

SES[▲]

SAT > IPTM

Protocol Specification

Version 1.2.2

<i>Author</i>	<i>Date</i>	<i>Version</i>	<i>Comment</i>
SES S.A.	08.01.2015	1.2.2	
British Sky Broadcasting Ltd			
Craftwork ApS			

Intellectual Property Notice

To the extent that any intellectual property rights subsist in this specification, the parties hereby agree not to assert any intellectual property rights that may vest in them in this specification.

In this instance, "the parties" means British Sky Broadcasting Limited, SES S.A. and Craftwork ApS, and "this specification" means the SAT>IP specification included in this document.

The information contained herein is provided on an "AS IS" basis, and to the maximum extent permitted by applicable law, the authors and developers of this specification hereby disclaim all other warranties and conditions, either express or implied, including but not limited to, any (if any) implied warranties, duties or conditions of merchantability and/or satisfactory quality, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, of lack of negligence.

Table of Contents

1	Introduction	6
1.1	SAT>IP Concept	6
1.2	Network Topology	7
1.3	Client Functionality	7
1.4	Specification Compliance	7
2	Usage Scenarios	8
2.1	IP Adapter / IP Multiswitch	8
2.2	IP LNB	8
2.3	Master STB	8
2.4	Universal Service Gateway	9
2.5	IP based SMATV / Multi-Dwelling Units	9
3	Protocol Specification	10
3.1	General	10
3.2	Addressing	11
3.2.1	DHCP	11
3.2.2	Auto-IP	11
3.3	Discovery	11
3.3.1	SSDP	11
3.3.2	Server Advertisements	12
3.3.3	DEVICE ID Negotiation	14
3.3.4	Client Search Requests	20
3.4	Description	21
3.4.1	XML Device Description	21
3.5	Control	24
3.5.1	RTSP	24
3.5.2	Setting up a new session (SETUP)	26
3.5.3	Starting the playout of a media stream (PLAY)	32
3.5.4	Maintaining a session (OPTIONS)	35
3.5.5	Modifying a media stream	36
3.5.6	Joining an existing stream	36
3.5.7	Listing available media streams (DESCRIBE)	37
3.5.8	Closing the session and stopping the playout (TEARDOWN)	40
3.5.9	RTSP Method Summary	41
3.5.10	Uniform Resource Identifier URI	43
3.5.11	Query Syntax	43
3.5.12	Example RTSP Sequence Diagram	46
3.5.13	IGMP	47
3.5.14	Status Code Definitions	50
3.5.15	RTCP Announcements	54
3.5.16	HTTP	56
3.6	Media Transport	57
3.6.1	RTP Transport	57
3.6.2	HTTP Streaming	57
3.7	Media Formats	57
4	Dynamic vs Static Server Operation	58
4.1	Dynamic Operation (default)	58
4.2	Static Operation	58
4.3	Mixed Operation	58
	Appendix A: Client Implementation Guidelines	59
	Appendix B: Example SAT>IP Message Exchanges	64
	Appendix C: Support for DVB-T/T2 (optional)	66
	Appendix D: Support for DVB-C/C2 (optional)	68
	Appendix E: Unicast Only Profile	70

Acronyms

CSV	Comma Separated Values
DHCP	Dynamic Host Configuration Protocol
DiSEqC	Digital Satellite Equipment Control
DLNA	Digital Living Network Alliance
DSL	Digital Subscriber Line
DVB	Digital Video Broadcasting
DVB-S	DVB for Satellite
DVR	Digital Video Recorder
FEC	Forward Error Correction
GENA	General Event Notification Architecture
HTML	HyperText Markup Language
HTTP	Hyper Text Transfer Protocol
HSPA	High Speed Packet Access
IF	Intermediate Frequency
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LAN	Local Area Network
LNB	Low Noise Block
MDU	Multi Dwelling Unit
MIME	Multipurpose Internet Mail Extensions
MPEG	Moving Picture Experts Group
MPTS	Multiple Program Transport Stream
MUDP	Multicast UDP
NAS	Network Attached Storage
PLC	Power Line Communication
PoE	Power over Ethernet
PSK	Phase Shift Keying
PVR	Personal Video Recorder
QPSK	Quaternary Phase Shift Keying
RFC	Request For Comments
RTP	Real-time Transport Protocol
RTCP	Real-time Transport Control Protocol
RTSP	Real Time Streaming Protocol
SDES	Source Description
SDP	Session Description Protocol
SI	Service Information
SMATV	Satellite Master Antenna Television
SOAP	Simple Object Access Protocol
SPTS	Single Program Transport Stream
SR	Sender Report
SSDP	Simple Service Discovery Protocol
STB	Set-Top-Box
TCP	Transport Control Protocol
TS	Transport Stream
UDP	User Datagram Protocol
UPnP	Universal Plug and Play
UPnP AV	UPnP Audio and Video
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF	UCS Transformation Format
WLAN	Wireless LAN
XML	Extensible Markup Language

References

UPnP Forum

- [1] UPnP Device Architecture 1.1

DLNA

- [2] DLNA Networked Device Interoperability Guidelines

IETF

- [3] RFC 2131 – DHCP (Dynamic Host Configuration Protocol)
- [4] RFC 2250 – RTP Payload Format for MPEG1/MPEG2 Video
- [5] RFC 2279 – UTF-8, a transformation format of ISO 10646
- [6] RFC 2326 – Real Time Streaming Protocol (RTSP)
- [7] RFC 3376 – Internet Group Management Protocol, Version 3
- [8] RFC 3550 – RTP: A Transport Protocol for Real-Time Applications
- [9] RFC 3927 – Dynamic Configuration of IPv4 Link-Local Addresses
- [10] RFC 4566 – SDP: Session Description Protocol
- [11] draft-cai-ssdp-v1-03 – Simple Service Discovery Protocol/1.0

ITU / ISO

- [12] ITU-T Recommendation H.222.0 / ISO/IEC 13818-1: "Information Technology - Generic Coding of moving pictures and associated audio information: Systems"

ETSI / DVB

- [13] ETSI TS 101 154 V1.9.1; Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream

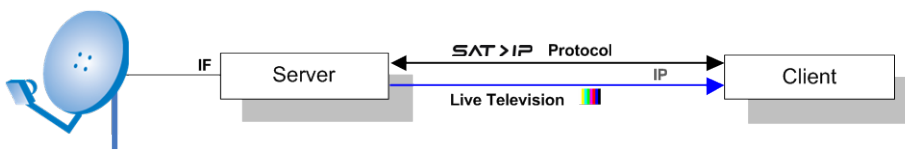
1 Introduction

This document describes the SAT>IP protocol and its usage.

The SAT>IP protocol provides a standardised way for IP clients to access live media broadcasts from satellite reception servers on IP networks.

SAT>IP specifies a **communication protocol**. SAT>IP is **not** a device specification. The SAT>IP protocol may be applied in different devices. Industry is left to come up with new and innovative devices using the SAT>IP protocol.

The SAT>IP protocol distinguishes between SAT>IP clients and SAT>IP servers. Actual devices may be clients or servers or both depending on their feature set.



SAT>IP Clients

SAT>IP clients provide the possibility of selecting and receiving live television programs. SAT>IP clients may be – DVB compliant Set-Top-Boxes (STBs) with an IP interface or – Software applications running on programmable hardware such as Tablets, PCs, Smartphones, Connected Televisions, NAS, Routers, etc.

SAT>IP Servers

SAT>IP servers answer requests from SAT>IP clients and forward live television programs to these clients. Servers may take various forms from simple in-home IP Adapters / Multiswitches, Master STBs to ultimately IP LNBs and large MDU headends covering whole buildings or cities.

1.1 SAT>IP Concept

Unlike in today's satellite distribution schemes, the SAT>IP architecture allows the reception of satellite television programs also on devices which do not have a satellite tuner directly built-in. Satellite tuners and demodulators are moved or "remoted" into SAT>IP server devices. Clients control SAT>IP servers via the SAT>IP protocol. SAT>IP is a remote tuner control protocol which provides the possibility of remotely controlling tuning devices.

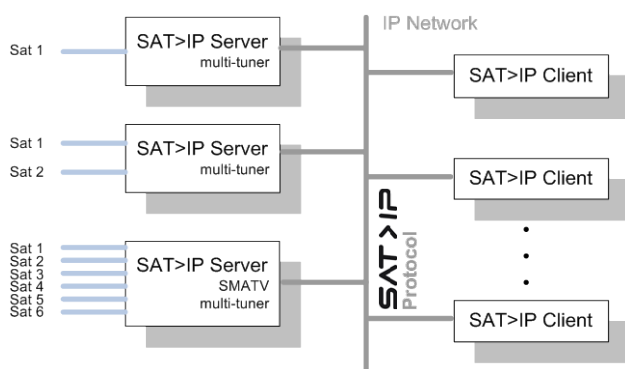
This means that the reception of satellite delivered programming can be dealt with by clients purely in software, provided a SAT>IP server is available on a network. Satellite programs become available on devices which would never be capable of supporting satellite TV otherwise e.g. Tablets.

From the distribution point of view, satellite distribution becomes physical layer agnostic and satellite services can be forwarded over all the latest types of IP wired or wireless technologies such as Powerline (PLC), Wireless LANs, Optical Fiber Distribution, etc.

1.2 Network Topology

The SAT>IP protocol is designed to suit different application scenarios. The same SAT>IP client can communicate with various types of live media servers ranging from single satellite, single tuner servers to multi-satellite multi-tuner servers. SAT>IP clients can be designed to work in single home scenarios as well as in SMATV or large community systems.

The number of clients that can be simultaneously supported depends on the particular server implementation. Large servers can potentially serve an unlimited number of SAT>IP clients. SAT>IP servers can also be stacked and run in parallel on the same network.



1.3 Client Functionality

SAT>IP is a client driven architecture. Clients send requests to servers. Servers execute these requests and forward live television programs to clients.

SAT>IP servers ideally do not need to be configured. They are purely connected to IF satellite signals on their input and the IP network on their output.

Clients specify what they would like to receive. In this sense SAT>IP is very much comparable to today's satellite distribution architectures which are also not aware of the particular signals being received and watched by clients.

SAT>IP servers are flexible in the way in which signals are transported to clients, however the logic of what is being received resides in the clients.

SAT>IP does not specify the mechanisms through which SAT>IP clients learn about the streams which are available to them. Clients can parse DVB Service Information / Program Specific Information for learning about services available, but they can also rely on service lists containing this information.

1.4 Specification Compliance

In order to be compliant with the SAT>IP specification, SAT>IP servers need to fully implement the following specification. All protocol mechanisms and tools shall be implemented.

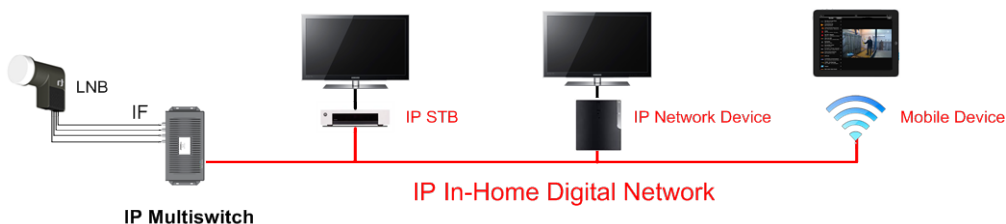
SAT>IP clients on the other hand only need to implement the mechanisms which allow them to properly operate in a UPnP and RTSP environment.

2 Usage Scenarios

The SAT>IP protocol has been designed with the following use cases in mind. This listing is however not exhaustive, it is simply meant to describe the most common usage scenarios:

2.1 IP Adapter / IP Multiswitch

The IP Adapter / IP Multiswitch is a device which is located close to the satellite antenna(s) or traditional RF multiswitches and converts satellite programming towards IP for in-home distribution.



The IP Adapter / Multiswitch converts (tunes, demodulates and demultiplexes) satellite IF signals towards IP. It does not feature a built-in video decoder. IP Adapter / Multiswitches generally have multiple tuners in order to serve multiple concurrent programs to multiple clients. The IP Adapter / Multiswitch understands the SAT>IP protocol on its IP interface and is capable of answering requests coming from IP network clients.

2.2 IP LNB

With increasing levels of integration, the SAT>IP concept enables the creation of new IP LNB devices which feature direct Ethernet connectivity and are powered via Power of Ethernet (PoE). This new generation of LNBs will no longer be connected via coaxial cable and will function as live media servers for the in-home network.



2.3 Master STB

A Master STB receives DVB-S2 satellite programming via its standard IF interfaces and forwards live television programs to one or more client STBs via an in-home IP network. Clients can also be dedicated software applications running on programmable devices (PCs, smartphones, tablets, etc.).

The SAT>IP protocol specifies how clients communicate with the master STB in order to access live streams.

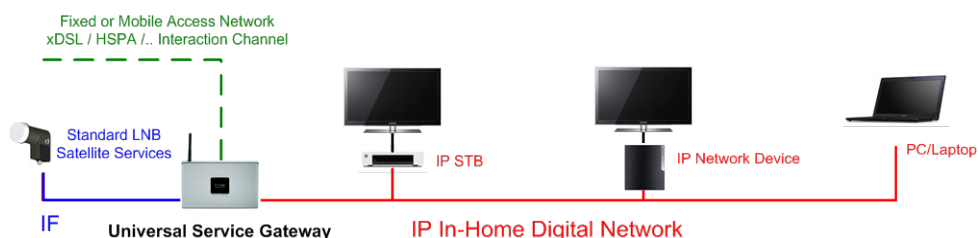


The master STB often features the possibility of recording content. This document does not specify how recorded content is accessed nor how content scheduling is handled. Such use cases are described in the DLNA protocol specifications. The SAT>IP protocol purely describes access to live media content.

The Master STB may include additional functions which are not part of the present document such as transcribing of programs from a proprietary CA solution to a DRM solution and transcoding from the over-the-air codec to an alternative codec and different bitrate.

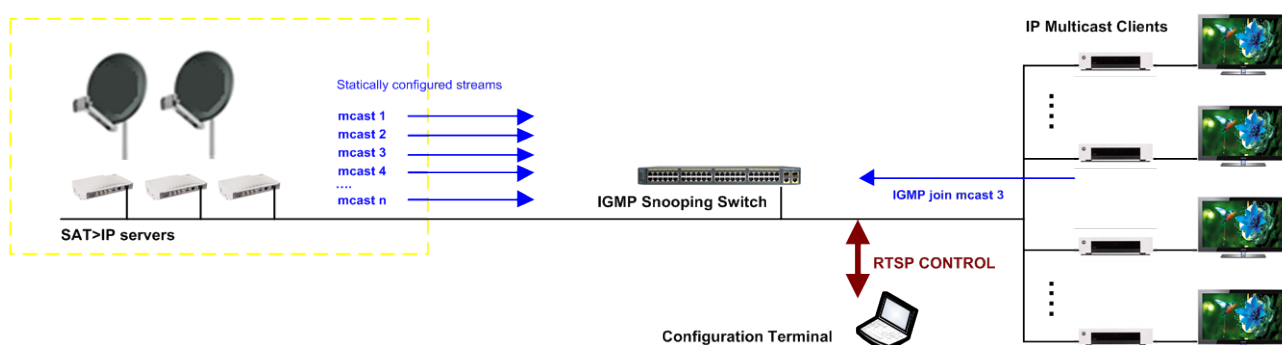
2.4 Universal Service Gateway

The Universal Service Gateway is an Internet Access device that features Fixed or Mobile Access Modems plus Satellite Reception Hardware. It combines access to both broadband and broadcast networks in one device.



2.5 IP based SMATV / Multi-Dwelling Units

The SAT>IP protocol was designed in order to also support very large satellite reception installations. Such installations are often found in Multi-Family Housing Units, Communal Dish Installations, Hotels, Hospitals, Fiber based Greenfield Satellite Deployments. SAT>IP STBs can be designed to work in small as well as in very large environments.



3 Protocol Specification

3.1 General

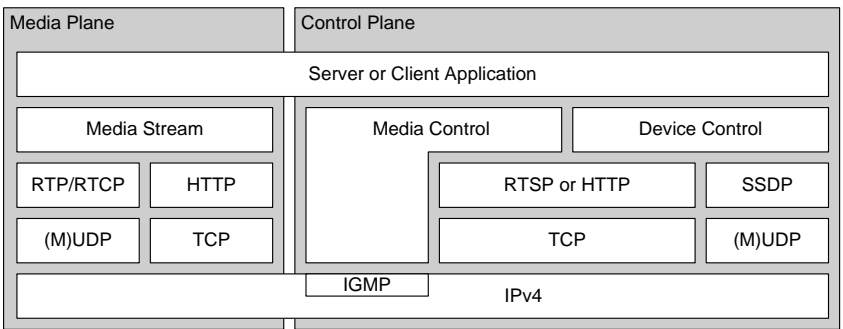
The SAT>IP protocol allows IP based clients to interact/communicate with IP based satellite servers for live media forwarding.

SAT>IP builds on industry standards and does complement those only where necessary i.e. where there are no established satellite specific extensions.

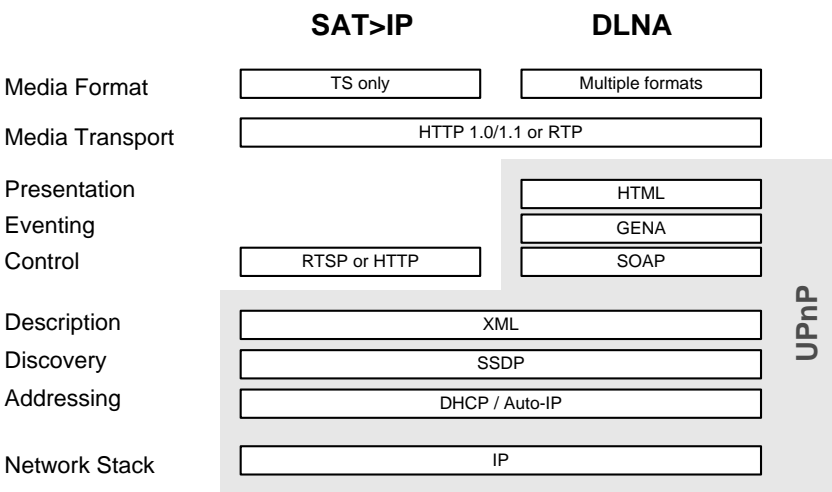
The SAT>IP protocol makes use of:

- UPnP for Addressing, Discovery and Description,
- RTSP or HTTP for Control,
- RTP or HTTP for Media Transport.

The SAT>IP protocol stack is organised in the following way:



SAT>IP uses a subset of the UPnP/DLNA architecture and protocols [1] [2] and SAT>IP devices can be extended to also become DLNA devices. As an example a SAT>IP client could access live media streams through the SAT>IP protocol and access recorded media streams through DLNA.



SAT>IP devices successively go through the following phases: Addressing, Discovery, Description, Control and finally Media Transport.

3.2 Addressing

Addressing is the process for a SAT>IP device (client or server) to obtain a network address. This is a pre-requisite before any communication can take place. SAT>IP follows the UPnP specification [1] by offering two options for address acquisition:

3.2.1 DHCP

Every SAT>IP device shall have a Dynamic Host Configuration (DHCP) client and search for a DHCP server when the device is first connected to the network [3]. The device from that point onwards shall use the IP address assigned to it.

3.2.2 Auto-IP

If no DHCP server can be located, the network is unmanaged and SAT>IP devices shall autoconfigure their IP address [9] from the Auto-IP range 169.254/16. In Auto-IP mode, SAT>IP servers shall periodically check for the existence of a DHCP server. All SAT>IP protocol mechanisms (SSDP, RTSP, etc.) shall work whichever way the IP address is obtained.

3.3 Discovery

During the discovery phase SAT>IP servers advertise their presence to other SAT>IP servers and clients.

When joining a network, SAT>IP clients search the network for available SAT>IP servers.

3.3.1 SSDP

Discovery in SAT>IP relies on the Simple Service Description Protocol (SSDP) [11] as specified in the UPnP Device Architecture 1.1 [1].

As a minimum:

- a SAT>IP server is a UPnP Device and a UPnP Control Point,
- a SAT>IP client is a UPnP Control Point.

DEVICE TYPE / URN

Every UPnP device has a specific type corresponding to a certain category of device. This is expressed with a Unique Resource Name (URN) in UPnP. The device type of a SAT>IP server is:

urn:**ses-com**:device:**SatIPServer**:1

where:

- "ses-com" is the domain-name,
- "SatIPServer" is the deviceType name,
- and "1" is the version of the device type.

UUID

Each instance of a SAT>IP server is uniquely identified by its Universally Unique Identifier (UUID) string. The UUID string shall have the format specified in [1]: 4B-2B-2B-2B-6B where B represents a Byte written as two hexadecimal digits. The UUID of a device shall always remain fixed.

Example of a UUID string:

"2fac1234-31f8-11b4-a222-08002b34c003"

3.3.2 Server Advertisements

SAT>IP servers joining a network multicast **three** different NOTIFY ssdp:alive messages to the SSDP address 239.255.255.250 port 1900. This is a requirement for UPnP root devices according to the UPnP Device Architecture 1.1 [1].

ssdp:alive

Method	NOTIFY	
Request	Protocol	MUDP (Multicast UDP)
	Headers	As specified in the UPnP Device Architecture 1.1. Announcement messages from SAT>IP servers shall include the "DEVICEID.SES.COM:" header extension field with the DEVICE ID value of a particular server.
Response	No response is sent.	
Example	Request	NOTIFY * HTTP/1.1 HOST: 239.255.255.250:1900 CACHE-CONTROL: max-age=1800 LOCATION: http://<SatIPServer_IP_Address>/<description>.xml NT: <notification type> NTS: ssdp:alive SERVER: OS/version UPnP/1.1 product/version USN: <unique service name> BOOTID.UPNP.ORG: <bootID> CONFIGID.UPNP.ORG: <configID> SEARCHPORT.UPNP.ORG: <searchPort> DEVICEID.SES.COM: <DEVICEID> <CRLF>

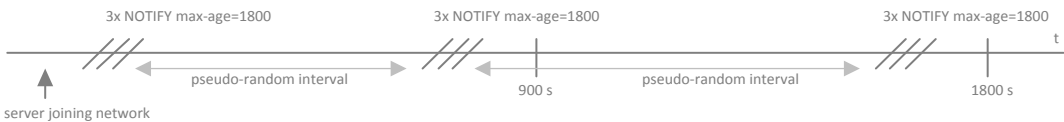
The **NTS** header value of these packets is ssdp:alive or ssdp:byebye

The **NT** (Notification Type) header value is different in each of the three NOTIFY messages. It successively announces the root device, its device uuid and the urn (please see the example below).

The **USN** (Unique Service Name) also changes between the three announcement packets and carries a composite identifier in concordance with the NT value. Please check table 1-1 of the UPnP Device Architecture 1.1 [1] for more information.

The **LOCATION** header announces the URL to the UPnP device description (Section 3.4). The URL should have a simple format e.g. http://<SatIPServer_IP_Address>/desc.xml.

The **CACHE-CONTROL** header announces the value during which the advertisement remains valid and before it needs to be re-sent. (1800 seconds in the example above). Multicast NOTIFY ssdp:alive messages are sent regularly by all SAT>IP servers in order to signal their continued presence. The refreshing of the NOTIFY advertisement set (consisting of the three messages) has to be done at a randomly-distributed interval of less than one half of the advertisement expiration date (e.g. 900 seconds for a max-age of 1800 seconds). This guarantees that the announcement set is repeated at least twice before it expires.



The **BOOTID.UPNP.ORG** value shall increase each time that the device first announces on the network. It shall be stored in non-volatile memory.

The **CONFIGID.UPNP.ORG** value represents the configuration number of a root device. This value needs to be identical to the "**configId=**" value in the <root> section of the XML description file (Section 3.4).

The **SEARCHPORT.UPNP.ORG** field is optional. It's use is **not** recommended in SAT>IP. SAT>IP devices are expected to listen to unicast M-SEARCH messages on the standard port 1900. Only if port 1900 is not available may a device select a different listening port for unicast M-SEARCH messages. If a device sends the SEARCHPORT.UPNP.ORG header field it must respond to unicast M-SEARCH messages sent to this advertised port.

In SAT>IP networks several SAT>IP server devices may coexist on the same network. In order to distinguish themselves from each other, every device assigns itself a unique DEVICE ID on the network. This DEVICE ID is carried in the mandatory **DEVICEID.SES.COM** header field. Its use is explained in Section 3.3.3. The DEVICEID.SES.COM header field is only required for servers implementing multicast transport delivery.

Please note that in UPnP,

- header field names are case insensitive and header field values are case sensitive,
- the order of the header fields does not matter,
- linear white spaces following the colon after a header field and in front of a header value shall be ignored by clients.

The TTL value of each IP multicast SSDP packet should default to 2.

Example Announcement of a SAT>IP server joining the network:

```

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: upnp:rootdevice
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c::upnp:rootdevice
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: urn:ses-com:device:SatIPServer:1
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

```

A SAT>IP server **present** on the network has to re-announce itself on a regular basis as described under CACHE-CONTROL above.

A SAT>IP server **leaving** the network needs to signal its departure by sending three different NOTIFY messages with the NTS value ssdp:byebye.

Please note that the **ssdp:byebye** messages should not include the CACHE-CONTROL, LOCATION, SERVER and DEVICEID.SES.COM headers. An example of such ssdp:byebye messages is shown in Section 3.3.3.

A SAT>IP server **changing** the network e.g. when passing from an Auto-IP network to a network with a DHCP assigned address shall signal its departure from one network by sending three NOTIFY messages with the NTS value ssdp:byebye on that network and shall announce its presence on the new network by sending three NOTIFY ssdp:alive messages.

SAT>IP **clients** (being at a minimum only UPnP Control Points) do not announce their presence. For this reason, a client leaving the network is not detectable at this level. The SAT>IP protocol however implements RTSP and IGMP which permit to detect the presence or absence of a client (RTSP session timeout, IGMP membership queries).

3.3.3 DEVICE ID Negotiation

Every SAT>IP server on a network carries its own non-clashing DEVICE ID value. The DEVICE ID value is negotiated between different SAT>IP servers when they join the network. The DEVICE ID value allows each SAT>IP server to uniquely associate itself with a different IP multicast address range on the network (see below).

When a server performs a cold start it reads the initial DEVICE ID value from its **non-volatile memory**. This value shall be the last value that the server had used. The server then multicasts this DEVICE ID value as part of the announcement NOTIFY messages that were described in section 3.3.2.

Other SAT>IP servers already on the network (acting as control points) listen for these announcements.

As long as the DEVICE ID value received by these servers does not conflict with their own DEVICE ID value, they will not react.

In case of a [DEVICE ID conflict](#), however the server that notices that another server would like to use its own DEVICE ID (DEVICE ID clash) needs to defend its own DEVICE ID, by sending (within 1 second) a **unicast** M-SEARCH message containing the in-use DEVICE ID to the IP address of the joining server and **port 1900** or the port contained in the SEARCHPORT.UPNP.ORG field if present.

The joining server acknowledges receipt via a 200 OK response repeating the current DEVICE ID value.

Method	M-SEARCH	
Request	Protocol	UDP unicast
	Headers	As specified in the UPnP Device Architecture 1.1. A request from a server shall include a "DEVICEID.SES.COM:" field name with the DEVICE ID value of that server. The target server acknowledges receipt in repeating the DEVICE ID value in the response. A "DEVICEID.SES.COM:" field shall not be included in a client request.
Response	Line	HTTP/1.1 200 OK
	Protocol	UDP
	Headers	As specified in the UPnP Device Architecture 1.1. The "DEVICEID.SES.COM:" field name with a DEVICE ID value shall be returned only if the request includes the "DEVICEID.SES.COM:" field name.
Example	Request	M-SEARCH * HTTP/1.1 HOST: <Target_SatIPServer_IP_Address> MAN: "ssdp:discover" ST: urn:ses-com:device:SatIPServer:1 USER-AGENT: OS/version UPnP/1.1 product/version DEVICEID.SES.COM: 3 <CRLF>
	Response	HTTP/1.1 200 OK CACHE-CONTROL: max-age=1800 DATE: <date> EXT: LOCATION: http://<Target_SatIPServer_IP_Address>/<description>.xml SERVER: OS/version UPnP/1.1 product/version ST: urn:ses-com:device:SatIPServer:1 USN: uuid:01234567-0123-0123-0123-0123456789ab::urn:ses-com:device:SatIPServer:1 BOOTID.UPNP.ORG: <bootID> CONFIGID.UPNP.ORG: <configID> SEARCHPORT.UPNP.ORG: <searchPort> DEVICEID.SES.COM: 3 <CRLF>

Please note that the [EXT](#) field in 200 OK responses is required for backwards compatibility with UPnP 1.0. It consists of a header name only and has no field value.

As a consequence the joining server may leave the network by multicasting three NOTIFY ssdp:byebye messages. These messages are optional. They do not contain the DEVICEID.SES.COM field.

The joining server then has to generate a new DEVICE ID value (it should increment the former value by one) and re-announces itself on the network by sending three NOTIFY ssdp:alive messages containing the new DEVICE ID value.

If within a 5 second timeout period no unicast M-SEARCH message with a conflicting DEVICE ID field is received by the joining server, the joining server allocates itself the new DEVICE ID value and stores it in non-volatile memory.



Installation Scenario

The process above is designed for the following mode of operation: on initial installation SAT>IP servers are connected to the same network and powered up one after the other. Each server will therefore sequentially acquire its own DEVICE ID value. After a power failure, server devices will simply read the last configured DEVICE ID value from non-volatile memory and re-boot more rapidly.

In order for installers to assess when to power up the next device during initial installation, it is recommended that the device provides visual feedback when the timeout period has expired (e.g. LED changing colour or pattern).

Example SSDP message flow:

The message flow and sequence diagram below show the packet sequence when a second server arrives and claims a DEVICE ID already in use by a first server:

Server A joins an empty network and announces its DEVICE ID 1.

To: 239.255.255.250 Port 1900

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: upnp:rootdevice
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c::upnp:rootdevice
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.21/desc.xml
NT: urn:ses-com:device:SatIPServer:1
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2318
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1
```

As Server A is alone on the network initially it assigns itself DEVICE ID 1 (after the 5 second timeout period).

Server B joins the network and also wants to take DEVICE ID 1. It sends three NOTIFY messages.

To: 239.255.255.250 Port 1900

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: upnp:rootdevice
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::upnp:rootdevice
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: urn:ses-com:device:SatIPServer:1
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

Server A hears the announcements of Server B and sees that it needs to counter the request as the DEVICE IDs are clashing

Server A sends a unicast M-SEARCH message directly to the IP address of server B: Port 1900

M-SEARCH * HTTP/1.1
HOST: 192.168.178.21:1900
MAN: "ssdp:discover"
ST: urn:ses-com:device:SatIPServer:1
USER-AGENT: Linux/1.0 UPnP/1.1 IDL4K/1.0
DEVICEID.SES.COM: 1

Server B acknowledges the message with a 200 OK

HTTP/1.1 200 OK
CACHE-CONTROL: max-age=1800
DATE: Sat Jan 1 00:00:20 2000
EXT:
LOCATION: http://192.168.178.41/desc.xml
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
ST: urn:ses-com:device:SatIPServer:1
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 1

Server B gives up DEVICE ID 1 and may send 3 NOTIFY ssdp:byebye messages

To: 239.255.255.250 Port 1900

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
NT: upnp:rootdevice
NTS: ssdp:byebye
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::upnp:rootdevice
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
NT: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
NTS: ssdp:byebye
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
NT: urn:ses-com:device:SatIPServer:1
NTS: ssdp:byebye
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2
CONFIGID.UPNP.ORG: 0
```

Server B then has to announce that it takes DEVICE ID 2 by sending 3 NOTIFY ssdp:alive messages

To: 239.255.255.250 Port 1900

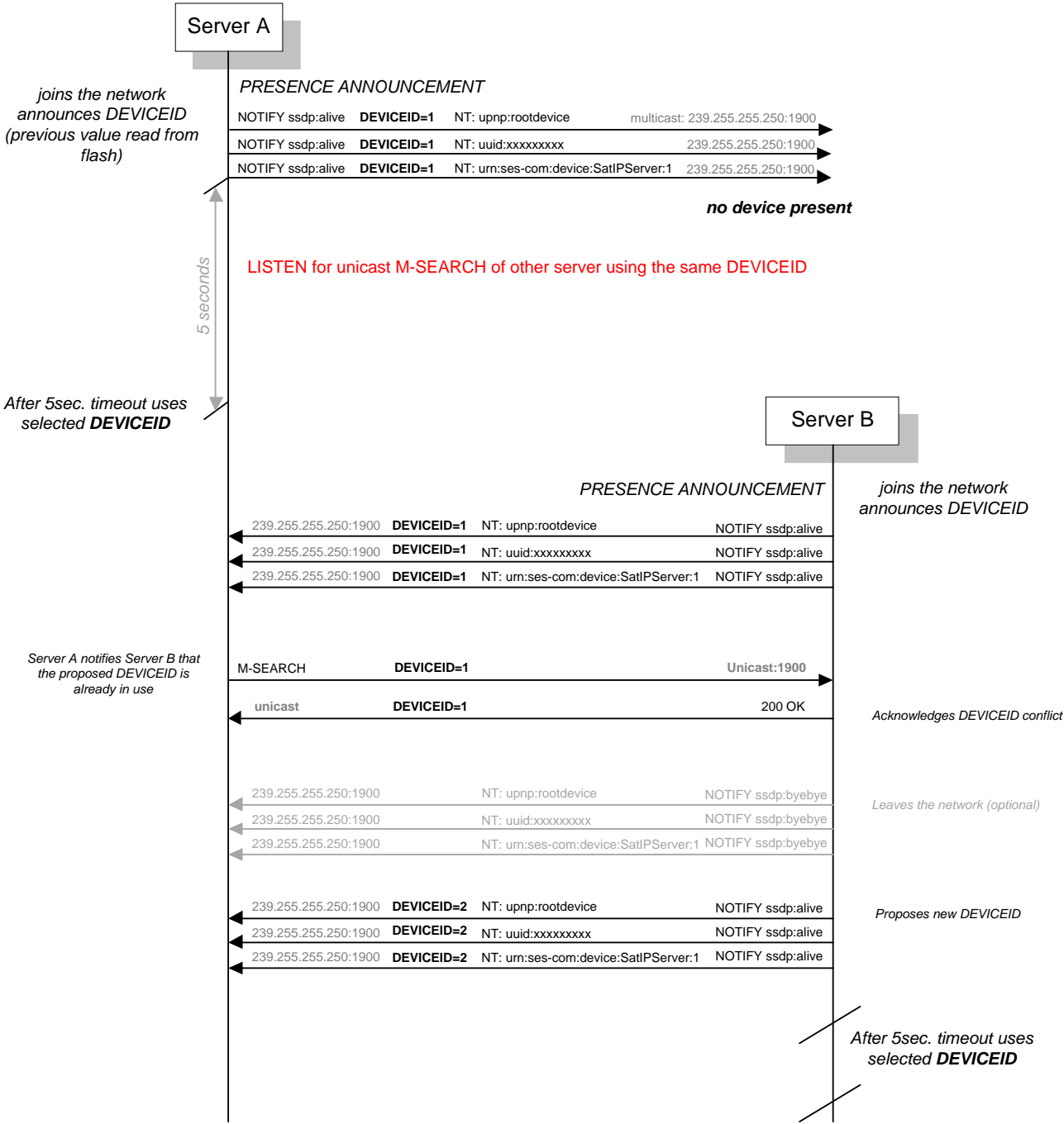
```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: upnp:rootdevice
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::upnp:rootdevice
BOOTID.UPNP.ORG: 3
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 2
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63
BOOTID.UPNP.ORG: 3
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 2
```

```
NOTIFY * HTTP/1.1
HOST: 239.255.255.250:1900
CACHE-CONTROL: max-age=1800
LOCATION: http://192.168.178.41/desc.xml
NT: urn:ses-com:device:SatIPServer:1
NTS: ssdp:alive
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
USN: uuid:5a0c857b-add1-42d0-8673-b5ca60df4a63::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 3
CONFIGID.UPNP.ORG: 0
DEVICEID.SES.COM: 2
```

After the 5 second timeout period Server B allocates itself DEVICE ID 2

SEQUENCE DIAGRAM



3.3.3.1 DEVICE ID Header Field

The following table summarizes when the DEVICEID.SES.COM field is required in SSDP messages.

SAT>IP	UPnP	SSDP Message	Protocol	Header Field
DEVICEID.SES.COM				
Server	Device	NOTIFY ssdp:alive	MUDP	required
		NOTIFY ssdp:byebye	MUDP	not required
	Control Point	M-SEARCH	UDP	required
	Device	200 OK	UDP	required
Client	Control Point	M-SEARCH	MUDP/UDP	not required
	Server Device	200 OK	UDP	not required

Please note that the response to an M-SEARCH message contains the "DEVICEID.SES.COM:" field only if the M-SEARCH request itself included the "DEVICEID.SES.COM:" field.

3.3.3.2 Multicast Address Range

In SAT>IP each server issues multicast addresses from its own multicast address range (which is not conflicting with multicast address ranges from other SAT>IP servers). This multicast address range is uniquely determined by the DEVICE ID value that each server allocates to itself during the device advertisement phase. The DEVICEID has a value in the range 0-255.

The address range for a given SAT>IP server is given as follows:

239. <DEVICEID> . <0-254> . <0-254>

Example: the SAT>IP server with DEVICEID=3 will issue addresses from the range: 239.3.x.x

3.3.4 Client Search Requests

SAT>IP clients issue a **multicast** M-SEARCH message to **discover** SAT>IP servers on the local network. Due to the unreliable nature of UDP, the recommendation is to send three identical M-SEARCH messages in 100 milliseconds. The TTL value of all multicast M-SEARCH messages is 2.

Method	M-SEARCH	
Request	Protocol	MUDP (Multicast UDP)
	Headers	As specified in the UPnP Device Architecture 1.1.
Response	Line	HTTP/1.1 200 OK
	Protocol	UDP
	Headers	As specified in the UPnP Device Architecture 1.1.
Example	Request	M-SEARCH * HTTP/1.1 HOST: 239.255.255.250:1900 MAN: "ssdp:discover" MX: 2 ST: urn:ses-com:device:SatIPServer:1 USER-AGENT: OS/version UPnP/1.1 product/version <CRLF>
	Response	HTTP/1.1 200 OK CACHE-CONTROL: max-age=1800 DATE: <date> EXT: LOCATION: http://<SatIPServer_IP_Address>/<description>.xml SERVER: OS/version UPnP/1.1 product/version ST: urn:ses-com:device:SatIPServer:1 USN:uuid:01234567-0123-0123-0123-0123456789ab::urn:ses-com:device:SatIPServer:1 BOOTID.UPNP.ORG: <bootID> CONFIGID.UPNP.ORG: <configID> <CRLF>

The [MAN](#) header value of these request packets is "ssdp:discover"

The [ST](#) header value contains the Search Target of the request. The value normally corresponds to the SAT>IP server URN.

The [MX](#) field contains the maximum wait time in seconds. The recommended value in SAT>IP is 2 seconds. Device responses should be delayed a random duration between 0 and this many seconds to balance load for the control point when it processes responses.

Example Search Request of a SAT>IP Client and Server Response:

In order to discover servers on the network a client sends a multicast M-SEARCH ssdp:discover message to 239.255.255.250 Port 1900

```
M-SEARCH * HTTP/1.1
HOST: 239.255.255.250:1900
ST: urn:ses-com:device:SatIPServer:1
MAN: "ssdp:discover"
MX: 2
```

Servers respond in unicast UDP to IP address and port where request came from

```
HTTP/1.1 200 OK
CACHE-CONTROL: max-age=1800
DATE: Sat Jan 1 00:01:50 2000
EXT:
LOCATION: http://192.168.178.21/desc.xml
SERVER: Linux/1.0 UPnP/1.1 IDL4K/1.0
ST: urn:ses-com:device:SatIPServer:1
USN: uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c::urn:ses-com:device:SatIPServer:1
BOOTID.UPNP.ORG: 2399
CONFIGID.UPNP.ORG: 0
```

SAT>IP servers shall not issue multicast M-SEARCH messages to detect other SAT>IP servers.

3.4 Description

The Description phase allows SAT>IP devices to provide more information about themselves to UPnP control points on the network. The device description is done via an XML file.

The location of this file is acquired during the discovery phase: the LOCATION field in the headers of the multicast NOTIFY ssdp:alive messages contains the URL of the description file. The XML file should have a simple name such as e.g. "desc.xml".

3.4.1 XML Device Description

```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0" configId="configuration number">
  <specVersion>
    <major>1</major>
    <minor>1</minor>
  </specVersion>
  <device>
    <deviceType>urn:ses-com:device:SatIPServer:1</deviceType>
    <friendlyName>short user-friendly title</friendlyName>
    <manufacturer>manufacturer name</manufacturer>
    <manufacturerURL>URL to manufacturer site</manufacturerURL>
    <modelDescription>long user-friendly title</modelDescription>
    <modelName>model name</modelName>
    <modelNumber>model number</modelNumber>
    <modelURL>URL to model site</modelURL>
    <serialNumber>manufacturer's serial number</serialNumber>
    <UDN>uuid:UUID</UDN>
    <UPC>Universal Product Code</UPC>
    <iconList>
      <icon>
        <mimetype>image/format</mimetype>
        <width>horizontal pixels</width>
        <height>vertical pixels</height>
        <depth>color depth</depth>
        <url>URL to icon</url>
      </icon>
    </iconList>
    <presentationURL>URL for presentation</presentationURL>
  </device>
</root>
```

The "**configId**=" value in the <root> section of the XML description file needs to be identical to the CONFIGID.UPNP.ORG value used during the UPnP Discovery phase.

A SAT>IP device shall provide two JPEG icons and two PNG icons:

- the image width and height of the small JPEG and PNG icons must be **48** pixels,
- the image width and height of the large JPEG and PNG icons must be **120** pixels.

The icon <url> must be relative to the URL at which the device description is located. Example for LOCATION: <http://192.168.128.192:40912/desc.xml> the url element should be <url>/icons/small.jpg</url>.

If the SAT>IP server integrates an HTTP server with an end-user interface, the URL to the presentation root shall be provided in the <presentationURL> field. Again the presentation URL shall be relative to the URL at which the device description is located.

Advertising the number of available tuners

A <satip:X_SATIPCAP> capabilities element should be included in the XML description of a SAT>IP server in order to provide the SAT>IP supported modulation systems with the number of corresponding front-ends.

This additional XML element shall be put at the end of the <device> elements in the XML Device Description.

Example of a SAT>IP server including eight DVB-S2 front-ends and four DVB-T front-ends:

<satip:X_SATIPCAP xmlns:satip="urn:ses-com:satip">DVBS2-8,DVBT-4</satip:X_SATIPCAP>

The namespace for <satip:X_SATIPCAP> must be "urn:ses-com:satip" and the namespace prefix must be "satip:".

Syntax:

The value of the <satip:X_SATIPCAP> element is a comma separated list as defined below:

satipcap-value	=	satipres "(" , " satipres)	; CSV list
satipres	=	satipmsys "-" satipfe	; Modulation system followed by the number of front-ends
satipmsys	=	"DVBS2" / "DVBT" / "DVBT2" / "DVBC" / "DVBC2"	; Supported modulation system
satipfe	=	1*DIGIT	; Number of front-ends supporting this modulation system
DIGIT	=	%x30-39	; 0-9

Server providing a link to a channel list

A <satip:X_SATIPM3U> capabilities element may be included in the XML description of a SAT>IP server in order to provide clients with a pointer to a channel list in the m3u format (described in Appendix A).

The advantage of a server provided channel list is that it allows clients to automatically set themselves up for all receivable channels without each client having to run through a channel search and antenna configuration process.

The namespace for <satip:X_SATIPM3U> must be "urn:ses-com:satip" and the namespace prefix must be "satip:".

This additional XML element shall be put at the end of the <device> elements in the XML Device Description. (after the <satip:X_SATIPCAP> element if present)

Example of a SAT>IP server pointing to a local channel list on the server:

<satip:X_SATIPM3U xmlns:satip="urn:ses-com:satip">/channellist.m3u</satip:X_SATIPM3U>

Example of a SAT>IP server pointing to an external channel list:

<satip:X_SATIPM3U xmlns:satip="urn:ses-com:satip">http://www.example.com/channellist.m3u</satip:X_SATIPM3U>

Example Description File

```

<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0" configId="0">
  <specVersion>
    <major>1</major>
    <minor>1</minor>
  </specVersion>
  <device>
    <deviceType>urn:ses-com:device:SatIPServer:1</deviceType>
    <friendlyName>SATIPBOX</friendlyName>
    <manufacturer>Manufacturer</manufacturer>
    <manufacturerURL>http://www.manufacturer.com</manufacturerURL>
    <modelDescription>SATIPBOX 500 4.0</modelDescription>
    <modelName>SATIPBOX</modelName>
    <modelName>SATIPBOX</modelName>
    <modelNumber>1.0</modelNumber>
    <modelURL>http://www.manufacturer.com/satipbox</modelURL>
    <serialNumber>1S81A31231000007</serialNumber>
    <UDN>uuid:50c958a8-e839-4b96-b7ae-8f9d989e136c</UDN>
    <iconList>
      <icon>
        <mimetype>image/png</mimetype>
        <width>48</width>
        <height>48</height>
        <depth>24</depth>
        <url>/icons/sm.png</url>
      </icon>
      <icon>
        <mimetype>image/png</mimetype>
        <width>120</width>
        <height>120</height>
        <depth>24</depth>
        <url>/icons/lr.png</url>
      </icon>
      <icon>
        <mimetype>image/jpeg</mimetype>
        <width>48</width>
        <height>48</height>
        <depth>24</depth>
        <url>/icons/sm.jpg</url>
      </icon>
      <icon>
        <mimetype>image/jpeg</mimetype>
        <width>120</width>
        <height>120</height>
        <depth>24</depth>
        <url>/icons/lr.jpg</url>
      </icon>
    </iconList>
    <presentationURL>/index.htm</presentationURL>
    <satip:X_SATIPCAP xmlns:satip="urn:ses-com:satip">DVBS2-8,DVBT-4</satip:X_SATIPCAP>
    <satip:X_SATIPM3U xmlns:satip="urn:ses-com:satip">/channellist.m3u</satip:X_SATIPM3U>
  </device>
</root>

```

3.5 Control

Control provides the functionality necessary for SAT>IP clients to request the delivery of media streams from SAT>IP servers. Device Control in SAT>IP can be handled through the use of RTSP or HTTP protocol mechanisms.

SAT>IP servers shall fully implement all protocol mechanisms specified in the current specification. Clients only need to implement those SAT>IP protocols important to their own proper operation.

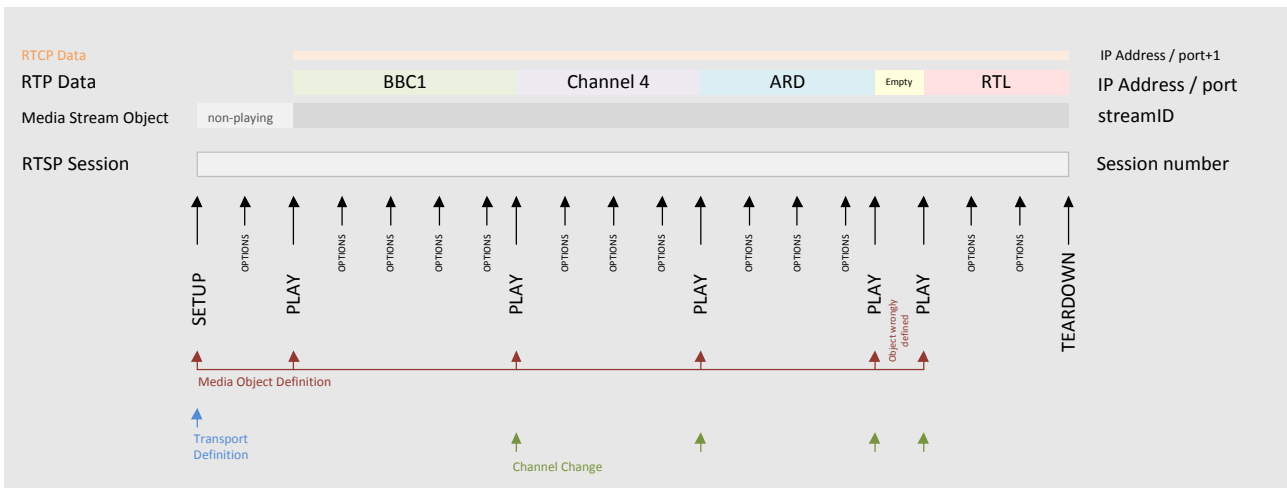
3.5.1 RTSP

SAT>IP Clients use RTSP over TCP to setup **RTSP sessions** with a SAT>IP server. An RTSP session starts with a an RTSP SETUP request and ends with an RTSP TEARDOWN message. A session is uniquely identified by a [session number](#) assigned by the server.

When setting up an RTSP session, clients define the transport mode which will be used for delivering the actual media stream. In the SETUP message they also define or start defining through a URI query the media stream object that they want to be delivered. SAT>IP media stream objects are identified through a [streamID](#). A media stream object is only modified through URI queries and no modification shall occur from any associated RTSP method used to invoke these queries.

Actual stream play-out is started in a session by invoking a PLAY message containing the streamID. During the course of the session, clients can change channels by invoking further PLAY messages with the URI query parameters corresponding to different requested channels.

In order to keep sessions with a server alive, clients need to issue regular RTSP messages within the timeout period (announced by the server in the original reply to the SETUP message). In SAT>IP RTSP OPTIONS messages shall be used as the default keep alive messages.



Clients and servers shall support RTSP version 1.0 as described by Appendix D of RFC 2326 [6].

RTSP is a text-based protocol and uses the ISO 10646 character set in UTF-8 encoding. Header field names are case-insensitive and header field values are case-sensitive in RTSP. Lines are terminated by CRLF.

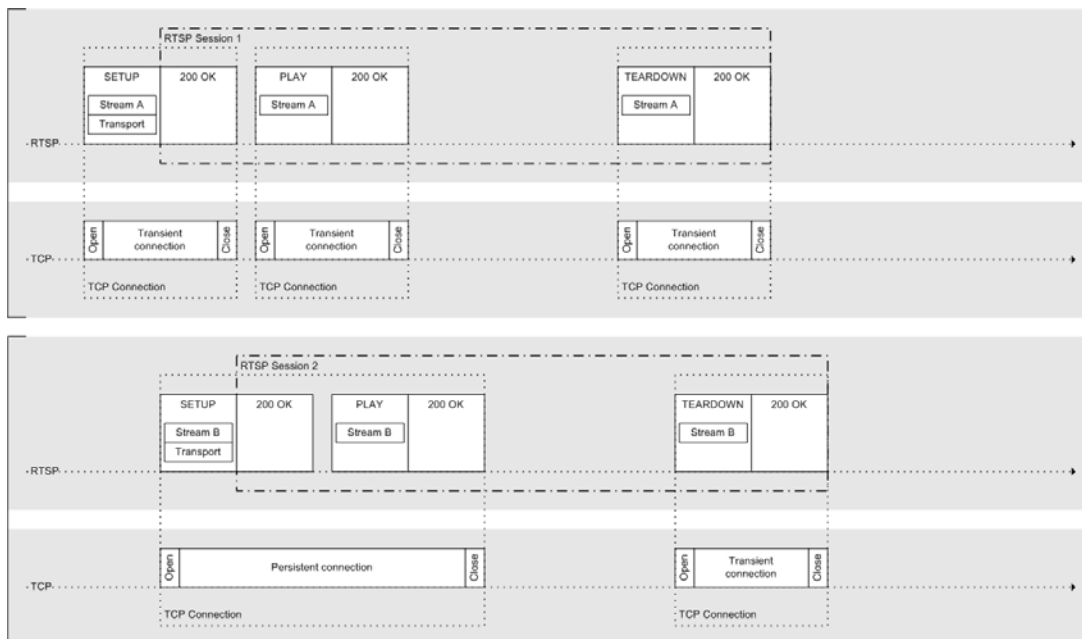
SAT>IP uses the standard RTSP server port number **554**.

RTSP and TCP

The RTSP control channel in SAT>IP must be implemented using TCP. It is important to note that RTSP sessions are independent of the underlying TCP connections.

An RTSP session generally spans multiple TCP connections (e.g. one TCP connection per request-response). It is however also possible for an RTSP session to consist of a single TCP connection (persistent mode) including several RTSP request-response pairs.

Example of a client with two RTSP sessions on the same SAT>IP server carried in different concurrent TCP connections:



In SAT>IP a single TCP connection does not carry more than one RTSP session

Please note that the server shall close a TCP connection:

- After all RTSP sessions being managed through the connection have timed out.
- 10 seconds after responding to a session level **TEARDOWN** request for the last RTSP session being controlled through this connection. This is to ensure that a client has time to issue a **SETUP** for a new session on this existing connection after having torn down the last one.

3.5.2 Setting up a new session (SETUP)

In order to setup a new session with a SAT>IP server, the client issues an RTSP SETUP request.

The SETUP message contains:

- the tuning and demultiplexing parameters of a TS media stream in an **RTSP URI** query string. This query string is carried in the request line (first line) of the message.
- the specification of the **Transport** delivery parameters (unicast/multicast, ports, etc.) in the header of the RTSP SETUP message.

In response and if the SETUP is successful the server allocates resources for the stream and returns a 200 OK message containing in its header:

- the new RTSP **session number**,
- the actual **Transport** delivery parameters such as IP destination address and ports,
- a **streamID** uniquely identifying the TS media stream.

Every RTSP session between a server and a client is identified by a randomly generated [session number](#).

The server communicates the actual IP unicast or multicast address on which the media stream will be delivered in the [Transport](#) header field.

The [streamID](#) in SAT>IP uniquely identifies a TS media stream object. The client should use the streamID in order to refer to a stream which has been previously setup. Every RTSP session contains a single TS media stream. The TS media stream itself may consist of various television/radio or data channels. Channel changes generally occur within the same session.

The client which sets up the session and defines the parameters of the media stream is the **owner** of the stream.

The creation of the media stream object and its streamID is not directly related to the actual tuning. The physical tuning may or may not intervene directly after the RTSP SETUP request message. Successful tuning is related to the availability of all the tuning parameters and the correct reception of the satellite signal.

Method		SETUP
RTSP_URI		rtsp://<ip_address>/?src=<srcID>&fe=<feID>&freq=<frequency>&... rtsp://<ip_address>/stream=<streamID>?src=<srcID>&... rtsp://<ip_address>/stream=<streamID>
Request	Headers	CSeq Session ⁽¹⁾ Transport
	Body	n/a
Response	Headers	CSeq Session ⁽²⁾ Transport com.ses.streamID
	Body	n/a
Status Codes		400, 403, 404, 414, 454, 461, 500, 503, 505, 551
Remarks		¹ This header field is only required if the client is in a session. ² The session timeout time value is specified by the "timeout" parameter in the "Session:" header field of the response. The default timeout value in RTSP is 60 seconds.

Example **Unicast Transport Delivery**

Request `SETUP rtsp://192.168.128.5/?src=1&fe=1&freq=12402&pol=v&msys=dvbs&sr=27500&fec=34&pids=0,16,50,104,166,1707 RTSP/1.0`
 CSeq: 1
 Transport: RTP/AVP;unicast;client_port=1400-1401
 <CRLF>

Response `RTSP/1.0 200 OK`
 CSeq: 1
 Session: 12345678;timeout=60
 Transport: RTP/AVP;unicast;client_port=1400-1401
 com.ses.streamID: 1
 <CRLF>

Example **Multicast Transport Delivery**

Request `SETUP rtsp://192.168.128.5/?src=1&fe=1&freq=12402&pol=v&msys=dvbs&sr=27500&fec=34&pids=0,16,50,104,166,1707 RTSP/1.0`
 CSeq: 1
 Transport: RTP/AVP;multicast;port=5004-5005
 <CRLF>

Response `RTSP/1.0 200 OK`
 CSeq: 1
 Session: 12345678;timeout=60
 Transport: RTP/AVP;multicast;destination=239.0.0.9;port=5004-5005;t1=5
 com.ses.streamID: 1
 <CRLF>

The **CSeq** header is carried in each RTSP request and response. This number is incremented by one for each distinct request transmitted. For every RTSP request containing the given sequence number there must be a corresponding response having the same number.

The **Session** header identifies an RTSP session started by the server in a **SETUP response** and concluded by a **TEARDOWN** message. The session identifier is chosen by the server. The session identifier is a string with a minimum length of 8 octets. Once a client receives a session identifier, it must return it for any request related to that session. The session identifier identifies the RTSP session across TCP transport sessions or connections. Please note that a given client can setup more than one session. When setting up a new session the Session header is not present in the request. However the Session header is present in the request if a client wants to modify an existing session (e.g. if the clients wants to change the Transport delivery parameters).

When the Session header is present in a response it may contain the **timeout** parameter after a semicolon. The server uses the timeout parameter to indicate to the client how long it is prepared to wait between RTSP commands before closing the session due to lack of activity. The timeout mechanism thus allows a server to detect the presence or absence of clients. In the absence of other messages to send, clients send RTSP **OPTIONS** methods as keep-alive signal to servers. The timeout is measured in seconds with a default value of 60 seconds.

If SAT>IP servers include the timeout parameter, the following recommendations apply:

SETUP unicast: the timeout value shall not be less than 30 seconds. The recommended value is **60** seconds.

SETUP multicast: the recommended timeout value is **0**. This special value corresponds to “**never**” meaning that the session will not automatically expire unless the client issues a **TEARDOWN** message. This mode of operation is useful in an environment where a single control terminal sets up streams for a large number of multicast clients. Other values are nevertheless possible. Clients carry the responsibility of maintaining sessions alive.

The **com.ses.streamID** header value uniquely identifies a TS media stream object. The streamID is a 16-bit numeric value. It is generated by the server and communicated to the client in the 200 OK response. The client should use the streamID in order to refer to streams which have been previously setup.

The [Transport](#) header field in the request indicates which transport protocol is to be used and specifies parameters such as destination address, ports, multicast time-to-live (ttl). Parameters are separated by a semicolon. Only a single transport mode is allowed (no preference list). The transport header may also be used to change certain parameters of an existing stream. The table below lists the supported variants:

Header Field	
Transport	<p>Specifies the transport of the requested stream. Only a single transport mode is allowed per SETUP request.</p> <p>RTP/AVP;unicast;client_port=<client RTP port>-<client RTCP port></p> <p>RTP/AVP;multicast;destination=<IP multicast address>;port=<RTP port>-<RTCP port>;ttl=<ttl></p> <p>RTP/AVP;multicast RTP/AVP;multicast;ttl=<ttl> RTP/AVP;multicast;port=<RTP port>-<RTCP port> RTP/AVP;multicast;port=<RTP port>-<RTCP port>;ttl=<ttl> RTP/AVP;multicast;destination=<IP multicast address> RTP/AVP;multicast;destination=<IP multicast address>;ttl=<ttl> RTP/AVP;multicast;destination=<IP multicast address>;port=<RTP port>-<RTCP port> RTP/AVP;multicast;destination=<IP multicast address>;port=<RTP port>-<RTCP port>;ttl=<ttl></p>

SAT>IP servers in the response to a SETUP request from a client automatically fill in those transport configuration parameters which are not specified in the request. The response always contains all the information necessary by the client in order to listen to the stream.

The configuration parameters associated with transport are:

RTP/AVP	indicates the stream delivery format. In SAT>IP only RTP/AVP is supported.
unicast multicast	is a mutually exclusive indication of whether unicast or multicast delivery is requested.
destination	indicates the address to which the stream will be sent. Only available for multicast.
client_port	provides the unicast RTP/RTCP port pair on which the client has chosen to receive the media stream and control information.
port	provides the RTP/RTCP port pair for a multicast session. It specifies a range e.g. 3456-3457.
ttl	multicast specific: indicates the time-to-live.

Please note that:

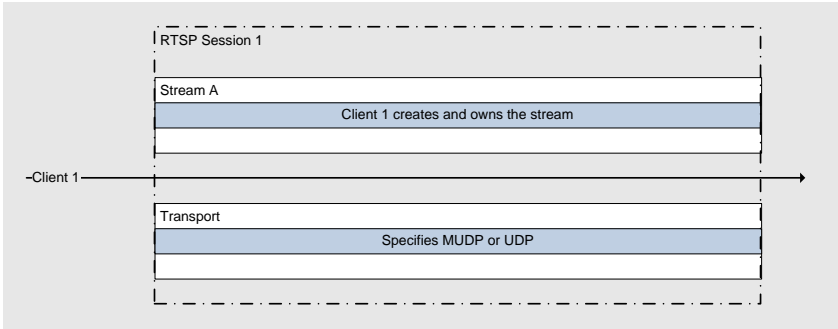
- an RTP delivered media stream is originated and received on an even port number and the associated RTCP announcement stream uses the next higher odd port number.
- the port names differ between UDP unicast transport and UDP multicast transport:

For unicast:	client_port= <client RTP port> - <client RTCP port> server_port= <server RTP port> - <server RTCP port>
For multicast:	port= <RTP port> - <RTCP port>

Joining an existing stream without becoming stream owner

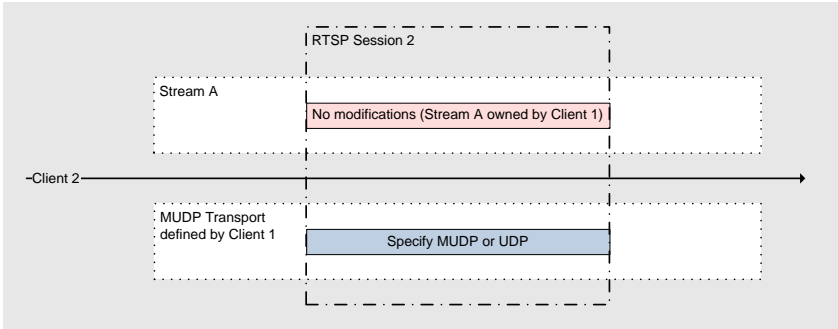
In SAT>IP a given media stream may be used in more than one RTSP session e.g. if a second client joins an existing media stream. Clients in these other sessions do not have the ownership of the media stream (see below).

Example of a Client 1 setting up a new session with the definition of a TS media stream and its associated IP transport (unicast or multicast). Client 1 becomes the stream owner.

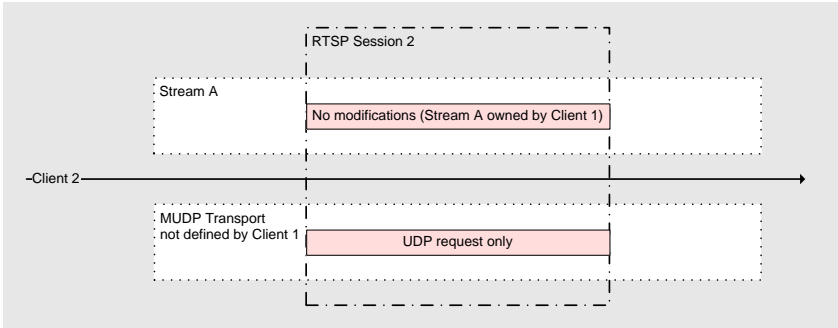


Client 2 may join the existing media stream, however cannot modify the stream itself.

If the original transport was multicast, Client 2 may join the original multicast or request an additional unicast transport for itself.



If the original transport was unicast, Client 2 may only request an additional unicast (not multicast) for itself.



Example RTSP SETUP messages

Standard SETUP of a unicast stream

Request>

```
SETUP rtsp://192.168.178.21:554/?src=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000&fec=56
&pids=0,400,401,402 RTSP/1.0
CSeq:1
Transport: RTP/AVP;unicast;client_port=11992-11993
Connection:close
```

Response>

```
RTSP/1.0 200 OK
CSeq: 1
Transport: RTP/AVP;unicast;client_port=11992-11993
Session: E9C18AA1
com.ses.streamID: 1
```

SETUP of a unicast stream with request for a particular frontend

Request>

```
SETUP rtsp://192.168.178.57:554/?src=1&fe=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,400,401,402 RTSP/1.0
CSeq:1
Transport: RTP/AVP;unicast;client_port=8220-8221
Connection: close
```

Response>

```
RTSP/1.0 200 OK
Session:2166e663b4be550;timeout=30
com.ses.streamID:2
Transport:RTP/AVP;unicast;client_port=8220-8221
CSeq:1
```

SETUP of a multicast stream without specifying the multicast address and ports

Request>

```
SETUP rtsp://192.168.178.57:554/?src=1&fe=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,400,401,402 RTSP/1.0
CSeq:1
Transport: RTP/AVP;multicast
Connection:close
```

Response>

```
RTSP/1.0 200 OK
Session:21dbf7c5873c9ff;timeout=30
com.ses.streamID:8
Transport:RTP/AVP;multicast;destination=239.0.0.3;port=1500-1501;ttl=2
CSeq:1
```

SETUP of a multicast stream specifying the multicast address, ports and ttl

Request>

```
SETUP rtsp://192.168.178.57:554/?src=1&fe=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,400,401,402 RTSP/1.0
CSeq:1
Transport: RTP/AVP;multicast;destination=224.16.16.1;port=42128-42129;ttl=5
Connection:close
```

Response>

```
RTSP/1.0 200 OK
Session:220b5b82dc21781;timeout=30
com.ses.streamID:14
Transport:RTP/AVP;multicast;destination=224.16.16.1;port=42128-42129;ttl=5
CSeq:1
```

SETUP used in an existing session to modify the transport parameters

Request>

SETUP rtsp://192.168.178.57:554/?src=1&fe=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off
&sr=22000&fec=56&pids=0,400,401,402 RTSP/1.0
CSeq:1
Transport: RTP/AVP;multicast
Connection:close

Response>

RTSP/1.0 200 OK
Session:222626d35aa05f4;timeout=30
com.ses.streamID:17
Transport:RTP/AVP;multicast;destination=239.0.0.8;port=1500-1501;tll=2
CSeq:1

Request>

SETUP rtsp://192.168.178.57:554/stream=17?src=1&fe=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,400,401,402 RTSP/1.0
CSeq:2
Session:222626d35aa05f4
Transport: RTP/AVP;multicast;destination=224.16.16.1;port=11722-11723;tll=11
Connection:close

Response>

RTSP/1.0 200 OK
Session:222626d35aa05f4;timeout=30
com.ses.streamID:17
Transport:RTP/AVP;multicast;destination=224.16.16.1;port=11722-11723;tll=11
CSeq:2

3.5.3 Starting the playout of a media stream (PLAY)

Once a session has been established with a server through a successful SETUP request, a client can start the actual play-out / delivery / data transmission of a TS media stream.

The **unicast** data transmission is started by sending:

- an RTSP PLAY message with the required streamID and an optional URI query string.

The **multicast** data transmission can be started either:

- by sending an RTSP PLAY message with the required streamID (and optional URI query string),
- or by sending an IGMPv3 membership report [7] to the multicast group.

The PLAY message starts the delivery of the actual RTP and RTCP data streams into the session. From that point onwards RTP and RTCP packets will be delivered to the destination IP address.

The content of the delivered data depends on the actual tuning process. If the media stream object was fully and correctly defined in the SETUP and PLAY process, the requested satellite data will be available. If however no signal is available, the signal is lost or the media object was not correctly defined (e.g. no PIDs available), the server will nevertheless deliver an empty RTP packet (not containing a TS packet) every 100 ms. The RTCP data stream will also be available in parallel and indicate the data and signal characteristics.

Method		PLAY
RTSP_URI		rtsp://<ip_address>/stream=<streamID> rtsp://<ip_address>/stream=<streamID>?src=<srcID>&...
Request	Headers	CSeq Session
	Body	n/a
Response	Headers	CSeq Session RTP-Info
	Body	n/a
	Status Codes	400, 403, 404, 408, 453, 454, 500, 505, 551
Example	Request	PLAY rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 2 Session: 12345678 <CRLF>
	Response	RTSP/1.0 200 OK CSeq: 2 Session: 12345678 RTP-Info: url=rtsp://192.168.128.5/stream=1 <CRLF>

The **RTP-Info** header is present in the 200 OK response to a PLAY message. It indicates the stream URL.

Please note that the PLAY message cannot be invoked outside of an RTSP session.

The PLAY message is also commonly used to modify the media stream URI e.g. when changing channels.

Example PLAY message

Request>

PLAY rtsp://192.168.178.57:554/stream=2 RTSP/1.0
CSeq:2
Session:2166e663b4be550
Connection: close

Response>

RTSP/1.0 200 OK
RTP-Info:url=rtsp://192.168.178.57/stream=2
CSeq:2
Session:2166e663b4be550

Example PLAY message with full stream definition (e.g. channel change)

Request>

PLAY rtsp://192.168.178.57:554/stream=5?src=1&freq=11538&pol=v&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,611,621,631 RTSP/1.0
CSeq:5
Session:21a15c02c1ee244
Connection:close

Response>

RTSP/1.0 200 OK
RTP-Info:url=rtsp://192.168.178.57/stream=5
CSeq:5
Session:21a15c02c1ee244

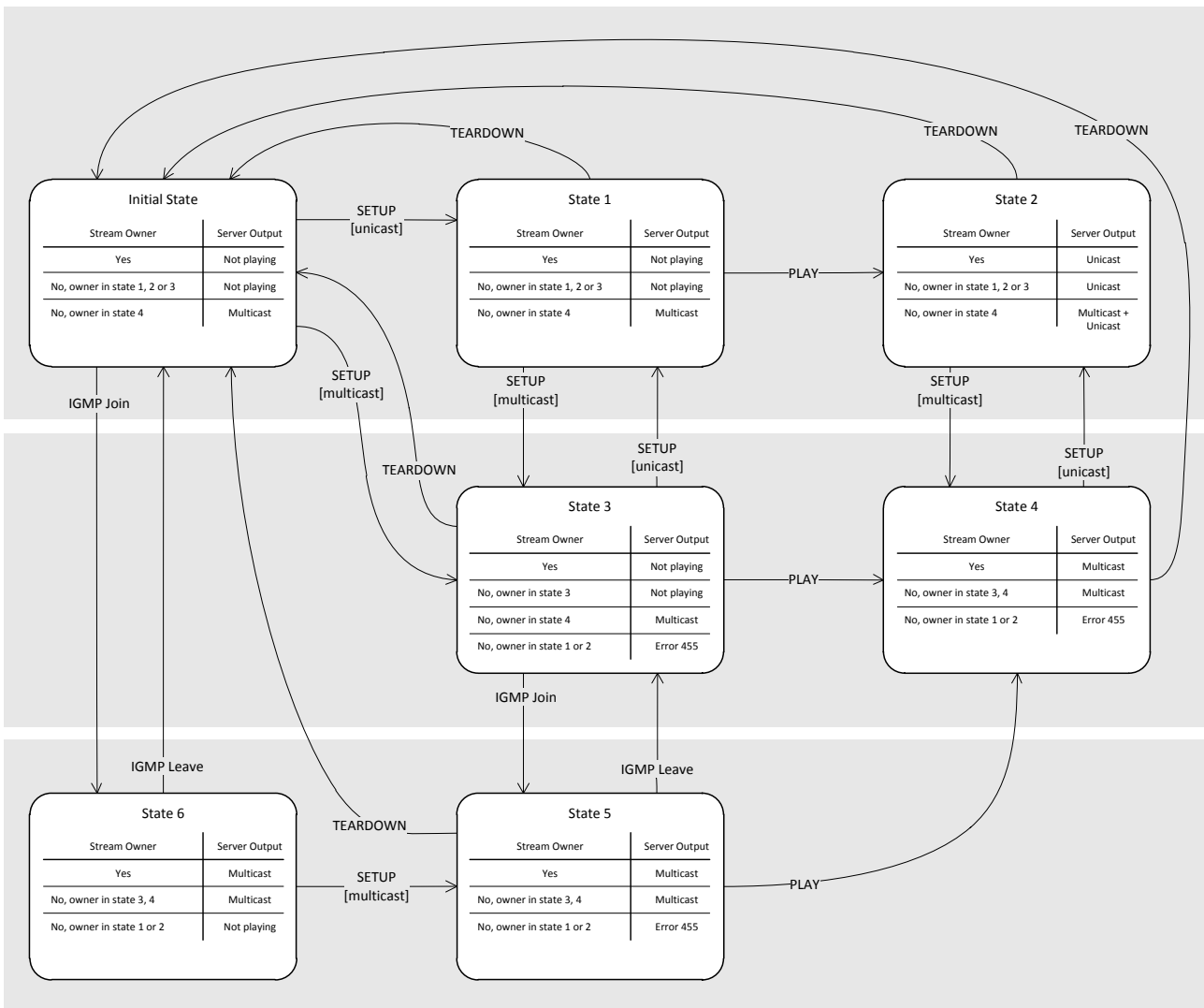
Server Stream Output

The SAT>IP server output depends on the server state (resulting from the sequence of RTSP and/or IGMP methods invoked by a client) and the state of the media stream owner.

If the client has the ownership of a media stream, the server output is given in line 1.

If the client does not have the ownership of the media stream, the server output also depends on the actual state of the media stream owner (lines 2, 3 or 4).

Server Stream Output State Machine



Any RTSP/IGMP method not shown in the diagram above will not modify the server output state.

3.5.4 Maintaining a session (OPTIONS)

Once an RTSP session has been setup, the client needs to maintain the RTSP session by sending an RTSP request before the end of each timeout period (60 seconds by default). A SAT>IP server may define a different timeout period in the Session Header of the response to a SETUP message.

Through the receipt of OPTIONS messages from clients, the server knows about the presence or absence of each RTSP client.

Clients need to send an OPTIONS message at the latest a few seconds before each expiration period. As OPTIONS messages are delivered using reliable TCP transport, there is no need to send excess OPTIONS messages.

The OPTIONS method may also be invoked out-of-session by clients to query servers on supported RTSP methods.

Method		OPTIONS
RSTP_URI		rtsp://<ip_address>/
Request	Headers	CSeq Session ⁽¹⁾
	Body	n/a
Response	Headers	CSeq Session ⁽¹⁾ Public
	Body	n/a
	Status Codes	400, 403, 454, 500, 505, 551
Remarks		This method may be issued at any time within or outside a session. ¹ This header field is only required if the client is in a session (e.g. for a keep-alive).
Example	Request	OPTIONS rtsp://192.168.128.5/ RTSP/1.0 CSeq: 4 <CRLF>
	Response	RTSP/1.0 200 OK CSeq: 4 Public: OPTIONS, DESCRIBE, SETUP, PLAY, TEARDOWN <CRLF>

The [Session](#) header is required when the OPTIONS request is used for keeping a session alive.

The [Public](#) header in the response lists the methods supported by the server.

If no OPTIONS message is received by the server within the timeout period, the session will be closed, and the consequences will be identical to a session TEARDOWN. The stream playout will immediately be stopped by the server.

Example OPTIONS message

```

Request>
OPTIONS rtsp://192.168.178.57:554/ RTSP/1.0
CSeq:5
Session:2180f601c42957d
Connection:close

Response>
RTSP/1.0 200 OK
Public:OPTIONS,SETUP,PLAY,TEARDOWN,DESCRIBE
CSeq:5
Session:2180f601c42957d

```

3.5.5 *Modifying a media stream*

A TS media stream may only be modified by its owner. The owner generally invokes the RTSP PLAY method inside the session for modifying the media stream. Under certain circumstances, e.g. if the owner wants to modify the session transport, an in-session SETUP message may also be used.

The parameters of the new TS media stream are passed to the server in the RTSP URI query string.

A change in parameters shall not interrupt the playing stream and shall not lead to any packet loss within the stream.

Example of a stream modification

Request>

```
PLAY rtsp://192.168.178.57:554/stream=5?src=1&fe=1&freq=12603&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000
&fec=56&pids=0,1290,2290,6290,7290 RTSP/1.0
CSeq:8
Session:21a15c02c1ee244
Connection:close
```

Response>

```
RTSP/1.0 200 OK
RTP-Info:url=rtsp://192.168.178.57/stream=5;seq=22857
CSeq:8
Session:21a15c02c1ee244
```

3.5.6 *Joining an existing stream*

Clients may join an already defined stream (existing streamID) owned by another client:

- by setting up a new session themselves (invoking SETUP and PLAY on the streamID) or
- by purely joining an existing stream through an IGMPv3 membership report.

The joining client will **not** get the ownership of the media stream. The joining client cannot modify the stream itself. The joining client may however request a unicast transport for itself if the stream had been originally requested by the stream owner as a multicast stream. Conversely it may not request a multicast stream if the stream had been SETUP as unicast by the stream owner.

Clients joining an existing stream, risk losing access to their stream if the owner of the media stream does a TEARDOWN. Therefore this mainly applies to managed scenarios where streams have been statically joined and are therefore always present (e.g. certain multicast distribution systems).

Example SETUP of a client joining an existing stream for which it will not have ownership

Request>

```
SETUP rtsp://192.168.128.192:554/stream=27 RTSP/1.0
CSeq:1
Transport: RTP/AVP;unicast;client_port=42494-42495
Connection:close
```

Response>

```
RTSP/1.0 200 OK
Session:43ae7c353e77bc;timeout=60
com.ses.streamID:27
Transport:RTP/AVP;unicast;destination=192.168.128.100;client_port=42494-42495
CSeq:1
```

3.5.7 Listing available media streams (DESCRIBE)

SAT>IP provides the possibility for clients to enquire servers about the TS media streams currently configured. A client can invoke the RTSP DESCRIBE request anytime (in or out of an RTSP session) to enquire about one particular media stream or all existing media streams configured on the SAT>IP server.

The RTSP DESCRIBE request may be done on the IP address of the SAT>IP server. In this case the SDP describes all TS streams available from the server.

The RTSP DESCRIBE request may also be done on a particular streamID. In this case the SDP describes only that particular stream.

In response to the DESCRIBE request and under the condition that at least one RTSP session has been setup, SAT>IP servers provide an SDP (Session Description Protocol) [10] formatted description file. The SDP implementation is based on RFC 4566 [10]. The description file provides details about the configured media streams and their reception characteristics. The SDP file consists of a session level section followed by one or more media level sections describing each a configured TS media stream.

If no session has been previously setup on the SAT>IP server (and thus no stream has been created), the response to the DESCRIBE request will be error message 404 (stream not found).

Method		DESCRIBE
RTSP_URI		rtsp://<ip_address>/ rtsp://<ip_address>/stream=<streamID>
Request	Headers	CSeq Session ⁽¹⁾ Accept
	Body	n/a
Response	Headers	CSeq Session ⁽¹⁾ Content-Type: application/sdp Content-Base Content-Length
	Body	application/sdp
	Status Codes	400, 404, 406, 500, 505, 551
Remarks		This method may be issued at any time. ¹ This header field is only required if the client is in a session.

The [Accept](#) header in the request specifies the content type acceptable for the response. Its value shall be set to: application/sdp

The [Content-Type](#) header in the response shall be correspondingly set to: application/sdp

The [Content-Base](#) header in the response carries the base url of the server

The [Content-Length](#) header contains the length of the content of the method (i.e. after the double CRLF following the last header)

The example below shows an SDP listing all the streams currently available from the SAT>IP server. The top part of the SDP (shown in red) provides session level information. It consists of the v,o,s and t attributes.

The sections below (blue and grey) contain information about the media streams configured. Each section corresponds to a media stream and has an identical structure m,c,a,a,a.

Please note that if a client joins an existing TS media stream via RTSP or IGMP, it shall not create a new media section in the SDP. Only the definition of a TS media stream (identified by a streamID) leads to the creation of a media section in the SDP.

Example	Request	DESCRIBE rtsp://192.168.128.5/ CSeq: 5 Accept: application/sdp <CRLF>
	Response	RTSP/1.0 200 OK CSeq: 5 Content-Type: application/sdp Content-Base: rtsp://192.168.128.5/ Content-Length: 724 <CRLF> v=0 o=- 5678901234 7890123456 IN IP4 192.168.128.5 s=SatIPServer:1 4 t=0 0 m=video 5004 RTP/AVP 33 c=IN IP4 239.0.0.8/5 a=control:stream=0 a=fmtp:33 ver=1.0;src=1;tuner=1,240,1,7,12402,v,dvbs,,,27500,34;pids=0,16,56,112,168,1709 a=inactive m=video 5006 RTP/AVP 33 c=IN IP4 239.0.0.9/5 a=control:stream=1 a=fmtp:33 ver=1.0;src=1;tuner=1,240,1,7,12402,v,dvbs,,,27500,34;pids=0,16,50,104,166,1707 a=sendonly m=video 0 RTP/AVP 33 c=IN IP4 0.0.0.0 a=control:stream=2 a=fmtp:33 ver=1.0;src=1;tuner=1,240,1,7,12402,v,dvbs,,,27500,34;pids=all a=sendonly m=video 5010 RTP/AVP 33 c=IN IP4 239.0.0.11/5 a=control:stream=3 a=fmtp:33 ver=1.0;src=2;tuner=2,221,1,6,11758,h,dvbs2,8psk,off,25,27500,56;pids=all a=sendonly

“v=” gives the version of the Session Description Protocol. This shall be set to 0.

“o=” gives the originator of the session “-“ <session-id> <session-version> IN for internet and the IP address of the server. The session-id and session-version values shall be present but are not used currently.

“m=” signals the start of each media-level section. “m=” provides information about the media type, port, protocol: RTP/AVP, media format description: 33 for SAT>IP. For unicast streams the port should be set to 0.

“c=” contains information about the connection. For multicast streams the IP multicast address is signalled followed by the ttl value. For unicast streams the IP address is set to 0.0.0.0.

TS media streams in non-playing mode are tagged with the a=inactive attribute. Media streams in playing mode are tagged with a=sendonly.

The RFC defined attributes are complemented with the following SAT>IP specific attributes:

<i>Session Level:</i> s=SatIPServer:1 <frontends> <i>Media level:</i> a=control:stream=<streamID> a=fmtp:33 ver=<major>.<minor>;src=<srcID>;tuner=<feID>,<level>,<lock>,<quality>,<frequency>,<pol arisation>,<system>,<type>,<pilots>,<roll_off>,<symbol_rate>,<fec_inner>;pids=<pid0>,...,<pidn>

Most label fields have been specified already as part of the query syntax. Additional fields are specified below:

Name	Value		Example
	Name	Range	
No. of frontends	frontends	Contains the total number of frontends available on the server	4
Major version	major	Contains the major version number of the fntp string syntax	1.0
Minor version	minor	Contains the minor version number of the fntp string syntax	
Signal level	level	Numerical value between 0 and 255 An incoming L-band satellite signal of -25dBm corresponds to 224 -65dBm corresponds to 32 No signal corresponds to 0	240
Frontend lock	lock	Set to one of the following values: "0" the frontend is not locked "1" the frontend is locked	1
Signal quality	quality	Numerical value between 0 and 15 Lowest value corresponds to highest error rate The value 15 shall correspond to -a BER lower than 2×10^{-4} after Viterbi for DVB-S -a PER lower than 10^{-7} for DVB-S2	7

Example DESCRIBE Message

Request>

```
DESCRIBE rtsp://192.168.178.57:554/ RTSP/1.0
CSeq:7
Accept: application/sdp
Connection:close
```

Response>

```
RTSP/1.0 200 OK
Content-Length:256
Content-Type:application/sdp
Content-Base:rtsp://192.168.178.57/
CSeq:7
```

```
v=0
o=- 35781181 35781181 IN IP4 192.168.178.57
s=SatIPServer:1 4
t=0 0
m=video 11784 RTP/AVP 33
c=IN IP4 239.0.0.7/1
a=control:stream=16
a=fntp:33 ver=1.0;src=1;tuner=1,123,1,4,10744.00,h,dvbs,qpsk,off,0.35,22000,56;pids=0,400,401,402
a=sendonly
```

3.5.8 Closing the session and stopping the playout (TEARDOWN)

In order to close an RTSP session and stop the playout of a **unicast** or a **multicast** TS media stream the client issues:

- an RTSP TEARDOWN message.

The TEARDOWN message always stops the playout of a **unicast** stream, whether the TEARDOWN message is issued by the stream owner or the non-owner. (Each unicast stream is of course unique to each client).

The server stops the **multicast** TS media stream only when receiving the RTSP session TEARDOWN message of the stream owner. A server has to be able to handle the concurrent access to TS media streams from multiple clients.

Please note that an out-of-session (non-stream-owning) multicast client may leave a stream by only issuing an IGMP leave message.

The TEARDOWN method is used to terminate the RTSP session.

Method		TEARDOWN
RTSP_URI		rtsp://<ip_address>/stream=<streamID>
Request	Headers	CSeq Session
	Body	n/a
Response	Headers	CSeq Session
	Body	n/a
	Status Codes	400, 403, 404, 408, 454, 500, 505, 551
Example	Request	TEARDOWN rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 3 Session: 12345678 <CRLF>
	Response	RTSP/1.0 200 OK CSeq: 3 Session: 12345678 <CRLF>

Example TEARDOWN message

Request>
TEARDOWN rtsp://192.168.178.57:554/stream=2 RTSP/1.0
CSeq:5
Session:2166e663b4be550
Connection: close

Response>
RTSP/1.0 200 OK
Content-Length:0
CSeq:5
Session:2166e663b4be550

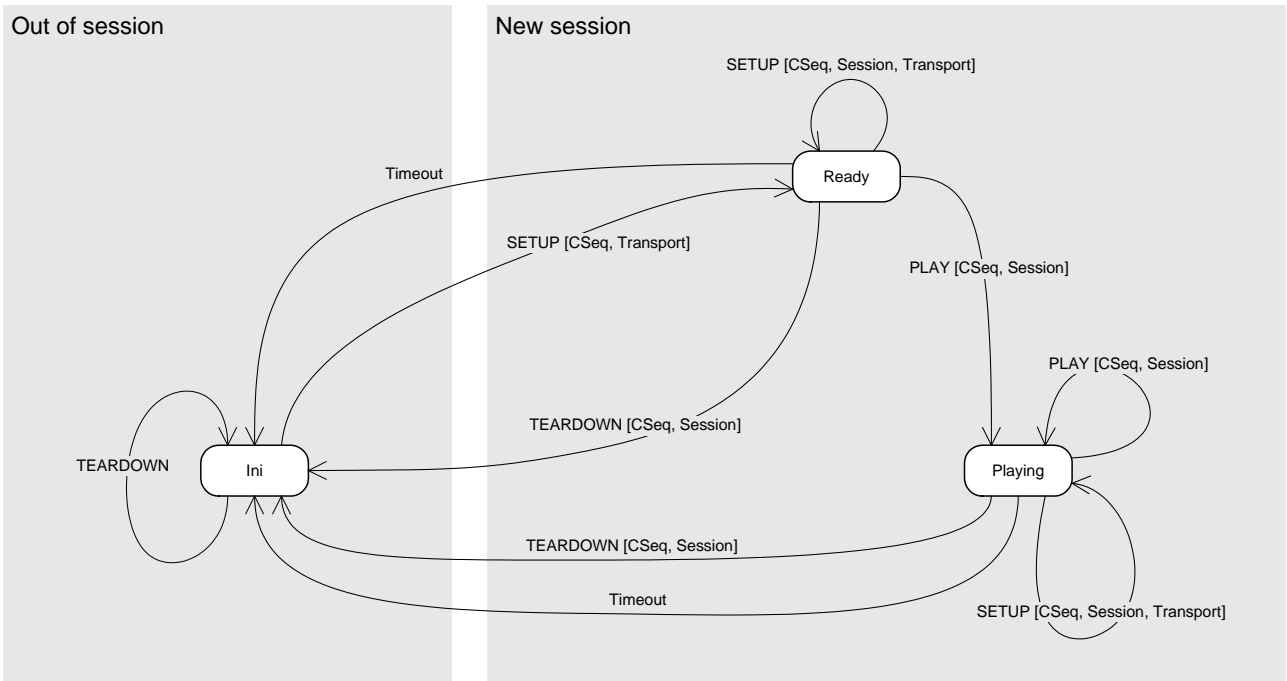
3.5.9 RTSP Method Summary

SAT>IP clients and servers shall support the following RTSP methods:

SETUP, PLAY, TEARDOWN, OPTIONS, DESCRIBE.

Methods	RTSP URI	Usage
SETUP	rtsp://<ip_address>/?src=<srcID>&fe=<feID>&freq=<frequency>&...	Setting up a new session and media stream
	rtsp://<ip_address>/stream=<streamID>	Joining an existing media stream
	rtsp://<ip_address>/stream=<streamID>?src=<srcID>&...	Modifying a media stream
PLAY	rtsp://<ip_address>/stream=<streamID>	Playing a media stream
	rtsp://<ip_address>/stream=<streamID>?src=<srcID>&...	Playing and modifying a media stream
TEARDOWN	rtsp://<ip_address>/stream=<streamID>	Closing the session
OPTIONS	rtsp://<ip_address>/	Maintaining a session alive
DESCRIBE	rtsp://<ip_address>/	Describing all the media streams configured
	rtsp://<ip_address>/stream=<streamID>	Describing a particular media stream

RTSP State machine



3.5.9.1 RTSP Headers in Request and Response Messages

Header	Message	Methods					
		SETUP	PLAY	TEARDOWN	OPTIONS	DESCRIBE	NOT IMPLEMENTED
CSeq	Request	req.	req.	req.	req.	req.	req.
	Response	all	all	all	all	all	501
Session	Request	opt. ⁽¹⁾	req.	req.	opt. ⁽²⁾	opt. ⁽³⁾	
	Response	200	200	200	200 ⁽²⁾	200 ⁽³⁾	
Transport	Request	req.					
	Response	200					
com.ses.streamID	Response	200					
RTP-Info	Response		200				
Accept	Request					req.	
Content-Base	Response					200	
Content-Type ⁽⁴⁾	Response	400, 403, 503	400, 403, 503	400, 403		200	
Content-Length	Response	400, 403, 503	400, 403, 503	400, 403		200	
Public	Response				200		501
Allow	Response	405	405	405			
Require	Request	opt.	opt.	opt.	opt.	opt.	
Unsupported	Response	551	551	551	551	551	
		req.	required header	opt.	optional header	all	apply to all responses

¹ A SETUP request that references an existing RTSP session shall include the "Session:" header field with the session value of that existing session.

² When using the OPTIONS request as a session keep alive mechanism, the request shall include a "Session:" header field with the session value of the session that is being kept alive. The response carries also the "Session:" header field with the same session value.

³ A DESCRIBE request that references an existing RTSP session shall include the "Session:" header field with the session value of that existing session. The response carries also the "Session:" header field with the same session value.

⁴ The Content-Type field takes the value:

- "application/sdp" in response to a successful DESCRIBE request containing an SDP description,
- "text/parameters" when the message body of the 400 response contains the "Check-Syntax:" parameter followed by the malformed syntax, when the message body of the 403 response contains the "Out-of-Range:" parameter followed by a space separated list of the attributes that are not understood or when the message body of the 503 response contains the "No-More:" parameter followed by a space separated list of the missing resources.

Please note that linear white spaces between header field names and header field values shall be ignored.

3.5.10 Uniform Resource Identifier URI

The RTSP request consists of an RTSP method followed by an RTSP URI.

```
PLAY rtsp://<ip_address>:554/?src=1&freq=10744&pol=h&sr=22000&fec=56&pids=0,400,401,402
```



The URI is composed of the IP address followed by a query or a stream identifier (if a stream has already been previously defined) or both (if an existing stream is modified).

RTSP_URI = "rtsp://" ip_address:port "/" ["stream="streamID] ["?"query]

3.5.11 Query Syntax

The query specifies the signal source selection (i.e. a given satellite position) followed by the physical tuning and demultiplexing parameters of a TS media stream. Queries are composed of a series of attribute value pairs separated by the "&" (ampersand) sign. The attribute value pairs are each separated by the "=" (equal) sign: **<attribute1>=<value1>&<attribute2>=<value2>&<attribute3>=<value3>...**

Query attribute value pairs may appear in any order. The parser on the SAT>IP server shall be sufficiently robust to correctly delineate values and identify the different parameters. Queries from SAT>IP clients need to contain all the mandatory parameters. Only complete queries allow a perfect interoperability between all clients and servers. Unknown attributes shall be ignored by the server. This allows future revisions of the SAT>IP specification to be extended without interfering with today's implementations.

Name	Attribute	Value	Example
		Name	Range
Frontend identifier	fe	feID	Numerical value between 1 and 65535. <i>Not required in normal client queries</i>
Signal source	src	srcID	Numerical value between 1 and 255. Default value is "1".
Frequency	freq	frequency	Transponder frequency expressed in MHz as fixed point type or integer value.
Polarisation	pol	polarisation	Set to one of the following values: "h" horizontal linear, "v" vertical linear, "l" circular left, "r" circular right.
Roll-Off	ro	roll_off	Set to one of the following values: "0.35", "0.25", "0.20". <i>For DVB-S this value shall be set to "0.35" in client queries</i>
Modulation system	msys	system	Set to one of the following values: "dvbs", "dvbs2".
Modulation type	mttype	type	Set to one of the following values: "qpsk", "8psk". <i>For DVB-S this value shall be set to "qpsk" in client queries</i>
Pilot tones	plts	pilots	Set to one of the following values: "on", "off". <i>For DVB-S this value shall be set to "off" in client queries</i>
Symbol rate	sr	symbol_rate	Value in kSymB/s.
FEC inner	fec	fec_inner	Set to one of the following values: "12", "23", "34", "56", "78", "89", "35", "45", "910".
List of PIDs	pids	CSV list of PIDs: (Num. values betw. 0 and 8191) for spts streams, "all" for mpts streams, "none" for no demux output.	
	addpids ⁽¹⁾	Opens new PID filters on the demux for streaming on the network. CSV list of PIDs.	
	delpids ⁽¹⁾	Removes PID filters from the demux. CSV list of PIDs.	

¹ The "addpids" or "delpids" parameters shall not be used in combination with the "pids" parameter in the same RTSP query. The "addpids" and "delpids" parameters may be combined in the same RTSP query.

Frontend Selection

SAT>IP servers generally feature multiple physical frontends (DVB-S/S2 tuners and demodulators). SAT>IP servers automatically assign these physical frontends to client requests as they come in. In normal operation SAT>IP clients should not specify a particular frontend in their requests.

In certain scenarios however (e.g. in fully managed systems) it maybe useful for a client to request access to a particular frontend. This allows the system to be configured in a static way which always guarantees access to specific physical tuning resources. For this purpose SAT>IP clients may add the “fe” attribute in their request.

Signal Source Selection

Clients need to specify the source (satellite position) from which they intend to receive their signals. The source parameter communicates to the frontend the selection of a satellite orbital position.

Different antenna installations feature different numbers of sources available. The components making up the antenna installation are generally not aware as to which satellite they have been pointed to and which position they are capable of receiving. SAT>IP was designed to operate in a way similar to today's installations.

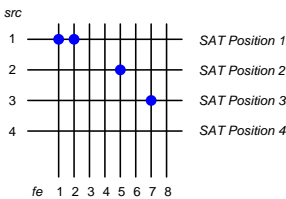
When a SAT>IP client is configured initially, it is told which satellite positions are available (in the local installation menu) and to which src parameters these have been connected. On the server side there will be a fixed or configurable mapping:



Multiswitch servers implementing DiSEqC shall always map the source parameter as follows:

“src”	DiSEqC
1	DiSEqC A = Position A Option A
2	DiSEqC B = Position B Option A
3	DiSEqC C = Position A Option B
4	DiSEqC D = Position B Option B

Every source may feed one or more frontends. The server device therefore needs to implement an internal switching matrix. Example showing the source and frontend selection matrix:



Physical Tuning

Standard DVB-S/S2 tuning parameters are passed to the SAT>IP server frontend in order to tune to a given transponder. For DVB-S and DVB-S2 the mandatory parameters in client requests are: “src”, “freq”, “pol”, “ro”, “msys”, “mtype”, “plts”, “sr” and “fec”.

Demultiplexing

The demux is informed about all the PIDs to be filtered through the “pids” comma separated values (CSV) list. Filtered PIDs may be changed, added or removed during normal operation via additional “pids”, “addpids” or “delpids” queries.

The PIDs filtering process is not started when creating a new TS media stream without any “pids” attribute in the query.

When the “pids=” statement is issued it replaces/erases any previously configured pids list. The “addpids” and “delpids” statements are used to add or remove pids from the list.

Please note that the “addpids” or “delpids” parameters shall not be used in combination with the “pids” parameter in the same RTSP query. The “addpids” and “delpids” parameters may be combined though in the same RTSP query.

Dynamic addition or removal of PIDs shall not interrupt continuous playback from the in-play demux and shall not lead to packet loss for any remaining PID values.

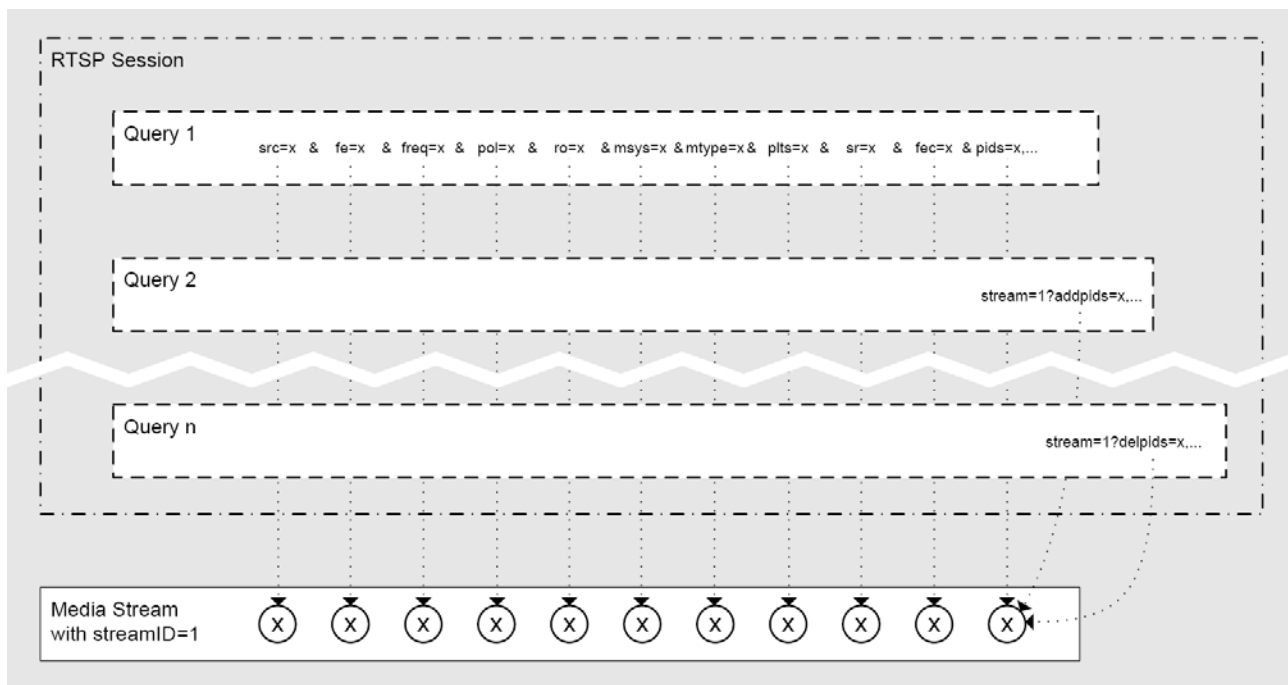
An in-session SETUP without a query in the RTSP URI shall not modify the list of PIDs to be filtered.

Clients are responsible for ignoring PIDs that they may have no interest in and which are present in the TS media stream.

The client should issue a “pids=” statement at least when changing transponders in order to clear the configured pid list and avoid that hanging pids are carried through without ever being removed.

When PIDs values have been defined, it is always possible to remove all PIDs from the filtering process through the “pids=none” attribute-value pair.

A media stream may be defined by a single RTSP query listing all the required parameters, or by a succession of RTSP methods with the succession of queries leading to its full definition.



URI Examples:

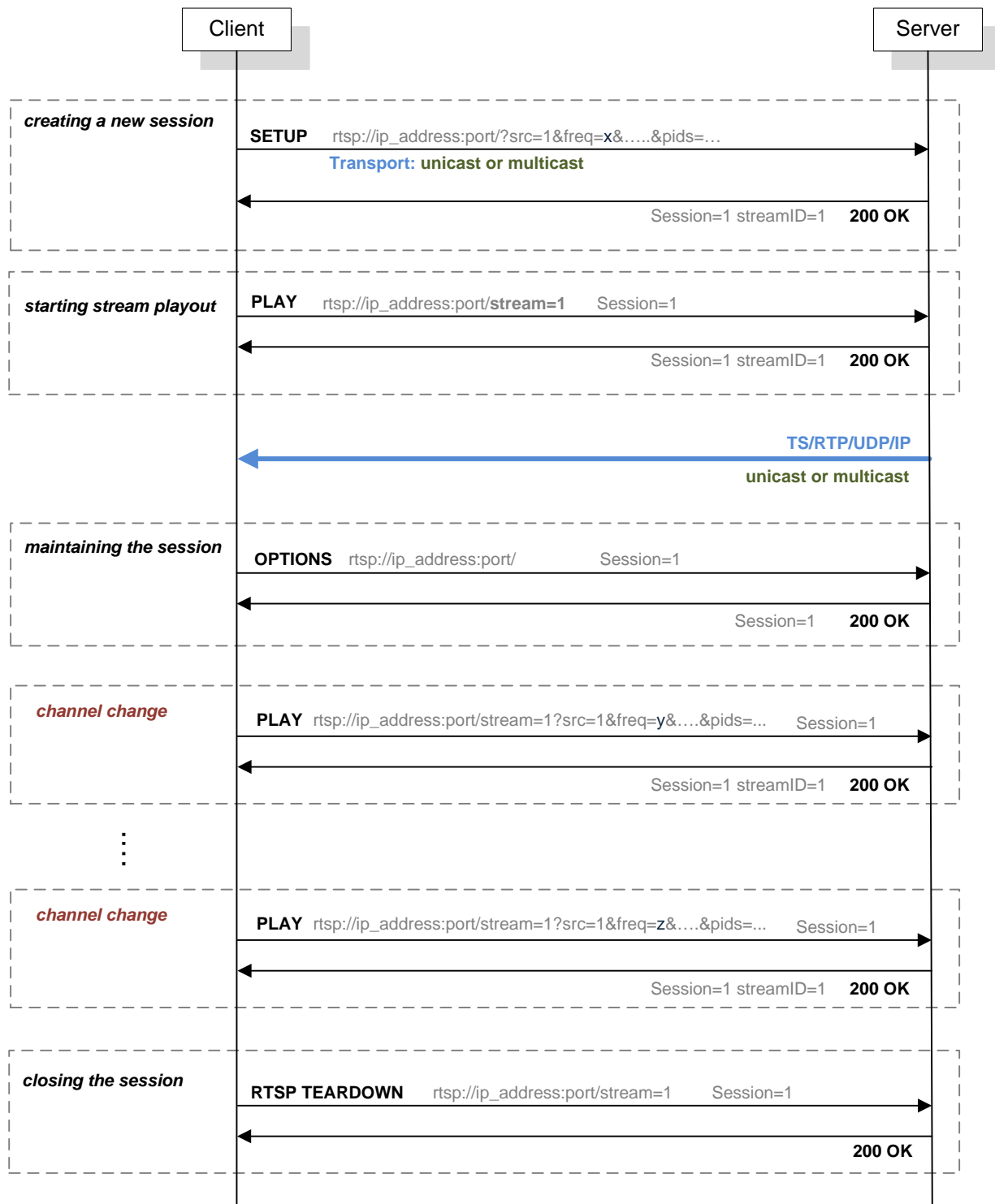
```
rtsp://192.168.128.1/?src=1&freq=12402&pol=v&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=27500
&fec=34&pids=0,16,50,104,166,1707

rtsp://192.168.128.1/stream=0?freq=12402&pol=v&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=27500
&fec=34&pids=0,16,50,104,166,1707

rtsp://192.168.128.1/stream=1?addpids=17,51&delpids=166,1707

rtsp://192.168.128.5/?src=1&fe=1&freq=12720&pol=v&msys=dvbs2&mtype=8psk&ro=0.35&plts=on&sr=30000
&fec=910&pids=0.16.50.104.166.1707
```

3.5.12 Example RTSP Sequence Diagram



3.5.13 IGMP

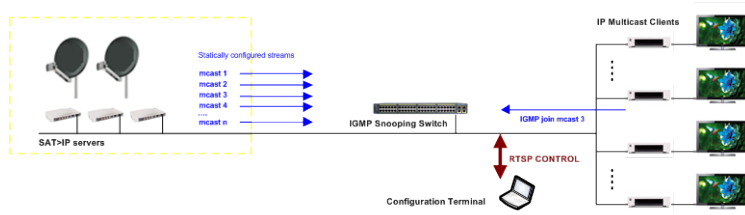
The implementation of IGMP is a requirement for any IP multicast host. As SAT>IP servers are capable of sourcing multicast traffic, they need to implement a basic set of IGMP features described in this chapter.

SAT>IP servers shall support **IGMPv3** as specified in RFC3376 [7]. Source-Specific Multicast (SSM) is not supported in SAT>IP. IGMPv3 mandates backwards compatibility with IGMPv2 but SAT>IP servers do not have to understand IGMPv2 messages.

All IGMP messages originating from SAT>IP servers are sent with an IP TTL of 1 and have to carry an IP Router Alert option in their IP header [RFC2113].

Multicast Operation

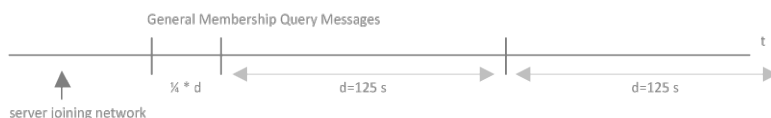
SAT>IP servers operating in a multicast environment, are typically setup using a configuration terminal. The configuration terminal sends RTSP SETUP messages defining the multicast transport followed each time by a PLAY message. Clients on the other hand purely join the sessions at the multicast level and do not issue RTSP requests.



General Membership Queries

During normal operation SAT>IP routers send an IGMPv3 General Membership Query message [7] to the multicast address **224.0.0.1** every 125 seconds in order to signal their presence to multicast aware switches and to learn about group memberships on the network.

When the server starts up this delay is reduced to one quarter that value.



Multicast hosts respond to these queries within a randomly distributed delay of 10 seconds. It is sufficient for one host to respond per group.

The “max response code” shall thus be set to 10 seconds in general queries. As it is expressed in units of 1/10 second, the actual encoded value is 100. The QRV (Querier’s Robustness Variable) shall be set to 2. The QQIC (Querier’s Query Interval Code) to 125.

Example: General Membership Query

```

Internet Protocol Version 4, Src: 192.168.128.193 (192.168.128.193), Dst: 224.0.0.1 (224.0.0.1)
Internet Group Management Protocol
  [IGMP Version: 3]
  Type: Membership Query (0x11)
  Max Response Time: 10.0 sec (0x64)
  Header checksum: 0xec1e [correct]
  Multicast Address: 0.0.0.0 (0.0.0.0)
  QRV=2
  QQIC: 125
  Num Src: 0

```

Querier Election

In order to reduce the number of multicast queries and reports on a network in the presence of more than one SAT>IP server, IGMP provides a Querier Election mechanism. This algorithm elects the router with the smallest IP address as the network's Querier. The other routers assume the role of Non-Querier and do not generate query messages but act on reports received via their network interface.

A SAT>IP server starts as a Querier on the network to which it is attached. If the server hears a message from another SAT>IP server with a lower IP address, it becomes a Non-Querier on the network. If no other message is heard from the elected Querier within 255 seconds (Other Querier Present Interval: $2 \times 125 + 10/2$) it becomes a Querier again.

Membership Reports

Membership Reports are sent by SAT>IP clients to signal that they are or want to become member of a group or that they would like to leave a group.

Joining a Multicast Group

When an IGMPv3 host wants to join a group it sends an IGMPv3 Membership Report message to the destination IP address **224.0.0.22** (CHANGE_TO_EXCLUDE_MODE) listing the group(s) it wants to be become a member of.

Example: Membership Report / Join group 239.43.0.1 for any sources

```
Internet Protocol Version 4, Src: 192.168.128.100 (192.168.128.100), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
  [IGMP Version: 3]
  Type: Membership Report (0x22)
  Header checksum: 0xead1 [correct]
  Num Group Records: 1
Group Record : 239.43.0.1 Change To Exclude Mode
```

Leaving a Multicast Group

When an IGMPv3 host wants to leave a group it sends an IGMPv3 Membership Report message to the destination IP address **224.0.0.22** (CHANGE_TO_INCLUDE_MODE).

The leave latency of the server shall be around 2 seconds.

Example: Membership Report / Leave group 239.43.0.1

```
Internet Protocol Version 4, Src: 192.168.128.100 (192.168.128.100), Dst: 224.0.0.22 (224.0.0.22)
Internet Group Management Protocol
  [IGMP Version: 3]
  Type: Membership Report (0x22)
  Header checksum: 0xebd1 [correct]
  Num Group Records: 1
Group Record : 239.43.0.1 Change To Include Mode
```

Multicast join and leave messages shall always be repeated twice.

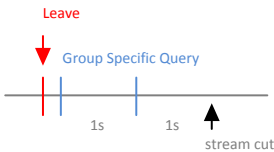
Please note an IGMP leave message does not stop the stream after a PLAY was issued for a multicast group by the stream owner.

Group-Specific Queries

Group Specific Membership Queries are sent by SAT>IP servers in response to changes in group membership. Group-Specific Queries are sent with an IP destination address equal to the multicast address of interest.

Thus if a SAT>IP client sends a leave message, the server, which is the querier, immediately makes a group specific query to check whether another host is still present for the same group.

After one second the querier re-sends the group specific query. If no group membership is received within one second after the second group specific query, the server assumes that there is no longer a member for this group and the multicast traffic shall be stopped.



The “Max Response Code” (time) in these queries shall be set to 1 second (10).

The “last member query interval” parameter sets the interval to wait for a response to a “group specific” query. The default value is 1 second.

The “last member query count” is the number of group specific queries sent before the SAT>IP server assumes that there are no members. The default value is 2.

Example: Membership Query, specific for group 239.43.0.1

Internet Protocol Version 4, Src: 192.168.128.193 (192.168.128.193), Dst: 239.43.0.1 (239.43.0.1)
Internet Group Management Protocol
[IGMP Version: 3]
Type: Membership Query (0x11)
Max Response Time: 1.0 sec (0x0a)
Header checksum: 0xfd4b [correct]
Multicast Address: 239.43.0.1 (239.43.0.1)
QRV=2
QQIC: 125
Num Src: 0

3.5.14 Status Code Definitions

Status code messages are sent in reply to RTSP requests. The following table lists the status code messages that are used in the SAT>IP protocol.

Status code	Reason Phrase
100	Continue
200	OK
400	Bad Request
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
408	Request Timeout
414	Request-URI Too Long
453	Not Enough Bandwidth
454	Session Not Found
455	Method Not Valid in This State
461	Unsupported Transport
500	Internal Server Error
501	Not Implemented
503	Service Unavailable
505	RTSP Version Not Supported
551	Option Not Supported

3.5.14.1 Client Error Response Messages (4xx)

Status code	400 – Bad Request	
Description	The request could not be understood by the server due to a malformed syntax. Returned when missing a character, inconsistent request (duplicate attributes), etc.	
Response	Headers	CSeq, Content-Type: text/parameters, Content-Length
	Body	The message body of the response may contain the " Check-Syntax: " parameter followed by the malformed syntax.
Example	Request	PLAY rtsp://192.168.128.5/strem=1 RTSP/1.0 CSeq: 6 Session: 12345678 <CRLF>
	Response	RTSP/1.0 400 Bad Request CSeq: 6 Content-Type: text/parameters Content-Length: 19 <CRLF> Check-Syntax: strem
Status code	403 – Forbidden	
Description	The server understood the request, but is refusing to fulfil it. Returned when passing an attribute value not understood by the server in a query, or an out-of-range value.	
Response	Headers	CSeq, Content-Type: text/parameters, Content-Length
	Body	The message body of the response may contain the " Out-of-Range: " parameter followed by a space-separated list of the attribute names that are not understood: "src" "fe" "freq" "pol" "msys" "mtype" "plts" "ro" "sr" "fec" "pids" "addpids" "delpids" "mcast"
Example	Request	SETUP rtsp://192.168.128.5/?src=1&fe=1&freq=22402&pol=v&msys=dvbs&sr=27500&fec=34 &pids=0,16,50,104,166,1707,8192 RTSP/1.0 CSeq: 7 <CRLF>
	Response	RTSP/1.0 403 Forbidden CSeq: 7 Content-Type: text/parameters Content-Length: 23 <CRLF> Out-of-Range: freq pids

Status code	404 – Not Found	
Description	The server has not found anything matching the Request-URI. Returned when requesting a stream with a streamID that does not exist.	
Response	Headers	CSeq
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/stream=7 RTSP/1.0 CSeq: 8 Session: 12345678 <CRLF>
	Response	RTSP/1.0 404 Not Found CSeq: 8 <CRLF>

Status code	405 – Method Not Allowed	
Description	The method specified in the request is not allowed for the resource identified by the Request-URI. Returned when applying a SETUP, PLAY or TEARDOWN method on an RTSP URI identifying the server.	
Response	Headers	CSeq, Allow
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/ RTSP/1.0 CSeq: 9 Session: 12345678 <CRLF>
	Response	RTSP/1.0 405 Method Not Allowed CSeq: 9 Allow: OPTIONS, DESCRIBE <CRLF>

Status code	406 – Not Acceptable	
Description	The resource identified by the request is only capable of generating response message bodies which have content characteristics not acceptable according to the accept headers sent in the request. Issuing a DESCRIBE request with an accept header different from "application/sdp".	
Response	Headers	CSeq
	Body	n/a
Example	Request	DESCRIBE rtsp://192.168.128.5/ RTSP/1.0 Cseq: 10 Accept: text/plain <CRLF>
	Response	RTSP/1.0 406 Not Acceptable Cseq: 10 <CRLF>

Status code	408 – Request Timeout	
Description	The client did not produce a request within the time that the server was prepared to wait. The client may repeat the request without modifications at any later time. E.g. issuing a PLAY request after the communication link had been idle for a period of time. The time interval has exceeded the value specified by the "timeout" parameter in the "Session:" header field of a SETUP response.	
Response	Headers	CSeq
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 11 Session: 12345678 <CRLF>
	Response	RTSP/1.0 408 Request Timeout CSeq: 11 <CRLF>

Status code	414 – Request-URI Too Long	
Description	The server is refusing to service the request because the Request-URI is longer than the server is willing to interpret. The RTSP protocol does not place any limit on the length of a URI. Servers should be able to handle URIs of unbounded length.	
Response	Headers	CSeq
	Body	n/a
Example	Request	SETUP rtsp://192.168.128.5/?src=1&fe=1&freq=12402&pol=v&mssys=dvbs&sr=27500&fec=34 &pids=0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25 RTSP/1.0 Cseq: 12 <CRLF>
	Response	RTSP/1.0 414 Request-URI Too Long Cseq: 12 <CRLF>

Status code	453 – Not Enough Bandwidth	
Description	The request was refused because there is insufficient bandwidth on the in-home LAN. Returned when clients are requesting more streams than the network can carry.	
Response	Headers	CSeq
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/stream=25 RTSP/1.0 CSeq:13 Session: 12345678 <CRLF>
	Response	RTSP/1.0 453 Not Enough Bandwidth CSeq: 13 <CRLF>

Status code	454 – Session Not Found	
Description	The RTSP session identifier value in the "Session:" header field of the request is missing, invalid, or has timed out. Returned when issuing the wrong session identifier value in a request.	
Response	Headers	CSeq
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/stream=1 RTSP/1.0 Cseq: 14 Session: 0 <CRLF>
	Response	RTSP/1.0 454 Session Not Found Cseq: 14 <CRLF>

Status code	455 – Method Not Valid in This State	
Description	The client or server cannot process this request in its current state. Returned e.g. when trying to change transport parameters while the server is in the play state (e.g. change of port values, etc.).	
Response	Headers	CSeq
	Body	n/a
Example	Request	SETUP rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 15 Transport: RTP/AVP;multicast;port=1402-1403 <CRLF>
	Response	RTSP/1.0 455 Method Not Valid in This State CSeq: 15 <CRLF>

Status code	461 – Unsupported Transport	
Description	The "Transport:" header field of the request did not contain a supported transport specification. Returned e.g. when issuing a profile that is different from "RTP/AVP".	
Response	Headers	CSeq
	Body	n/a
Example	Request	SETUP rtsp://192.168.128.5/?src=1&fe=1&freq=12402&pol=v&msys=dvbs&sr=27500&fec=34 &pids=0,16,50,104,166,1707 RTSP/1.0 CSeq: 16 Transport: RTP/SAVP;multicast;port=1400-1401 <CRLF>
	Response	RTSP/1.0 461 Unsupported Transport CSeq: 16 <CRLF>

3.5.14.2 Server Error Response Messages (5xx)

Status code	500 – Internal Server Error	
Description	The server encountered an error condition preventing it to fulfil the request. Returned when the server is not functioning correctly due to a hardware failure or a software bug or anything else that can go wrong.	
Response	Headers	CSeq
	Body	n/a
Example	Request	DESCRIBE rtsp://192.168.128.5/ CSeq: 17 Accept: application/sdp <CRLF>
	Response	RTSP/1.0 500 Internal Server Error CSeq: 17 <CRLF>
Status code	501 – Not Implemented	
Description	The server does not support the functionality required to fulfil the request. Issuing a request that is not implemented (e.g. PAUSE).	
Response	Headers	CSeq, Public
Example	Request	PAUSE rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 18 Session: 12345678 <CRLF>
	Response	RTSP/1.0 501 Not Implemented CSeq: 18 Public: OPTIONS, DESCRIBE, SETUP, PLAY, TEARDOWN <CRLF>
Status code	503 – Service Unavailable	
Description	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. Returned when reaching the maximum number of hardware and software resources, the maximum number of sessions.	
Response	Headers	CSeq
	Body	The message body of the response may contain the "No-More:" parameter followed by a space-separated list of the missing resources: "sessions" "frontends" "pids"
Example	Request	SETUP rtsp://192.168.128.5/?src=2&freq=12402&pol=v&msys=dvbs&sr=27500&fec=34&pids=0,16,50,104,166,1707 RTSP/1.0 CSeq: 19 Transport: RTP/AVP;multicast;port=1400-1401 <CRLF>
	Response	RTSP/1.0 503 Service Unavailable CSeq: 19 <CRLF> No-More: frontends
Status code	505 – RTSP Version Not Supported	
Description	The server does not support the RTSP protocol version that was used in the request message.	
Response	Headers	CSeq
	Body	n/a
Example	Request	OPTIONS rtsp://192.168.128.5/ RTSP/2.0 CSeq: 20 <CRLF>
	Response	RTSP/1.0 505 RTSP Version Not Supported CSeq: 20 <CRLF>
Status code	551 – Option Not Supported	
Description	A feature-tag given in the "Require:" header field of the request was not supported. Issuing a request with a "Require:" header field.	
Response	Headers	CSeq, Unsupported
	Body	n/a
Example	Request	PLAY rtsp://192.168.128.5/stream=1 RTSP/1.0 CSeq: 21 Require: specific-feature Specific-parameter: parameter <CRLF>
	Response	RTSP/1.0 551 Option Not Supported CSeq: 21 Unsupported: specific-feature <CRLF>

3.5.15 RTCP Announcements

3.5.15.1 Description

Every RTP delivered media stream (unicast or multicast) is accompanied by an RTCP announcement stream. The RTCP stream becomes available as soon as the RTP stream starts playing. This announcement stream carries information about the SAT>IP configuration for that particular TS media stream plus information about the **real-time** reception conditions.

These announcements use an RTCP (RFC 3550) [8] control channel that is delivered on the same IP address as the corresponding RTP stream. However the port number of the RTCP stream is the port number of RTP + 1.

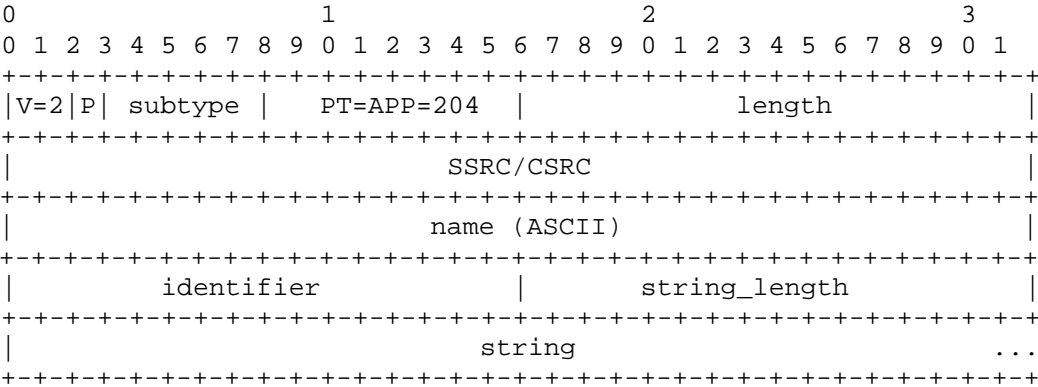
The RTCP stream carries real-time information about the **signal level**, **tuner lock**, **signal quality** and configured DVB reception parameters of the corresponding RTP stream. RTCP should provide **5 updates** of the reception parameters **per second**. The RTCP stream shall also be available in the presence of empty RTP packets i.e. when no satellite signal is being received.

In the context of this specification clients are required not to send any Receiver Reports back to the SAT>IP server.

3.5.15.2 Syntax

RTCP announcements are carried in RTCP Application-Defined APP packets as specified in RFC 3550 [8].

The APP packets have the following syntax:



The APP packet fields are specified below:

Field	Value	Example
name	Set to "SES1".	SES1
identifier	A 16-bit version field to allow multiple interoperating packet formats. Set to "0000".	0000
string_length	A 16-bit field specifying the number of bytes of the string field.	0048
string	Field carrying the payload format specified below.	See below.

APP Packet String Payload Format:

```
ver=<major>.<minor>;src=<srcID>;tuner=<feID>,<level>,<lock>,<quality>,<frequency>,<polarisation>,<system>,<type>,<pilots>,<roll_off>,<symbol_rate>,<fec_inner>;pids=<pid0>,...,<pidn>
```

The string payload format is identical to the format following the "a=fmtp:33" attribute defined in the SDP file.

The text is encoded using UTF-8 encoding as specified in RFC 2279 [5].

- If the text string fills the packet to the next 32-bit boundary, the string is not null terminated.
- If not, the APP packet shall be padded with null bytes to the next 32-bit boundary (This padding is separate from that indicated by the P bit in the RTCP header).

Example String:

```
ver=1.0;src=1;tuner=1,240,1,7,12402,v,dvbs,,,27500,34;pids=0,16,56,112,168,1709
```

Please note:

The APP packet is always part of a compound RTCP packet. According to the rules of RFC 3550 [8], a compound RTCP packet must contain, in addition to the APP packet, a Sender Report (SR) packet plus a Source Description Items (SDS) packet.

3.5.16 HTTP

SAT>IP Media servers shall also be controllable via HTTP based requests.

HTTP protocol support shall rely on HTTP version 1.1.

If a SAT>IP client would like to get a stream delivered via the HTTP transport protocol it shall use the HTTP GET method to initiate a stream playout.

The HTTP URI syntax is as follows:

HTTP_URI = "http://" ip_address [":"port] "/"? query

The HTTP query syntax is identical to the RTSP query syntax specified in the table under 3.5.11.

A streaming HTTP client should implement the "stop" media operation by disconnecting the TCP connection of the HTTP transaction.

Standard HTTP status code messages are returned in reply to HTTP requests.

Status code messages 400, 403 and 503 can make use of the syntactical extensions ("Check-Syntax", "Out-of-Range", "No-More") described for RTSP.

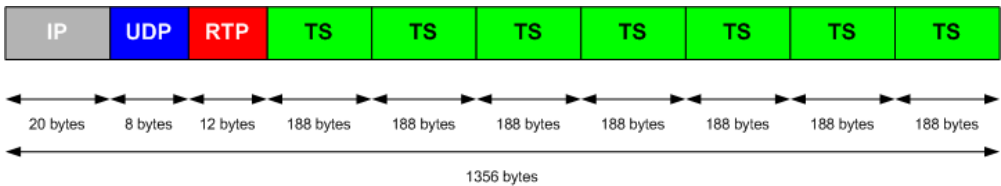
Example http get message:

```
http://192.168.128.1/?src=1&fe=1&freq=12402&pol=v&msys=dvbs&mtype=qpsk&sr=27500&fec=34&pids=0,16,50,104,166,1707
```


3.6 Media Transport

3.6.1 RTP Transport

SAT>IP uses the industry standard RFC 2250 [4] for carrying transport streams. Generally, respecting the overall MTU size of 1500 bytes, 7 TS packets can be carried per IP datagram. However an IP packet may also contain less than 7 TS packets. RTP encapsulation is mandatory.



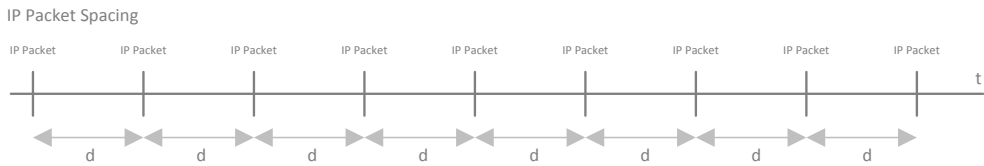
SAT>IP devices can carry MPTS or SPTS streams according to the session setup parameters. MPTS streams contain the entire original transport stream as received off the satellite. Partial Transport Streams or Single Program Transport Streams only carry a selection of the PIDs making up the original transport stream.

The SAT>IP receiver requests the PIDs required in order to present a requested program to the end-user. The SAT>IP receiver implements a DVB service logic and thus is capable of parsing DVB Service Information and Program Specific Information in order to correctly tune to DVB services.

Please note that when no signal is being received, the signal is being lost, or no PID is available (also e.g. when "pids=none"), the SAT>IP server shall nevertheless issue an empty RTP packet at least every 100ms (accompanied by an RTCP announcement stream).

Empty RTP packets shall **not** contain dummy generated TS null packets. Actual null packets broadcast over satellite with PID value 8191 shall be demultiplexed and streamed only if the CSV list of PIDs contains the value 8191.

RTP packets shall not be buffered by SAT>IP servers and shall not be delivered in a single burst. As soon as an RTP packet is filled with up to seven MPEG-2 TS packets, the RTP packet shall be released by the server. Buffering of RTP packets by the server will lead to increased packet jitter and may result in the malfunctioning of clients. SAT>IP servers shall therefore send packets at a constant rate, not in bursts, and packets shall be evenly paced.



3.6.2 HTTP Streaming

The SAT>IP server shall also implement HTTP as a mandatory media transport protocol.

The HTTP header shall contain the following MIME type: "video/MP2T".

3.7 Media Formats

SAT>IP devices shall support the carriage of A/V content and related information in an MPEG-2 Transport Stream as specified in clause 4 of TS 101 154 [13].

The SAT>IP architecture does not require server-side A/V format transcoding.

4 Dynamic vs Static Server Operation

4.1 Dynamic Operation (default)

In normal operation SAT>IP clients request the delivery of TS media streams from SAT>IP servers. SAT>IP servers dynamically tune to satellite transponders as requested. This mode of operation is called dynamic.

Each time that a client requests the setup of a stream it creates a session with the server. Within the session the client can request modifications to the streams as needed e.g. when changing channels.

Tuner resources are dynamically allocated by the server based on these requests.

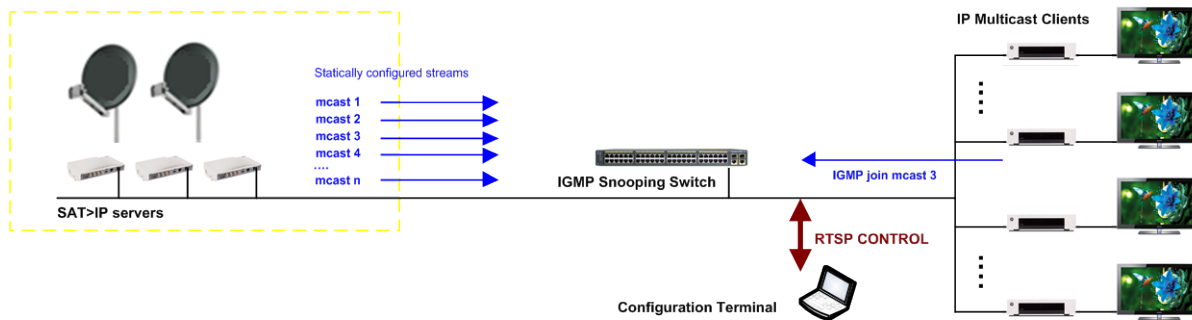
4.2 Static Operation

The SAT>IP protocol also supports a so-called static operation. This type of operation is suitable for very large installations with many tens, hundreds or even more clients. Examples for such deployments are hospitals, large hotels, new build areas where satellite signals are distributed via fiber.

In these installations a configuration terminal sets up the streams through RTSP requests and creates a session for each stream with the server. The configuration terminal as the virtual master or owner of the sessions has total control over all the TS media streams.

Clients join and leave the streams without entering into an RTSP session by simply issuing IGMP join and leave messages as required. In this mode of operation SAT>IP resembles the operation of a traditional managed IPTV platform.

The diagram below shows an example of such a system:



4.3 Mixed Operation

In mixed operation, SAT>IP servers have certain streams permanently configured (monopolising a part of the server's tuning resources) and another part of the resources available for dynamic tuning.

Appendix A: Client Implementation Guidelines

This section contains guidelines for the implementation of SAT>IP clients. Two major categories of SAT>IP clients are considered below: RTSP clients and IGMPv3 clients.

A1: RTSP Clients

RTSP clients always setup RTSP sessions with SAT>IP servers. RTSP sessions may be defined using **unicast** or **multicast** transport parameters.

A1.1 Client Setup and Configuration Settings

Detecting available SAT>IP servers

SAT>IP clients shall automatically detect the availability of all SAT>IP servers on the network by listening to Server Announcement messages (SSDP NOTIFY) and by sending multicast Search Requests (SSDP M-SEARCH). In SAT>IP several servers can co-exist (be stacked) in the same network.

Automatic binding to a server

SAT>IP clients should automatically bind to a SAT>IP server with available tuning resources. If a given server is out of resources for fulfilling a client request, the client should try other servers on the network. After a client application re-start, the UPnP discovery process shall be re-launched before the automatic binding takes place.

Binding to a particular server

In addition to the automatic binding, it is recommended that SAT>IP clients provide a settings menu entry in order to manually select a particular server. Unlike the automatic binding which is only temporary, it should be possible to permanently force the selection of a particular server. This allows the creation of a network in which clients are setup with guaranteed server and tuning resources.

Addressing servers through their IP address

In addition to the DHCP and Auto-IP mechanisms described above, it is recommended that SAT>IP clients provide the option of manually configuring the IP address of a SAT>IP server. This is useful in instances where a SAT>IP server cannot be detected via UPnP.

Binding to a particular frontend

SAT>IP servers mostly provide several frontends. A client should generally let the server automatically select a frontend. It does this by **not** including the “fe” parameter in its requests.

Clients can however provide an option in the settings menu for specifying a fixed frontend in their requests. This allows a client to get access to a dedicated frontend resource. In a controlled network environment it thus becomes possible to associate guaranteed tuning resources (servers and frontends) to each and every client.

Antenna Configuration

Before a SAT>IP RTSP client can be used for the first time it needs to be setup and configured with information about the actual satellite antenna and installation that the SAT>IP server(s) on the network is(are) connected to. From a client perspective this is identical to the way that traditional DVB Set-Top-Boxes are setup. Within the client application settings menu, the end user tells the application which satellite orbital

positions are available in the actual installation and to which source they are associated. When designing client applications it is reasonable to assume that all frontends available from SAT>IP servers in the network are connected to the same satellite installation and that every frontend provides access to the same signals.

SAT>IP clients use the “src” parameter in their requests to specify the satellite position that they would like to access. In SAT>IP implementations that support legacy satellite installations, the recommendation is to map the “src” parameter to the criteria used in the DiSEqC™ control system. The mapping shall be as follows:

“src”	DiSEqC
1	DiSEqC A = Position A Option A
2	DiSEqC B = Position B Option A
3	DiSEqC C = Position A Option B
4	DiSEqC D = Position B Option B

A1.2 Service Discovery

The SAT>IP specification does not mandate a particular service discovery mechanism i.e. a particular way for clients to learn about the availability of audiovisual services.

Standard DVB Service Discovery

DVB clients generally get information about services available via standardised DVB tables (Service Information / Program Specific Information) carried as part of the Network Transport Streams. This mechanism can also be used by SAT>IP clients. The advantage of this mechanism is that it is always up-to-date, relies on the actual broadcast information, works generically on any DVB network in the world and does not need an internet connection.

The disadvantage is that it can be slow to run the initial scan when the client is setup for the first time. For this reason it is recommended that clients are delivered with a factory preset service list.

Discovery based on Service Lists

Additionally or alternatively SAT>IP clients can also rely on service lists made available offline (as files) or online (from a web server) which list all the services available from one or more satellite positions. Clients implementing this type of scheme are referred to as thin clients.

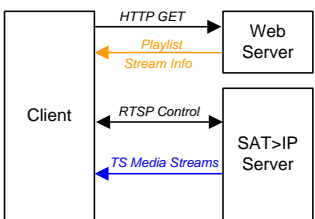
When implementing this latter scheme it is strongly recommended that SAT>IP clients understand at least the **Extended M3U Channel list** format described below.

SAT>IP clients relying on service lists should feature an option in their settings menu to specify:

- the path towards the M3U file in the directory structure
- a URL with the IP address or domain of a web server carrying such lists

http://sat-ip_server/channel_file.m3u

A web server exporting the channel list could run on the SAT>IP server itself, on a separate device on the same LAN, or simply remotely on the internet.



A1.3 Channel Change Implementation

There are different possibilities for initiating a channel change in SAT>IP. Below we provide a recommendation for a method which is suitable in most instances.

In-session channel change

Request 1

```
SETUP rtsp://192.168.178.21:554/?src=1&freq=10744&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000&fec=56&pids=0,400,401,402
RTSP/1.0
CSeq:1
Transport: RTP/AVP;unicast;client_port=11992-11993
```

Server Response in 13 ms

```
RTSP/1.0 200 OK
CSeq: 1
Date: Fri, Jan 07 2000 00:57:14 GMT
Transport: RTP/AVP;unicast;destination=192.168.178.39;source=192.168.178.21;client_port=11992-11993;server_port=6970-6971
Session: ADB555E0
com.ses.streamID: 18
```

Request 2

```
PLAY rtsp://192.168.178.21:554/stream=18 RTSP/1.0
CSeq:2
Session:ADB555E0
```

Server Response in 8 ms

```
RTSP/1.0 200 OK
CSeq: 2
Date: Fri, Jan 07 2000 00:57:15 GMT
Session: ADB555E0
RTP-Info: url=rtsp://192.168.178.21/stream=18;seq=55446;rtptime=411856963
```

Request 3

```
PLAY rtsp://192.168.178.21:554/stream=18?src=1&freq=11538&pol=v&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000&fec=56&pids=0,201,605,625
RTSP/1.0
CSeq:3
Session:ADB555E0
```

Server Response in 19 ms

```
RTSP/1.0 200 OK
CSeq: 3
Date: Fri, Jan 07 2000 00:57:25 GMT
Session: ADB555E0
RTP-Info: url=rtsp://192.168.178.21/stream=18;seq=61344;rtptime=2499037766
```

Request 4

```
PLAY
rtsp://192.168.178.21:554/stream=18?src=1&freq=12603&pol=h&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000&fec=56&pids=0,1290,2290,6290,7
290,8190 RTSP/1.0
CSeq:4
Session:ADB555E0
```

Server Response in 21 ms

```
RTSP/1.0 200 OK
CSeq: 4
Date: Fri, Jan 07 2000 00:57:35 GMT
Session: ADB555E0
RTP-Info: url=rtsp://192.168.178.21/stream=18;seq=64053;rtptime=2499950790
```

Request 5

```
PLAY
rtsp://192.168.178.21:554/stream=18?src=1&freq=12551&pol=v&ro=0.35&msys=dvbs&mtype=qpsk&plts=off&sr=22000&fec=56&pids=0,2170,2171,2172
RTSP/1.0
CSeq:5
Session:ADB555E0
```

Server Response in 20 ms

```
RTSP/1.0 200 OK
CSeq: 5
Date: Fri, Jan 07 2000 00:57:45 GMT
Session: ADB555E0
RTP-Info: url=rtsp://192.168.178.21/stream=18;seq=1618;rtptime=2500864089
```

Request 6
TEARDOWN rtsp://192.168.178.21:554/stream=18 RTSP/1.0
CSeq:6
Session:ADB555E0

Server Response in 8 ms
RTSP/1.0 200 OK
CSeq: 6
Session:ADB555E0
Date: Fri, Jan 07 2000 00:57:55 GMT

A2 IGMPv3 Clients

IGMP clients do not setup RTSP sessions with the server. They purely join and leave multicast sessions setup by other clients. IGMP clients are mainly used in larger and fully managed network environments.

A2.1 Client Setup and Configuration Settings

IGMP clients do not need to be configured with parameters relating to the satellite installation, as they are not setting up sessions themselves and are not controlling the tuning process in SAT>IP servers.

IGMP clients however need to be configured with a path (URL/file link) to the channel list which carries all the services configured and available on the local network together with the multicast addresses on which these services are carried. IGMP clients use this information to provide a channel list to the end-user and allow the end-user to select/join certain channels.

IGMP clients shall implement at least the **Extended M3U Channel list** described below.

A typical multicast channel list for an IGMP client will look as follows:

M3U playlist format (for IGMP clients)

```
#EXTM3U
#EXTINF:0,1. TF1
rtsp://232.0.1.17:8200
#EXTINF:0,2. France 2
rtsp://232.0.1.1:8200
#EXTINF:0,3. France 3
rtsp://232.0.1.2:8200
#EXTINF:0,4. Das Erste
rtsp://239.35.129.11:10000
#EXTINF:0,5. ZDF
rtsp://239.35.86.11:10000
#EXTINF:0,6. KI.KA
rtsp://239.35.205.12:10000
#EXTINF:0,9. EinsFestival
rtsp://239.35.193.11:10000
#EXTINF:0,10. ZDFinfokanal
rtsp://239.35.214.11:10000
#EXTINF:0,11. ZDFdoku kanal
rtsp://239.35.150.11:10000
....
```

Extended M3U Channel List

SAT>IP RTSP unicast clients should and SAT>IP IGMP multicast clients shall support the extended M3U playlist file format for acquiring information about available streams.

Two examples of extended M3U files are shown below:

M3U playlist format (Unicast Streams)

```
#EXTM3U
#EXTINF:0,13. BBC World
rtsp://192.168.1.202/?src=1&freq=11597&pol=v&msys=dvbs&mtype=qpsk&sr=22000&fec=56&pids=0,16,17,92,163,1858
#EXTINF:0,14. Sky News
rtsp://192.168.1.202/?src=1&freq=12604&pol=h&msys=dvbs&mtype=qpsk&sr=22000&fec=56&pids=0,1290,2290,7290,6290,5290
#EXTINF:0,15. France24
rtsp://192.168.1.202/stream=18
```

M3U playlist format (Multicast Streams)

```
#EXTM3U
#EXTINF:0,1. TF1
rtp://232.0.1.17:8200
#EXTINF:0,2. France 2
rtp://232.0.1.1:8200
#EXTINF:0,3. France 3
rtp://232.0.1.2:8200
```

The Extended M3U file is encoded as follows using the Latin-1 charset:

#EXTM3U: The Extended M3U file starts with this header (all capital letters)

#EXTINF:0,10. ZDFinfokanal

The #EXTINF line is encoded as follows:

#EXTINF:

This indicates that this is an Extended Information field. It ends with a colon ":".

0,

This is the time of the stream. For live streams which do not have an end time a zero value is used. This is followed by a comma.

10. ZDFinfokanal

This is the logical channel number (LCN) "10" followed by a **dot** "." followed by a **space** and then the channel name. No characters follow the channel name.

Appendix B: Example SAT>IP Message Exchanges

Example 1: Unicast Session Setup (no front-end selected) plus three additional channel changes

```
Client Request 1 >
53 45 54 55 50 20 72 74 73 70 3a 2f 2f 31 39 32 SETUP rtsp://192.
2e 31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 .168.128.192:554
2f 3f 72 72 63 3d 31 26 66 72 65 71 3d 31 30 37 /?src=1&freq=107
34 34 26 70 6f 6c 3d 68 26 72 6f 3d 30 2e 33 35 44&pol=h&ro=0.35
26 6d 73 79 73 3d 64 76 62 73 26 6d 74 79 70 65 &msys=dvbs&mtype
32 71 70 73 6b 26 70 6c 74 73 3d 6f 66 66 26 73
72 6d 32 32 30 30 30 26 66 65 63 3d 35 36 26 70
69 64 73 3d 30 2c 3a 30 30 2c 3a 30 31 2c 3a 30
32 20 52 54 53 50 2f 31 2e 30 0d 0a 43 53 65 71 2 RTSP/1.0..CSeq
3a 31 0d 0a 54 72 61 6e 73 70 6f 72 74 3a 20 52 :1..Transport: R
54 50 2f 41 56 50 3b 75 6e 69 63 61 73 74 3b 0b TP/AVP:unicast;c
6c 69 65 6e 74 5f 70 6f 72 74 3d 6c 33 37 37 38 client_port=37786
2d 33 37 37 38 37 0d 0a 43 4f 6e 6e 65 63 74 69 -37787..Connecti
6f 6e 3a 63 6c 6f 73 65 0d 0a 0d 0a on:close....
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 31 0d 0a 44 61 74 65 3a 20 .CSeq: 1..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 33 31 3a 3a 37 20 47 4d 54 0d 0a 54 01:31:47 GMT..T
72 61 6e 73 70 6f 72 74 3a 37 20 52 54 50 2f 41 56 ransport: RTP/AV
50 3b 75 6e 69 63 61 73 74 3b 64 65 73 74 69 6e P:unicast;destin
61 74 69 6f 6e 3d 31 39 32 2e 31 36 38 2e 31 32 ation=192.168.12
38 2e 31 30 30 3b 73 6f 75 72 63 65 3d 31 39 32 8.100;source=192
2e 31 36 38 2e 31 32 38 2e 31 39 32 3b 63 6c 69 .168.128.192:cli
65 6e 74 5f 70 6f 72 74 3d 33 37 37 38 36 2d 33 ent_port=37786-3
37 37 38 37 3b 73 6f 72 76 65 72 5f 70 6f 72 74 7787;server_port
32 36 39 37 30 2d 36 39 37 31 0d 0a 53 65 73 73 =6970-6971..Sess
69 6f 6e 3a 20 46 44 32 43 32 39 38 45 0d 0a 63 ion:FD2C298E..c
6f 6d 2e 73 65 73 2e 73 74 72 65 61 6d 49 44 3a om.ses.streamID:
20 32 30 0d 0a 0d 0a 20....
Client Request 2 >
50 4c 41 59 20 72 74 73 70 3a 2f 2f 31 39 32 2e PLAY rtsp://192.
31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 2f 168.128.192:554/
73 74 72 65 61 6d 3d 32 30 20 52 54 53 50 2f 31 stream=20 RTSP/1
2e 30 0d 0a 43 53 65 71 3a 32 0d 0a 53 65 73 73 ..CSeq:2..Sess
69 6f 6e 3a 46 44 32 43 32 39 38 45 0d 0a 43 4f ion:FD2C298E..Co
6e 6e 65 63 74 69 6f 6e 3a 63 6c 6f 73 65 0d 0a nnection:close..
0d 0a
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 32 0d 0a 44 61 74 65 3a 20 .CSeq: 2..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 33 31 3a 3a 37 20 47 4d 54 0d 0a 53 01:31:47 GMT..S
65 73 73 69 6f 6e 3a 20 46 44 32 43 32 39 38 45 session:FD2C298E
0d 0a 52 54 50 2d 49 6e 66 6f 3a 20 75 72 6c 3d ..RTP-Info: url=
72 74 73 70 3a 2f 2f 31 39 32 2e 31 36 38 2e 31 rtsp://192.168.1
32 38 2e 31 39 32 2f 73 74 72 65 61 6d 3d 32 30 28.192/stream=20
3b 73 65 71 3d 33 38 32 35 34 3b 72 74 70 74 69 ;seq=38254;rtpti
6d 65 3d 32 38 32 31 39 38 37 34 33 0d 0a 0d me=28219877543...
0a
Client Request 3 >
50 4c 41 59 20 72 74 73 70 3a 2f 2f 31 39 32 2e PLAY rtsp://192.
31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 2f 168.128.192:554/
73 74 72 65 61 6d 3d 32 30 3f 73 72 63 3d 31 26 stream=20?src=1&
66 72 65 71 3d 31 31 35 33 38 26 70 6f 6c 3d 76 freq=11538&pol=v
26 72 6f 3d 30 2e 33 35 26 6d 73 79 73 3d 64 76 kro=0.35&msys=dv
62 73 26 6d 74 79 70 65 3d 71 70 73 6b 26 70 6c bs&mtype=qps&pl
74 73 3d 6f 66 66 26 73 72 3d 32 32 30 30 30 26 ts=off&sr=22000&
66 65 63 3d 35 36 26 70 69 64 73 3d 30 2c 36 31 fec=56&pids=0,61
31 2c 36 32 31 2c 36 33 31 20 52 54 53 50 2f 31 1,621,631 RTSP/1
2e 30 0d 0a 43 53 65 71 3a 37 20 47 4d 54 0d 0a 53 01:31:57 GMT..S
69 6f 6e 3a 46 44 32 43 32 39 38 45 0d 0a 43 4f ion:FD2C298E..Co
6e 6e 65 63 74 69 6f 6e 3a 63 6c 6f 73 65 0d 0a nnection:close..
0d 0a
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 33 0d 0a 44 61 74 65 3a 20 .CSeq: 3..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 33 31 3a 35 37 20 47 4d 54 0d 0a 53 01:31:57 GMT..S
65 73 73 69 6f 6e 3a 20 46 44 32 43 32 39 38 45 session:FD2C298E
0d 0a 52 54 50 2d 49 6e 66 6f 3a 20 75 72 6c 3d ..RTP-Info: url=
74 73 70 3a 2f 2f 31 39 32 2e 31 36 38 2e 31 rtsp://192.168.1
32 38 2e 31 39 32 2f 73 74 72 65 61 6d 3d 32 30 28.192/stream=20
3b 73 65 71 3d 35 31 30 36 38 3b 72 74 70 74 69 ;seq=51068;rtpti
6d 65 3d 32 38 32 34 36 36 39 37 36 31 0d 0a 0d me=2824669761...
0a
Client Request 4 >
50 4c 41 59 20 72 74 73 70 3a 2f 2f 31 39 32 2e PLAY rtsp://192.
31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 2f 168.128.192:554/
73 74 72 65 61 6d 3d 32 30 3f 73 72 63 3d 31 26 stream=20?src=1&
6
```


Example 2: Multicast Session Setup with front-end selection and destination address/port

```
Client Request 1 >
53 45 54 55 50 20 72 74 73 70 3a 2f 2f 31 39 32 SETUP rtsp://192
2e 31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 .168.128.192:554
2f 3f 73 72 63 3d 31 26 66 65 3d 31 26 66 72 65 //src=l&fe=l&fre
71 3d 31 30 37 34 34 26 70 6f 6c 3d 68 26 72 6f q=10744&pol=h&ro
3d 30 2e 33 35 26 6d 73 79 73 3d 64 76 62 73 26 =0.35&msys=dvbs&
6d 74 79 70 65 3d 71 70 73 6b 26 70 6c 74 73 3d mtype=qpsk&plts=
6f 66 66 26 73 72 3d 32 32 30 30 30 26 66 65 63 off&sr=22000&fec
3d 35 36 26 70 69 64 73 3d 30 2c 34 30 30 2c 34 =56&pids=0,400,4
30 31 2c 34 30 32 20 52 54 53 50 2f 31 2e 30 0d 01,402 RTSP/1.0.
0a 43 53 65 71 3a 31 0d 0a 54 72 61 6e 73 70 6f .CSeq:1..Transpo
72 74 3a 20 52 54 50 2f 41 56 50 3b 6d 75 6c 74 rt: RTP/AVP;mult
69 63 61 73 74 3b 64 65 73 74 69 6e 61 74 69 6f icast;destinatio
6e 3d 32 32 34 2e 31 36 2e 31 36 2e 31 3b 70 6f n=224.16.16.1;po
72 74 3d 34 32 31 32 38 2d 34 32 31 32 39 0d 0a rt=42128-42129..
43 6f 6e 6e 65 63 74 69 6f 6e 3a 63 6c 6f 73 65 Connection:close
0d 0a 0d 0a ....
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 31 0d 0a 44 61 74 65 3a 20 .CSeq: 1..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 32 39 3a 33 39 20 47 4d 54 0d 0a 54 01:29:39 GMT..T
72 61 6e 73 70 6f 72 74 3a 20 52 54 50 2f 41 56 ransport: RTP/AV
50 3b 6d 75 6c 74 69 63 61 73 74 3b 64 65 73 74 P;multicast;dest
69 6e 61 74 69 6f 6e 3d 32 32 34 2e 31 36 2e 31 ination=224.16.1
36 2e 31 3b 73 6f 75 72 63 65 3d 31 39 32 2e 31 6.1;source=192.1
36 38 2e 31 32 38 2e 31 39 32 3b 70 6f 72 74 3d 68.128.192;port=
34 32 31 32 38 2d 34 32 31 32 39 3b 74 74 6c 3d 42128-42129;ttl=
35 0d 0a 53 65 73 73 69 6f 6e 3a 20 36 32 45 42 5..Session: 62EB
43 32 44 38 0d 0a 63 6f 6d 2e 73 65 73 2e 73 74 C2D8..com.ses.st
72 65 61 6d 49 44 3a 20 32 35 0d 0a 0d 0a reamID: 25....

Client Request 2 >
50 4c 41 59 20 72 74 73 70 3a 2f 2f 31 39 32 2e PLAY rtsp://192.

31 36 38 2e 31 32 38 2e 31 39 32 3a 35 35 34 2f 168.128.192:554/
73 74 72 65 61 6d 3d 32 35 20 52 54 53 50 2f 31 stream=25 RTSP/1
2e 30 0d 0a 43 53 65 71 3a 32 0d 0a 53 65 73 73 .0..CSeq:2..Sess
69 6f 6e 3a 36 32 45 42 43 32 44 38 0d 0a 43 6f ion:62EBC2D8..Co
6e 6e 65 63 74 69 6f 6e 3a 63 6c 6f 73 65 0d 0a nnection:close..
0d 0a ..
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 32 0d 0a 44 61 74 65 3a 20 .CSeq: 2..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 32 39 3a 33 39 20 47 4d 54 0d 0a 53 01:29:39 GMT..S
65 73 73 69 6f 6e 3a 20 36 32 45 42 43 32 44 38 ession: 62EBC2D8
0d 0a 52 54 50 2d 49 6e 66 6f 3a 20 75 72 6c 3d ..RTP-Info: url=
72 74 73 70 3a 2f 2f 31 39 32 2e 31 36 38 2e 31 rtsp://192.168.1
32 38 2e 31 39 32 2f 73 74 72 65 61 6d 3d 32 35 28.192/stream=25
3b 73 65 71 3d 33 37 37 34 39 3b 72 74 70 74 69 ;seq=37749;rtpti
6d 65 3d 33 33 33 30 38 30 36 31 34 37 0d 0a 0d me=3330806147...
0a .

Client Request 3 >
54 45 41 52 44 4f 57 4e 20 72 74 73 70 3a 2f 2f TEARDOWN rtsp://
31 39 32 2e 31 36 38 2e 31 32 38 2e 31 39 32 3a 192.168.128.192:
35 35 34 2f 73 74 72 65 61 6d 3d 32 35 20 52 54 554/stream=25 RT
53 50 2f 31 2e 30 0d 0a 43 53 65 71 3a 33 0d 0a SP/1.0..CSeq:3..
53 65 73 73 69 6f 6e 3a 36 32 45 42 43 32 44 38 Session:62EBC2D8
0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 63 6c 6f ..Connection:clo
73 65 0d 0a 0d 0a se....
Server Response >
52 54 53 50 2f 31 2e 30 20 32 30 30 20 4f 4b 0d RTSP/1.0 200 OK.
0a 43 53 65 71 3a 20 33 0d 0a 44 61 74 65 3a 20 .CSeq: 3..Date:
53 61 74 2c 20 4a 61 6e 20 30 31 20 32 30 30 30 Sat, Jan 01 2000
20 30 31 3a 32 39 3a 34 39 20 47 4d 54 0d 0a 0d 01:29:49 GMT...
0a .
```

Appendix C: Support for DVB-T/T2 (optional)

Some satellite distribution solutions include the possibility of carrying terrestrial transmissions over the same cable infrastructure that carries satellite signals. In order to allow for a one-to-one porting of all of today's functionality into the SAT>IP environment, this appendix describes how the SAT>IP mechanism can be used in a mixed satellite terrestrial environment.

This section is fully optional and does not mandate the inclusion of DVB-T/T2 functionality into SAT>IP products. However if a manufacturer opts to include DVB-T/T2 into a SAT>IP product it shall be implemented as shown below in order to guarantee interoperability amongst products.

Name	Attribute	Range	Example
Frequency	freq	Channel centre frequency expressed in MHz as fixed point type or integer value.	freq=754 freq=191.5
Bandwidth	bw	Set to one of the following values: "5", "6", "7", "8", "40", "1.712"	bw=7
Modulation system	msys	Set to one of the following values: "dvbt", "dvbt2".	msys=dvbt
Transmission mode	tmode	Indicates the number of carriers in an OFDM frame Set to one of the following values: "2k", "4k", "8k", "1k", "16k", "32k"	tmode=4k
Modulation type	mtype	Set to one of the following values: "qpsk", "16qam", "64qam", "256qam"	mtype=64qam
Guard interval	gi	Set to one of the following values: "14", "18", "116", "132", "1128", "19128", "19256".	gi=18
FEC inner	fec	Set to one of the following values: "12", "35", "23", "34", "45", "56", "78".	fec=34
PLP	plp	Uniquely identifies a PLP. Numerical value between 0 and 255. Not required for DVB-T	plp=1
T2_system_id	t2id	Uniquely identifies the T2 system within the DVB network. Numerical value between 0 and 65535. Not required for DVB-T	t2id=8
SISO/MISO	sm	Indicates the SISO/MISO mode. "1" means MISO, "0" means SISO. Not required for DVB-T	sm=0
List of PIDs	pids	CSV list of PIDs: (Num. values betw. 0 and 8191) for spts streams, "all" for mpts streams, "none" for no demux output.	pids=0,16,201,302
	addpids	Opens new PID filters on the demux for streaming on the network. CSV list of PIDs.	addpids=307,309
	delpids	Removes PID filters from the demux. CSV list of PIDs.	delpids=201,302

Example DVB-T Query:

```
rtsp://192.168.128.5/?freq=754&bw=8&msys=dvbt&tmode=2k&mtype=64qam&gi=132&fec=23&pids=0,16,50,201,301
```

RTSP DESCRIBE

When supporting DVB-T/T2, the attribute syntax in the SDP implementation is extended as follows:

```

Session Level:

s=SatIPServer:1 <sat frontends>,<terr frontends>

Media level:

a=control:stream=<streamID>

a=fmtp:33 ver=1.1;tuner=<feID>,<level>,<lock>,<quality>,<freq>,<bw>,<msys>,<tmode>,<mtype>,<gi>,<fec>,<plp>,<t2id>,<sm>;pids=<pid0>,...,<pidn>

```

Please note that the version number is set to 1.1

<sat frontends> provides the number of satellite frontends available from the server

<terr frontends > provides the number of terrestrial frontends available on the server (DVB-T+DVB-T2)

RTCP APP Packet String Payload Format for DVB-T/T2:

Similarly to the SDP extension, the APP Packet payload format for RTCP Announcements is extended as follows:

```

ver=1.1;tuner=<feID>,<level>,<lock>,<quality>,<freq>,<bw>,<msys>,<tmode>,<mtype>,<gi>,<fec>,<plp>,<t2id>,<sm>;pids=<pid0>,...,<pidn>

```

Please note that the version number is set to 1.1.

Example Text String:

```

ver=1.1;tuner=1,240,1,7,754.00,8,dvbt,4k,16qam,116,23,,,;pids=0,16,56,112,168,1709

```

Appendix D: Support for DVB-C/C2 (optional)

This section is fully optional and does not mandate the inclusion of DVB-C/C2 functionality into SAT>IP products. However if a manufacturer opts to include DVB-C/C2 into a SAT>IP product it shall be implemented as shown below in order to guarantee interoperability amongst products.

Name	Attribute	Range	Example
Frequency	freq	Frequency expressed in MHz as fixed point type or integer value. DVB-C: Channel centre DVB-C2: System Tuning Frequency (see attribute c2tft)	freq=623.25
C2 Tuning Frequency Type	c2tft	"0": Data Slice tuning frequency "1": C2 System center frequency "2": Initial tuning position for a dependent static data slice DVB-C2 only	c2tft=0
Bandwidth	bw	Set to one of the following values: "6", "8" DVB-C2 only	bw=8
Modulation system	msys	Set to one of the following values: "dvbc", "dvbc2".	msys=dvbc
Modulation type	mtype	Set to one of the following values: "16qam", "32qam", "64qam", "128qam", "256qam" DVB-C only	mtype=64qam
Symbol rate	sr	Symbol rate value in kSymb/s. DVB-C only	sr=6900
ds	ds	Uniquely identifies a data slice Numerical value between 0 and 255. DVB-C2 only	ds=0
PLP	plp	Uniquely identifies a PLP. Numerical value between 0 and 255. DVB-C2 only	plp=1
Spectrum inversion	specinv	Spectrum inversion. Set to "0" (spectrum inversion off) or "1" (spectrum inversion on) DVB-C only	specinv=0
List of PIDs	pids	CSV list of PIDs: (Num. values betw. 0 and 8191) for spts streams, "all" for mpts streams, "none" for no demux output.	pids=0,16,201,302
	addpids	Opens new PID filters on the demux for streaming on the network. CSV list of PIDs.	addpids=307,309
	delpids	Removes PID filters from the demux. CSV list of PIDs.	delpids=201,302

Example DVB-C Query:

```
rtsp://192.168.128.5/?freq=623.25&msys=dvbc&mtype=256qam&sr=6900&specinv=0&pids=0,16,50,201,301
```

Example DVB-C2 Query:

```
rtsp://192.168.128.5/?freq=793.982&bw=8&msys=dvbc2&c2tft=0&ds=0&plp=0&pids=0,16,100,308,256
```

RTSP DESCRIBE

When supporting DVB-C/C2, the attribute syntax in the SDP implementation is extended as follows:

```

Session Level:

s=SatIPServer:1 <sat frontends>,<terr frontends>,<cable frontends>

Media level:

a=control:stream=<streamID>

a=fmtp:33 ver=1.2;tuner=<feID>,<level>,<lock>,<quality>,<freq>,<bw>,<msys>,<mtype>,<sr>,<c2tft>,<ds>,<plp>,<specinv>;pids=<pid0>,...,<pidn>

```

Please note that the version number is set to 1.2

<sat frontends> provides the number of satellite frontends available from the server
 <terr frontends > provides the number of terrestrial frontends available on the server (DVBT+DVBT2)
 <cable frontends > provides the number of cable frontends available on the server (DVBC+DVBC2)

RTCP APP Packet String Payload Format for DVB-C/C2:

Similarly to the SDP extension, the APP Packet payload format for RTCP Announcements is extended as follows:

```

ver=1.2;tuner=<feID>,<level>,<lock>,<quality>,<freq>,<bw>,<msys>,<mtype>,<sr>,<c2tft>,<ds>,<plp>,<specinv>;pids=<pid0>,...,<pidn>

```

Please note that the version number is set to 1.2

Example Text String:

```

ver=1.2;tuner=1,240,1,7,444.00,8,dvbc,256qam,6900,,,0;pids=0,16,56,112,168,1709

```

Appendix E: Unicast Only Profile

Under circumstances where only a limited number of tuners are available, e.g. in SmartTVs or STBs, it is possible to design simpler SAT>IP servers that do not support multicast functionality. The requirements for such basic Unicast Only SAT>IP servers are listed below:

Chapter	Unicast Only Profile
3 Protocol Specification	x
3.1 General	x
3.2 Addressing	x
3.2.1 DHCP	x
3.2.2 Auto-IP	x
3.3 Discovery	x
3.3.1 SSDP	x
3.3.2 Server Advertisements	x
3.3.3 DEVICE ID Negotiation	not required
3.3.4 Client Search Requests	x
3.4 Description	x
3.4.1 XML Device Description	x
3.5 Control	x
3.5.1 RTSP	x
3.5.2 Setting up a new session (SETUP)	Transport: unicast only
3.5.3 Starting the playout of a media stream (PLAY)	x
3.5.4 Maintaining a session (OPTIONS)	x
3.5.5 Modifying a media stream	x
3.5.6 Joining an existing stream	not required
3.5.7 Listing available media streams (DESCRIBE)	x
3.5.8 Closing the session and stopping the playout (TEARDOWN)	x
3.5.9 RTSP Method Summary	x
3.5.10 Uniform Resource Identifier URI	x
3.5.11 Query Syntax	x
3.5.12 Example RTSP Sequence Diagram	x
3.5.13 IGMP	not required
3.5.14 Status Code Definitions	x
3.5.15 RTCP Announcements	x
3.5.16 HTTP	not required
3.6 Media Transport	x
3.6.1 RTP Transport	x
3.6.2 HTTP Streaming	not required
3.7 Media Formats	x