3000 series control channel signaling

Version 1.3

2007/12/7

Vivotek may make changes to specifications and product descriptions at any time, without notice.

The following is trademarks of Vivotek Inc., and may be used to identify Vivotek products only: Vivotek. Other product and company names contained herein may be trademarks of their respective owners.

Revision History

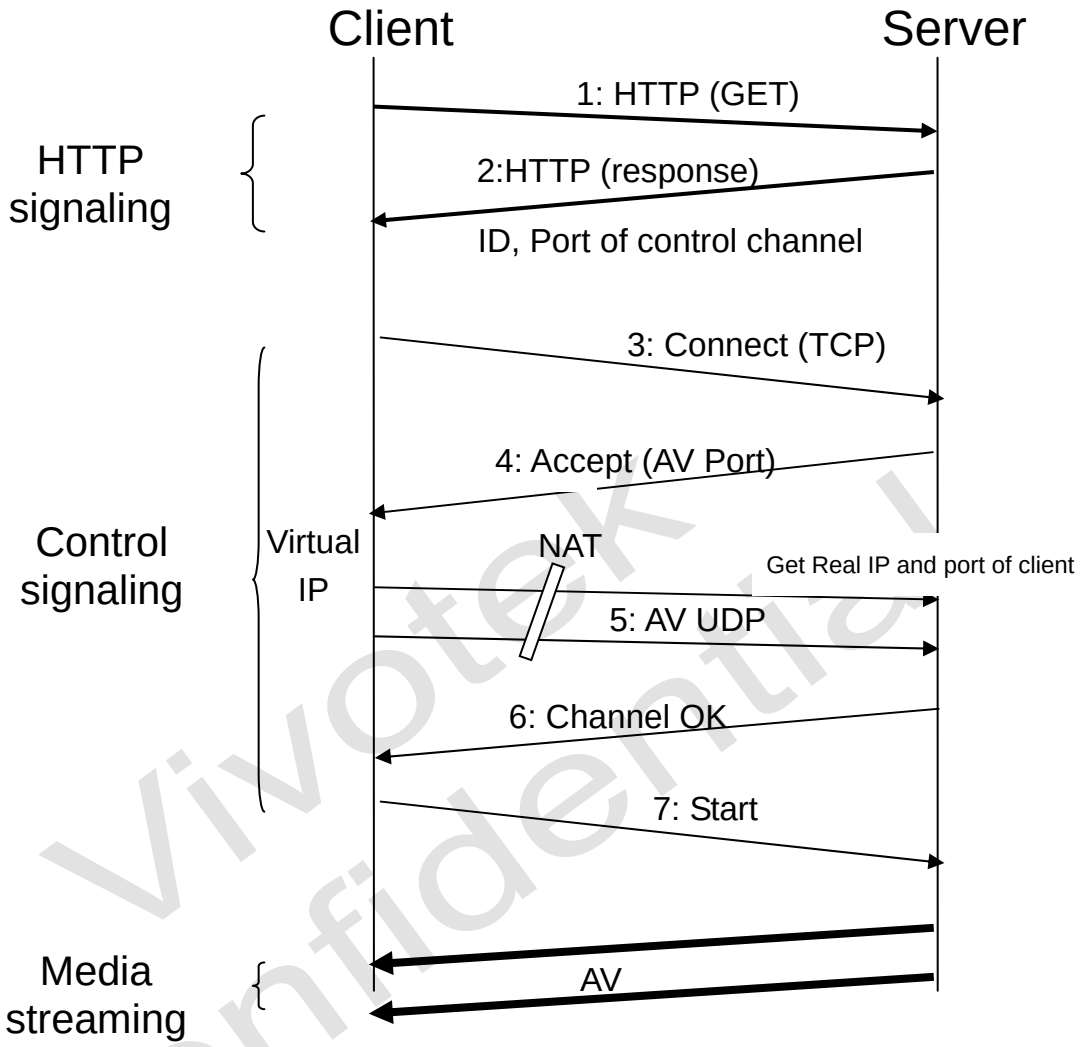
version	Issue date	author	comment
1.0	2005/06/01	ShengFu	
1.1	2007/09/05	Mei-Yun Hsu	Correct the meaning of length in media data format
1.2	2007/11/30	Mei-Yun Hsu	Correct the length of millisecond in media data format
1.3	2007/12/7	Mei-Yun Hsu	Correct the length and unit of millisecond in media data format

TABLE of CONTENTS

Overview

This document describes signaling sequence and format of the control messages of control channel of Vivotek 3000 series products. Control channel exchange the capabilities of codec between server and client as well as the transport information that tells how media data can be delivered via network.

UDP streaming for video and audio



STEP 1:

Direction: client -> server

Message type: HTTP GET method

Purpose:

1. use standard HTTP protocol to get main.html
2. server will return web page

Format:

Standard format of HTTP GET. See example below:

GET /main.html HTTP/1.1
Accept-Language: zh-tw
Accept-Encoding: gzip, deflate
Host: 192.168.1.118
Connection: Keep-Alive

STEP 2:

Direction: server->client

Message type: HTTP 200 OK reply

Purpose:

1. server will put connection ID in the web page as well as control channel listening port
2. Pass connection ID and listening port to control channel for further action

Format:

1. Standard HTML format with specific content below

```
document.write("<PARAM NAME=\"RemotePort\" VALUE=\"5001\">");
document.write("<PARAM NAME=\"RemoteID\" VALUE=\"11255\">");
```

2. 5001 is the port number of control channel. 11255 is the connection ID for control channel

STEP 3:

Direction: client -> server

Message type: CONNECT

Purpose:

1. Authenticate client to server by connection ID
2. Express client's decoder capability
3. Chose streaming protocol (UDP/TCP/HTTP). According to protocol client choose, server will know where and how to receive probe packets from client (see step 5)

Format

1. Format of CONNECT message (total 10 bytes)

Message Type	Connection ID	Streaming Protocol	Reserved	Codec capability	Reserved
8 bits	32 bits	8 bits	16 bits	8 bits	8 bits

Field name	Field length	Field description
Message Type	8 bits	0x01
Connection ID	32 bits	Four byte random number acquired by HTTP GET message. This is for client authentication. The connection ID is Little Endian order
Streaming protocol	8 bits	1. 0x50…client ask video/audio as UDP packets 2. 0x51…client ask video/audio as TCP packets 3. 0x52…client ask video only and streaming via HTTP
Codec capability	8 bits	See below

2. Format of codec capability

Reserved	H.263	Reserved	G.723	G.7221
Bit 1~3	Bit 4	Bit 5~6	Bit 7	Bit 8

STEP 4:

Direction: server->client

Message type: ACCPET

Purpose:

1. Express which codec server decide to use
2. Express which video and audio local port server decides to bind. This is for NAT traverse. Client will send out each probe packet for video and audio to these ports where server will try to receive. Once server receive these probe packets. Server will know where to send media data without blocked by NAT.

Format

1. Format of ACCEPT message (total 10 bytes)

Message Type	Connection ID	Audio port of server	Video port of server	Codec capability
8 bits	32 bits	16 bits	16 bits	8 bits

Field name	Field length	Field description
Message Type	8 bits	0x02
Connection ID	32 bits	The same connection ID as CONNECT message

		The connection ID is Little Endian order
Audio port of server side	16 bits	The audio port number server will bind and try to receive the probe packet. This value is Little Endian
Video port of server side	16 bits	The video port number server will bind and try to receive the probe packet. This value is Little Endian
Codec capability	8 bits	See below

2. Format of codec capability

HTTP only	Reserved	H.263	Improved audio quality	Reserved	G.723	G.7221
Bit 1	Bit 2~3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8

Note1: HTTP only means server will chose to send video only via HTTP mode

Note2: Improved audio quality means Server will send 10 audio samples in one packet. This would improve audio quality but less real time. Recommend enable this feature in the internet environment

STEP 5:

Direction: client->server

Message type: PROBE

Purpose:

1. Traverse NAT. Client will know where to send probe packets by parsing the ACCEPT message. The source port client used to send probe packets would be the port for receiving media data.
2. Client will try UDP first. Server will reply CHANNEL OK message to notify probing is OK. If Server reply CONTROL_CHANGE_TO_TCP message, client needs to try TCP. If Server reply CONTROL_CHANGE_TO_HTTP message. Control channel will abort signaling and let HTTP client take over it.
3. Basically client will try from UDP->TCP->HTTP to check with server the feasibility of transportation of network. However If server notify that server will stream video only via HTTP mode (by ACCEPT message). Client should skip everything and abort control channel to let HTTP client take over HTTP mode streaming

Format (total 10 bytes)

Message Type	Connection ID	Reserved
8 bits	32 bits	40 bits

Field name	Field length	Field description
Message Type	8 bits	0x08
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order

STEP 6: (if UDP or TCP probe packets succeed)

Direction: server->client

Message type: CHANNEL OK

Purpose: notify client that server received the probe packets and is ready to stream media data

Format (total 10 bytes)

Message Type	Connection ID	Reserved
8 bits	32 bits	40 bits

Field name	Field length	Field description
Message Type	8 bits	0x03
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order

STEP 7:

Direction: client->server

Message type: start streaming

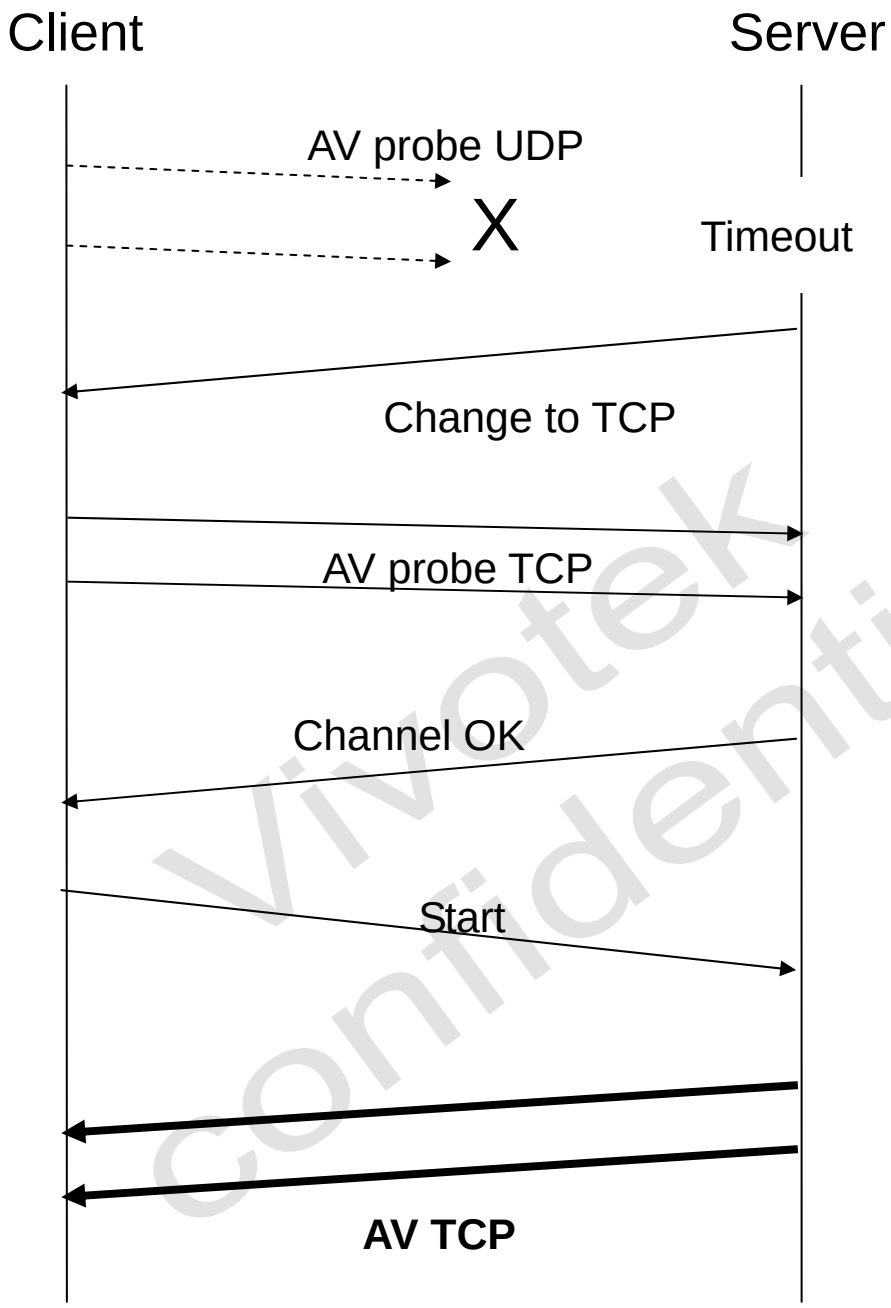
Purpose: request streaming

Format (total 10 bytes)

Message Type	Connection ID	Reserved
8 bits	32 bits	40 bits

Field name	Field length	Field description
Message Type	8 bits	0x06
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order

TCP streaming for video and audio



If UDP probe packets didn't reach server for some reason (NAT blocking), server will send a CONTROL_CHANGE_TO_TCP message to notify client to switch protocol after timeout. Client can set protocol option in CONNECT message to use TCP streaming directly

Direction: server->client

Message type: CONTROL_CHANGE_TO_TCP

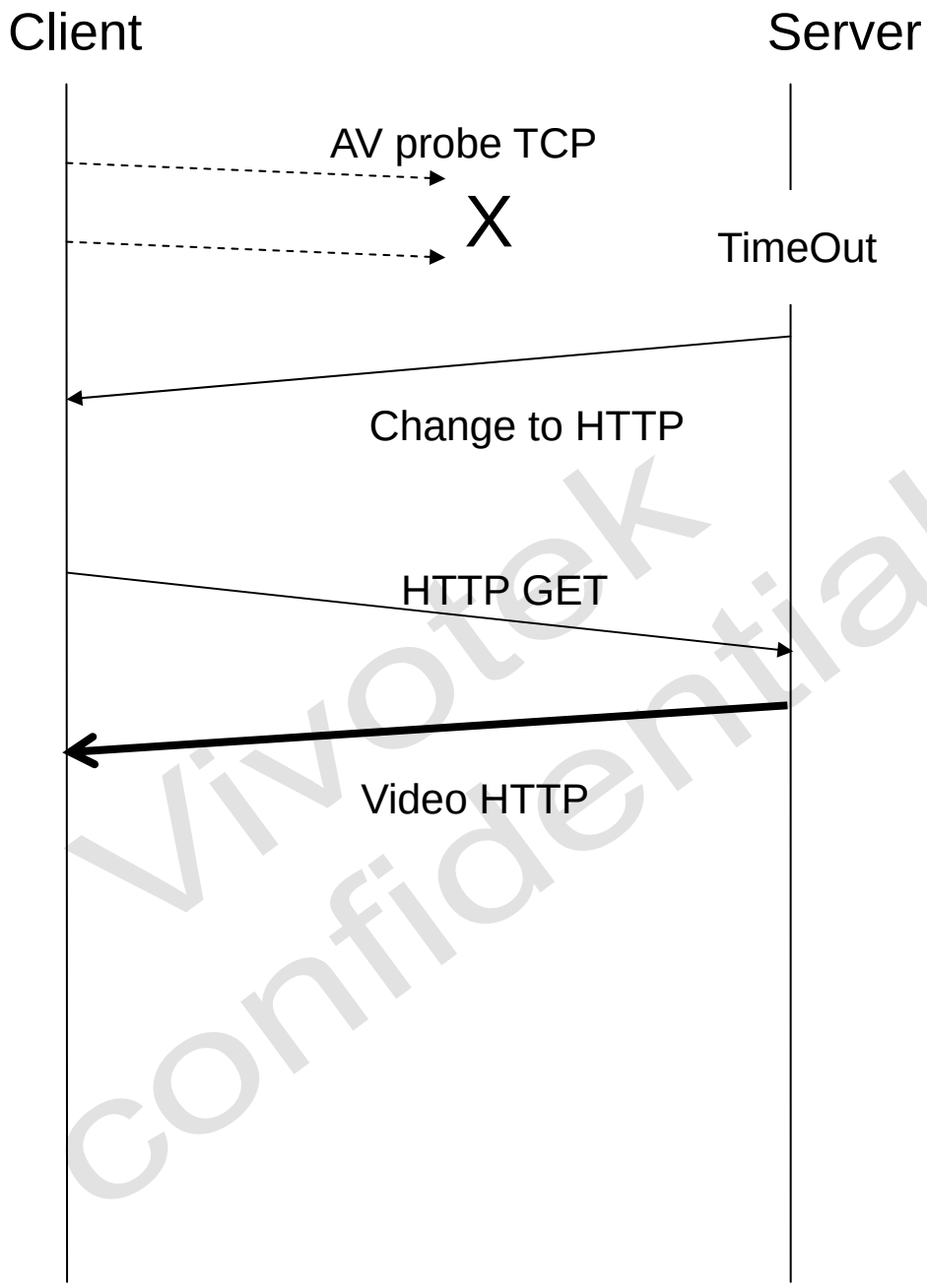
Purpose: notify client that server fail to receiver UDP probe packets and ready to take TCP probe packets

Format (total 10 bytes)

Message Type	Connection ID	Reserved
8 bits	32 bits	40 bits

Field name	Field length	Field description
Message Type	8 bits	0x04
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order

HTTP streaming for video only



If TCP probe packets didn't reach server for some reason (NAT blocking), server will send a CONTROL_CHANGE_TO_HTTP message to notify client to switch protocol after timeout. Client can set protocol option in CONNECT message to use HTTP streaming directly.

Direction: server->client

Message type: CONTROL_CHANGE_TO_HTTP

Purpose: notify client that server fail to receiver TCP probe packets and ask client to

switch to HTTP mode

Format (total 10 bytes)

Message Type	Connection ID	Reserved
8 bits	32 bits	40 bits

Field name	Field length	Field description
Message Type	8 bits	0x05
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order

During streaming

After signaling is completed and streaming begins, Client needs to send CONTROL_COMMAND_KEEP_ALIVE message every 15 seconds to keep notify server its availability.

Direction: client->server

Message type: CONTROL_COMMAND_KEEP_ALIVE

Purpose: (1) notify server to client is alive

(2) Client can fill specific value in this message to notify server to send out I frame immediately

Format (total 10 bytes)

Message Type	Connection ID	Force I frame	Reserved
8 bits	32 bits	8 bits	32 bits

Field name	Field length	Field description
Message Type	8 bits	0x0a
Connection ID	32 bits	The same connection ID as CONNECT message The connection ID is Little Endian order
Force I frame	8 bits	0x09 If client want a I frame immediately

Media data format

Media data can be divided as two parts: header and payload. As regarding to payload, please refer to 3000_videostream_userdata_format.pdf for detail information. See below for detail of header format

Media data header format (total 8 Bytes)

Time stamp	Sequence number
6 Bytes	2 Bytes

Field name	Field length	Field description
Reserved	1 bit	
Millisecond	7 bits	0~127, the unit is 10 ms
Reserved	2 bits	
Second	6 bits	0~60
Year	12 bits	0~4095 (little endian)
Month	4 bits	1~12
Date	5 bits	1~31
Hour	5 bits	1~24
Minutes	6 bits	1~60
Total bits	48 bits	6 bytes of timestamp

Note1: Sequence number is stored in little endian

Note2: For TCP transportation of media data, there will be 2 bytes of length information ahead of media data. For example: the first packet of video is 1300 bytes including header. Client will receive 2 byte data first which is 1302 in little endian, then comes the 1300 video packet which contains 8 bytes header and 1292 bytes video payload. This way client can always know how many bytes to receive for next media packet since TCP comes in bit stream instead of packets in UDP.