulfill client requests
loop Each segment
21: GET /session9876/prd1/adp1/rep1/1234.m4s
Host: mg.subscriber9876.isp.net
(L)
22: Check cache
opt Gateway cache miss
23: GET /s1/prd1/adp1/rep1/1234.m4s
Host: content-provider.cdn.isp.net
(A)
200 OK
<Media segment 1234>
200 OK
<Media segment>
24: LEAVE multicast group(s)
http://msc-generator.sourceforge.net v6.5

Figure 5.1.3-1: Multicast gateway playback sequence

The steps in the playback sequence are as follows:

10. A playback session begins when a presentation manifest URL referencing the address of the *Multicast rendezvous service* is supplied to the *Content playback* function.

11. The *Content playback* function makes a unicast HTTP(S) request for the manifest at reference point **B**, as specified in clause 7.5.1 of [i.1].

12. The *Multicast rendezvous service* function examines the source IP address of the request and determines that it belongs to a subnet with an operational *Multicast gateway* that satisfies the requirements specified in clause 7.5.2.0 of [i.1].

13. The *Multicast rendezvous service* issues an HTTP redirect to the *Multicast gateway* function as specified in clause 7.5.2.1 of [i.1].

Table 5.1.3-1 below shows an example of the presentation manifest URL requested from the *Multicast gateway* at reference point **B**, and the reference point **L** URL to which the requesting *Content playback function* is redirected.

Table 5.1.3-1: Manifest URLs over reference points B and L

Multicast rendezvous service configuration	
Manifest redirect base URL *Nominated by Multicast gateway in step 5*	https://mg.subscriber9876.isp.net/
Manifest redirect path suffix *Nominated by Provisioning*	**/content-provider/s1/client_manifest.mpd**
Example outcome	
Presentation manifest URL at reference point **B**	https://dvb-rv.isp.net/content-provider/s1/client_manifest.mpd
Redirected presentation manifest URL at reference point **L**	https://mg.subscriber9876.isp.net/**content-provider/s1/client_manifest.mpd**

14. The *Content playback* function requests the presentation manifest from the *Multicast gateway* nominated by the *Multicast rendezvous service* in step 13 at reference point **L**.

NOTE: For brevity, the presentation manifest is requested only once by the *Content playback* function. This would be the case, for example, if the `MPD@minimumUpdatePeriod` attribute is omitted from the presentation manifest, and the media segment URLs are constructed using a Segment Template as specified in [i.3].

15. The *Multicast gateway* examines its multicast session configuration (acquired during step 5 in clause 5.1) and determines that there is an active multicast session corresponding to the reference point **L** presentation manifest URL.

NOTE: In the case where there is no configured multicast session, the behaviour may be as described in clause 4.3.1.

Using the content playback manifest request path suffix from the multicast gateway configuration acquired in step 5, the *Multicast gateway* matches redirected presentation manifest requests at reference point **L** to services present in the multicast gateway configuration. Table 5.1.3-2 below shows the URL of the presentation manifest signalled in the multicast session configuration, as well as the manifest path that the *Multicast gateway* will match, alongside the various URLs then used at reference points **L** and **A**.

Table 5.1.3-2: Manifest URLs over reference points L and A

Multicast session configuration	
Session `PresentationManifestLocator`	https://content-provider.cdn.isp.net/s1/client_manifest.mpd

Content playback manifest request path pattern
`PresentationManifestLocator`
`@contentPlaybackPathPattern`
	/content-provider/s1/client_manifest.mpd

Example outcome	
Redirected presentation manifest URL at reference point **L**	https://mg.subscriber9876.isp.net/manifests/**content-provider/s1/client_manifest.mpd**
Presentation manifest URL at reference point **A**	https://content-provider.cdn.isp.net/s1/client_manifest.mpd

16. If the *Multicast gateway* does not have a valid cached copy of the requested presentation manifest in its cache, it requests it from the *Content hosting* via reference point **A** as specified in clause 8.4.1.0 of [i.1].

17. The *Multicast gateway* may modify the presentation manifest for the reasons given in clause 8.4.1 and clause 10.2.2.0 of [i.1]. The modifications needed are signalled in the multicast session configuration, and in this example include adding a session identifier to the **`MPD/BaseURL`** element for the *Multicast gateway* to track individual client sessions. The modification to the **`MPD/BaseURL`** is given in Table 5.3-3 below.

NOTE: Another implementation may choose to include the session identifier in requests made at reference point **A**, as either a path element or as a query parameter.

18. The *Content playback* function makes a request for the MPEG-DASH initialisation segment(s) for its chosen representation(s) via reference point **L**.

19. If the *Multicast gateway* does not have a cached copy of the requested MPEG-DASH initialisation segment(s), it may request it/them from the *Content hosting* function at reference point **A**.

Table 5.1.3-3 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. This process is dependent on how the manifest was modified before returning to the client. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **A** from the information present in the MPD.

Table 5.1.3-3: Mapping of MPEG-DASH initialisation segment URLs

MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://content-provider.cdn.isp.net/s1/**
`MPD/BaseURL` at reference point **L**	**https://mg.subscriber9876.isp.net/session9876/**
Example outcome	
Initialisation segment URL at reference point **L**	**https://mg.subscriber9876.isp.net/session9876/** prd1/adp1/IS.mp4
Initialisation segment URL at reference point **A**	**https://content-provider.cdn.isp.net/s1/** prd1/adp1/IS.mp4

20. The *Multicast gateway* joins the multicast transport sessions indicated in its multicast session configuration as specified in clause 8.4.2 of [i.1].

21. The *Content playback* function makes requests for MPEG-DASH media segments at reference point **L**.

22. The *Multicast gateway* checks whether it has a cached copy of the requested media object in its *Asset storage*, either received over reference point **M** or previously requested via reference point **A** by another *Content playback* function.

23. If the requested media object is not cached in the *Asset storage* subfunction, then the *Multicast gateway* makes a unicast request for the media object from the *Content hosting* via reference point **A**.

Table 5.1.3-4 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. This process is dependent on how the manifest was modified before returning to the client. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **A** from the information present in the MPD.

Table 5.1.3-4: Mapping of MPEG-DASH media segment URLs

MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://content-provider.cdn.isp.net/s1/**
`MPD/BaseURL` at reference point **L**	**https://mg.subscriber9876.isp.net/session9876/**
Example outcome	
Media segment URL at reference point **L**	**https://mg.subscriber9876.isp.net/session9876/** prd1/adp1/rep1/1234.m4s
Media segment URL at reference point **A**	**https://content-provider.cdn.isp.net/s1/** prd1/adp1/rep1/1234.m4s

24. The *Multicast gateway* leaves the multicast transport sessions when the *Content playback* function stops requesting segments and the session is deemed finished.

5.1.4 Multicast reception sequence

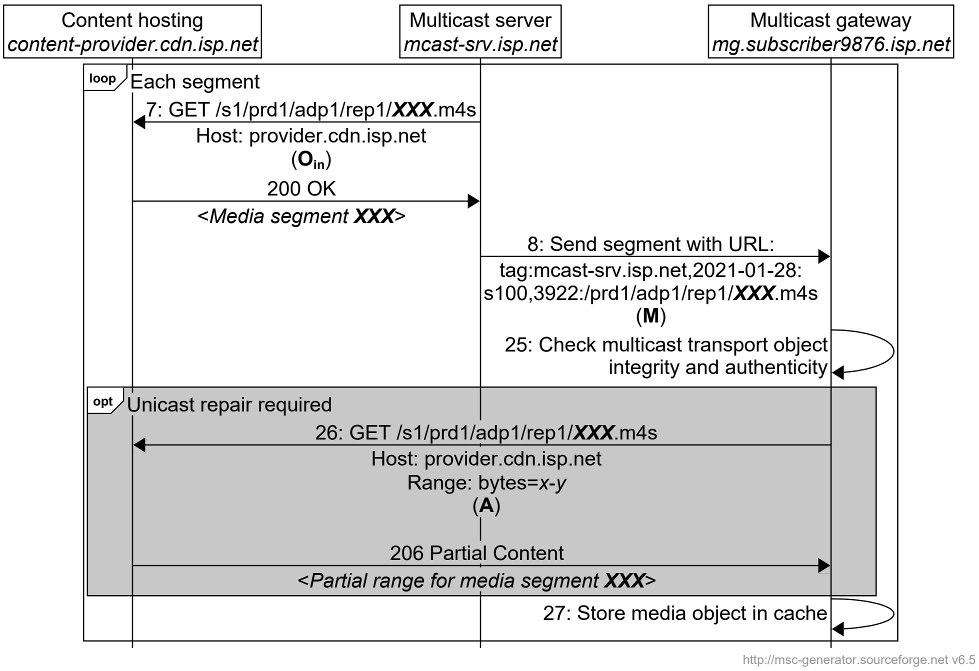

Content hosting
content-provider.cdn.isp.net
Multicast server
mcast-srv.isp.net
Multicast gateway
mg.subscriber9876.isp.net
loop
Each segment
7: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: provider.cdn.isp.net
(O_in)
200 OK
<Media segment XXX>
8: Send segment with URL:
tag:mcast-srv.isp.net,2021-01-28:
s100,3922:/prd1/adp1/rep1/XXX.m4s
(M)
25: Check multicast transport object
integrity and authenticity
opt
Unicast repair required
26: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: provider.cdn.isp.net
Range: bytes=x-y
(A)
206 Partial Content
<Partial range for media segment XXX>
27: Store media object in cache
http://msc-generator.sourceforge.net v6.5

Figure 5.1.4-1: Multicast reception sequence

The steps in the *Multicast gateway* reception sequence are as follows:

25. Once the *Multicast gateway* has fully received a multicast transport object on the multicast transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

26. Should the *Multicast gateway* not receive a complete intact multicast transport object over reference point **M** (see clause 5.2), or if the integrity check performed in step 25 should fail, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1].

Table 5.1.4-1 below shows how the *Multicast gateway* uses the **`UnicastRepairParameters`** in the multicast gateway configuration to transform the multicast transport object URI at reference point **M** to the unicast repair URL at reference point **A**.

Table 5.1.4-1: Mapping of multicast transport object URIs to unicast repair URLs

Multicast session configuration	
`UnicastRepairParameters` `@transportObjectBaseURI`	**tag:mcast-srv.isp.net,2021-01-28:mts100,3922:**
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.isp.net/s1**
Example outcome	
Multicast transport object URI at reference point **M**	**tag:mcast-srv.isp.net,2021-01-28:mts100,3922:**/prd1/adp1/rep1/1234.m4s
Unicast repair URL at reference point **A**	**https://content-provider.cdn.isp.net/s1**/prd1/adp1/rep1/1234.m4s

27. The media segment carried as a multicast transport object is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1].

5.2 Pull ingest from a third party Content hosting function with Multicast gateway in the network edge

5.2.0 Introduction

Figure 5.2.0-1 below illustrates deployment in which the system relies on a Content hosting (CDN) function located in a third party's domain.

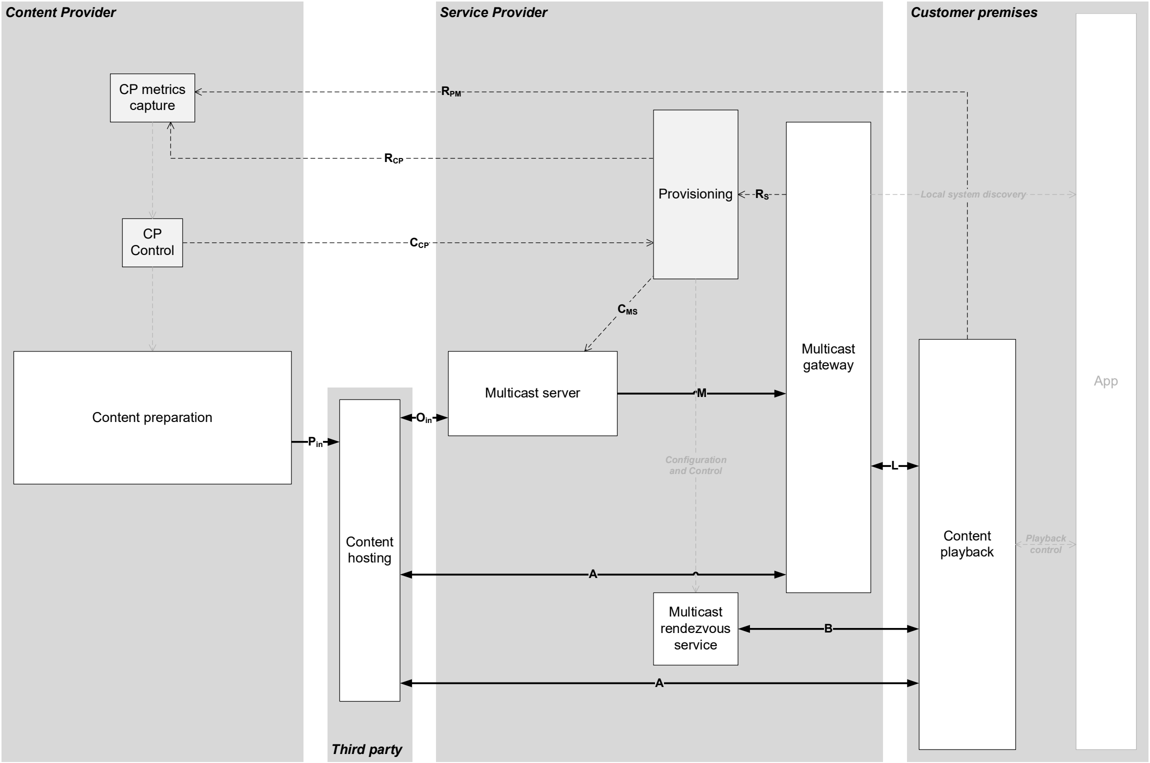


Figure 5.2.0-1: Deployment architecture diagram with two domains

Table 5.2.0-1 below shows example fully-qualified domain names for the key functions:

Table 5.2.0-1: Logical function owners and FQDNs for vertically integrated deployment

Logical function	Domain	FQDN
Provisioning	Service provider	N/A
Multicast rendezvous service		dvb-rnrv.isp.net
Content hosting	Third party	content-provider.cdn.net
Multicast server	Service provider	mcast-srv.isp.net
Multicast gateway		mg1234.isp.net
Content playback	Customer premises	N/A

In this example, the following assumptions are made:

- The multicast service carries an MPEG-DASH media presentation with relative media segment URLs carried either in a `SegmentList` element of the MPEG-DASH MPD, as described by clause 5.3.9.3 of [i.3]; or generated using the substitution rules applied to a `SegmentTemplate` element in association with the `SegmentTimeLine` element of the MPD as specified in clause 5.3.9.4 and 5.3.9.6 respectively of [i.3]. In both cases, the *Multicast gateway* and *Content playback* functions need to refresh the manifest frequently (at most once per media segment duration) in order to receive information about the latest media segments.

- The *Multicast server* operates the pull-based content ingest method specified in clause 8.3.2 of [i.1].

- The presentation manifest, initialisation segments and media segments are delivered over multicast reference point **M** and may also be retrieved over reference point **A**.

- The service provider allows consumers to subscribe to sufficient numbers of multicast groups that each service component is carried on its own multicast group.
- The *Multicast gateway* is configured according to the just-in-time configuration method as specified in clause 10.1.2 of [i.1]. The multicast gateway configuration transport session described in clause 10.2.5 of [i.1] is therefore not used.
- The *Multicast gateway* behaves as an HTTP(S) reverse caching proxy, and the *Asset storage* subfunction of the *Multicast gateway* acts as a HTTP object cache.
- The rendezvous service is not co-located with the *Multicast gateway*.

5.2.1 Provisioning sequence

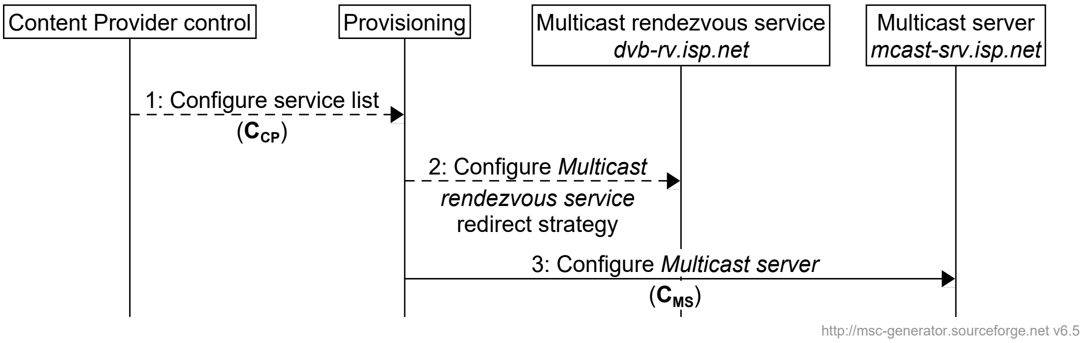

Content Provider control
Provisioning
Multicast rendezvous service
dvb-rv.isp.net
Multicast server
mcast-srv.isp.net
1: Configure service list
(C_{CP})
2: Configure Multicast rendezvous service redirect strategy
3: Configure Multicast server
(C_{MS})
http://msc-generator.sourceforge.net v6.5

Figure 5.2.1-1: Provisioning sequence

The steps in the provisioning sequence are as follows:

1. The *Content Provider control* function configures the *Provisioning* function with the list of linear services to be delivered via multicast.
2. The *Provisioning* function configures the *Multicast rendezvous service* to redirect *Content playback* functions to the *Multicast gateway* (shown in step 14 in clause 6.3).

NOTE: The fully-qualified domain name of the *Content hosting* function is known per peering agreement between the service provider and the third party CDN operator. Per this agreement, this can be based on the hostname optionally joined with the HTTP redirect response from the third-party CDN broker to the *Multicast rendezvous service* as depicted in clauses 7.4 and 7.5.1 of [i.1].

3. The *Provisioning* function uses the out-of-band pushed multicast server configuration method defined in clause 10.4.2.2 of [i.1] over reference point $\mathbf{C_{MS}}$ to supply a multicast server configuration instance document to the *Multicast server* that describes the set of multicast sessions and multicast transport sessions that it is to transmit. This configuration references the presentation manifests needed to fetch ingest media objects and start transmitting multicast transport objects over reference point **M**.

5.2.2 Multicast server transmission sequence

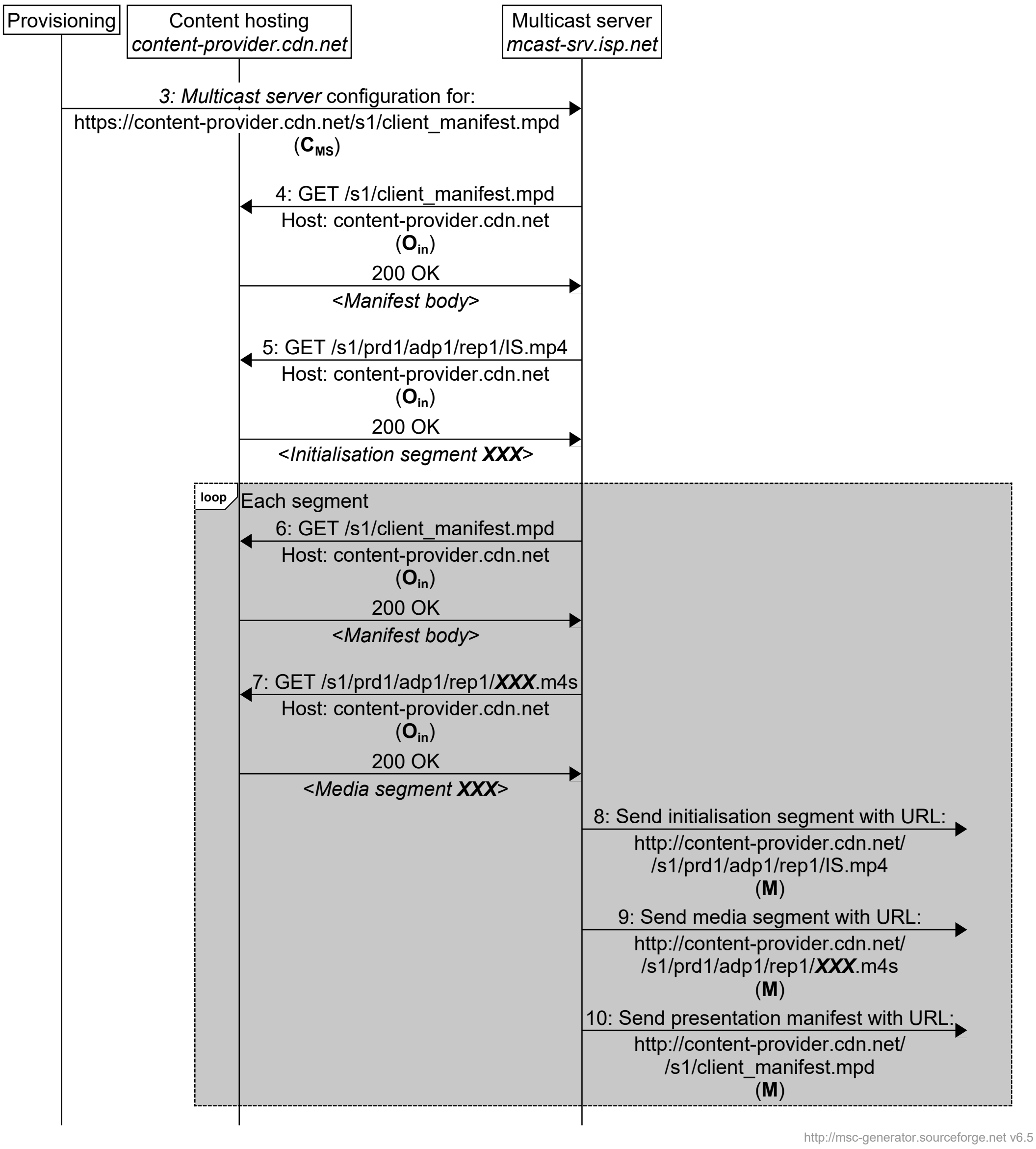

Provisioning
Content hosting
content-provider.cdn.net
Multicast server
mcast-srv.isp.net
3: Multicast server configuration for:
https://content-provider.cdn.net/s1/client_manifest.mpd
(C_MS)
4: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(O_in)
200 OK
<Manifest body>
5: GET /s1/prd1/adp1/rep1/IS.mp4
Host: content-provider.cdn.net
(O_in)
200 OK
<Initialisation segment XXX>
loop
Each segment
6: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(O_in)
200 OK
<Manifest body>
7: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: content-provider.cdn.net
(O_in)
200 OK
<Media segment XXX>
8: Send initialisation segment with URL:
http://content-provider.cdn.net/
/s1/prd1/adp1/rep1/IS.mp4
(M)
9: Send media segment with URL:
http://content-provider.cdn.net/
/s1/prd1/adp1/rep1/XXX.m4s
(M)
10: Send presentation manifest with URL:
http://content-provider.cdn.net/
/s1/client_manifest.mpd
(M)
http://msc-generator.sourceforge.net v6.5

Figure 5.2.2-1: Multicast server transmission sequence

The steps in the *Multicast server* transmission sequence are as follows:

4. Using the presentation manifest URL from the multicast server configuration acquired in step 3 (see clause 6.1 above), the *Multicast server* requests the presentation manifest via reference point **O$_{in}$** as specified in clause 8.3.2 of [i.1].

5. The *Multicast server* requests the initialisation segment object(s) using the pull-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point **O$_{in}$** for the purpose of transmission over reference point **M** as shown in step 8.

For each segment:

6. The *Multicast server* requests the presentation manifest via reference point **O$_{in}$** as specified in clause 8.3.2 of [i.1] for the purpose of transmission over reference point **M**.

7. The *Multicast server* requests media objects using the pull-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point **O$_{in}$**.

8. The *Multicast server* payloads the initialisation segment(s) ingested in step 5 above as multicast transport objects and transmits them over reference point **M**, as specified in clause 8.3.4 of [i.1].

NOTE: The initialisation segment can alternatively be retrieved by the by the *Multicast gateway* over reference point **A**.

9. The *Multicast server* payloads ingested media segments as multicast transport objects and transmits them over reference point **M**, as specified in clause 8.3.4 of [i.1].

10. The *Multicast server* payloads the latest presentation manifest object as a multicast transport object and transmits it over reference point **M**, as specified in clause 8.3.4 of [i.1].

NOTE: The manifest can also be retrieved by the by the *Multicast gateway* over reference point **A**. Sending the manifest after the segment avoids a potential race condition where the *Multicast gateway* may present a new presentation manifest to the *Content playback* function that lists media objects that it has yet to receive over reference point **M**. If the manifest has not changed, this is a way to repeat (carousel) it.

The *Multicast server* uses the **`UnicastRepairParameters/BaseURL`** and **`UnicastRepairParameters`** `@transportObjectBaseURI` carried in the multicast server configuration to convert the URL used to request media objects over reference point **O$_{in}$** into a multicast transport object URI at reference point **M**. Per clause 8.3.3 of [i.1], this mapping is implementation-specific; table 5.2.2-1 below shows one possible example.

Table 5.2.2-1: Mapping of media object URL to multicast transport object URI in the Multicast server

Multicast session configuration	
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.net/**
`UnicastRepairParameters` `@transportObjectBaseURI`	**https://content-provider.cdn.net/**
Example outcome	
Content ingest URL at reference point **O$_{in}$**	**https://content-provider.cdn.net/**s1 /prd1/adp1/rep1/1234.m4s
Multicast transport object URI at reference point **M**	**https://content-provider.cdn.net/**s1 /prd1/adp1/rep1/1234.m4s

NOTE: Because the **`UnicastRepairParameters/BaseURL`** element content is identical to the value of the **`UnicastRepairParameters`**`@transportObjectBaseURI` attribute in this example, the mapping is the identity transform.

5.2.3 Multicast gateway playback sequence

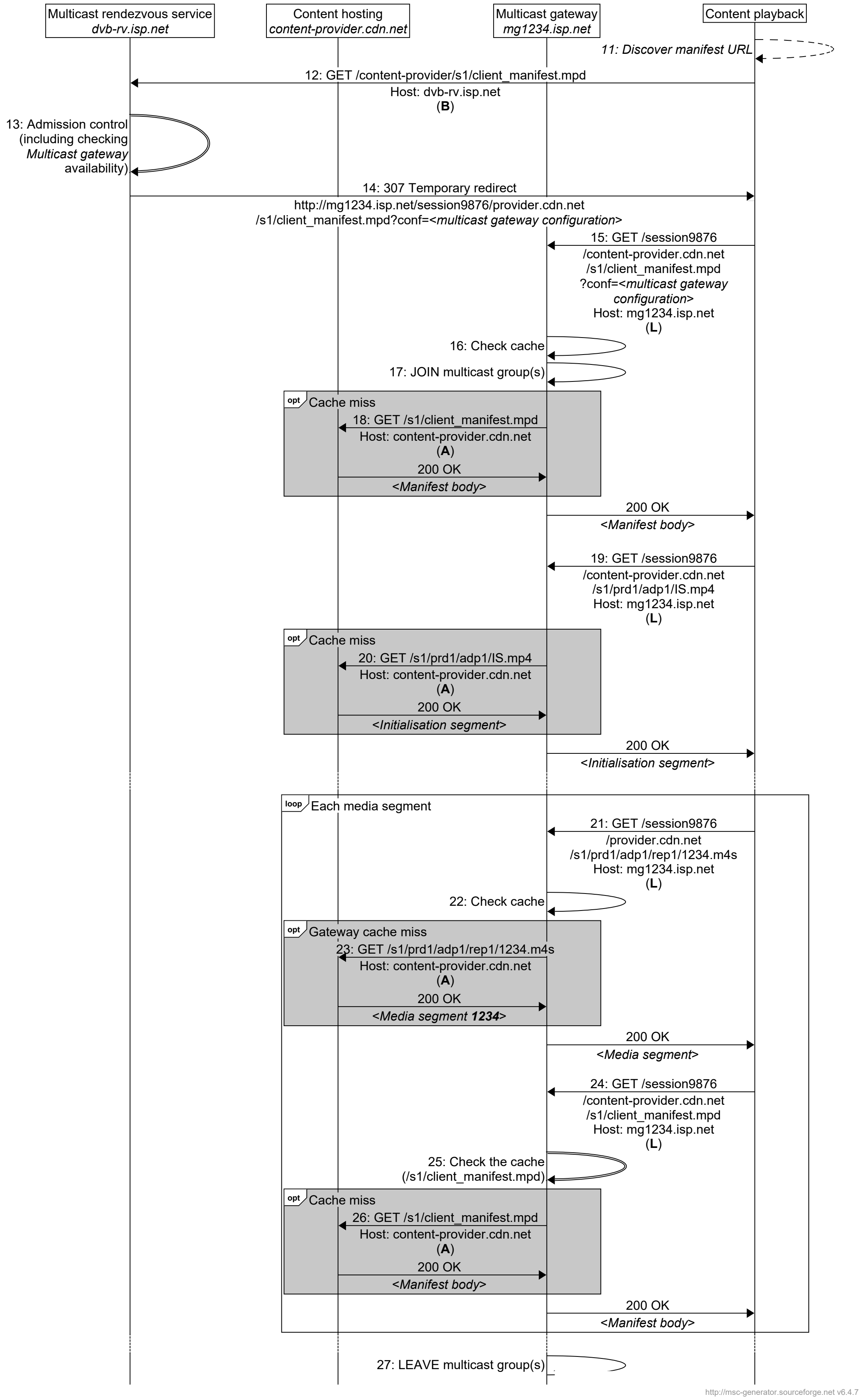

Multicast rendezvous service
dvb-rv.isp.net
Content hosting
content-provider.cdn.net
Multicast gateway
mg1234.isp.net
Content playback
11: Discover manifest URL
12: GET /content-provider/s1/client_manifest.mpd
Host: dvb-rv.isp.net
(**B**)
13: Admission control
(including checking
Multicast gateway
availability)
14: 307 Temporary redirect
http://mg1234.isp.net/session9876/provider.cdn.net
/s1/client_manifest.mpd?conf=<*multicast gateway configuration*>
15: GET /session9876
/content-provider.cdn.net
/s1/client_manifest.mpd
?conf=<*multicast gateway*
configuration>
Host: mg1234.isp.net
(**L**)
16: Check cache
17: JOIN multicast group(s)
opt
Cache miss
18: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(**A**)
200 OK
<*Manifest body*>
200 OK
<*Manifest body*>
19: GET /session9876
/content-provider.cdn.net
/s1/prd1/adp1/IS.mp4
Host: mg1234.isp.net
(**L**)
opt
Cache miss
20: GET /s1/prd1/adp1/IS.mp4
Host: content-provider.cdn.net
(**A**)
200 OK
<*Initialisation segment*>
200 OK
<*Initialisation segment*>
loop
Each media segment
21: GET /session9876
/provider.cdn.net
/s1/prd1/adp1/rep1/1234.m4s
Host: mg1234.isp.net
(**L**)
22: Check cache
opt
Gateway cache miss
23: GET /s1/prd1/adp1/rep1/1234.m4s
Host: content-provider.cdn.net
(**A**)
200 OK
<*Media segment **1234***>
200 OK
<*Media segment*>
24: GET /session9876
/content-provider.cdn.net
/s1/client_manifest.mpd
Host: mg1234.isp.net
(**L**)
25: Check the cache
(/s1/client_manifest.mpd)
opt
Cache miss
26: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(**A**)
200 OK
<*Manifest body*>
200 OK
<*Manifest body*>
27: LEAVE multicast group(s)
http://msc-generator.sourceforge.net v6.4.7

Figure 5.2.3.1-1: Multicast gateway playback sequence

The steps in the *Multicast gateway* reception sequence are as follows:

11. A playback session begins when a presentation manifest URL referencing the address of the *Multicast rendezvous service* is supplied to the *Content playback* function.

NOTE: The presentation manifest URL may also include as additional path elements or as query parameters the fully-qualified domain name of the *Content hosting* function as well as a token provided by a third party CDN broker (see clause 7.5.1 of [i.1]).

12. The *Content playback* function makes a unicast HTTP(S) request for the manifest at reference point **B**, as specified in clause 7.5.1 of [i.1].

13. The *Multicast rendezvous service* function examines the source IP address of the request and determines that it belongs to a subnet with an operational *Multicast gateway* that satisfies the requirements specified in clause 7.5.2.0 of [i.1]. In addition, the *Multicast rendezvous service* detects that the requested content is available from a third party CDN for which there is a peering agreement (see clause 7.4 of [i.1])

14. The *Multicast rendezvous service* issues an HTTP redirect to the *Multicast gateway* function as specified in clause 7.5.2.1 of [i.1]. The redirection URL contains a ***conf*** query parameter that corresponds to the just-in-time multicast gateway configuration instance document as specified in clause 10.1.2 of [i.1].

15. The *Content playback* function requests the presentation manifest from the *Multicast gateway* nominated by the *Multicast rendezvous service* in step 14 at reference point **L**.

16. The *Multicast gateway* checks for a valid presentation manifest in its *Asset storage* cache.

17. The *Multicast gateway* examines the just-in-time multicast gateway configuration supplied in the request made in step 15 and joins the multicast transport session(s) indicated, as specified in clause 8.4.2 of [i.1].

18. If the *Multicast gateway* does not have a valid cached copy of the requested presentation manifest in its *Asset storage*, the *Multicast gateway* requests the presentation manifest from the *Content hosting* via reference point **A** as specified in clause 8.4.1.0 of [i.1].

Table 5.2.3-1 below shows the URLs used at reference points **L** and **A**. Having received the manifest request, the multicast gateway matches the URL suffix with the corresponding entry in the multicast gateway configuration.

Table 5.2.3-1: Manifest URLs mapping over reference points L and A

Multicast session configuration	
Session **`PresentationManifestLocator`**	https://content-provider.cdn.net/s1/client_manifest.mpd
`PresentationManifestLocator` `@contentPlaybackPathSuffix`	/*/content-provider.cdn.net/s1/client_manifest.mpd
Example outcome	
Presentation manifest URL at reference point **L**	https://mg1234.isp.net/session9876/content-provider.cdn.net/s1/client_manifest.mpd?conf=(...)
Presentation manifest URL at reference point **A**	https://content-provider.cdn.net/s1/client_manifest.mpd

19. The *Content playback* function makes a request for the MPEG-DASH initialisation segment(s) for its chosen representation(s) via reference point **L**.

20. If the *Multicast gateway* does not have a cached copy of the requested MPEG-DASH initialisation segment(s), it may request it/them from the *Content hosting* function at reference point **A**. Table 5.2.3-2 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. In this example, the unicast repair URL at reference point **A** is derived relative to the URL of the presentation manifest.

Table 5.2.3-2: Mapping of MPEG-DASH initialisation segment URLs

MPD URLs	
MPD URL at reference point **L**	**https://mg1234.isp.net/session9876/content-provider.cdn.net/s1/**client_manifest.mpd?conf=(...)
MPD URL at reference point **A**	**https://content-provider.cdn.net/s1/**client_manifest.mpd
MPD configurations	
`MPD/BaseURL` at reference point **L**	Not present or “./”
`MPD/BaseURL` at reference point **A**	Not present or “./”
Example outcome	
Initialisation segment URL at reference point **L**	**https://mg1234.isp.net/session9876/content-provider.cdn.net/s1/**prd1/adp1/IS.mp4
Initialisation segment URL at reference point **A**	**https://content-provider.cdn.net/s1/**prd1/adp1/IS.mp4

For each MPEG-DASH media segment:

21. The *Content playback* function makes requests for a media segment at reference point **L**.

22. The *Multicast gateway* checks whether it has a cached copy of the requested media object in its *Asset storage*, either received over reference point **M** or previously requested via reference point **A** due to a cache miss resulting from a request by another *Content playback* function consuming the same service.

23. If the requested media object is not cached in the *Asset storage* subfunction, then the *Multicast gateway* makes a unicast request for the media object from the *Content hosting* via reference point **A**.

24. The *Content playback* function requests the updated presentation manifest from the *Multicast gateway*.

25. The *Multicast gateway* checks whether it has a valid cached copy of the requested presentation manifest in its *Asset storage*, either received over reference point **M** or previously requested via reference point **A** due to a cache miss resulting from a request by another *Content playback* function consuming the same service.

26. If the *Multicast gateway* does not have a valid cached copy of the requested presentation manifest in its cache, then the *Multicast gateway* makes a unicast request for the media object from the *Content hosting* via reference point **A**.

Table 5.2.3-3 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. This process is dependent on how the manifest was modified before returning to the client. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **L** from the information present in the MPD.

Table 5.2.3-3: Mapping of MPEG-DASH media segment URLs

MPD URLs	
MPD URL at reference point **L**	**https://mg1234.isp.net/session9876/content-provider.cdn.net/s1/**client_manifest.mpd?conf=(...)
MPD URL at reference point **A**	**https://content-provider.cdn.net/s1/**client_manifest.mpd
MPD configurations	
`MPD/BaseURL` at reference point **L**	**Not present** or **“./”**
`MPD/BaseURL` at reference point **A**	**Not present** or **“./”**
Example outcome	
Media segment URL at reference point **L**	**https://mg1234.isp.net/session9876/provider.cdn.net/s1/**prd1/adp1/rep1/1234.m4s
Media segment URL at reference point **A**	**https://provider.cdn.isp.net/s1/**prd1/adp1/rep1/1234.m4s

27. The *Multicast gateway* leaves the multicast transport session(s) when the *Content playback* function stops requesting segments and the session is deemed finished.

5.2.4 Multicast reception sequence

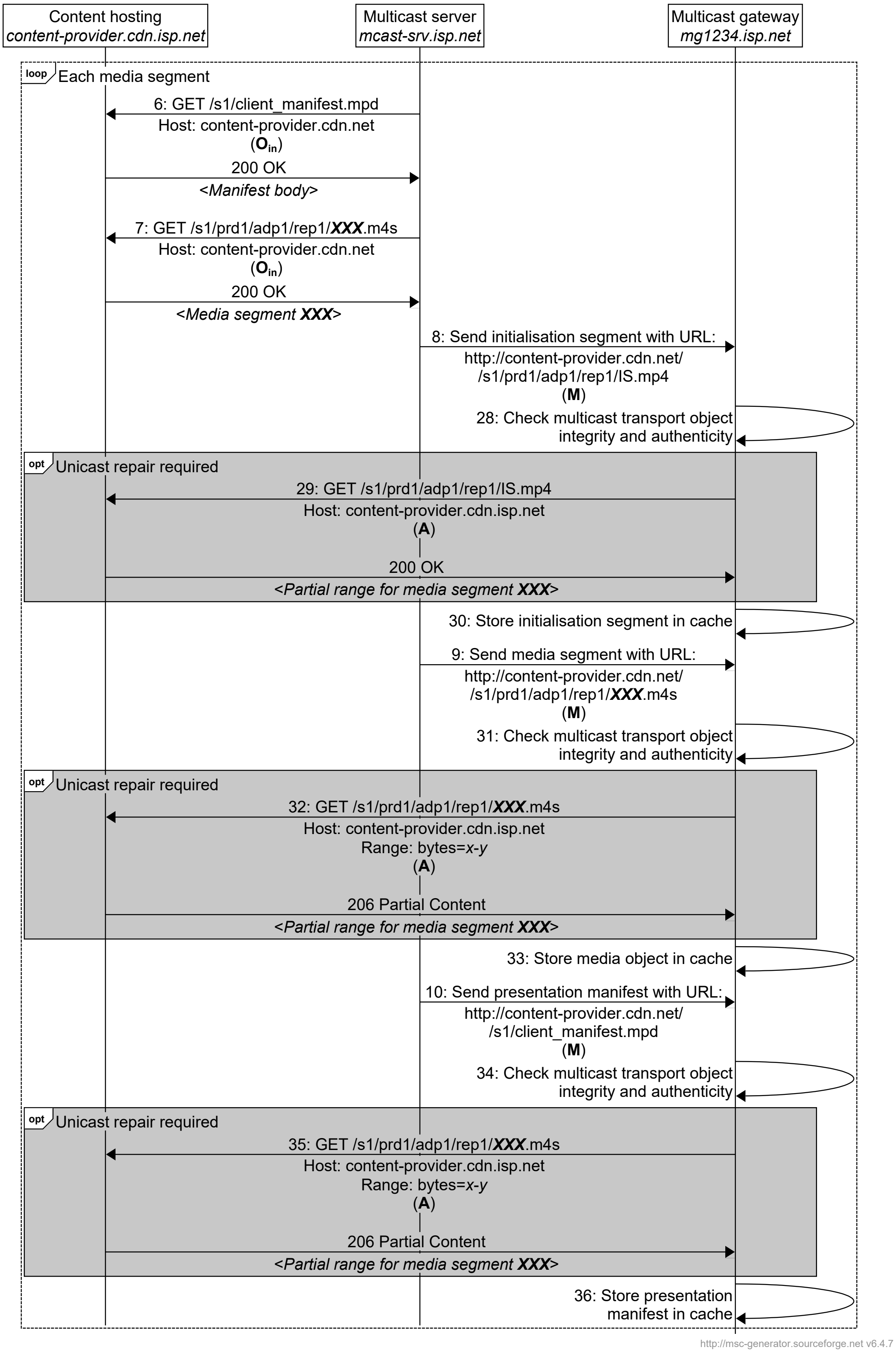

Content hosting
content-provider.cdn.isp.net
Multicast server
mcast-srv.isp.net
Multicast gateway
mg1234.isp.net
loop Each media segment
6: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(O_in)
200 OK
<Manifest body>
7: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: content-provider.cdn.net
(O_in)
200 OK
<Media segment XXX>
8: Send initialisation segment with URL:
http://content-provider.cdn.net/
/s1/prd1/adp1/rep1/IS.mp4
(M)
28: Check multicast transport object
integrity and authenticity
opt Unicast repair required
29: GET /s1/prd1/adp1/rep1/IS.mp4
Host: content-provider.cdn.isp.net
(A)
200 OK
<Partial range for media segment XXX>
30: Store initialisation segment in cache
9: Send media segment with URL:
http://content-provider.cdn.net/
/s1/prd1/adp1/rep1/XXX.m4s
(M)
31: Check multicast transport object
integrity and authenticity
opt Unicast repair required
32: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: content-provider.cdn.isp.net
Range: bytes=x-y
(A)
206 Partial Content
<Partial range for media segment XXX>
33: Store media object in cache
10: Send presentation manifest with URL:
http://content-provider.cdn.net/
/s1/client_manifest.mpd
(M)
34: Check multicast transport object
integrity and authenticity
opt Unicast repair required
35: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: content-provider.cdn.isp.net
Range: bytes=x-y
(A)
206 Partial Content
<Partial range for media segment XXX>
36: Store presentation
manifest in cache
http://msc-generator.sourceforge.net v6.4.7

Figure 5.2.4-1: Multicast reception sequence

The steps in the *Multicast gateway* reception sequence are as follows:

28. Once the *Multicast gateway* has fully received the multicast transport object carrying the MPEG-DASH initialisation segment on the multicast transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

29. Should the *Multicast gateway* not receive a complete intact multicast transport object over reference point **M** (see clause 5.2), or if the integrity check performed in step 28 should fail, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1].

Table 5.2.4-1 below shows how the multicast transport object URI at reference point **M** is identical to the unicast repair URL at reference point **A** because the `UnicastRepairParameters` in the multicast gateway configuration does not explicitly provide a unicast repair base URL or transport object base URI.

Table 5.2.4-1: Mapping of multicast transport object URIs to unicast repair URLs

Multicast session configuration	
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.net/**
`UnicastRepairParameters` `@transportObjectBaseURI`	**https://content-provider.cdn.net/**
Example outcome	
Content ingest URL at reference point **O$_{in}$**	**https://content-provider.cdn.net/**s1 /prd1/adp1/rep1/IS.mp4
Multicast transport object URI at reference point **M**	**https://content-provider.cdn.net/**s1 /prd1/adp1/rep1/IS.mp4

30. The MPEG-DASH initialisation segment carried as a multicast transport object (see step 8 in clause 6.2) is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* then uses this cached object to serve the MPEG-DASH initialisation segment to *Content playback* devices, as shown by step 19 in clause 6.3.1.

A multicast transport object carrying a media segment is received from the *Multicast server*:

31. Once the *Multicast gateway* has fully received the multicast transport object carrying the media object on the multicast transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

32. Should the *Multicast gateway* not receive a complete intact multicast transport object over reference point **M** (see clause 5.2), or if the integrity check performed in step 31 should fail, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1].

33. The media object carried as a multicast transport object is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* then uses this cached object to serve the MPEG-DASH initialisation segment to *Content playback* devices as shown by step 22 in clause 6.3.

A multicast transport object carrying an updated presentation manifest is received from the *Multicast server*:

34. Once the *Multicast gateway* has fully received the multicast transport object carrying the presentation manifest describing the media object received in step 10 and step 31 on the multicast transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

35. Should the *Multicast gateway* not receive a complete intact multicast transport object over reference point **M** (see clause 5.2), or if the integrity check performed in step 31 should fail, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1].

36. The presentation manifest carried as a multicast transport object is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* then uses this cached object to serve the MPEG-DASH initialisation segment to *Content playback* devices as step 15 or step 24 in clause 6.3.1. The media object received in step 10 is now available for Content playback devices to request as shown by step 22 in clause 6.3.

5.3 Push ingest with fused edge Multicast gateway

5.3.0 Introduction

Figure 5.3.0-1 below illustrates a vertically integrated deployment in which the *Multicast gateway* is deployed in the Service Provider domain and is co-located with the *Multicast rendezvous service*.

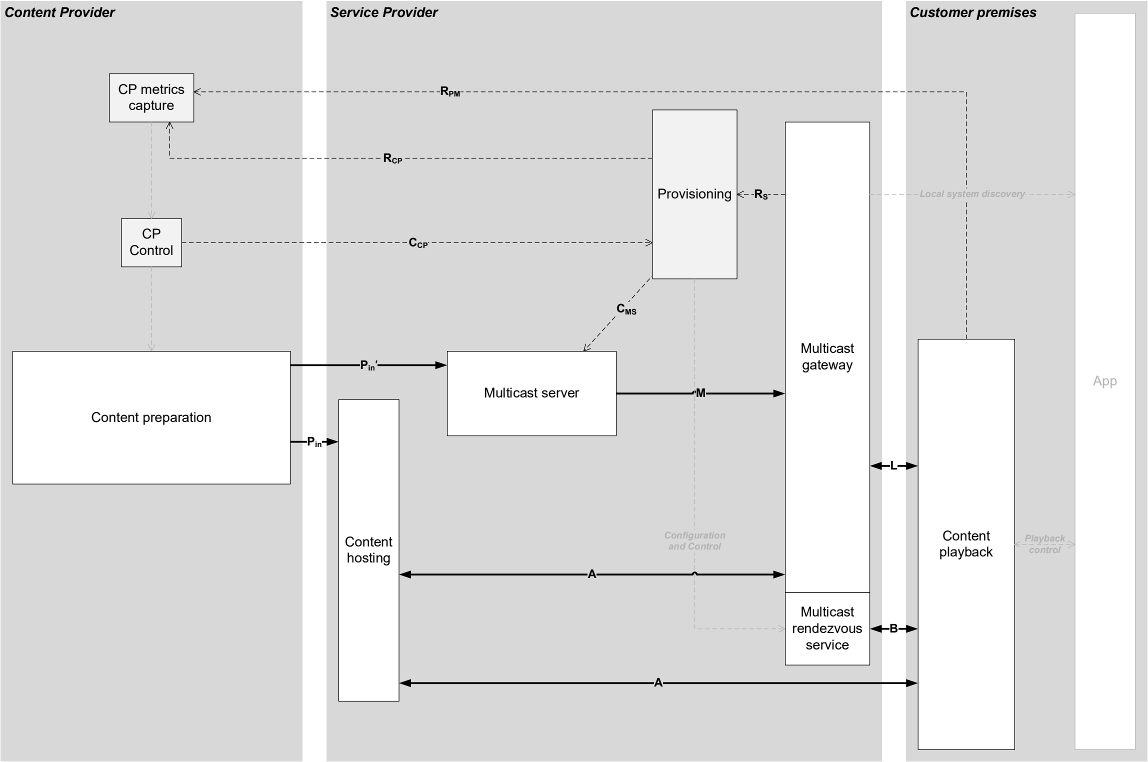


Figure 5.3.0-1: Deployment architecture diagram for vertically integrated deployment

Table 5.3.0-1 below shows example fully-qualified domain names for the key functions:

Table 5.3.0-1: Logical function owners and FQDNs for vertically integrated deployment

Logical function	Domain	FQDN
Provisioning	Service provider	N/A
Multicast rendezvous service		dvb-rv.isp.net
Content hosting		content-provider.cdn.isp.net
Multicast server		mcast-srv.isp.net
Multicast gateway		mg1234.isp.net
Content playback	Customer premises	N/A

In this example, the following assumptions are made:

- The multicast service carries an MPEG-DASH media presentation.
- The *Multicast server* operates the push-based content ingest method specified in clause 8.3.1 of [i.1].
- The media segments, presentation manifest and initialisation segments are delivered over multicast reference point **M** and may also be retrieved over reference point **A**.
- The service provider allows consumers to subscribe to sufficient numbers of multicast groups that each service component is carried on its own multicast group.
- The multicast gateway configuration transport session described in clause 10.2.5 of [i.1] is used.
- The *Multicast rendezvous service* is co-located with the *Multicast gateway* as described in clause 7.2 of [i.1].

- The *Multicast gateway* behaves as an HTTP(S) reverse caching proxy, and the *Asset storage* subfunction of the *Multicast gateway* acts as a HTTP object cache.

5.3.1 Provisioning sequence

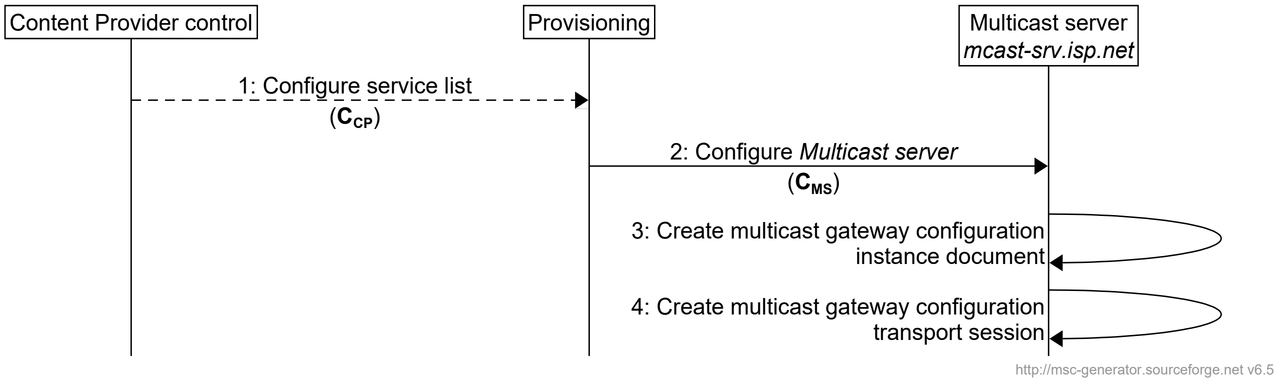

Content Provider control
Provisioning
Multicast server
mcast-srv.isp.net
1: Configure service list
(CCP)
2: Configure Multicast server
(CMS)
3: Create multicast gateway configuration instance document
4: Create multicast gateway configuration transport session
http://msc-generator.sourceforge.net v6.5

Figure 5.3.1-1: Multicast server provisioning sequence

The *Multicast server* is first provisioned using the following steps:

1. The *Content Provider control* function configures the *Provisioning* function with the list of linear services to be delivered via multicast.

2. **C_MS** procedures are used to supply a multicast server configuration instance document to the *Multicast server* that describes the set of multicast sessions and multicast transport sessions that it is to transmit. This configuration references the presentation manifests needed to fetch ingest media objects and start transmitting multicast transport objects over reference point **M**.

3. The *Multicast server* creates a multicast gateway configuration instance document from the supplied multicast server configuration, per the in-band configuration method described in clause 10.1.2 of [i.1].

4. The *Multicast server* creates a multicast gateway configuration transport session as described in clause 8.3.5 of [i.1].

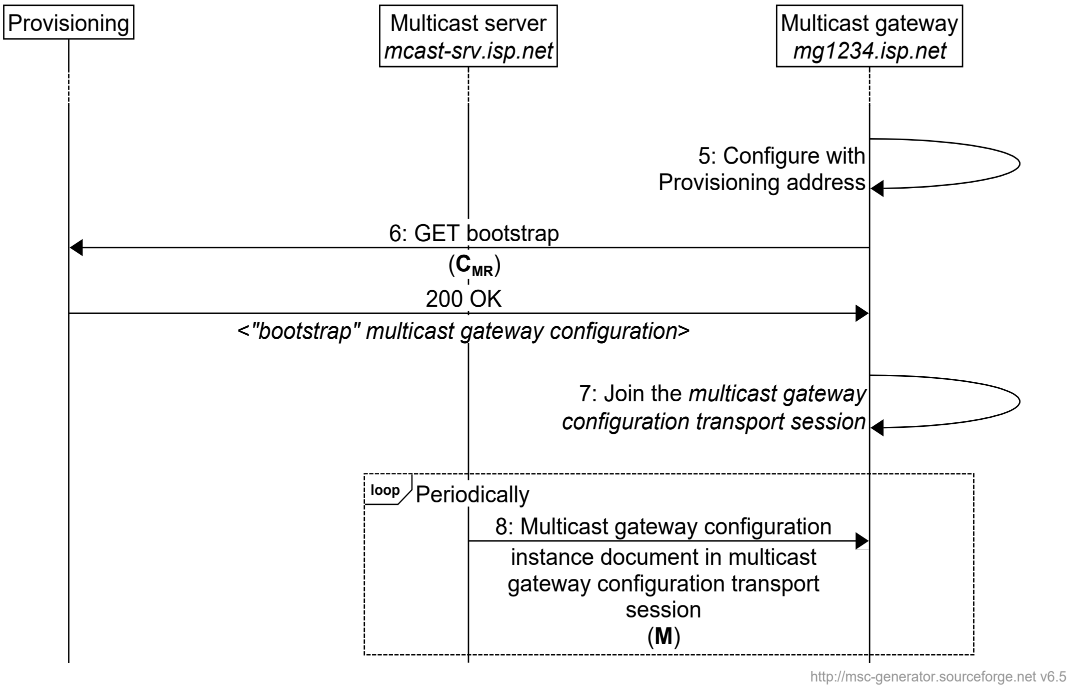

Provisioning
Multicast server
mcast-srv.isp.net
Multicast gateway
mg1234.isp.net
5: Configure with Provisioning address
6: GET bootstrap
(CMR)
200 OK
<"bootstrap" multicast gateway configuration>
7: Join the multicast gateway configuration transport session
loop
Periodically
8: Multicast gateway configuration instance document in multicast gateway configuration transport session
(M)
http://msc-generator.sourceforge.net v6.5

Figure 5.3.1-2: Multicast gateway provisioning sequence

The provisioning of the *Multicast gateway* can now start.

5. The *Multicast gateway* is statically configured with the address of the *Provisioning* function, for example using TR-069 [i.4] or TR-369 [i.5].

6. Using the interface **C$_{MR}$**, the *Multicast gateway* requests the simple "bootstrap" multicast gateway configuration instance document as described in clause 10.4.5 of [i.1].

7. The *Multicast gateway* joins the multicast gateway configuration transport session using the information collected in step 6.

8. The *Multicast gateway* acquires the full multicast gateway configuration from the multicast gateway configuration transport session at reference point **M** as described in clause 8.3.5 of [i.1].

5.3.2 Multicast server transmission sequence

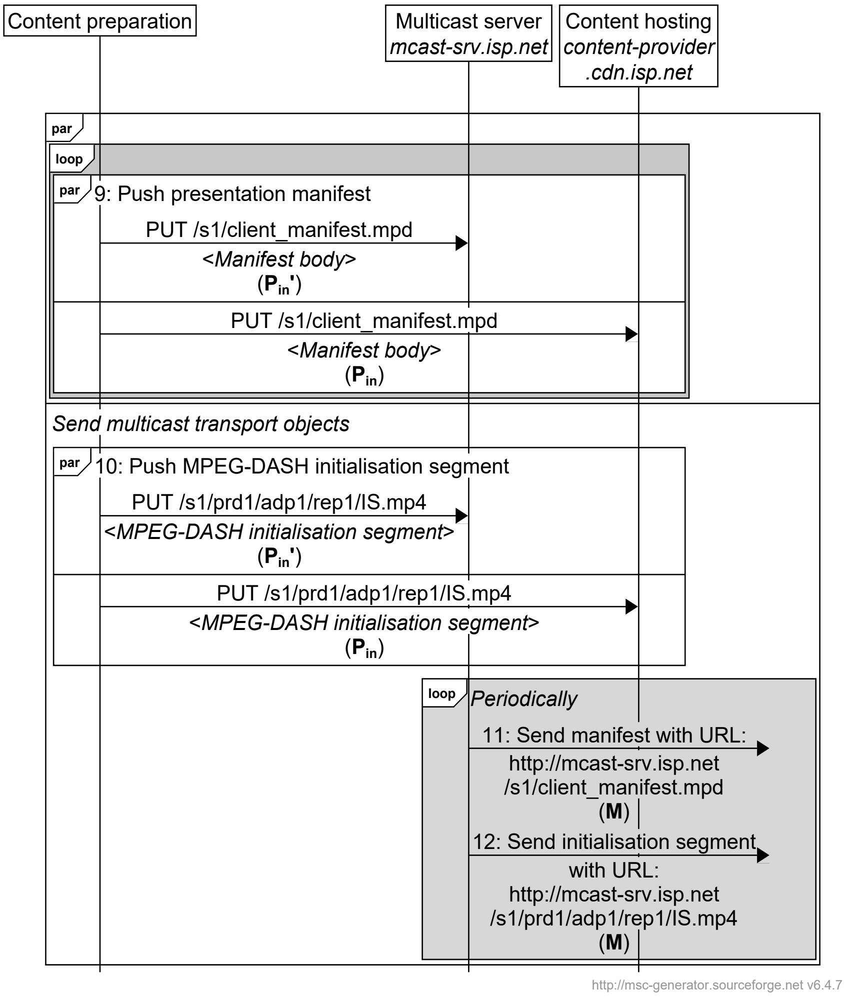

Content preparation
Multicast server
mcast-srv.isp.net
Content hosting
content-provider
.cdn.isp.net
par
loop
par
9: Push presentation manifest
PUT /s1/client_manifest.mpd
<Manifest body>
(P_in')
PUT /s1/client_manifest.mpd
<Manifest body>
(P_in)
Send multicast transport objects
par
10: Push MPEG-DASH initialisation segment
PUT /s1/prd1/adp1/rep1/IS.mp4
<MPEG-DASH initialisation segment>
(P_in')
PUT /s1/prd1/adp1/rep1/IS.mp4
<MPEG-DASH initialisation segment>
(P_in)
loop
Periodically
11: Send manifest with URL:
http://mcast-srv.isp.net
/s1/client_manifest.mpd
(M)
12: Send initialisation segment
with URL:
http://mcast-srv.isp.net
/s1/prd1/adp1/rep1/IS.mp4
(M)
http://msc-generator.sourceforge.net v6.4.7

Figure 5.3.2-1: Multicast gateway configuration transport session transmission sequence

The presentation manifest for the session, as well as all the initialisation segments it references, are transmitted over the multicast gateway configuration transport session as follows:

9. The *Content preparation* function creates and then pushes the presentation manifest for the service to the *Multicast server* at the path indicated in the multicast server configuration delivered in step 1 using the push-based content ingest mode via reference point **P$_{in}$′** as specified in clause 8.3.1 of [i.1]. In addition, the presentation manifest is pushed to the *Content hosting* function via reference point **P$_{in}$**.

NOTE: This operation is subsequently repeated as and when the presentation manifest is updated by the *Content preparation* function.

10. The *Content preparation* function creates and then pushes all the MPEG-DASH initialisation segments for the service to the *Multicast server* at the paths indicated in the multicast server configuration delivered in step 2 using the push-based content ingest mode via reference point **P$_{in}$′** as specified in clause 8.3.1 of [i.1]. In

addition, the MPEG-DASH initialisation segments are pushed to the *Content hosting* function at the path indicated in the presentation manifest via reference point $\mathbf{P_{in}}$.

11. The *Multicast server* regularly transmits the current presentation manifest(s) as multicast transport objects over reference point **M** using the multicast gateway configuration transport session as specified in clause 8.3.5 of [i.1].

12. The *Multicast server* regularly transmits initialisation segment(s) as multicast transport objects over reference point **M** using the multicast gateway configuration transport session as specified in clause 8.3.5 of [i.1].

In parallel with steps 11–12 above, the media objects are transmitted over a multicast transport session as follows:

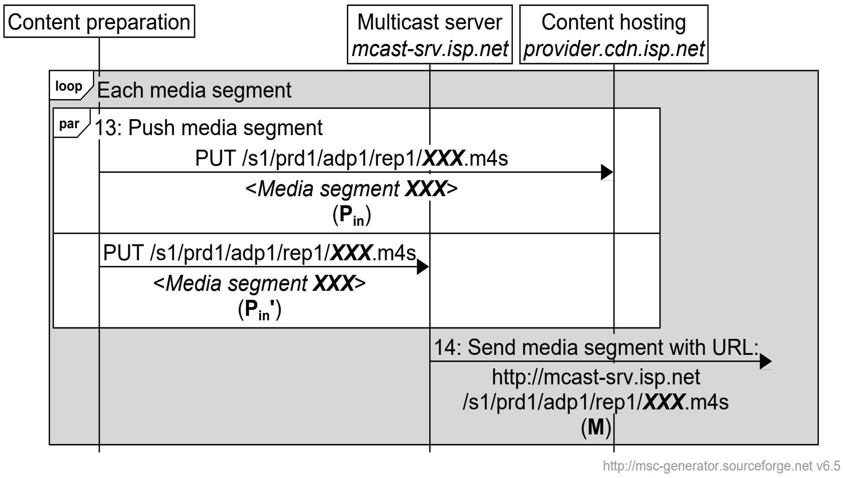


Figure 5.3.2-2: Multicast transport session transmission sequence

13. The *Content preparation* function regularly creates and then publishes media objects to the *Multicast server* at the paths indicated in the multicast server configuration delivered in step 2 using the push-based content ingest mode via reference point $\mathbf{P_{in}'}$ as specified in clause 8.3.1 of [i.1]. In addition, the media objects are pushed to the *Content hosting* function at the path indicated in the presentation manifest via reference point $\mathbf{P_{in}}$.

14. The *Multicast server* payloads ingest media objects as multicast transport objects and transmits them over reference point M, as specified in clause 8.3.4 of [i.1], on the multicast group address(es) assigned to multicast transport session(s) described in the multicast server configuration delivered in step 2.

The *Multicast server* uses the **`UnicastRepairParameters/BaseURL`** and **`UnicastRepairParameters`** `@transportObjectBaseURI` carried in the multicast server configuration instance document to convert the URL used to push media objects over reference point $\mathbf{P_{in}'}$ into a multicast transport object URI at reference point **M**. Per clause 8.3.3 of [i.1], this mapping is implementation-specific; table 5.3.2-1 below shows one possible example.

Table 5.3.2-1: Mapping of media object URL to multicast transport object URI in the Multicast server

Multicast session configuration	
`UnicastRepairParameters/BaseURL`	http://content-provider.cdn.isp.net/
`UnicastRepairParameters` `@transportObjectBaseURI`	**http://mcast-srv.isp.net/**
Example outcome	
Content ingest URL at reference point $\mathbf{P_{in}'}$	**http://mcast-srv.isp.net/**s1/prd1/adp1/rep1/XXX.m4s
Multicast transport object URI at reference point **M**	**http://mcast-srv.isp.net/**s1/prd1/adp1/rep1/XXX.m4s

5.3.3 Multicast gateway playback sequence

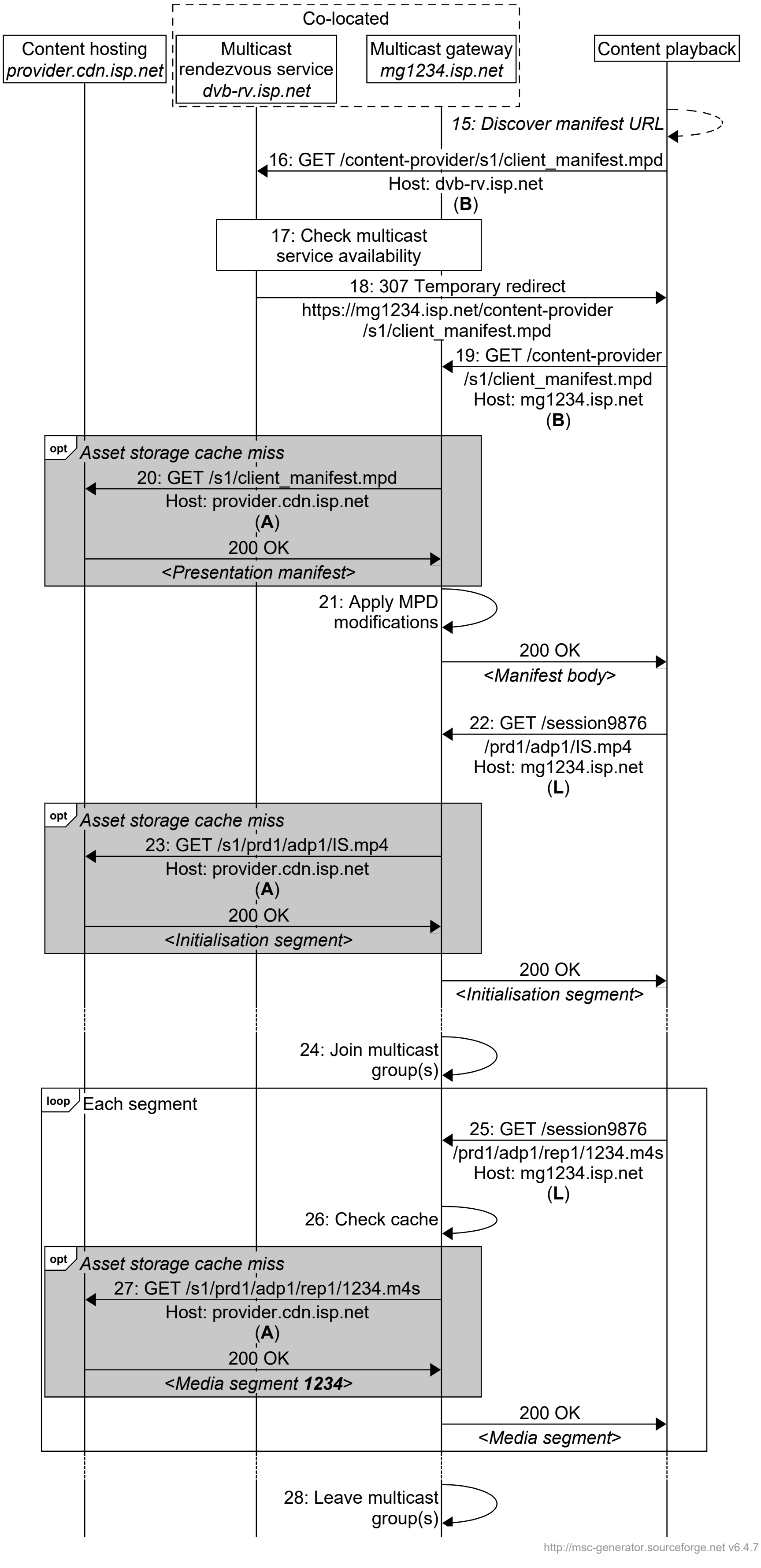

Co-located
Content hosting
provider.cdn.isp.net
Multicast rendezvous service
dvb-rv.isp.net
Multicast gateway
mg1234.isp.net
Content playback
15: Discover manifest URL
16: GET /content-provider/s1/client_manifest.mpd
Host: dvb-rv.isp.net
(B)
17: Check multicast service availability
18: 307 Temporary redirect
https://mg1234.isp.net/content-provider /s1/client_manifest.mpd
19: GET /content-provider /s1/client_manifest.mpd
Host: mg1234.isp.net
(B)
opt
Asset storage cache miss
20: GET /s1/client_manifest.mpd
Host: provider.cdn.isp.net
(A)
200 OK
<Presentation manifest>
21: Apply MPD modifications
200 OK
<Manifest body>
22: GET /session9876 /prd1/adp1/IS.mp4
Host: mg1234.isp.net
(L)
opt
Asset storage cache miss
23: GET /s1/prd1/adp1/IS.mp4
Host: provider.cdn.isp.net
(A)
200 OK
<Initialisation segment>
200 OK
<Initialisation segment>
24: Join multicast group(s)
loop
Each segment
25: GET /session9876 /prd1/adp1/rep1/1234.m4s
Host: mg1234.isp.net
(L)
26: Check cache
opt
Asset storage cache miss
27: GET /s1/prd1/adp1/rep1/1234.m4s
Host: provider.cdn.isp.net
(A)
200 OK
<Media segment 1234>
200 OK
<Media segment>
28: Leave multicast group(s)
http://msc-generator.sourceforge.net v6.4.7

Figure 5.3.3-1: Multicast gateway playback sequence

The steps in the playback sequence are as follows:

15. A playback session begins when a presentation manifest URL referencing the address of the *Multicast rendezvous service* and *Multicast gateway* is supplied to the *Content playback* function.

16. The *Content playback* function makes a unicast HTTPS request for the manifest at reference point **B**, as specified in clause 7.5.1 of [i.1].

17. The *Multicast gateway* co-located with the *Multicast rendezvous service* examines its multicast session configuration (acquired during step 8 in clause 5.3.1) and determines that there is an active multicast session corresponding to the reference point **L** presentation manifest URL. The *Multicast gateway* verifies that an active multicast session exists for the requested presentation manifest by finding an instance of **`PresentationManifestLocator`**`@contentPlaybackPathPattern` in the multicast gateway configuration where the path pattern matches the path of the presentation manifest URL received in step 16.

NOTE: In the case where there is no configured multicast session, the behaviour may be as described in clause 4.3.1.

Table 5.3.3-2: Manifest URLs over reference points L and A

Multicast session configuration	
Session **`PresentationManifestLocator`**	https://content-provider.cdn.isp.net /content-provider/s1/client_manifest.mpd
Content playback manifest request path pattern **`PresentationManifestLocator`** `@contentPlaybackPathPattern`	**/content-provider/s1/client_manifest.mpd**
Example outcome	
Redirected presentation manifest URL at reference point **L**	https://mg1234.isp.net **/content-provider/s1/client_manifest.mpd**
Presentation manifest URL at reference point **A**	https://content-provider.cdn.isp.net **/content-provider/s1/client_manifest.mpd**

18 When the requested content is available over multicast the *Multicast rendezvous service* redirects the *Content playback* function to its co-located *Multicast gateway* functions.

19. The *Content playback* function makes a unicast HTTPS request for the manifest at reference point **B**.

20. If the *Multicast gateway* does not have a cached copy of the requested MPEG-DASH presentation manifest, it may request it from the *Content hosting* function at reference point **A**.

21. The *Multicast gateway* may modify the presentation manifest for the reasons given in clause 8.4.1 and clause 10.2.2.0 of [i.1]. The modifications needed are signalled in the multicast session configuration, and in this example include adding a session identifier to the **`MPD/BaseURL`** element for the *Multicast gateway* to track individual client sessions. The modification to the **`MPD/BaseURL`** is given in Table 7.3-3 below.

22. The *Content playback* function makes a request for the MPEG-DASH initialisation segment(s) for its chosen representation(s) via reference point **L**.

23. If the *Multicast gateway* does not have a cached copy of the requested MPEG-DASH initialisation segment(s), it may request it/them from the *Content hosting* function at reference point **A**.

Table 5.3.3-3 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. This process is dependent on how the manifest was modified before returning to the client. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **A** from the information present in the MPD.

Table 5.3.3-3: Mapping of MPEG-DASH initialisation segment URLs

MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://content-provider.cdn.isp.net/s1/**
`MPD/BaseURL` at reference point **L**	**https://mg1234.isp.net/session9876/**
Example outcome	
Initialisation segment URL at reference point **L**	**https://mg1234.isp.net/session9876/** prd1/adp1/IS.mp4
Initialisation segment URL at reference point **A**	**https://content-provider.cdn.isp.net/s1/** prd1/adp1/IS.mp4

24. The *Multicast gateway* joins the multicast transport sessions indicated in its multicast session configuration as specified in clause 8.4.2 of [i.1].

25. The *Content playback* function makes requests for MPEG-DASH media segments at reference point **L**.

26. The *Multicast gateway* checks whether it has a cached copy of the requested media object in its *Asset storage*, either received over reference point **M** or previously requested via reference point **A** in response to a cache miss when servicing a request by another *Content playback* function as detailed in step 27 below.

27. If the requested media object is not cached in the *Asset storage* subfunction, then the *Multicast gateway* makes a unicast request for the media object from the *Content hosting* via reference point **A**.

Table 5.3.3-4 below shows how the *Multicast gateway* transforms the URL at reference point **L** to the URL at reference point **A**. This process is dependent on how the manifest was modified before returning to the client. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **A** from the information present in the MPD.

Table 5.3.3-4: Mapping of MPEG-DASH media segment URLs

MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://provider.cdn.isp.net/s1/**
`MPD/BaseURL` at reference point **L**	**https://mg1234.isp.net/session9876/**
Example outcome	
Media segment URL at reference point **L**	**https://mg1234.isp.net/session9876/** prd1/adp1/rep1/1234.m4s
Media segment URL at reference point **A**	**https://provider.cdn.isp.net/s1/** prd1/adp1/rep1/1234.m4s

28. The *Multicast gateway* leaves the multicast transport sessions when the *Content playback* function stops requesting segments and the session is deemed finished.

5.3.4 Multicast reception sequence

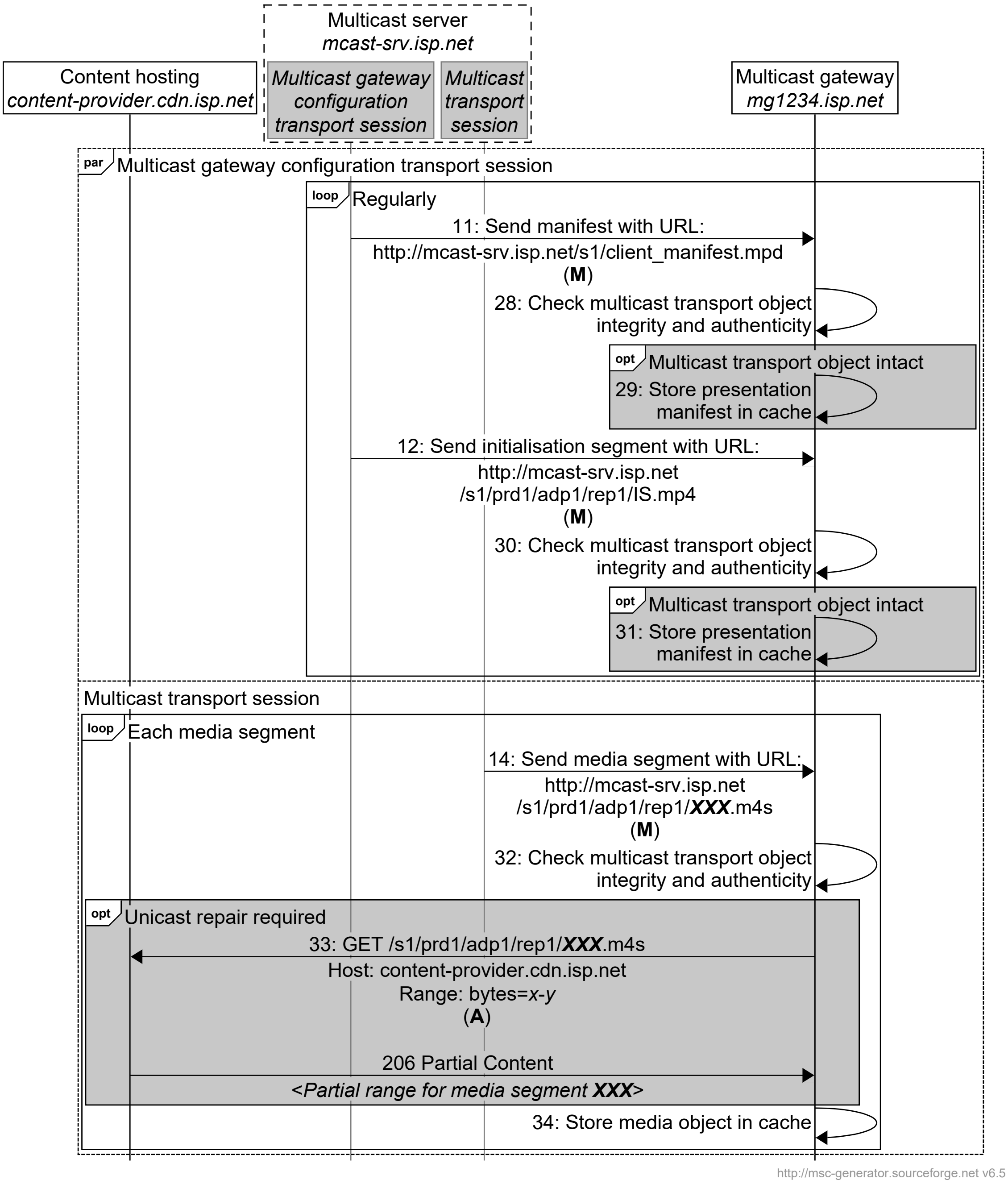

Multicast server
mcast-srv.isp.net
Content hosting
content-provider.cdn.isp.net
Multicast gateway configuration transport session
Multicast transport session
Multicast gateway
mg1234.isp.net
par Multicast gateway configuration transport session
loop Regularly
11: Send manifest with URL:
http://mcast-srv.isp.net/s1/client_manifest.mpd
(M)
28: Check multicast transport object integrity and authenticity
opt Multicast transport object intact
29: Store presentation manifest in cache
12: Send initialisation segment with URL:
http://mcast-srv.isp.net
/s1/prd1/adp1/rep1/IS.mp4
(M)
30: Check multicast transport object integrity and authenticity
opt Multicast transport object intact
31: Store presentation manifest in cache
Multicast transport session
loop Each media segment
14: Send media segment with URL:
http://mcast-srv.isp.net
/s1/prd1/adp1/rep1/XXX.m4s
(M)
32: Check multicast transport object integrity and authenticity
opt Unicast repair required
33: GET /s1/prd1/adp1/rep1/XXX.m4s
Host: content-provider.cdn.isp.net
Range: bytes=x-y
(A)
206 Partial Content
<Partial range for media segment XXX>
34: Store media object in cache
http://msc-generator.sourceforge.net v6.5

Figure 5.3.4-1: Multicast reception sequence

The reception sequence for the presentation manifest and MPEG-DASH initialisation segments is as follows:

28. Once the *Multicast gateway* has fully received the multicast transport object carrying the presentation manifest on the multicast gateway configuration transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

29. If the integrity check performed in step 28 succeeds, then the received presentation manifest is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* uses this cached object to serve the presentation manifest to *Content playback* devices as shown in step 19 of clause 5.3.3.

30. Once the *Multicast gateway* has fully received the multicast transport object carrying the MPEG-DASH initialisation segment on the multicast gateway configuration transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

31. If the integrity check performed in step 30 succeeds, then the received MPEG-DASH initialisation segment is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* uses this cached object to serve the presentation manifest to *Content playback* devices as shown in step 21 of clause 5.3.3.

If the *Multicast gateway* does not receive a complete multicast transport object containing either the presentation manifest or MPEG-DASH initialisation segment, then it does not store the respective object in its cache and will instead wait for the presentation manifest to be sent on the object carousel again.

NOTE 1: In this example, there is no unicast repair performed for the presentation manifest nor the MPEG-DASH initialisation segment received over reference point **M**. The *Multicast gateway* will be unable to serve the presentation manifest to the *Content playback* function as shown in step 19 of clause 5.3.3 until it has received a complete presentation manifest from the multicast gateway configuration transport session.

NOTE 2: If it is not available in the *Asset storage* cache, the MPEG-DASH initialisation segment may be requested over reference point **A** in response to a request from the *Content playback* function as shown in step 22 of clause 5.3.3.

The reception sequence for media objects is as follows:

32. Once the *Multicast gateway* has fully received a multicast transport object carrying a media object on the multicast transport session, it performs any applicable integrity and authenticity checks, such as verifying an object digest hash.

33. Should the *Multicast gateway* not receive a complete intact multicast transport object over reference point **M** (see clause 5.2), or if the integrity check performed in step 32 should fail, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1].

Table 5.3.4-1 below shows how the *Multicast gateway* uses the `UnicastRepairParameters` in the multicast gateway configuration to transform the multicast transport object URI at reference point **M** to the unicast repair URL at reference point **A**.

Table 5.3.4-1: Mapping of multicast transport object URIs to unicast repair URLs

Multicast session configuration	
`UnicastRepairParameters` `@transportObjectBaseURI`	**http://mcast-srv.isp.net/s1**
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.isp.net/s1**
Example outcome	
Multicast transport object URI at reference point **M**	**http://mcast-srv.isp.net/s1**/prd1/adp1/rep1/1234.m4s
Unicast repair URL at reference point **A**	**https://content-provider.cdn.isp.net/s1** /prd1/adp1/rep1/1234.m4s

34. The media object received via reference point **M** (and possibly repaired via reference point **A**) is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4 of [i.1]. The *Multicast gateway* then uses this cached object to serve the MPEG-DASH initialisation segment to *Content playback* devices as shown by steps 24 and 25 in clause 5.3.3.

5.4 Push ingest with multiple consumers

5.4.0 Introduction

Figure 5.4.0-1 below illustrates an over-the-top (or horizontally integrated) deployment in which the system operates across several domains, comprising one content provider, one content distribution network, and two service providers. One of the service providers operates a conventional fixed-line IP network, and another operator providing digital television services via a unidirectional satellite service.

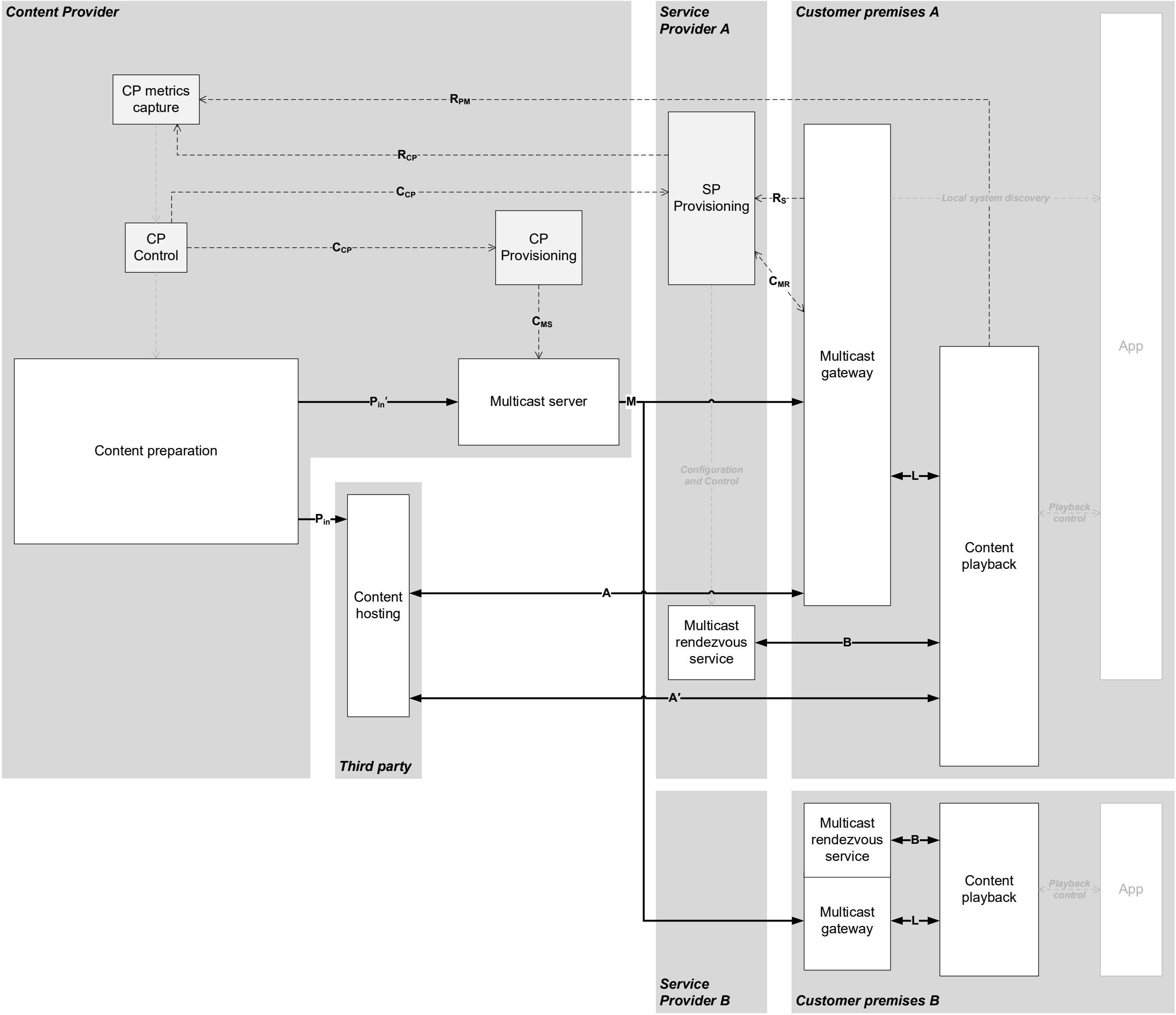


Figure 5.4.0-1: Deployment architecture diagram for over-the-top deployment

Table 5.4.0-1 below shows example Fully-Qualified Domain Names for those functions:

Table 5.4.0-1: Logical function owners and FQDNs for over-the-top deployment

Logical function	Owner	FQDN
CP Provisioning	Content Provider	dvb-provisioning.content-provider.com
Content preparation	Content Provider	enc.content-provider.com
Content hosting	Third party	content-provider.cdn.net
Multicast server	Content Provider	mcast-srv.content-provider.com
Multicast rendezvous service	Service Provider A	dvb-rndz.isp.net
Multicast gateway (Customer premises A)	Customer A	sub1234-net987.isp.net
Content playback (Customer premises A)	Customer A	N/A
Multicast gateway with co-located *Multicast rendezvous service* (Customerpremises B)	Service Provider B	N/A
Content playback (Customer premises B)	Customer B	N/A

In this example, the following assumptions are made:

1. The multicast service carries an MPEG-DASH media presentation, and all representations within each adaptation set use a common initialisation segment.

2. The *Content Provider Control* function is statically configured with the address of both the Content Provider’s *Provisioning* function and Service Provider A’s *Provisioning* function.

3. The third-party *Content hosting* function is also statically configured.

4. The Content Provider's *Multicast server* operates the push content ingest method specified in clause 8.3.1 of [i.1].

5. The presentation manifest and MPEG-DASH initialisation segments are delivered by the Content Provider's *Multicast server* over reference point **M** in a multicast gateway configuration transport session as described in clause 8.3.5 of [i.1].

   a. In the fixed-line case, the *Multicast gateway* does not use the multicast gateway configuration transport session and (in this example) it requests presentation manifests and MPEG-DASH initialisation segments over reference point **A** because these media objects are not delivered with the media segments.

6. Media segments are delivered by the Content Provider’s *Multicast server* over reference point **M** in multicast transport sessions that are common between all Service Provider networks that will deliver them.

   a. The multicast transport sessions carrying the media segments utilise Application-Level Forward Erasure Correction (AL-FEC) techniques as described in clause 8.3.4.2 of [i.1]. The AL-FEC scheme used is a systematic code and the source and repair blocks are sent on different multicast groups to enable receivers to opt in to receive them. In this example, only the unidirectional system (Service Provider B carries and makes use of the AL-FEC repair codes; AL-FEC is not advertised in the multicast gateway configuration sent to *Multicast gateway* instances in Service Provider A’s network.

   b. In the case of the fixed-line network, the *Multicast gateway* in customer premises A may also retrieve media components over reference point **A** from the *Content hosting* function deployed in the fixed-line operator's network, for example to affect unicast repair as described in clause 9 of [i.1].

   c.In the case of the satellite distribution system, the *Multicast gateway* in customer premises B has no unicast repair channel and as such cannot retrieve media components over reference point **A** as specified in clause 9.2 of [i.1]. The multicast transport objects are instead protected by a Forward Erasure Correction scheme operating on the satellite transport protocol.

7. The *Multicast gateway* behaves as an HTTP(S) reverse caching proxy, and the *Asset storage* subfunction of the *Multicast gateway* acts as a HTTP object cache.

   a. In the case of the fixed-line network, the *Multicast gateway* in customer premises A is deployed in the customer's home gateway equipment as described in clause 6.2 of [i.1].

b. In the case of the satellite distribution system, the *Multicast gateway* in customer premises B is deployed in the customer's terminal device as described in clause 6.3 of [i.1].

5.4.1 Provisioning sequences

5.4.1.1 Content Provider and Service Provider A provisioning sequence

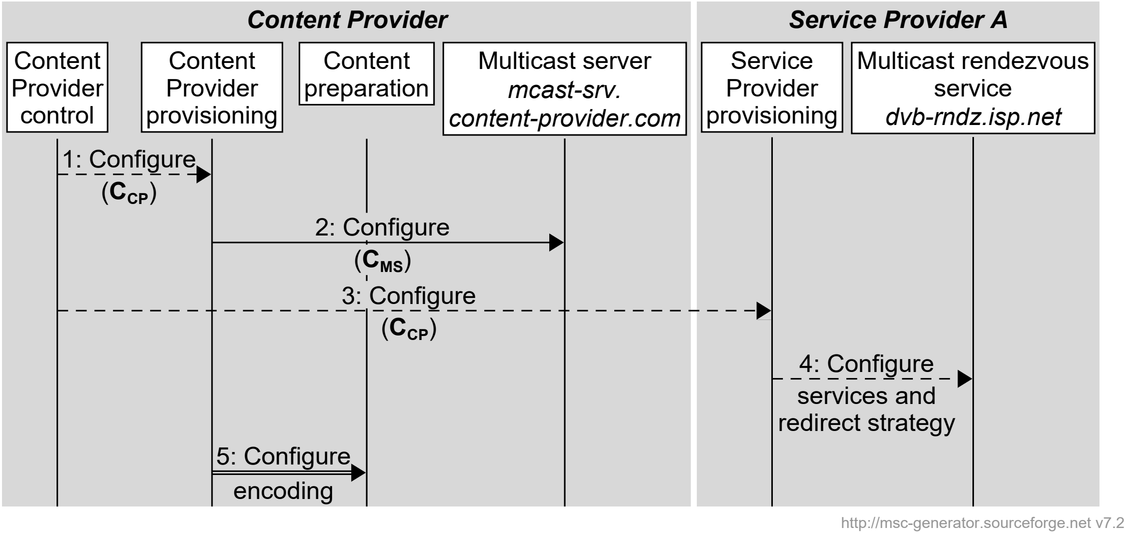


Figure 5.4.1.1-1: Content Provider and Service Provider A provisioning sequence

The steps in the configuration sequence are as follows:

1. The *Content Provider control* function configures the Content Provider's *Provisioning* function via reference point **C**$_{CP}$ with the set of services that the Content Provider wishes to provide. This includes details of each individual service to correctly configure the *Multicast server* function, as well as a list of subnets that those services are to be provided to using IP multicast.

2. The Content Provider's *Provisioning* function uses the out-of-band pushed multicast server configuration method defined in clause 10.4.2.2 over reference point **C**$_{MS}$ to supply a multicast server configuration instance document to the *Multicast server* that describes the set of multicast sessions and multicast transport sessions that it is to transmit.

   This configuration references the URLs used to deliver presentation manifests, media objects and any other ingest objects over reference point **P**$_{in}$**′** for transmission as multicast transport objects at reference point **M**.

3. The *Content Provider control* function configures Service Provider A's *Provisioning* function via reference point **C**$_{CP}$ with the set of services that the Content Provider wishes to provide to subscribers within Service provider A's fixed-line network.

NOTE: This step is not performed in Service Provider B's network.

4. Service Provider A's *Provisioning* function configures the *Multicast rendezvous service* to support redirecting *Content playback* functions to the media presentation.

NOTE: This step is not performed in Service Provider B's network.

5. The Content Provider's *Provisioning* function configures the *Content preparation* function and instructs it to begin encoding the media presentation.

5.4.1.2 Multicast gateway provisioning sequence for fixed-line consumer

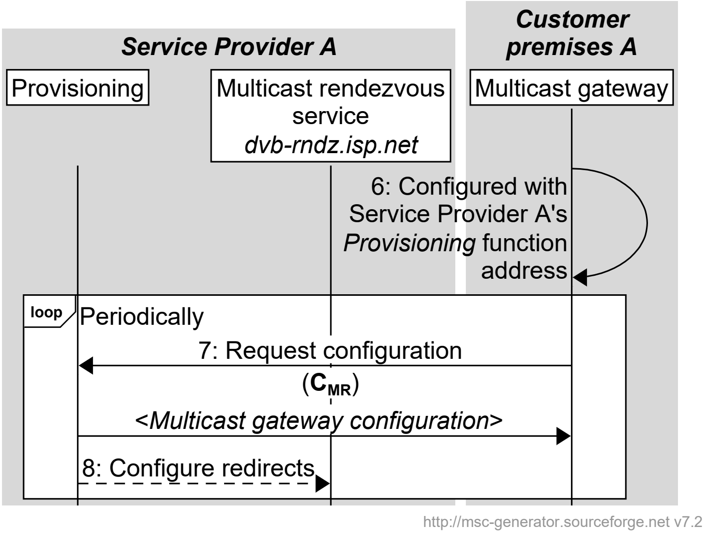

Service Provider A
Customer premises A
Provisioning
Multicast rendezvous service dvb-rndz.isp.net
Multicast gateway
6: Configured with Service Provider A's Provisioning function address
loop
Periodically
7: Request configuration
(C_MR)
<Multicast gateway configuration>
8: Configure redirects
http://msc-generator.sourceforge.net v7.2

Figure 5.4.1.2-1: Fixed-line consumer provisioning sequence

6. The *Multicast gateway* is configured with the address of Service Provider A's *Provisioning* function, for example using TR-069 [i.2] or TR-369 [i.3].

7. The *Multicast gateway* instance periodically requests an updated multicast gateway configuration using the out-of-band pulled multicast gateway configuration method including the redirect-subnet and redirect-base-url query parameters described in clause 10.4.4.2 of [i.1]. Service Provider A's *Provisioning* function returns a multicast gateway configuration describing the set of sessions that the *Multicast gateway* is expected to serve.

8. When Service Provider A's *Provisioning* function receives a configuration request from a *Multicast gateway* function that includes the query parameters specified for the second request URI form in clause 10.4.4.2 of [i.1], it pushes an updated configuration to the *Multicast rendezvous service* that includes the subnet and base URLs provided by the *Multicast gateway* in step 7.

5.4.1.3 Multicast gateway provisioning sequence for satellite consumer

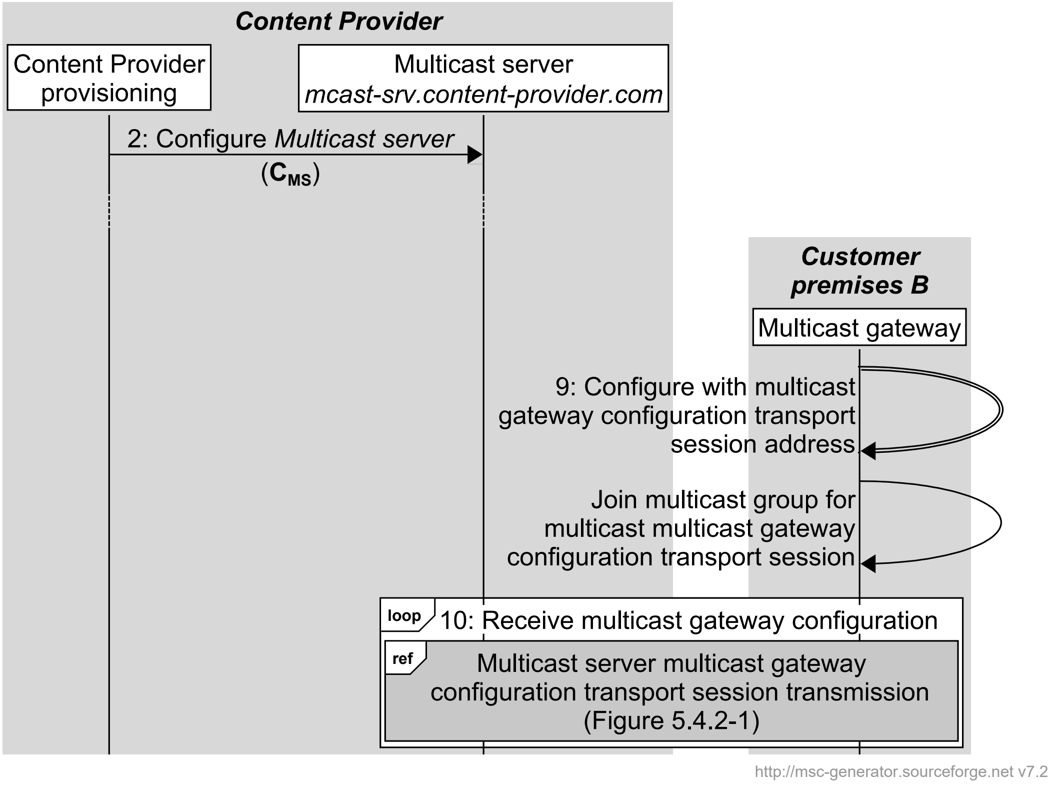

Content Provider
Content Provider provisioning
Multicast server
mcast-srv.content-provider.com
2: Configure Multicast server
(C_MS)
Customer premises B
Multicast gateway
9: Configure with multicast gateway configuration transport session address
Join multicast group for multicast multicast gateway configuration transport session
loop
10: Receive multicast gateway configuration
ref
Multicast server multicast gateway configuration transport session transmission (Figure 5.4.2-1)
http://msc-generator.sourceforge.net v7.2

Figure 5.4.1.3-1: Satellite consumer provisioning sequence

The steps in the configuration of the unidirectional Satellite customer’s *Multicast gateway* are as follows:

9. The *Multicast gateway* is configured with the address of the multicast gateway configuration transport session, for example by a “bootstrap” multicast gateway configuration carried out-of-band as described in clause 10.4.5 of [i.1]. The *Multicast gateway* joins the multicast group carrying the multicast gateway configuration transport session.

10. The *Multicast gateway* receives its multicast gateway configuration(s) from the multicast gateway configuration transport session as sent by the content provider’s *Multicast server*, described in step 13 in clause 5.4.2.

5.4.2 Multicast server transmission sequence

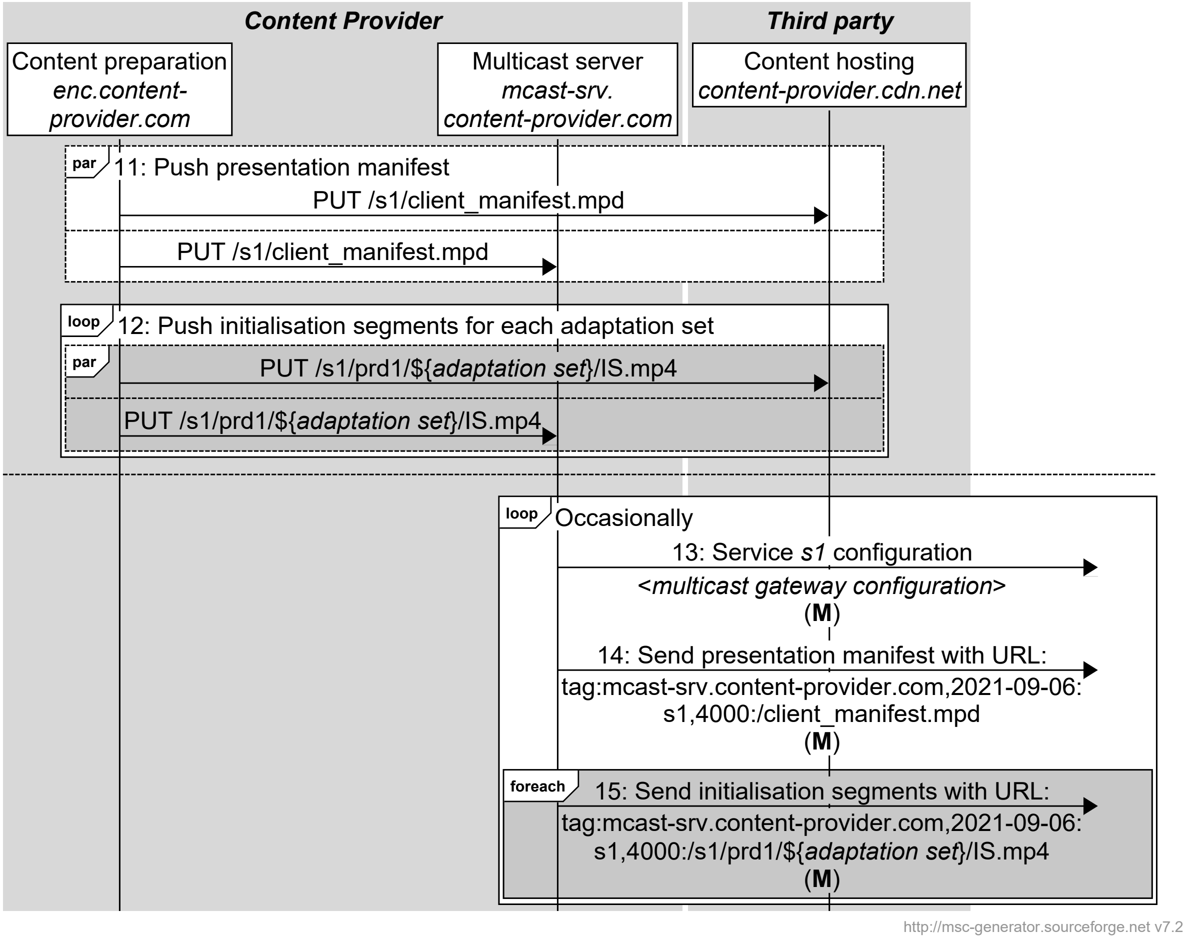

Content Provider
Third party
Content preparation
enc.content-provider.com
Multicast server
mcast-srv.
content-provider.com
Content hosting
content-provider.cdn.net
par
11: Push presentation manifest
PUT /s1/client_manifest.mpd
PUT /s1/client_manifest.mpd
loop
12: Push initialisation segments for each adaptation set
par
PUT /s1/prd1/${adaptation set}/IS.mp4
PUT /s1/prd1/${adaptation set}/IS.mp4
loop
Occasionally
13: Service s1 configuration
<multicast gateway configuration>
(M)
14: Send presentation manifest with URL:
tag:mcast-srv.content-provider.com,2021-09-06:
s1,4000:/client_manifest.mpd
(M)
foreach
15: Send initialisation segments with URL:
tag:mcast-srv.content-provider.com,2021-09-06:
s1,4000:/s1/prd1/${adaptation set}/IS.mp4
(M)
http://msc-generator.sourceforge.net v7.2

Figure 5.4.2-1: Multicast gateway configuration transport session transmission sequence

The presentation manifest for the session, as well as all the initialisation segments it references, are transmitted over the multicast gateway configuration transport session as follows:

11. The *Content preparation* function creates and then pushes the presentation manifest for the service to the *Multicast server* via reference point **P$_{in}$'** using the push-based content ingest mode (see clause 8.3.2 of [i.1]). The *Content preparation* function also pushes the presentation manifest to the *Content hosting* function over reference point **P$_{in}$**.

12. The *Content preparation* function creates and then pushes all the MPEG-DASH initialisation segments for all Adaptation Sets described in the presentation manifest that was generated in step 11 to the *Multicast server* using the push-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point **P$_{in}$'**. The *Content preparation* function also pushes the initialisation segments to the *Content hosting* function over reference point **P$_{in}$**.

13. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits the multicast gateway configurations for the services available over multicast as described in clause 8.3.5 of [i.1].

14. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits the presentation manifest for consumption by any *Multicast gateway* instances that are configured to receive their presentation manifest over reference point **M**, such as those in clause 5.4.3.2.

15. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits MPEG-DASH initialisation segments for all Adaptation Sets described in the presentation manifest sent in step 14 for consumption by any *Multicast gateway* instances that are configured to receive their initialisation segments over reference point **M**, such as those in clause 5.4.3.2.

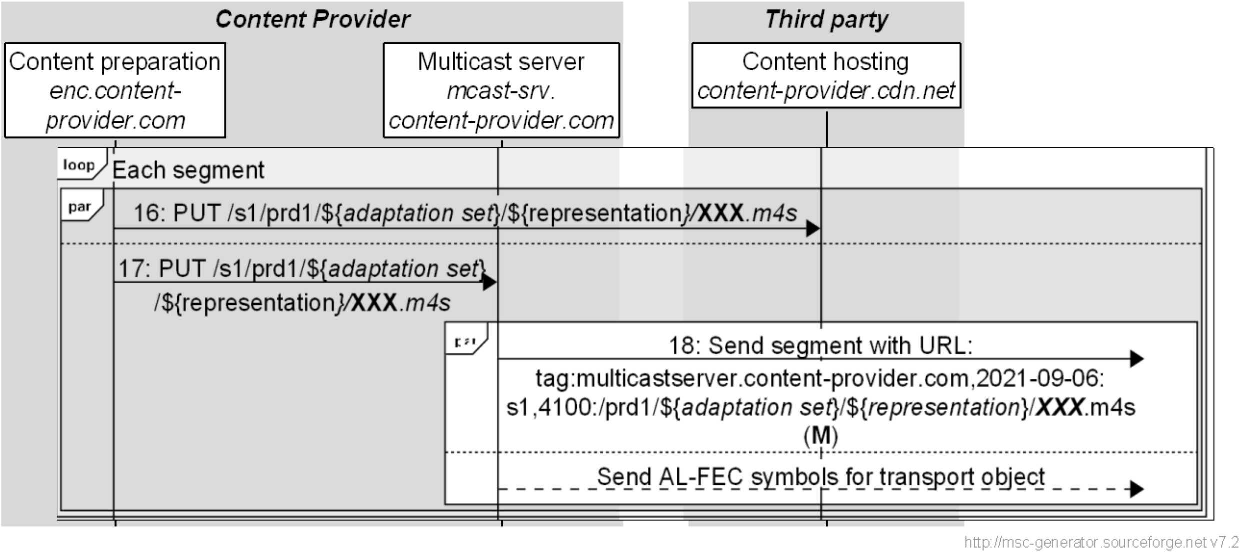

Content Provider
Third party
Content preparation
enc.content-provider.com
Multicast server
mcast-srv.content-provider.com
Content hosting
content-provider.cdn.net
loop
Each segment
par
16: PUT /s1/prd1/${adaptation set}/${representation}/XXX.m4s
17: PUT /s1/prd1/${adaptation set}/${representation}/XXX.m4s
18: Send segment with URL:
tag:multicastserver.content-provider.com,2021-09-06:s1,4100:/prd1/${adaptation set}/${representation}/XXX.m4s
(M)
Send AL-FEC symbols for transport object
http://msc-generator.sourceforge.net v7.2

Figure 5.4.2-2: Multicast transport session transmission sequence

In parallel with steps 13, 14 and 15 above, media segments are published as follows:

16. The *Content preparation* function creates and then pushes each media segment to the *Content hosting* function using reference point **P$_{in}$**.

17. In parallel with step 16, the *Content preparation* function pushes each media segment to the *Multicast server* using the push-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point **P$_{in}$′**.

18. The *Multicast server* serialises ingested media segments as multicast transport objects and transmits them over reference point **M**, as specified in clause 8.3.4 of [i.1], along with the corresponding AL-FEC symbols carried on another multicast address.

The *Multicast server* uses the **`UnicastRepairParameters/BaseURL`** and **`UnicastRepairParameters`** `@transportObjectBaseURI` carried in the multicast server configuration instance document to convert the URL on ingest media objects over reference point **P$_{in}$′** into a multicast transport object URI at reference point **M**. Per clause 8.3.3 of [i.1], this mapping is implementation-specific; table 5.4.2-1 below shows one possible example.

Table 5.4.2-1: Mapping of media object URL to multicast transport object URI in the Multicast server

Multicast session configuration	
`UnicastRepairParameters/BaseURL`	https://mcast-srv.content-provider.com/s1
`UnicastRepairParameters` `@transportObjectBaseURI`	**tag:mcast-srv.content-provider.com,2021-09-06:s1,4100:**
Example outcome	
Content ingest URL at reference point **O$_{in}$**	https://mcast-srv.content-provider.com/s1 /prd1/adp1/rep1/1234.m4s
Multicast transport object URI at reference point **M**	**tag:mcast-srv.content-provider.com,2021-09-06:s1,4100:** /prd1/adp1/rep1/1234.m4s

5.4.3 Multicast gateway playback sequences

5.4.3.1 Fixed-line customer playback sequence

The following example is described from the perspective of the *Multicast gateway* deployed in customer A's network. This *Multicast gateway* instance is deployed in the home gateway router device and has two host names: *sub1234-net987.isp.net* address facing the fixed-line service provider's access network, and *localrouter.net* facing the residential network. In order to support HTTPS, the service provider-facing host name is used exclusively by the *Content playback* function.

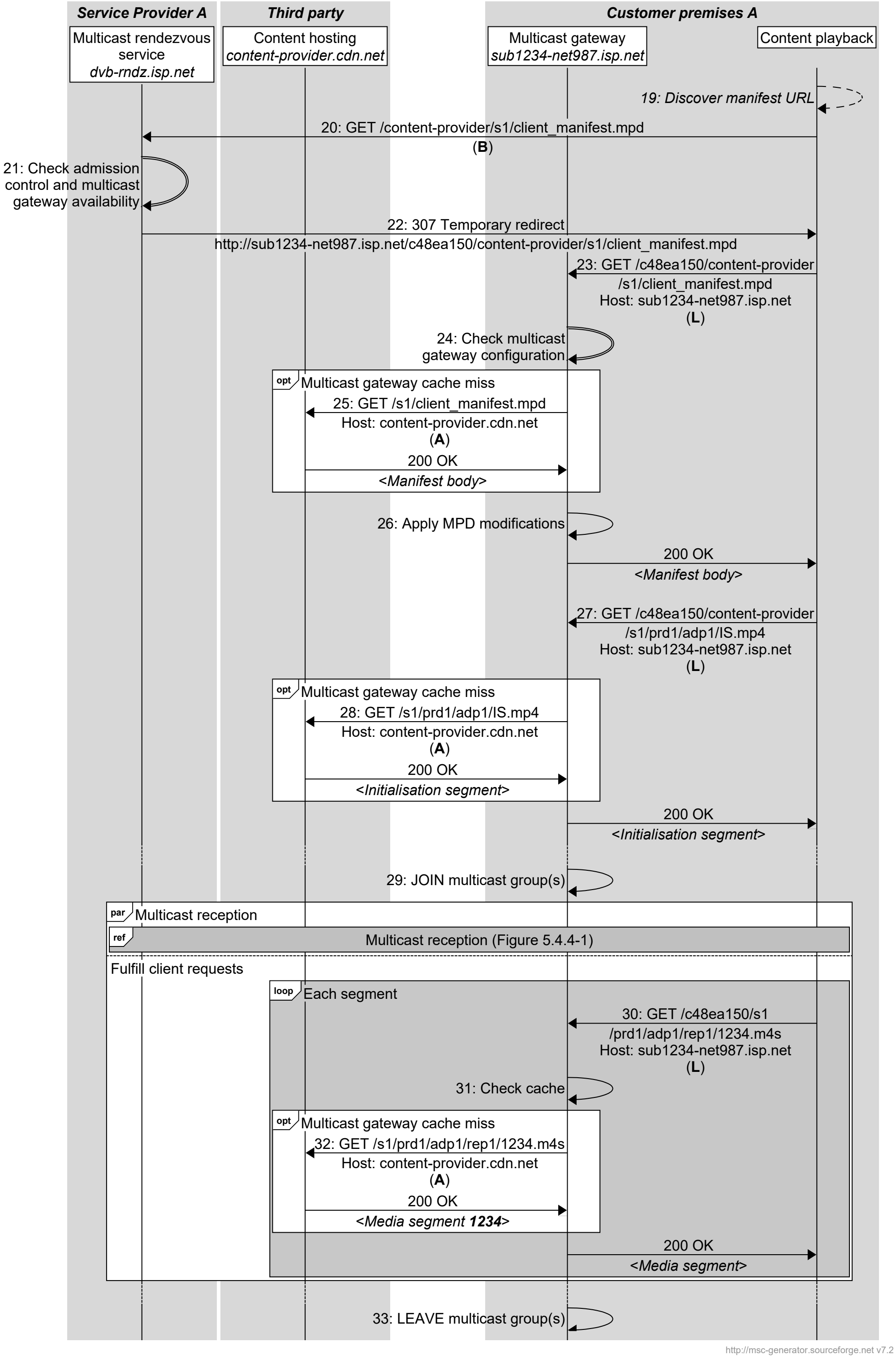

Service Provider A
Third party
Customer premises A
Multicast rendezvous service
dvb-rndz.isp.net
Content hosting
content-provider.cdn.net
Multicast gateway
sub1234-net987.isp.net
Content playback
19: Discover manifest URL
20: GET /content-provider/s1/client_manifest.mpd
(B)
21: Check admission control and multicast gateway availability
22: 307 Temporary redirect
http://sub1234-net987.isp.net/c48ea150/content-provider/s1/client_manifest.mpd
23: GET /c48ea150/content-provider /s1/client_manifest.mpd
Host: sub1234-net987.isp.net
(L)
24: Check multicast gateway configuration
opt
Multicast gateway cache miss
25: GET /s1/client_manifest.mpd
Host: content-provider.cdn.net
(A)
200 OK
<Manifest body>
26: Apply MPD modifications
200 OK
<Manifest body>
27: GET /c48ea150/content-provider /s1/prd1/adp1/IS.mp4
Host: sub1234-net987.isp.net
(L)
opt
Multicast gateway cache miss
28: GET /s1/prd1/adp1/IS.mp4
Host: content-provider.cdn.net
(A)
200 OK
<Initialisation segment>
200 OK
<Initialisation segment>
29: JOIN multicast group(s)
par
Multicast reception
ref
Multicast reception (Figure 5.4.4-1)
Fulfill client requests
loop
Each segment
30: GET /c48ea150/s1 /prd1/adp1/rep1/1234.m4s
Host: sub1234-net987.isp.net
(L)
31: Check cache
opt
Multicast gateway cache miss
32: GET /s1/prd1/adp1/rep1/1234.m4s
Host: content-provider.cdn.net
(A)
200 OK
<Media segment 1234>
200 OK
<Media segment>
33: LEAVE multicast group(s)
http://msc-generator.sourceforge.net v7.2

Figure 5.4.3.1-1: Multicast gateway playback sequence

The steps in the *Multicast gateway* playback sequence are as follows:

19. A playback session begins when a presentation manifest URL referencing the address of the content provider's *Multicast rendezvous service* is supplied to customer A's *Content playback* function.

20. The *Content playback* function makes a unicast HTTPS request for the manifest to the Service Provider's *Multicast rendezvous service* over reference point **B**, as specified in clause 7.5.1 of [i.1].

21. The Service Provider's *Multicast rendezvous service* function examines the source IP address of the request and determines that it belongs to a subscriber with an operational *Multicast gateway* that satisfies the requirements specified in clause 7.5.2.0 of [i.1].

22. The Service Provider's *Multicast rendezvous service* issues an HTTP redirect to the *Multicast gateway* function as specified in clause 7.5.2.1 of [i.1].

Table 5.4.3.1-1 below shows an example of the presentation manifest URL requested from the *Multicast rendezvous service* at reference point **B**, and the reference point **L** URL to which the requesting *Content playback* function is redirected, as well as the value of the `@contentPlaybackPathPattern` attribute in the multicast gateway configuration that is used to map the request URL to the value carried in the **`PresentationManifestLocator`** element.

Table 5.4.3.1-1: Manifest URLs over reference points B, L and A for customer A

Multicast session configuration	
Session **`PresentationManifestLocator`**	https://content-provider.cdn.net/s1/client_manifest.mpd
`PresentationManifestLocator` `@contentPlaybackPathPattern`	**/*/content-provider/s1/client_manifest.mpd**
Example outcome	
Presentation manifest URL at reference point **B**	https://dvb-rndz.isp.net/content-provider/s1/client_manifest.mpd
Redirected presentation manifest URL at reference point **L**	https://sub1234-net987.isp.net **/c48ea150/content-provider/s1/client_manifest.mpd**
Presentation manifest URL at reference point **A**	https://content-provider.cdn.net/s1/client_manifest.mpd

23. Customer A's *Content playback* function requests the presentation manifest from the *Multicast gateway* at reference point **L,** as nominated by the Service Provider's *Multicast rendezvous service* in step 22.

24. The *Multicast gateway* examines its multicast session configuration (acquired during step 7 in clause 5.4.1.2) and determines that there is an active multicast session corresponding to the reference point **L** presentation manifest URL.

Table 5.4.3.1-1 above shows the value of the **`PresentationManifestLocator`** element and associated `@contentPlaybackPathPattern` attribute manifest path that Service Provider A's *Multicast gateway* uses to match against the request URL at reference point **B** as specified in clause 8.4.1 of [i.1], alongside the various URLs then used at reference points **L** and **A**. In this example, the path element "*c48ea150*" is used as a session identifier, and as such is wildcarded by the `@contentPlaybackPathPattern` as specified in clause 10.2.2.2 of [i.1].

If the requested presentation manifest is available in the *Asset storage* cache from a previous playback session and is still valid according to the *Cache-Control* information received with it at reference point **A**, the following step (only) is skipped.

25. If the *Multicast gateway* does not have a valid cached copy of the requested presentation manifest in its cache, it requests it from the third party's *Content hosting* function via reference point **A** as specified in clause 8.4.1.0 of [i.1].

26. The *Multicast gateway* modifies the presentation manifest as permitted according to clause 8.4.1 and clause 10.2.2.0 of [i.1]. In this example, the *Multicast gateway* modifies the **`MPD/BaseURL`** element in the presentation manifest to replace the reference point **A** URL with the reference point **L** URL, which includes a session identifier for the *Multicast gateway* to track individual client sessions. The modification to the **`MPD/BaseURL`** is given in table 5.4.3.1-2 below.

NOTE: An implementation may also include the session identifier in requests made at reference point **A**, as either a path element or as a query parameter.

27. The *Content playback* function makes a request for the MPEG-DASH initialisation segment(s) for its chosen representation(s) via reference point **L**.

If the requested MPEG-DASH initialisation segment(s) are available in the *Asset storage* cache from a previous playback session and are still valid according to the *Cache-Control* information received with them at reference point **A**, the following step (only) is skipped.

28. If the *Multicast gateway* does not have a cached copy of the requested MPEG-DASH initialisation segment(s), it may request it/them from the *Content hosting* function at reference point **A**.

Table 5.4.3.1-2 below shows how the *Multicast gateway* replaces the **`MPD/BaseURL`** in the presentation manifest supplied to the *Content playback* function at reference point **L** as described in step 26, and the effect this has on the initialisation segment request URL that the *Content playback* function uses at reference point **L** in step 27. The table also shows how the *Multicast gateway* transforms the initialisation segment request URL at reference point **L** to the equivalent URL at reference point **A** in the case of a cache miss as described in step 28. In this example, because the initialisation segment is only available to the *Multicast gateway* at reference point **A**, the reference point **L** URL is mapped directly to the reference point **A** URL from the information present in the original presentation manifest.

Table 5.4.3.1-2: Mapping of MPEG-DASH initialisation segment URLs

Multicast session configuration	
Session **`PresentationManifestLocator`**	**https://content-provider.cdn.net/s1/**client_manifest.mpd
`PresentationManifestLocator` `@contentPlaybackPathPattern`	**/*/content-provider/s1/**client_manifest.mpd
Presentation manifest URL at reference point **L**	**https://sub1234-net987.isp.net/c48ea150/content-provider/s1/**client_manifest.mpd
MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://content-provider.cdn.net/s1/**
`MPD/BaseURL` at reference point **L**	**http://sub1234-net987.isp.net/c48ea150/content-provider/s1/**
Example outcome	
Initialisation segment URL at reference point **L**	**http://sub1234-net987.isp.net/c48ea150/content-provider/s1/**prd1/adp1/IS.mp4
Initialisation segment URL at reference point **A**	**https://content-provider.cdn.net/s1/**prd1/adp1/IS.mp4

29. The *Multicast gateway* joins the multicast transport sessions indicated in its multicast session configuration as specified in clause 8.4.2.

30. The *Content playback* function makes requests for MPEG-DASH media segments at reference point **L**.

31. The *Multicast gateway* checks whether it has a cached copy of the requested media object in its *Asset storage*, either received over reference point **M** or previously requested via reference point **A** by another *Content playback* function.

32. If the requested media object is not cached in the *Asset storage* subfunction, then the *Multicast gateway* makes a unicast request for the media object from the *Content hosting* (edge) via reference point **A**.

Table 5.4.3.1-3 below shows how the *Multicast gateway* transforms the media segment request URL at reference point **L** to the equivalent URL at reference point **A**, given the modifications to the presentation manifest performed in step 26. In this example, the unicast repair URL at reference point **A** directly matches the MPEG-DASH media segment URL accessed at reference point **A** from the information present in the original presentation manifest.

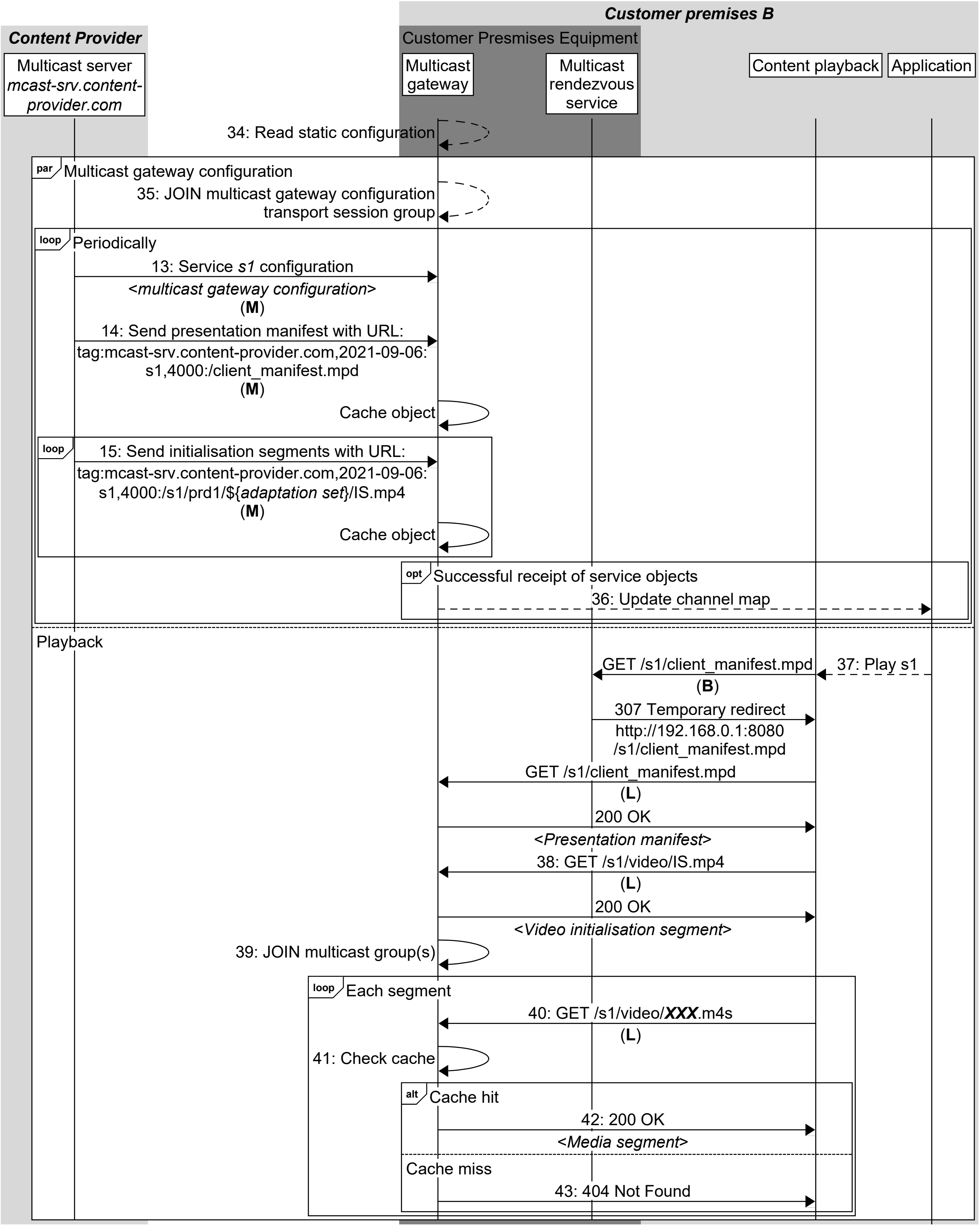
Table 5.4.3.1-3: Mapping of MPEG-DASH media segment URLs

Multicast session configuration	
Session **`PresentationManifestLocator`**	**https://content-provider.cdn.net/s1/**client_manifest.mpd
`PresentationManifestLocator` `@contentPlaybackPathPattern`	**/*/content-provider/s1/**client_manifest.mpd
Presentation manifest URL at reference point **L**	**https://sub1234-net987.isp.net /c48ea150/content-provider/s1/**client_manifest.mpd
MPD configurations	
`MPD/BaseURL` at reference point **A**	**https://content-provider.cdn.net/s1/**
`MPD/BaseURL` at reference point **L**	**https://sub1234-net987.isp.net /c48ea150/content-provider/s1/**
Example outcome	
Media segment URL at reference point **L**	**https://sub1234-net987.isp.net /c48ea150/content-provider/s1/**prd1/adp1/rep1/1234.m4s
Media segment URL at reference point **A**	**https://content-provider.cdn.net/s1/** prd1/adp1/rep1/1234.m4s

33. The *Multicast gateway* leaves the multicast transport sessions when the *Content playback* function stops requesting segments and the session is deemed finished.

5.4.3.2 Satellite customer playback sequence

The following example is from the perspective of the *Multicast gateway* deployed in Service Provider B's network. This *Multicast gateway* instance is deployed in the terminal device, alongside the *Content playback* function.


Customer premises B
Content Provider
Customer Presmises Equipment
Multicast server
mcast-srv.content-provider.com
Multicast gateway
Multicast rendezvous service
Content playback
Application
34: Read static configuration
par Multicast gateway configuration
35: JOIN multicast gateway configuration transport session group
loop Periodically
13: Service s1 configuration
<multicast gateway configuration>
(M)
14: Send presentation manifest with URL:
tag:mcast-srv.content-provider.com,2021-09-06:s1,4000:/client_manifest.mpd
(M)
Cache object
loop
15: Send initialisation segments with URL:
tag:mcast-srv.content-provider.com,2021-09-06:s1,4000:/s1/prd1/${adaptation set}/IS.mp4
(M)
Cache object
opt Successful receipt of service objects
36: Update channel map
Playback
GET /s1/client_manifest.mpd
(B)
37: Play s1
307 Temporary redirect
http://192.168.0.1:8080/s1/client_manifest.mpd
GET /s1/client_manifest.mpd
(L)
200 OK
<Presentation manifest>
38: GET /s1/video/IS.mp4
(L)
200 OK
<Video initialisation segment>
39: JOIN multicast group(s)
loop Each segment
40: GET /s1/video/XXX.m4s
(L)
41: Check cache
alt Cache hit
42: 200 OK
<Media segment>
Cache miss
43: 404 Not Found

https://gitlab.com/msc-generator v8.0

NOTE: The numbering of steps continues from the end of figure 5.4.3.1-1 to allow for unique referencing through clause 5.4. Ordering between the sequences is not implied: this sequence occurs in parallel with the one described in clause 5.4.3.1.

Figure 5.4.3.2-1: End-to-end worked playback example

34. Once started, the combined *Multicast rendezvous*/*Multicast gateway* instance checks its static configuration for details of available configuration services.

35. The *Multicast gateway* joins the multicast gateway configuration transport session group as indicated in its static configuration.

13. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits the multicast gateway configurations for the services available over multicast as described by clause 8.3.5 of [i.1].

14. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits the presentation manifest for consumption by any *Multicast gateway* instances that are configured to receive their presentation manifest over reference point **M**, such as those in clause 5.4.4.

    The *Multicast gateway* caches this media object. If the object is not received complete, the *Multicast gateway* waits for the next time the object is sent in order to correct the received multicast transport object before caching.

15. At a period specified in the multicast server configuration (see clause 10.2.3.14 of [i.1]), the *Multicast server* serialises and repeatedly transmits MPEG-DASH initialisation segments for all Adaptation Sets described in the presentation manifest sent in step 14 for consumption by any *Multicast gateway* instances that are configured to receive their initialisation segments over reference point **M**, such as those in clause 5.4.4.

    The *Multicast gateway* caches these media objects to allow for future fast channel change. If these objects are not received complete, the *Multicast gateway* waits for the next time each object is sent in order to correct the received multicast transport object before caching.

36. If the multicast gateway configuration, presentation manifest(s) and initialisation segments have been successfully received, the *Multicast gateway* may (using means beyond the scope of [i.1]) make an updated channel map available to the *Application* function describing the set of services advertised in the multicast gateway configuration received in step 13.

37. The *Application* function instructs the *Content playback* function to begin playback of a specific service, which prompts the *Content playback* function to request the presentation manifest from the *Multicast rendezvous service* that is co-located with the *Multicast gateway* using a URL delivered as part of the channel map in step 36. The *Multicast rendezvous service* function checks that the request is valid and returns a redirect to the *Multicast gateway* which serves the presentation manifest previously stored in its cache in step 14.

38. The *Content playback* function makes a request via reference point **L** for the MPEG-DASH initialisation segment(s) for a Representation that is available via reference point **M**. The *Multicast gateway* serves the initialisation segment previously stored in its cache in step 15.

39. The *Multicast gateway* joins the multicast transport session(s) indicated in its multicast session configuration as specified in clause 8.4.2.

In this example, the *Multicast gateway* requires the use of AL-FEC to provide repair to the transported multicast transport objects. Each multicast transport session in the multicast gateway configuration therefore contains a **`ForwardErrorCorrectionParameters`** element, and the *Multicast gateway* subscribes to all relevant multicast groups in order to receive objects for the chosen representations.

40. The *Content playback* function makes requests for MPEG-DASH media segments at reference point **L**.

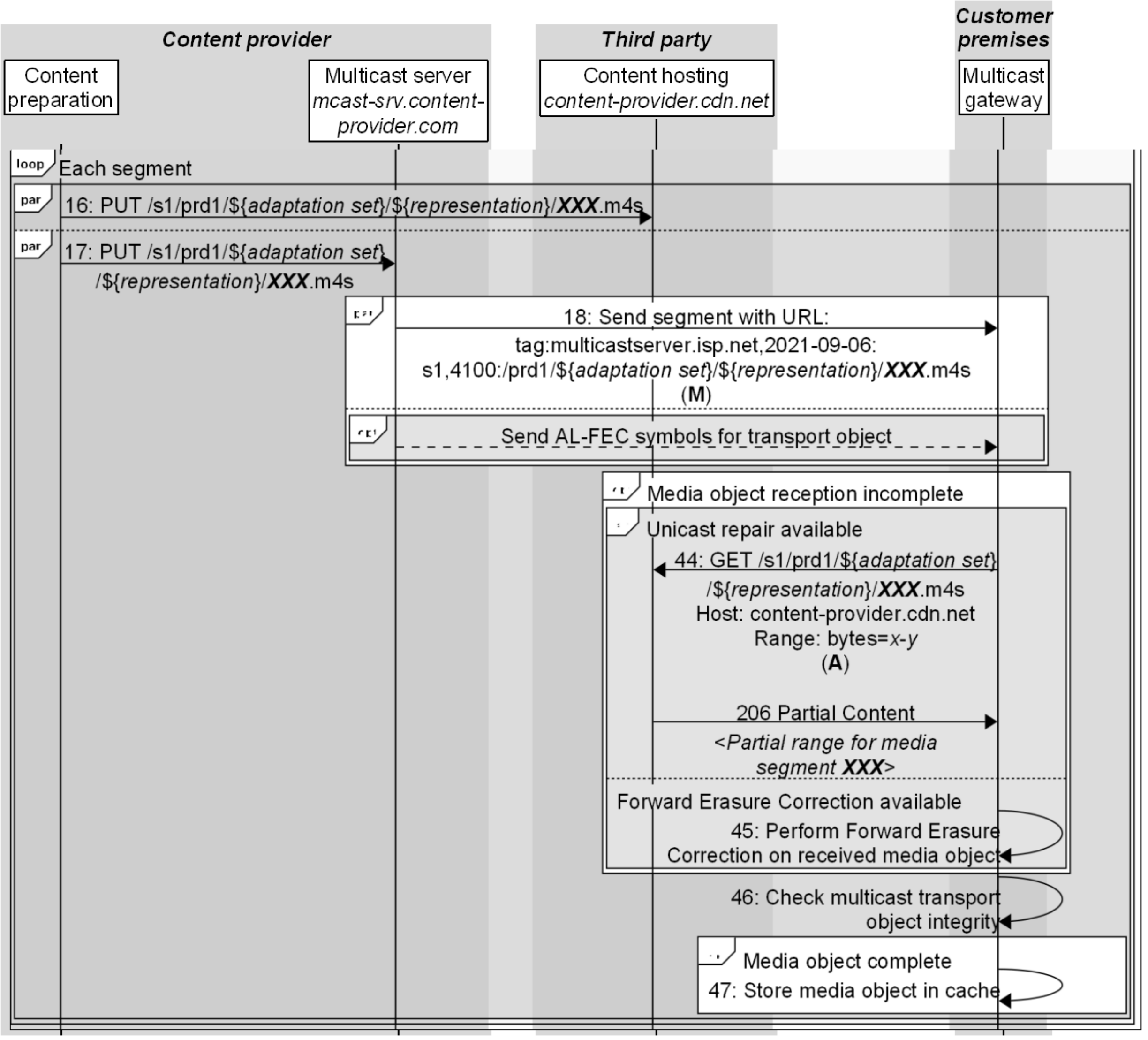
41. The *Multicast gateway* checks whether it has a cached copy of the requested media object in its *Asset storage*. If this is the first media request, then the *Multicast gateway* delays processing of the request until such a time that a full media segment has been received on the multicast transport session.

42. If the media object is complete in the *Asset storage*, then the media delivery object is returned to the *Content playback* function with a 200 OK response code.

43. If the media object is not complete in the *Asset storage*, then the *Multicast gateway* replies with a 404 Not Found response code.

Due to the unidirectional nature of the satellite-based deployment described in this clause, unicast repair over reference point **A** is not possible. Therefore, a multicast transport object may be only partially received in such a way that it is not possible to repair it using Application Level Forward Erasure Correction.

5.4.4 Multicast reception sequence


Content provider
Third party
Customer premises
Content preparation
Multicast server mcast-srv.content-provider.com
Content hosting content-provider.cdn.net
Multicast gateway
loop
Each segment
par
16: PUT /s1/prd1/${adaptation set}/${representation}/XXX.m4s
par
17: PUT /s1/prd1/${adaptation set}/${representation}/XXX.m4s
18: Send segment with URL: tag:multicastserver.isp.net,2021-09-06: s1,4100:/prd1/${adaptation set}/${representation}/XXX.m4s (M)
Send AL-FEC symbols for transport object
Media object reception incomplete
Unicast repair available
44: GET /s1/prd1/${adaptation set}/${representation}/XXX.m4s Host: content-provider.cdn.net Range: bytes=x-y (A)
206 Partial Content
<Partial range for media segment XXX>
Forward Erasure Correction available
45: Perform Forward Erasure Correction on received media object
46: Check multicast transport object integrity
Media object complete
47: Store media object in cache

http://msc-generator.sourceforge.net v7.2

Figure 5.4.4-1: Multicast reception example

The steps in the *Multicast gateway* playback sequence are as follows:

16. The *Content preparation* function creates and then pushes each media segment to the *Content hosting* function using reference point **P**$_{\textbf{in}}$.

17. In parallel with step 16, the *Content preparation* function pushes each media segment to the *Multicast server* using the push-based content ingest mode (see clause 8.3.2 of [i.1]) via reference point **P**$_{\textbf{in}}$**′**.

18. The *Multicast server* serialises ingested media segments as multicast transport objects and transmits them over reference point **M**, as specified in clause 8.3.4 of [i.1], along with the corresponding AL-FEC symbols carried on another multicast address (that are only used by the satellite customer's *Multicast gateway*).

44. Should the fixed-line customer's *Multicast gateway* not receive a complete intact multicast transport object over reference point **M**, it uses the unicast repair information present in the multicast session configuration to perform a unicast repair operation per clause 9 of [i.1], and then it continues directly to step 46.

Table 5.4.4-1 below shows how the *Multicast gateway* uses the **`UnicastRepairParameters`** in the multicast gateway configuration to transform the multicast transport object URI at reference point **M** to the unicast repair URL at reference point **A**.

Table 5.4.4-1: Mapping of multicast transport object URIs to unicast repair URLs

Multicast session configuration	
`UnicastRepairParameters``@transportObjectBaseURI`	**tag:mcast-srv.content-provider.com,2021-09-06:s1,4100:**
`UnicastRepairParameters/BaseURL`	**https://content-provider.cdn.net/s1**
Example outcome	
Multicast transport object URI at reference point **M**	**tag:mcast-srv.content-provider.com,2021-09-06:s1,4100:** /prd1/adp1/rep1/1234.m4s
Unicast repair URL at reference point **A**	**https://content-provider.cdn.net/s1** /prd1/adp1/rep1/1234.m4s

45. Should the satellite customer's *Multicast gateway* not receive a complete intact multicast transport object over reference point **M**, it uses the AL-FEC symbols received from the multicast group signalled in the appropriate **`ForwardErrorCorrectionParameters`** element in the multicast gateway configuration to attempt to repair the multicast transport object as specified in clause 8.3.4.2 of [i.1].

46. Once the *Multicast gateway* has fully received a multicast transport object on the multicast transport session or has completed any repair operations per steps 44 or 45, it performs any applicable integrity checks, such as verifying an object digest hash.

47. The media segment carried as a multicast transport object is cached by the *Asset storage* subfunction of the *Multicast gateway* as specified in clause 5.3.5.0 and 5.3.5.4.

Annex A
Baseline procedures and call flows

A.0 Overview

This annex documents the baseline procedures and call flows for a system compliant with [i.1]. The scenario additionally documents how unicast retrieval of DVB-I service discovery and programme metadata (as specified in TS 103 770 [i.11]) can be incorporated into such a system. Variations of the procedures and call flows for the DVB-NIP [i.12] instantiation of the System are noted.

A.1 Architecture

In this instantiation of the logical reference architecture specified in [i.1]:

- The *Content preparation* function realises both of the following DVB-I functions:
    - DVB-DASH MPD Server.
    - DVB-DASH Segment Server.
- The logical *Content hosting* function may be provided by a DVB-MABR Service Provider, by a Content Provider or by a third-party service provider in the form of a Content Delivery Network (CDN).
- On the client side, the *Application* function realises the DVB-I Client function to acquire and process DVB-I service metadata.

In general, a DVB-MABR Service Provider may deploy the required logical functions at its discretion. In the specific case where DVB-MABR is deployed as part of a DVB-NIP System per [i.12], certain functions are co-located as shown in figure A.1-1 and as described below:

- The *Multicast server* function and the *Provisioning* function are co-located at the DVB-NIP Head-end.
- The *Multicast gateway* function and the supporting *Multicast rendezvous service* function are co-located in the DVB-NIP Gateway.
- The DVB-NIP Gateway also hosts a proxy DVB-I Service List Server and (optionally) a proxy DVB-I Content Guide Server.

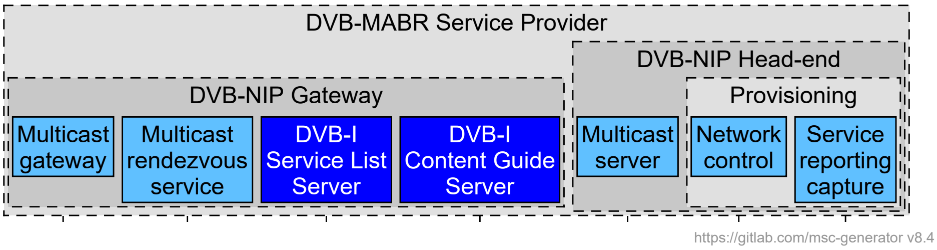

DVB-MABR Service Provider
DVB-NIP Head-end
DVB-NIP Gateway
Provisioning
Multicast gateway
Multicast rendezvous service
DVB-I Service List Server
DVB-I Content Guide Server
Multicast server
Network control
Service reporting capture
https://gitlab.com/msc-generator v8.4

Figure A.1-1: Co-location of logical functions in a DVB-NIP deployment

The DVB-NIP System is not considered further in this annex.

A.2 Procedures and call flows

A.2.1 System provisioning procedure

The procedures used to provision the System are depicted in figures A.2.1-1.

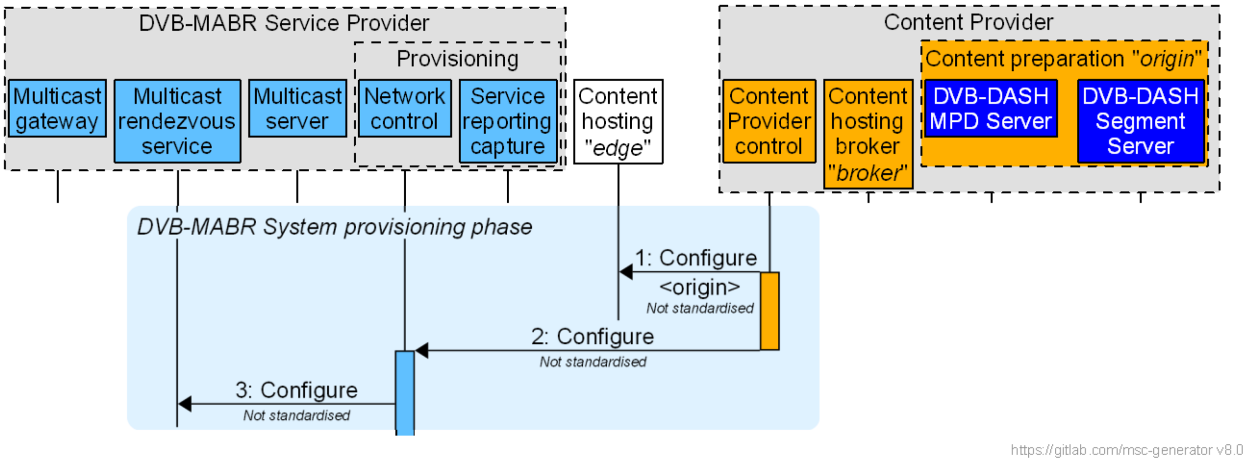


Figure A.2.1-1: Call flow sequence for provisioning DVB-I services

The steps for provisioning DVB-I services are as follows:

1: The Content Provider's *Content Provider control* subfunction configures the *Content hosting* function to act as a content origin for DVB-I services by informing it about the host address of the DVB-DASH MPD Server in the *Content preparation* function. The *Content hosting* function may be, for example, the edge of a Content Delivery Network (CDN).

NOTE 1: This interaction is not standardised by DVB.

2: The Content Provider's *Content Provider control* subfunction provides configuration information to the DVB-MABR Service Provider's *Network control* function about which services are to be made available via DVB-MBAR and the URLs of their presentation manifests on the *Content hosting* function.

NOTE 2: This interaction is not standardised by DVB.

3: As a consequence, the *Network control* subfunction configures the *Multicast rendezvous service* to inform it which services are currently configured in the System, including the URLs of their presentation manifests on the *Content hosting* function.

NOTE 3: This procedure is not standardised in [i.1]. However, [i.12] specifies that the *Multicast rendezvous service* deployed in the DVB-NIP Gateway function is configured indirectly by the *Network control* subfunction via the *Multicast server* by means of the (unidirectional) multicast gateway configuration transport session at reference point **M**, as described in step 8 below.

A.2.2 System configuration procedure

The **multicast session configuration** (see clause 10 of [i.1]) describes the set of multicast sessions currently configured in the System. It is realised as a **multicast server configuration** instance document used to configure the *Multicast server* and a **multicast gateway configuration** instance document used to configure *Multicast gateway* instances, both of which follow the same XML-based syntax specified in clause 10.2 of [i.1].

NOTE 1: A multicast session declared in the multicast session configuration is not necessarily active at all times, so the *Multicast server* may not always be transmitting packets at reference point **M** for its declared service components.

The procedures that a DVB-MABR Service Provider uses to configure DVB-I services in the *Multicast server* and in the population of *Multicast gateway* instances are depicted in figure A.2.2-1.

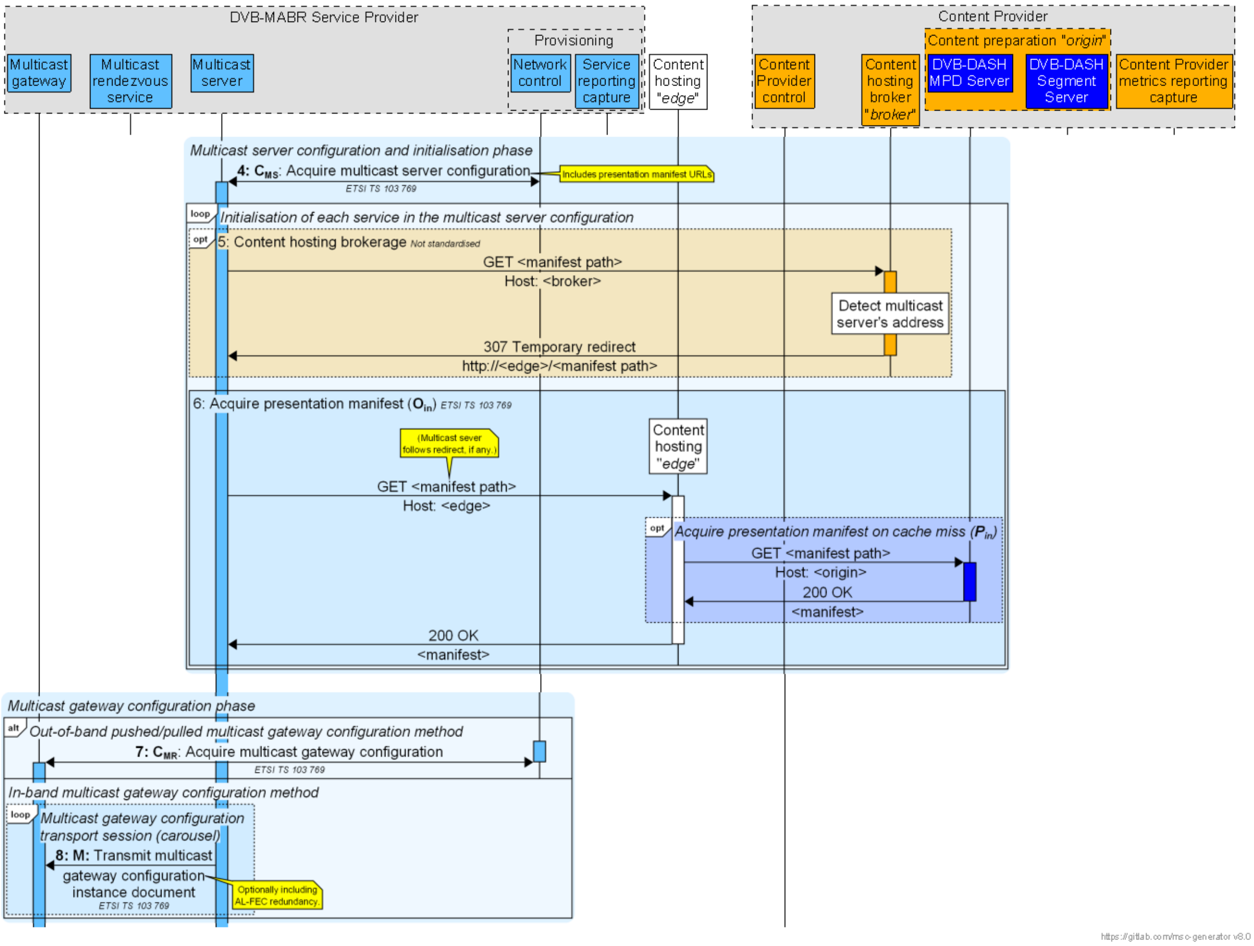

DVB-MABR Service Provider
Provisioning
Multicast gateway
Multicast rendezvous service
Multicast server
Network control
Service reporting capture
Content hosting "edge"
Content Provider
Content preparation "origin"
Content Provider control
Content hosting broker "broker"
DVB-DASH MPD Server
DVB-DASH Segment Server
Content Provider metrics reporting capture
Multicast server configuration and initialisation phase
4: C_MS: Acquire multicast server configuration
ETSI TS 103 769
Includes presentation manifest URLs
loop
Initialisation of each service in the multicast server configuration
opt
5: Content hosting brokerage Not standardised
GET <manifest path>
Host: <broker>
Detect multicast server's address
307 Temporary redirect
http://<edge>/<manifest path>
6: Acquire presentation manifest (O_in) ETSI TS 103 769
(Multicast sever follows redirect, if any.)
Content hosting "edge"
GET <manifest path>
Host: <edge>
opt
Acquire presentation manifest on cache miss (P_in)
GET <manifest path>
Host: <origin>
200 OK
<manifest>
200 OK
<manifest>
Multicast gateway configuration phase
alt
Out-of-band pushed/pulled multicast gateway configuration method
7: C_MR: Acquire multicast gateway configuration
ETSI TS 103 769
In-band multicast gateway configuration method
loop
Multicast gateway configuration transport session (carousel)
8: M: Transmit multicast gateway configuration instance document
ETSI TS 103 769
Optionally including AL-FEC redundancy.
https://gitlab.com/msc-generator v8.0

Figure A.2.2-1: Call flow sequence for provisioning DVB-I services in the Multicast server and Multicast gateway

The steps involved are as follows:

4: The *Multicast server* acquires its configuration via reference point **C$_{MS}$** using either the push- or pull-based acquisition protocol according to clause 10.4.2 of [i.1]. The multicast server configuration includes URLs of the presentation manifests for each configured multicast session. These URLs may point directly to the *Content hosting* function or to an intermediate *Content hosting broker* function.

The *Multicast server* then initialises itself by acquiring the presentation manifest for each multicast session described in the current multicast server configuration:

5: If the presentation manifest URL points at an intermediate *Content hosting broker* function, the *Multicast server* receives an HTTP redirect in response to its request for the presentation manifest and follows this to the appropriate *Content hosting* function.

NOTE 2: This interaction is not standardised by DVB.

6: The *Multicast server* fetches the presentation manifest from the *Content hosting* function via reference point **O$_{in}$**. If the *Content hosting* function does not already have a cached copy of the presentation manifest, it fetches one from the DVB-DASH MPD Server subfunction of the *Content preparation* function.

Multicast transport sessions for the service components of each configured multicast session are now provisioned in the *Multicast server*.

Each *Multicast gateway* instance in the deployment also needs to be configured by one of the following methods:

7: The *Multicast gateway* instance acquires its configuration via reference point **C$_{\textbf{MR}}$** using either the push- or pull-based acquisition protocol according to clause 10.4.4 of [i.1].

Or:

8: The *Multicast server* is configured to carousel the multicast gateway configuration instance document in a multicast gateway configuration transport session at reference point **M**, as specified in clause 8.3.5 of [i.1]. This instance document is generated by the *Multicast server* as a subset of the current multicast server configuration acquired in step 4 above.

The *Multicast gateway* instance (and, in the case of DVB-NIP deployments, also the *Multicast rendezvous server* instance co-located with it) acquires its configuration via reference point **M** using the in-band multicast gateway configuration method specified in clause 10.4.5 of [i.1].

A.2.3 Service discovery procedure

The procedures that a DVB-I Client uses to discover the set of available DVB-I services are depicted in figure A.2.3-1 below.

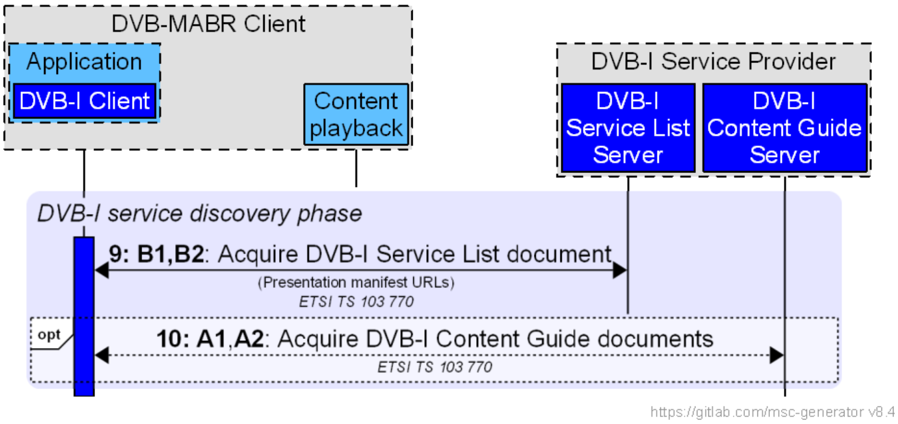

DVB-MABR Client
Application
DVB-I Client
Content playback
DVB-I Service Provider
DVB-I Service List Server
DVB-I Content Guide Server
DVB-I service discovery phase
9: B1,B2: Acquire DVB-I Service List document
(Presentation manifest URLs)
ETSI TS 103 770
opt
10: A1,A2: Acquire DVB-I Content Guide documents
ETSI TS 103 770
https://gitlab.com/msc-generator v8.4

Figure A.2.3-1: Call flow sequence for discovering DVB-I services

The service discovery steps are as follows:

9: The DVB-I Client acquires the DVB-I Service List document from the DVB-I Service List Server using interactions B1 and B2 as specified in clauses 6.2 and 5.5.1 of ETSI TS 103 770 [i.11]. All service instances appearing in the DVB-I Service List with a **`DASHDeliveryParameters`** element are available via unicast and (unless a DVB-I Playlist document is referenced) the URL of the presentation manifest document for the corresponding multicast session in the current multicast session configuration (in this case, a DVB-DASH MPD) is carried in the **`DASHDeliveryParameters/UriBasedLocation`** element, as specified in clause 5.2.7.2 of ETSI TS 103 770 [i.11].

NOTE: Each service instance in the DVB-I Service List corresponds to one session in the multicast session configuration.

10: The DVB-I Client may additionally acquire DVB-I Content Guide documents from the DVB-I Content Guide Server using interactions A1 and A2 as specified in clause 6 of ETSI TS 103 770 [i.11].

A.2.4 Service operation procedures

A.2.4.0 General

The following activities proceed in parallel:

i) Transmission of multicast transport objects by the *Multicast server*. See clause A.2.4.1.

ii) Periodic registration of a *Multicast gateway* instance with the *Network control* function (bidirectional deployments only). See clause A.2.4.2.

iii) Service selection by the DVB-I Client and reception of multicast transport objects by a *Multicast gateway* instance. See clause A.2.4.3.

iv) Reporting of metrics by the *Multicast gateway* instance (bidirectional deployments only). See clause A.2.4.4.

A.2.4.1 Multicast transmission procedure

For the service components declared in each currently configured multicast session, the *Multicast server* acquires and transmits multicast media objects in a continuous loop as shown in figure A.2.4.1-1:

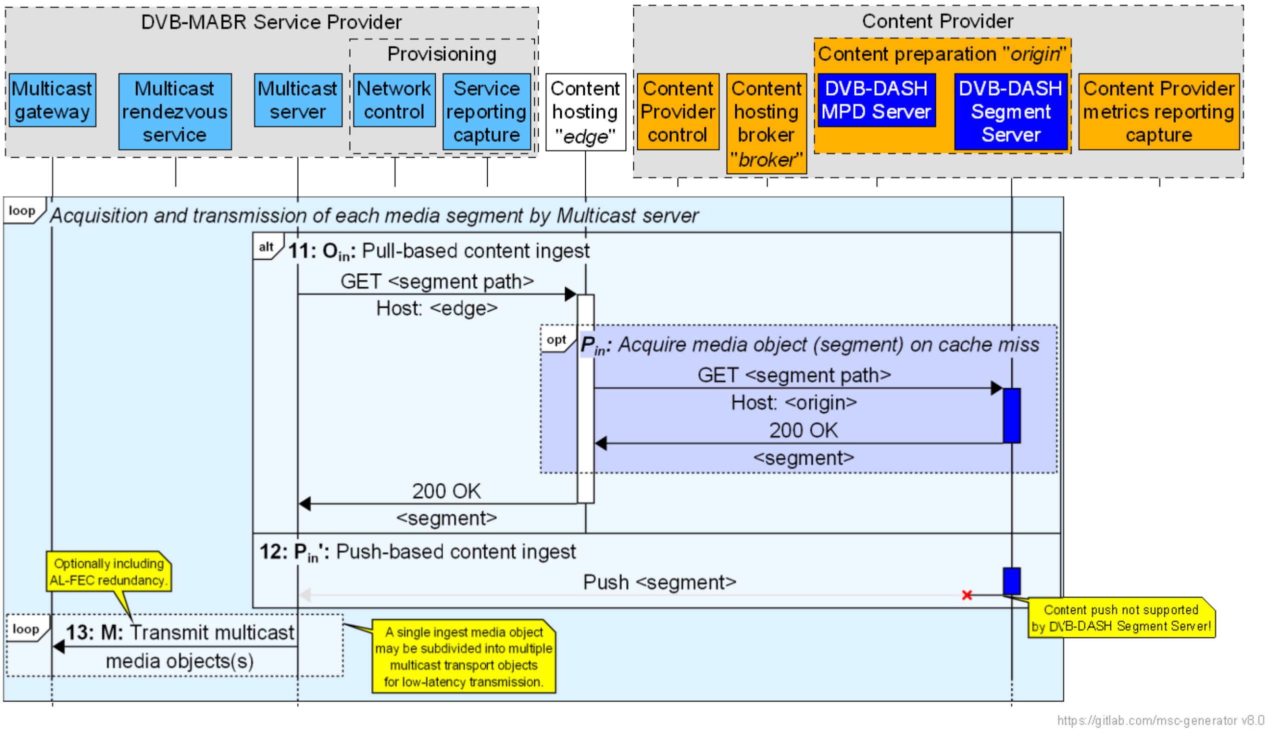


Figure A.2.4.1-1: Multicast transmission procedure

The steps are as follows:

11: The *Multicast server* retrieves an ingest object (media segment) from the *Content hosting* function via reference point **O$_{in}$**.

The *Content hosting* function may need to retrieve the media segment from the DVB-DASH Segment Server subfunction of the *Content preparation* function if it does not already have a copy cached.

12: Although the *Multicast server* supports a mode in which ingest objects are pushed to it via reference point **P$_{in}$'**, the DVB-DASH Segment Server is not required to implement push-based publication of media segments, and so this mode of operation is omitted from this scenario.

13: The ingest object is divided into one or more multicast transport objects by the *Multicast server*, as specified in clause 8.3.4 of [i.1].

NOTE 1: In chunked transmission mode, the multicast transport protocol may specify division of a single ingest object into multiple multicast transport objects, for example in order to facilitate low-latency ingest and multicast transmission of individual CMAF chunks comprising a media segment.

Each multicast transport object is then serialised according to the multicast transport protocol configured for the multicast transport session, and the resulting packets are transmitted to the corresponding multicast group address at reference point **M**.

NOTE 2: Additional Application Layer FEC (AL-FEC) redundancy may be added by the *Multicast server* to the same multicast transport session or to separate multicast transport session(s) dedicated to AL-FEC packets, per the current multicast server configuration.

A.2.4.2 Multicast gateway registration procedure

In bidirectional deployments only, each *Multicast gateway* instance periodically registers itself with the *Network control* function, as shown in figure A.2.4.2-1, to indicate that it is willing to serve streaming sessions to the *Content playback* function.

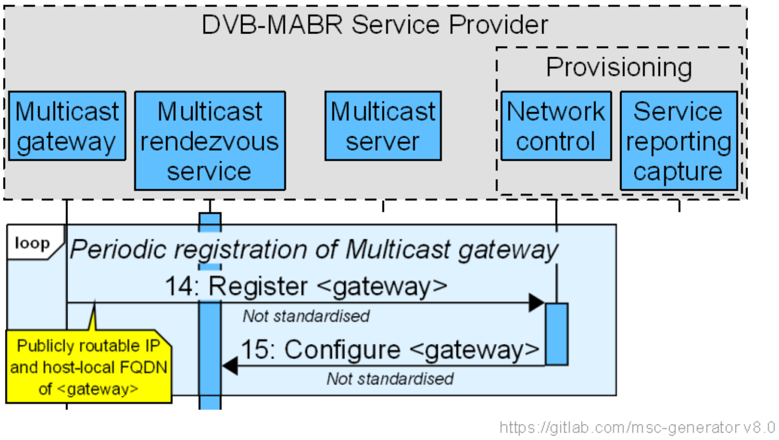

DVB-MABR Service Provider
Provisioning
Multicast gateway
Multicast rendezvous service
Multicast server
Network control
Service reporting capture
loop
Periodic registration of Multicast gateway
14: Register <gateway>
Not standardised
15: Configure <gateway>
Not standardised
Publicly routable IP and host-local FQDN of <gateway>
https://gitlab.com/msc-generator v8.0

Figure A.2.4.2-1: Multicast gateway registration procedure

The steps are as follows:

14: The *Multicast gateway* instance registers itself with the *Network control* function, passing its publicly routable IP address and a host-local Fully-Qualified Domain Name that it presents to the *Content playback* function.

15: The *Network control* function responds by configuring the *Multicast rendezvous service* with the publicly routable IP address of the *Multicast gateway* and its host-local Fully-Qualified Domain Name.

NOTE: This interaction is not standardised by DVB.

A.2.4.3 Service selection procedure

The procedures that a DVB-I Client in the *Application* uses to select and play back a DVB-I service for presentation are depicted in figure A.2.4.3-1 below.

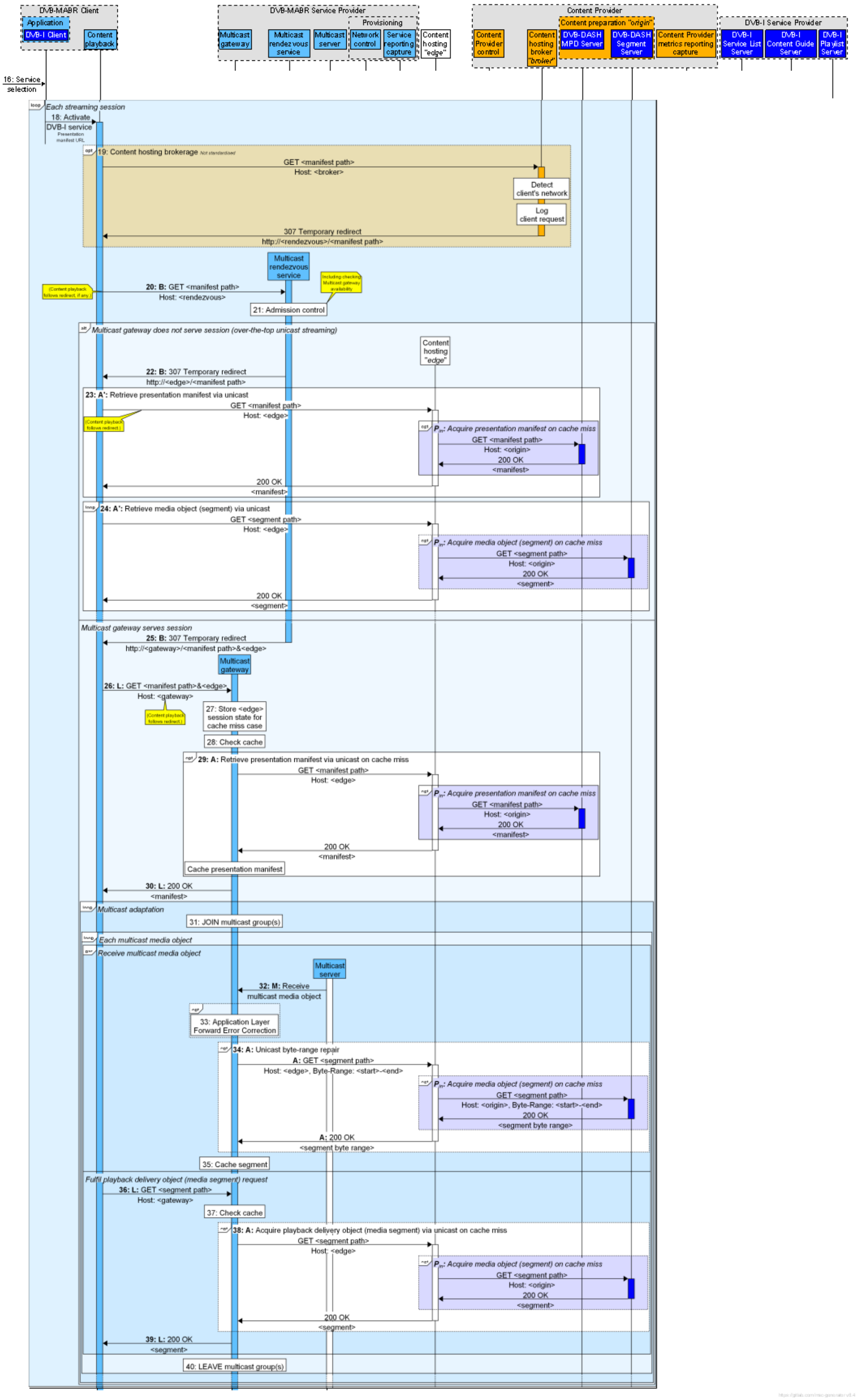


Figure A.2.4.3-1: Call flow sequence for DVB-I service selection

When a DVB-I service is selected for presentation (e.g. as a result of a user interaction with the *Application*), the steps are as follows:

16: The end user selects a DVB-I service in the user interface of the *Application*.

17: Void.

18: The DVB-I Client invokes the *Content playback* function, passing the URL of a presentation manifest document (i.e. a DVB-DASH MPD) as a parameter. This is either the URL of a presentation manifest selected from the DVB-I Playlist document in the previous optional step or else the contents of the **`DASHDeliveryParameters/UriBasedLocation`** element of the selected DVB-I service instance.

NOTE 1: This interaction is internal to the operation of the DVB-MABR Client and is not standardised by DVB.

19: If the chosen presentation manifest URL points at an intermediate *Content hosting broker* function, the apparent IP address of the requesting *Content playback* function (after any Network Address Translation) is inspected by the *Content hosting broker* and it issues an HTTP redirect in response pointing to the appropriate *Multicast rendezvous service*.

NOTE 2: This interaction is not standardised by DVB.

20: The *Content playback* function requests the presentation manifest from the *Multicast rendezvous service* (see clause 7.5 of [i.1]) via reference point **B**.

21: The *Multicast rendezvous service* performs session admission control. It determines whether the requested presentation manifest is present in the current multicast session configuration and whether there is an active *Multicast gateway* registered in the network of the *Content playback* function (per clause A.2.4.2 above). If both conditions are met, the call flow proceeds as a multicast-capable streaming session from step 25; otherwise, the call flow proceeds as a conventional unicast session (steps 22 to 24 inclusive).

In the case of a conventional unicast streaming session:

22: The *Multicast rendezvous service* issues an HTTP redirect pointing to the *Content hosting* function.

23: The *Content playback* function retrieves the presentation manifest from the *Content hosting* function via reference point **A′**.

If the *Content hosting* function does not already have a cached copy of the presentation manifest, it fetches one from the DVB-DASH MPD Server subfunction of the *Content preparation* function via reference point $\mathbf{P_{in}}$.

24: The *Content playback* function retrieves a media segment from the *Content hosting* function via reference point **A′**.

If the *Content hosting* function does not already have a cached copy of the requested media segment, it fetches one from the DVB-DASH Segment Server subfunction of the *Content preparation* function via reference point $\mathbf{P_{in}}$.

Playback of the selected DVB-I service continues in step 24 until further user interaction.

In the case of a multicast-capable streaming session:

25: The *Multicast rendezvous service* issues an HTTP redirect pointing to its chosen *Multicast gateway*. The URL in the redirect may additionally include (as an additional URL query parameter) the address of the *Content hosting* function to be used for HTTP-based unicast repair.

NOTE 3: Additional URL query parameters in the HTTP redirect URL are not standardised in [i.1].

26: Following the HTTP redirect, the *Content playback* function now requests the presentation manifest from the *Multicast gateway* via reference point **L**.

27: The *Multicast gateway* stores state about the active streaming session for later use, including the address of the requesting *Content playback* function and the address of the *Content hosting* function passed in step 25.

28: The *Multicast gateway* checks whether it already has a copy of the requested presentation manifest cached in its *Asset storage* subfunction.

29: If there is no copy of the presentation manifest in its *Asset storage* subfunction, the *Multicast gateway* retrieves it from the *Content hosting* function via reference point **A**.

If the *Content hosting* function does not already have a cached copy of the presentation manifest, it fetches one from the DVB-DASH MPD Server subfunction of the *Content preparation* function via reference point **P$_{\text{in}}$.**

The presentation manifest is cached in the *Asset storage* subfunction of the *Multicast gateway*.

30: The *Multicast gateway* returns the presentation manifest to the requesting *Content playback* function.

Steps 31 to 40 now proceed in a loop of multicast adaptation:

31: The *Multicast gateway* joins the multicast group corresponding to one of the multicast transport sessions declared in the multicast session configuration for the currently active multicast session.

A multicast media object is received by the *Multicast gateway* from the subscribed multicast transport session:

32: The *Multicast gateway* receives multicast packets on the multicast transport session and attempts to reassemble them into the transmitted multicast media object.

33: If configured for the multicast transport session in question, *Multicast gateway* may attempt to repair damage to the multicast media object caused by lost multicast packets using Application Layer FEC.

34: To repair any remaining damage to the multicast media object, the *Multicast gateway* employs unicast HTTP byte-range repair, as specified in clauses 9.1 and 9.2 of [i.1].

35: The *Multicast gateway* caches the intact multicast transport object in its *Asset storage* subfunction as a playback delivery object.

In parallel, the *Content playback* function retrieves playback delivery objects (i.e. media segments) from the *Multicast gateway* function:

36: The *Content playback* function requests the next playback delivery object from the *Multicast gateway* function at reference point **L**.

37: The *Multicast gateway* checks whether it already has a copy of the requested playback delivery object cached in its *Asset storage* subfunction.

38: If there is no copy of the playback delivery object in its *Asset storage* subfunction, the *Multicast gateway* retrieves it from the *Content hosting* function via reference point **A**.

If the *Content hosting* function does not already have a cached copy of the playback delivery object, it fetches one from the DVB-DASH Segment Server subfunction of the *Content preparation* function via reference point **P$_{\text{in}}$**.

39: The *Multicast gateway* returns the playback delivery object to the requesting *Content playback* function.

At some point during the multicast-capable streaming session , the *Multicast gateway* switches to a different multicast transport session:

40: The *Multicast gateway* leaves the multicast group corresponding to the currently subscribed multicast transport session.

Playback of the selected DVB-I service then continues from step 31 until further user interaction.

A.2.4.4 Metrics reporting procedure

In bidirectional deployments only, the *Multicast gateway* reports metrics using the procedure illustrated in figure A.2.4.4-1 below.

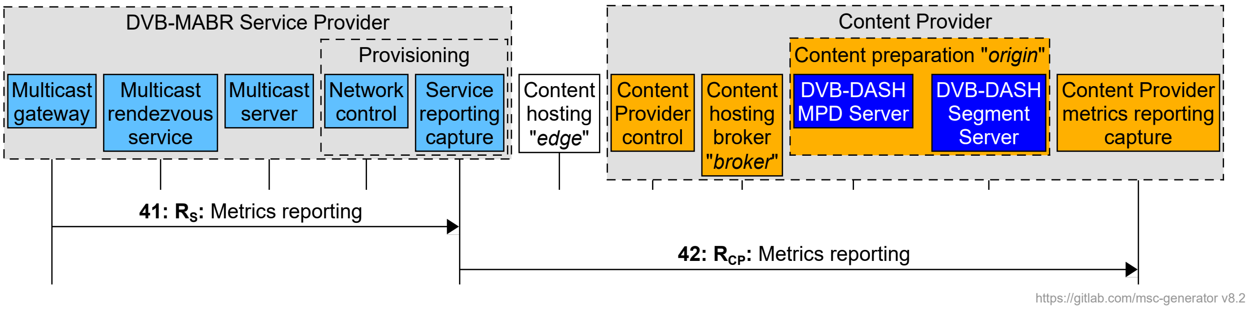

DVB-MABR Service Provider
Provisioning
Multicast gateway
Multicast rendezvous service
Multicast server
Network control
Service reporting capture
Content hosting "*edge*"
Content Provider
Content preparation "*origin*"
Content Provider control
Content hosting broker "*broker*"
DVB-DASH MPD Server
DVB-DASH Segment Server
Content Provider metrics reporting capture
41: R$_S$: Metrics reporting
42: R$_{CP}$: Metrics reporting
https://gitlab.com/msc-generator v8.2

Figure A.2.4.4-1: Call flow sequence for metrics reporting

The steps are as follows:

41: The *Multicast gateway* periodically sends a metrics report (formatted according to clause 11.1 of [i.1] to the *Service reporting capture* subfunction of *Provisioning*, using the reference point **R$_S$** procedure specified in clause 11.2 of [i.1].

42: The *Service reporting capture* subfunction may forward the metrics report to the *Content Provider metrics reporting capture* function via reference point **R$_{CP}$**.

NOTE: The interactions reference point **R$_{CP}$** are not further specified by [i.1].

History

Document history		
(A181)	January 2022	First BlueBook publication (A181).
(A181r1)	February 2023	Revised BlueBook publication (A181r1): - Added guidelines on the use of macro expansion to generate multiple multicast gateway configurations. - Added fourth worked example. - Added guidance in clause 4.1.2 on debugging the output of the Multicast Server when the macro expansion mechanism is in use. - Added guidance in clauses 4.1, 4.2 and 4.3 to describe how the Multicast server can dynamically enable/disable the transmission of individual multicast transport sessions in response to input from the Provisioning system.
(A181r2)	June 2023	Revised BlueBook publication (A181r2): - Added annex documenting baseline call flow.