Brandon Vo

New York University

December 13, 2021

RFC 8888: Video-Chat Streaming Protocol

Status of this Memo

This memo specifies the local standards defined for the Video-Chat Streaming Protocol service for the internet community. This protocol is up for discussion and requests suggestions for improvements. Distribution of this memo is unlimited.

1. Introduction

The objective of VCSP is to 1) facilitate and support an online streaming environment, 2) connect one or more user(s) to an environment that allows the use of remote computers to communicate amongst each other, 3) provide a basic level of security and efficiency to authenticate the identities of all users using this protocol.

2. Abstract

This memo describes the process used between two or more hosts who wish to form interconnected groups. The method proposed will allow any number of hosts to construct a set of underlying infrastructure of which would allow any number of people to communicate between one or all members of their respective groups provided the hardware, network, and infrastructure can support such actions.

This memo assumes knowledge of the following protocols:

Transmission Control Protocol (RFC 793)

User Datagram Protocol (RFC 768)

SIP Protocol (RFC 3261)

IPv4 Protocol (RFC 791)

IPv6 Protocol (RFC 2460)

3. Video-Chat Streaming Protocol

3.1 The VSCP Model

Provided below is a diagram of the Master-Slave model used by VCSP

```
          --------
          |Master|
          |Server|
          --------

             ^

        /    |    \

       /     |      \

     /       |       \

--------   --------   --------
|User  |   |User  |   |User  |
|Client|   |Client|   |Client|
--------   --------   --------
```

Figure 1 model for VSCP's Infrastructure

In the model shown above, one machine will establish itself as the master server which the other user clients will connect to. This connection will initiate and sustain its connection using UDP.

Operation:

As mentioned previously, the goal of VCSP is to provide a basic layer of infrastructure needed to facilitate an online session of communication among two or more parties. In order to establish this infrastructure, this system will have to fulfill the following list of requirements:

- Established network
- Basic level of authentication of messaging history
- Message reliability from sender to host

For the entirety of this memo, it will be assumed that all processes of communication will occur within one group with one or more clients connecting to one server for this scenario. It will be assumed all clients will be able to establish a connection with more than one group being maintained by their respective, independent servers. As such, it will be assumed that any server can create and maintain more than one chat sessions without interference between any two chat groups overlapping.

4. Communications through text

4.1 Group messaging

Communications through text will begin and behave differently from communications through online VOIP-based communications. As a result, online voice-based communications and communications through text will be based off different procedures.

All messages will be done using UDP as the server will have to message more than one user. Server-to-client communications will be performed through multicasting which is not possible on TCP's end to end based communications.

For this process of communication, it will be assumed that the client already knows the host's IP Address through any means provided to the user. In addition, all users will host and initiate their socket connections on port 50429 as their default ports. However, the host should be able to advertise and connect to other clients using any port number if specified by the user.

4.2 VSCP Request

When a client wants to start a connection with an endpoint server, the client will create a VSCP Request packet embedded within a UDP packet.

The VSCP request packet is a packet that contains the version number of VSCP protocol being used, the IP Address of the server, the port number to be used, 0xFF for the challenge number, a fragmentation offset of 0x00, and a flag containing 0x1F to identify this packet as the VSCP Request packet. This packet will be embedded in a UDP packet and sent towards the server.

Because this message will be the client message's first message sent to a server, the client will have to generate a random number and use that number as the client's sequence number. This sequence number is randomly generated to prevent a malicious party from using a remote Idle Scan attack to probe the network for the IP Addresses of the other members in the local chat session.

```
+--------------------------------+
|       VERSION NUMBER (4)       |
+--------------------------------+
|      Source IP ADDRESS (20)    |
+--------------------------------+
|   Destination IP Address (20)  |
+--------------------------------+
|     Shared Port Number (2)     |
+--------------------------------+
|     SEQUENCE NUMBER (4)        |
+--------------------------------+
|      CHALLENGE NUMBER (4)      |
+--------------------------------+
|       FLAG HEADER (2)          |
+--------------------------------+
|     FRAGMENTATION OFFSET (2)   |
+--------------------------------+
|         DATA (64)              |
+--------------------------------+
```

Figure 2 Model for VSCP’s header format

For the ASCII diagram above, the parenthesis represents the number of bytes granted to each part of the VSCP header. In the case of IP Address, the size may be 8 bytes or 20 bytes for IPv4 (RFC 2460) and IPv6 (RFC 2460) addresses. As a result, the payload size for each packet will be 122 bytes.

4.3 VSCP Reply

In this version of VSCP, the server will be designed to automatically respond to and accept an incoming VSCP Request messages unless specified otherwise or if the server is unable to establish a compatible connection with the client. If the server does not have a matching version number and/or IP Address with the specifications provided by the client, the server will automatically drop or reject the packet based on the server host’s preferences.

If the server owner chooses to reject packets, the server will respond with a VSCP reject packet. The packet will contain the flag 0xC9 to indicate a rejection flag, and the server will embed a message into the packet detailing the reason to reject the message, if the server owner chooses to provide any reasons for a rejection.

If the server owner instead opts to drop any incoming packets, the server will discard the packet and will not reply to any incompatible request packets.

If the request message matches the server and shares the same version number of VSCP, the server will create its own VSCP Response packet that contains the server's credentials and details needed to connect to the server. This is to inform the client about which server it is being connected to and to make sure if the client is connecting to the intended destination. The VSCP Response packet will also contain a flag of 0x2F to indicate that this packet is a response packet.

The server's VSCP Response packet will contain the following: Version Number of VSCP protocol that the server is using, IP Address of the server, and Port Number being used by the server. The server will also generate a random set of numbers to identify the client's new challenge number and associate that challenge number with the client.

The version number will identify what version of the VSCP protocol to utilize to maintain a compatible connection with the client. The IP Address and the port number will help the client verify what server to connect and through which socket port number.

Once the client has established a two-way connection with the server, the client will be officially recognized as a member of the chat session. As such, the client will be recognized as part of the same group of IP Addresses that the server will send its messages towards.

5.   Public Messaging

5.1 Terminology

Multicast

When a message is being multicasted from the client, the message will be sent to all recipients that share the same group with the sender. Any hosts that are on the same network but not in the same group will not receive the message.

5.2 Client-Server Communications

This stage will assume that the server is operational and has atleast one client connected to the host, and the client and server can communicate with each other. Such communications would not be possible if the host does not have any other receivers to speak to. This section will further detail how two or more clients will communicate with each other by using the server.

All further communications will be initiated by the client. Whenever the client wants to send a message to the chat group, the client will have to send a message to the server. This is because the client is not supposed know the IP Address of the other members in the same session, so all messages must be directed through the server which will hold a connection to every member in the group.

When a server sends a message to the group, the message will be embedded in a VSCP message containing the same header format from the previous VSCP packets, but the flag for this packet will be marked with 0x2E to identify this message as a VSCP SYN message. The message itself will be contained in the data section of the packet up to the allotted 64 bytes.

The client will use the previously generated sequence number used when the client requested to join the host's chat session. For every message sent by the client, the sequence number will be incremented by one to inform the server regarding how to rearrange the packets to be in-order.

5.3 Server-Client Multicasting

When the server receives the message, it will check the sequence number to see if this is the next message expected to be received from the sender. When it receives the message, the server will wait for approximately 100ms before multicasting the message to the rest of the members. This allows the server to send bulk messages all at once and check if the message being sent is a fragment of several more packets in transit.

When the wait time has expired and there are no more incoming messages, the server will create a VSCP MSG packet. The header of this packet will use the multicast IP Address to send to every member in the group, a flag containing 0x3F to identify this message as a message packet from the server, and the same sequence number received from the original sender.

However, because multicasting means the server will not be sending this message to every member at a time, the challenge number will not be useable in determining the legitimacy of a conversation. The server will have to mark the challenge number as 0xFFFF to mark the challenge number as empty due to this limitation.

Once the packet has been constructed, it will be multicasted to the remaining members of the chat instance who will read the message once received.

For this version of VSCP, the server will not verify if all packets sent out by the server have been received by all members in the group due to the limitations of using UDP protocol and multicasting. Only the server will verify if it has received the message, and if the server notices that a message received does not have the expected sequence number, then it will send an error message to the sender, informing them that the message has failed to be sent.

If the server notices that the challenge number does not match the associated challenge number assigned to that particular sender, then the server will drop the packet and ignore the contents of that message. The server will not reply with an error message because the sender's identity could not be verified.

6. Private Messaging

6.1 Terminology

DM

DM is the abbreviated message of direct message. Direct messaging refers to a private exchange of messages between specifically two people.

DoS

6.2 Establishing a private channel

VSCP will also be able to be repurposed to allow direct messaging between two users which will not require the assistance of a server. This step is independent of the group messaging service, and two clients that wish to communicate will not require a server to send DMs.

Users using private messaging will skip all steps previously used by a client to join a server's group. Neither clients will send a request to authenticate each other until the first message is sent and received by both parties.

Because clients should not be expecting to receive private messages, all VSCP direct messages will use UDP as it cannot be reasonably expected for two hosts to maintain a separate connection for a sporadic exchange of messages. It will also reduce the effectiveness of malicious parties using direct messages as a form of DoS attack against a particular user.

This step will assume that the sender of the private message already knows the receiver's IP Address.

If one user wishes to send a private message to another user, the user will create a VSCP Direct Message packet. The packet will use the flag of 0xD3 to indicate that this message is a direct message, and the user will randomly generate its own sequence number if this is the user’s first message to a particular receiver. If this message is intended to be the first message, the challenge number will be marked as 0xFFFF until the receiver returns a message to the sender. The user will also embed its own message in the packet’s data header before sending the message to another client.

When another client receives a message, the client will check the header to see the IP Address of the sender and check the challenge number and sequence number to see if this message was part of a conversation previously held by the receiver.

6.3 Responding to a DM message

If the receiver identifies this message as part of a previously held conversation, then the message will be read and recorded as part of the exchange. However, the receiver must recognize both the challenge number and the IP Address to accept this message. If one part of the header is missing or wrong, then the receiver will drop the message and not respond.

If the receiver does not recognize the sender’s IP Address and the challenge number is marked as 0xFFFF, then the receiver will create the VSCP Silent Message packet directed at the sender.

The VSCP Silent Message packet will use 0x2A as its flag to indicate that this is a silent message to be read by the client but not by the user. This message will copy the sender’s IP Address and sequence number to match the sender’s credentials. The challenge number, on the other hand, will be a randomly generated number to associate this DM instance with a form of identity. Furthermore, the data field will be empty as this message will not be read by the user.

Once the user receives the VSCP Silent Message packet, it will now associate any private messages towards that user with challenge number received. At this point, both users will now be able to recognize this specific conversation.

All messages sent from this point will use the VSCP Direct Message packet. Both the sender and the receiver will use the same challenge number and the same sequence number for their messages where every message sent from either party will increment the sequence number with every message sent.

7. Exiting a session

7.1 Terminology

FIN

FIN being an abbreviation of finish. A FIN packet refers to a packet that is used to indicate the end of a conversation where both the sender and receiver will terminate their connection.

7.2 Termination Request

While it is primarily the server's responsibility to monitor and maintain its conversation with every member in the chat group, all conversations will end only if a client wishes to leave the group.

Because neither the sender nor the receiver of DM conversation is actively upholding their respective conversations, a FIN message will do nothing if sent in a private conversation. The challenge number used for this private conversation will not be erased by either parties as well.

If the client wishes to leave either the video call or the text channel, the client will create a VSCP FIN packet. This packet will be directed towards the IP Address of the server maintaining the chat group of which the client wishes to leave.

The client will include its previously provided sequence number and challenge number as part of its FIN packet, and the client will include a flag of 0xFF to identify this packet as a FIN packet. The data and the fragmentation offset fields will be empty as there is no extra information required for a FIN packet.

7.3 Termination Response

Once the client sends its FIN packet to the server, the server will check the VSCP header to see if it matches the same credentials previously used by the client. If the credentials do not match, the server will perceive the packet to be a spoofed packet possibly sent by a malicious party and drop the packet.

If the credentials match the client's previous history of messages, then the server will create a VSCP Response message using the same challenge number, sequence number, and VSCP Response flag. This message is returned to the sender who will now effectively disconnect itself from the chat session by no longer sending any messages to the server. If the client was part of the session's voice call, then the client will no longer send and no longer read any messages sent to and from this instance until the client wishes to rejoin the server's session in the future.

The server, after sending the VSCP Response message will erase the client from the server's member list by removing the challenge number previously associated from the client. Any messages received by the server from this point on will no longer be sent towards any clients who have disconnected from the group.

7.4 Leaving a voice call

If a client wishes to end its voice call with the server but wants to remain in the server’s text-based chat group, then the client will create a VSCP FIN message similarly to the previous steps, but the client will instead use a flag of 0xEF to indicate that the client wishes to leave only the voice call.

When the server receives this modified VSCP FIN message, the server will check if the client was part of the server’s voice call by checking the challenge number and the client’s IP Address. The sequence number cannot be checked because all media streaming packets do not include sequence numbers for referencing.

If the server could not identify the client, then this packet is dropped as the server cannot remove a client that is not part of the video/voice call.

If the server correctly identifies the client, then the server will simply remove the client’s challenge number used for the video/audio call. The server will respond with a VSCP Response packet using the same format but with a flag of 0xEF as a response to the client’s request to leave the voice call.

Because the client will use a challenge number that is separate from the challenge number used for text communications, the client will remain as a member of the server’s group and will still be able to receive and send messages.

When the client receives this message, the client will disassociate from the server’s voice call by no longer reading or sending media streaming packets related to this server’s voice session. Any packets received from this point will be discarded until the client rejoins the server’s call.

8. VSCP Header utilities

8.1 Fragmentation Offset

All messages sent between a host and server will have a maximum size of 128 bytes. If a client/server intends to send a message larger than the maximum size, then the machine will break up the message into multiple parts and send them individually. Each message will mark an increment on the Fragmentation Offset header to indicate which part of the entire message is held in this individual fragment.

In addition to incrementing the fragmentation offset, the message will also increment the sequence number to indicate an additional message being sent to the server. All fragments will be incremented by 1; however, the final fragment will be marked as 0xFF to indicate that it is the final message being sent. When the entire message has been broken down in to fragments, each packet will be sent to the client which will reassemble the message based on the fragmentation number and the sequence number.

However, if the receiver notices that a fragment is missing due to the sequence numbers not matching the fragmentation offsets or if it never receives the final fragmented message, then the receiver will send a message to the sender, requesting the sender to resend the missing fragment.

For the duration of this exchange, the message will be considered lost until the message has been reassembled. As such, if the server is unable to receive the entire message, then the server will not distribute this message to the other clients of the chat session until the server is able to read the entire message.

8.2 Challenge Number

When sending messages between the client and server, a challenge number is used to as a basic level of authentication and integrity checking for the client messaging. In order to deter a malicious party from utilizing replay attacks or spoofing messages, a challenge number will be included which uniquely identifies the sender from one specific set of conversations.

If the client is speaking in the session for the first time, the client's messaging history will not be known by the server even if the server recognizes the client as a member. The client will send its first packet with the headers directed towards the host, but the message will also contain a challenge number of 0xFF to indicate that this is the client's first time speaking.

Whenever a server receives a packet of 0xFF from a client, the server will check its registry to see if the server has already assigned the client a challenge number before. If a corresponding challenge number was not found on the server's registry listings, then the server will randomly generate a new challenge number. The server will return an ACK containing the randomly generated challenge number in the packet's header, assigning the client a new challenge number and store that challenge number in its registry regarding users in its online chat session.

This, however, will lead to an issue that malicious third parties may spoof messages that contain a challenge number of 0xFF to attempt to hijack a member's identification in the session. As a result, the server will automatically reject any messages from a client whose packets do not have a challenge number which matches the number registered on the server's listings, including 0xFF challenge numbers.

The server will not remember the challenge numbers of any clients that are not part of the server's chat session. Whenever a client disconnects from the online chat session, his/her challenge number will be released from the server’s registry and be available to be used for another member’s messages. As a result, if a client disconnects and reconnects to the server, the client may be provided with a new randomly generated challenge number. As such, if the client attempts to communicate using its previously generated challenge number, those messages will be discarded unless the client uses the challenge number provided to it by the server at that given time.

9. Audio and Video Calling

8.1 Terminology

Codec

A host will use their own codec software to encode any audio or video input into a data stream that can be compressed and read by a computer. Similarly, any computer can use codec software to decode a data stream and convert it into an output for a user to listen to or watch.

8.2 Requesting to join a server’s voice call

This section assumes that the reader knows about the implementation of the SIP Protocol (RFC 3261). This memo will also use MPEG-2 (RFC 2250) for reference regarding header formatting.

This step requires that the client must first be a member of a server’s online chat session before being able to initiate an online call.

The server will drop all voice call requests from senders that are not part of the server’s list of participants. As a result, this means that all clients must join the server’s text-based communications instance in order to join the server’s voice call. This also means that private voice calls through DMs will not be possible for this version of VSCP. All clients will automatically drop any requests to join a voice call sent by other clients.

After a host has been connected to, a client may send a request to join the host’s online session by sending a VSCP packet that has a header of 0x25 to indicate that this client wants to join the server’s voice call.

Because this form of communication requires voice communication and, as such, may require special codec software, the packet’s data field will contain a list of audio codecs that the client has and can use to encode and decode audio streaming data.

8.3 Server Invitation

In order to set up a connection between the client and host, SIP will be used and will be send from the server. The host, upon reading the client’s request message, will either accept or reject the client’s session based on the host’s intentions. If the server does not have a matching compatible codec with the client, the server will reject the message and respond with an error message with a list of codec software that the server is using. If the server is compatible with the client and chooses to accept the client, the host may respond by forging a SIP Invite message and sending the invitation to the sender.

Once the client receives a SIP Invite message, the client will check the details in the request packet to see if the invitation message matches the credentials of the server that the client is trying to connect to. If the credentials do not match, the client will drop the packet. This is to avoid the possibility of a man-in-the-middle attack.

If the credentials match the intended server, the client will send a 200 OK SIP Reply message to the server, indicating that the client is ready to join the server’s voice call. This packet will be written in UDP with the IP Address of the client and use the default SIP port of 5060 when sending an invitation message.

In the SIP Invite packet, the server will also include the port number that the server intends to use to handle all audio and video communications. This port number will be randomly generated unless specified by the server, and the port number will be the same port number being used for the server’s text-based communications instance. The server will also generate a new challenge number different from the challenge number used by the client for text communication. This is so that the client’s method of communicating through text and communicating through audio and video remain independent of one another.

Once the client receives the message, the client will return a SIP ACK message to inform the server that the client is ready to begin the session. Once both hosts are prepared, the client will be considered part of the server’s video call and be allowed to send its own data to the server.

8.4 Media Streaming Communications

For audio and video calls, the client will continue directing messages towards the server to communicate with any other participants.

Clients will communicate in the server’s voice call by encoding all audio and video input into a data stream. This data stream is processed and compressed by the host’s codec software which would get embedded into a VSCP packet and transmitted to the server. However, due to the limitations of having to transmit consistently large payloads of data over a network, the client will create a modified VSCP packet.

```
+--------------------------------+
|       VERSION NUMBER (4)       |
+--------------------------------+
|      Source IP ADDRESS (20)    |
+--------------------------------+
|   Destination IP Address (20)  |
+--------------------------------+
|     Shared Port Number (2)     |
+--------------------------------+
|      CHALLENGE NUMBER (4)      |
+--------------------------------+
|       FLAG HEADER (2)          |
+--------------------------------+
|         DATA (188)             |
+--------------------------------+
```

Figure 3 Model for VSCP's header format when transmitting audio and video data

The clients will use a modified VSCP header with a flag of 0x3F to indicate that this packet is related to audio and video data streaming for the server and other members of the voice call to read. The client will use the same challenge number provided by the server when the client had requested to join the server's online chat session and will be verified by the server.

The server will not be expecting to receive every packet in order and will not reassemble any messages related to media streaming. As a result, the sequence number and fragmentation offset packet serve no purpose and will not be used this form of communication. Unlike the procedures for text-based messages, the server will not wait upon receiving any audio or video packets and instead immediately distribute all media streaming packets to every member of the voice call through multicasting.

Furthermore, media streaming data is much larger than packets holding chat messages, so this current version of VSCP will expand the maximum payload size to 240 bytes. This is to maintain compatibility with MPEG-2 Transport Stream's fixed size of 188 bytes

(RFC 2250). As a result, the payload size of this modified VSCP packet will be 228 bytes.

When encoding audio and video stream data, the client will record its input until the maximum buffer size granted by the client’s codec software is reached. Upon reaching the maximum size, the client will encode the input into data and send it out immediately. The client will not use fragmentation, and as a result, any excess data will be discarded if it cannot be formatted into the VSCP packet.

VCSP Functions

When Video-Chat Streaming Protocol is in service, the host server and client(s) will be granted access to a limited selection of commands based on whether they are recognized as a member of the chat session or the server owner of the session. Clients will send a message containing these commands to the server which will interpret these commands and provide info if given proper authorization.

For the following commands, the label of the command is provided alongside the keyword used to initiate the command.

VCSP Commands

It will be assumed that, unless specified otherwise, the server will be identified as the administrator of its own online chat instance, and as such, the server will be granted access to all private information accessible only to administrators. Clients, unless specified otherwise, will only have access to information that matches their level of privilege.

Connect to a server (INIT)

This command, when provided with an IP Address of a server to connect to, will send a VSCP Request packet to the target server. If successful, the client will receive a VSCP Response packet that informs the client that it can connect to the target server.

List All Users (LIST)

This command will be initially sent out by the user shortly after he/she has successfully joined the host's online session. This command will request the server to reply with a list of all users who are currently in the host's online session, including the client that is sending this message. This message will provide only a list usernames but not their respective IP Addresses unless proper permission has been granted to the user.

Direct message another user (DM | WHISPER | TELL)

When provided with the name or IP Address of another user, the user will forge a VSCP Direct Message packet to send to the other user, effectively putting the sender and the receiver into their own private session. This process is described the private messaging section.

Forcibly disconnect a client (KICK)

When provided with this command along with the identity of the target client machine, the server will send a VSCP disconnect message containing the sender's IP Address, Port Number, Challenge Number, and a flag of 0x2D to inform the client that it has been kicked out of the server’s current session. After sending the message, the server

Return Server Info (INFO)

This comment may be sent out by any users of any permission level. The server will reply with a message detailing the server that the user is currently connected to. Such details include but not limited to the server’s IP Address, server’s name, the server’s owner, and the number of users currently connected to the server. Further details can be removed or added by the host if they wish to do so.

Ping Server (PING)

The client will send an ICMP Ping Request packet to the user. The server will respond with an ICMP Ping Reply packet detailing the user’s connection status and the latency of sending and replying to the server in milliseconds.

Join Voice Chat (VC-INIT)

This command will prompt the client’s machine to request the server to join the chat session’s voice chat. After inputting this command, the client machine will undergo the procedures detailed in VSCP Video-Chat and connect to the server’s voice call if successful.

Leave Voice Chat (VC-LEAVE)

If the client is currently in the server’s voice chat session, this command will have the client send a FIN termination packet directed to the server’s voice call socket. This will have the client immediately stop its connection to the server regarding the voice call.

As the protocol used for the server’s voice call and the protocol used for the server’s chat session are independent procedures, ending the connection to the server’s voice call will not disrupt the client’s connection to the server’s chat session.

Restart Voice Chat (VC-RESTART)

This command will have the client emulate a restart procedure in its voice call by calling VC-LEAVE and VC-INIT sequentially to disconnect and reconnect to the server’s voice call.

Restart Connection (RES)

This will prompt the client to fully restart its connection to the server by sending a VSCP FIN packet to the server to disrupt its own connection, then establish its connection again by sending a VSCP Request packet to the same server.

If the client was connected to the server’s voice call, the client will utilize the VC-LEAVE command before terminating its connection to the server entirely. After reconnecting to the server, the client will use VC-INIT to reconnect to the server’s voice call.

Because this restart procedure will terminate the client’s connection, the client may end up with a new challenge number after restoring its connection to the server. This is because the server will not remember the challenge numbers of any clients that choose to leave the chat session.

Terminate Connection (QUIT)

The client will send a FIN packet to the server containing the client’s name, IP Address, challenge number, and connection ID to the server, informing the host that the client will be leaving the chat session and will be closing the socket connection being held with the server. In order to make sure that it was the client that sent the message and not a malicious party, the server will check the packet to see if the info provided by the message matches the TCP header information sent by the client.

If the IP Address, challenge number, and connection ID match the connection with the specified client, then the server will drop the socket connection. The server will not respond with an ACK message but instead, the server will immediately close the socket shared with the client which sent the message.