The Mechanism of Tencent QQ Video Communication

Xiaoxian Chen
School of Computer Science and Technology
Harbin University of Science and Technology, Harbin, China
xiaoxianchen73@gmail.com

Zhenlong Yuan
Department of Automation,
Tsinghua University,
Beijing, China
yuanzl11@mails.tsinghua.edu.cn

Yibo Xue *
Tsinghua National Laboratory for Information Science and Technology,
Beijing, 100084, China
yiboxue@tsinghua.edu.cn

***Abstract*— The supernormal growth of QQ has outdistanced all the expectations and the number of users has reached up to 780 million till the end of 2012. At the same time, QQ causes several intractable issues, such as generating huge traffic, especially for the great video traffic. However, owing to proprietary protocol and encrypted traffic, the QQ video mechanism is little to be known and very hard to be identified. In this paper, we reveal the QQ video mechanism and its protocol structure, and introduce its communication process in details. To the best of our knowledge, this is the first time to discover the QQ video mechanism. This will be very useful for optimizing network bandwidth and improving quality of service.**

Keywords-Tencent QQ; Video Communication Mechanism; Media transport protocol; Video codec.

I. INTRODUCTION

VoIP has been going through a rapid development in the recent years due to its low cost, high quality, and flexibility. More and more users switch to VoIP from their traditional phones. QQ video communication is one of the most popular and representative VoIP application in China. According to the statistics in 2012, the total number of QQ users has reached up to 780 million and the largest number of QQ online users is more than 176 million [1]. QQ video communication can provide a real-time face to face communication.

Due to huge users, QQ is generating huge traffic every day, especially video traffic. This results in several issues and challenges: (1) Bandwidth Consumption. A mass of video traffic takes up too many network resources. (2) Network Management and Optimization. Due to its proprietary protocol and encrypted traffic, it is very difficult to identify QQ video traffic and further manage and optimize the whole network effectively. (3) Quality of Service. Because of its real time online application, it is required to improve Quality of Service in order to improve users' experience.

However, the QQ video communication mechanism is still in secret till now because of its proprietary protocol standard and using encryption. It is very difficult to analyze the procedure of QQ video communication. In order to address this issue, we did a lot of experiments and observations, investigated the characteristics of QQ video traffic and analyzed QQ local video plugins with the help of reverse engineering, and finally dig out the communication mechanism, reveal the video communication process, and uncover the proprietary video protocol structure and video codec.

The main contributions of this paper are as follows:

1) *Revealing the QQ video communication mechanism:* We reveal the QQ video communication mechanism in details, including call setup and video data transport. Generally, call setup goes through servers to negotiate the parameters of video session, whereas video data is transferred directly between peers.

2) *Discovering the process of a call setup:* We analyzed the procedure of call setup and exposed several necessary parameters for video session.

3) *Analyzing Proprietary protocol:* The proprietary video protocol structure is figured out, which consists of two parts: the header and the video data.

4) Finding out the video codec: QQ adopts a new video compression format of the Google open source. Moreover, the process of QQ video data recovery is explained in details.

The rest of the paper is organized as follows. Section II describes the related work about the QQ video communication. Section III analyzes the video communication Architecture. Section IV describes QQ video proprietary protocol structure. Section V reveals the video codec and video data recovery. Section VI concludes the paper.

II. RELATED WORK

The standard-based video communication mechanism has been displayed in [2][3][4][5]. For the traditional VoIP applications, several standard-based protocols of multimedia communication are utilized, such as SIP (Session initialization protocol) /SDP (Session Description Protocol) [6] and RTP (Real-time Transport Protocol) [7], etc. With the rapid development of real-time communication technologies, more works focus on how to improve the quality of video (or voice) communication. Therefore, many real-time communication applications use their proprietary standard to transmit media. For MSN, the RTP and SIP protocols are extended to improve the call quality[8][9]. And the video communication of QQ 2009 also used the two protocols [10].

However, for the latest version of QQ 2012 and 2013, the call setup is encrypted to protect the negotiation parameters for video session, media transferring use proprietary

*Corresponding author. Tel: +86 010 62772393. E-mail: yiboxue@tsinghua.edu.cn

protocols to improve the compatibility and call quality. Our analysis of QQ video communication exploits the approach in [11], in which a large number of Skype communication experiments are performed. We have not only analyzed the basic architecture of QQ video communication, but also discover the protocol structure of QQ video transport and video codec. So we reveal the mechanism of QQ video communication completely.

III. ARCHITECTURE OF QQ VIDEO COMMUNICATION

In this section, we will provide an overview of the mechanism and some key technologies of QQ video communication, including basic architecture, call setup and media transport.

A. Basic Architecture

Conceptually, when client A wants to chat with his peer B, he just needs to set up a connection and follow sending video data. However, the situation in practice is not so uncomplicated. For instance, one of the problems is how to map called-peer's address to the actual machine of his using.

To make it work, QQ adopts a typical VoIP architecture, involving call setup (or video session initialization) and media transfer, as shown in Fig 1.

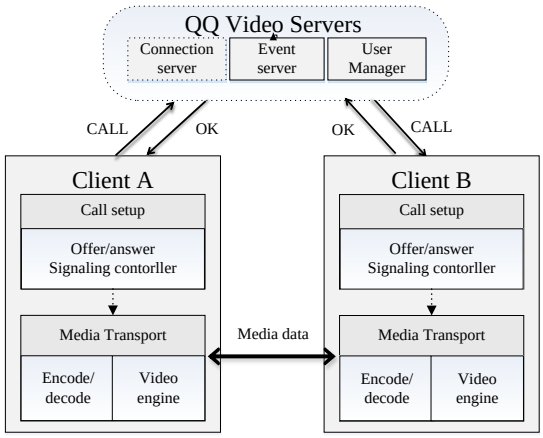

Figure 1. Basic Architecture of QQ Video Communication

The above basic architecture consists of three main entities, two clients and QQ video servers which are used to set up the call, including listening request of clients, managing video connections, rendezvousing peers' information and negotiating video parameters, etc. When starting a call, client A sends a request to a QQ event server which is always on listening. Then the user manager server determines where the call should be routed to and forwards the call to client B. After client B accepts the call, they negotiate with each other to agree on parameters via connection server. At last, the two clients can establish a new connection directly. Following, video data are captured, encoded (or decoded) by QQ video engine and are transmitted (or played).

Once one of them is ready to stop the communication, the disconnection message will to be sent to the video server to indicate the close of the connection. In summary, call setup goes through servers to initiate a video session, whereas video date transfers directly between clients.

B. Call Setup

For call setup, one client communicates with his server and asks to transmit a calling-request to intended peer. After that, they negotiate necessary information to initiate the video session. The process of the VSI (Video Session Initiation) is shown as in Fig.2

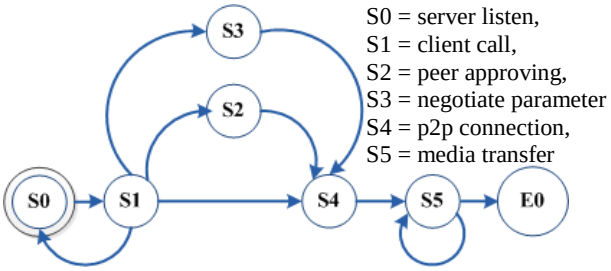

Figure 2. the Status of VSI

The whole process is explained in details as follows:

- Request a connection to his peer.
- Rout the request to the intended peer via the QQ video servers.
- The intended peer approves or refuses the calling-request.
- Negotiate with each other and agree on necessary parameters.
- Establish a new connection directly between the two clients.

During the VSI, one client gathers his information to contact QQ video servers which help to forward information to his intended peer. Then the two clients exchange necessary parameters with QQ proprietary video session description protocol, such as the address of the intended peer, the codecs for the video and the parameters for the codecs. After that, the two clients establish a new media channel with their shared capabilities.

To protect its confidential data in the communication effectively, QQ encrypts the VSI traffic. Moreover, VIS can support at least two transport protocols, TCP and UDP. UDP is usually the first choice owing to the higher efficiency. If the UDP connection failed, TCP is adopted.

In addition, another technical difficulty is how to establish connections directly between the two clients when they are located behind NATs (Network Address Translation) [12] or Firewalls. We conjecture that there are at least two ways for passing through NAT or FW.

- STUN. One possibility is the Session Traversal Utilities for NAT (STUN) protocol. The client uses QQ servers to discover the address mapping assigned by the NAT. The back trace is reported and forwarded to the client by QQ server when a STUN packet is sent to the server outside the NAT by client inside the NAT.
- ICE. The other possibility is Interactive Connectivity Establishment (ICE). The idea of ICE is to try all possible pairs of addresses in parallel until one is

reachable. It allows the client to use the best available UDP "connection".

C. *Media Transfer*

Comparing with the call setup, the media transfer is conceptually simple. Two clients can establish connection directly and transmit media data by using the agreed-upon video parameters. The overall structure describes video data transfer between clients, as shown in Fig 3. It mainly contains capturing video data, encoding/decoding and packaging (or unpackaging) with QQ proprietary protocol for transmitting (or playing).

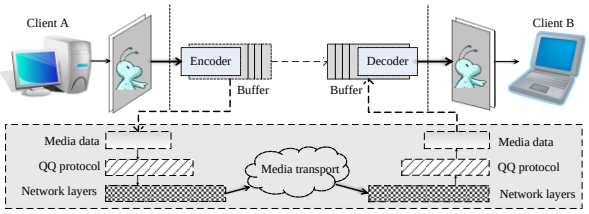

Figure 3. the Structure of QQ Media Transfer

At the sender side, the process of media transfer is described as follows:

- Capture the image as video frame.
- Encode the video frame at a particular codec and frame rate for video transmitting.
- The compressed video data exit the video encoder and queue in the buffer to wait for being sent.
- Transmit compressed video data to his peer while the size of the buffer is a relative value depending on the different network environment.

Inversely at the receiver side, the encoded video data is received in the decompressed buffer and then sent to the decoding engine. At last, the video frame which is decoded using a specific codec is played out at a particular size.

Moreover, in most of VoIP environments, media are carried over UDP. QQ proprietary video transport protocol is based on UDP and TCP. We will describe the protocol structure in detail later. If the two clients are in the same LAN, the video data transport over TCP. Otherwise, if they are in different network environments, they are transported over UDP.

In general, QQ video communication is divided into two steps, call setup and video transport. Although the QQ version is always updating, its communication mechanism doesn't change ever.

IV. THE PROTOCOL STRUCTURE OF MEDIA TRANSPORT

We summarize the protocol structure of QQ video data transport, which is divided into two parts, as shown in Table I, including Header and Body.

QQ header includes command part, RTP part and extension part. The RTP part takes advantage of typically real-time transport protocol which is explained in RFC 3550. The difference is that QQ extends it. The length of extension part is 8 bytes, which is mainly used to mark the fragment number of video frame. The command part based on UDP is different from that based on TCP. Following, we will describe the command part fields over TCP and UDP respectively.

Table I the Protocol Structure

Header			Body
Command	RTP	Extension	Video data
Unfix	12B	8B	Unfix

A. *Based on UDP*

The command part based on UDP has two cases which begin with \x05 and \x04 respectively, they can't appear in the same video session.

1) *Case 1*

The fields of command part beginning with \x05 are shown in Table II.

Table II Command Part Fields Based on UDP

Command part beginning with \x05					
BF	MS	VF	VS	FF	EF
1B	2B	3B	5B	4B	1B

The first 16 bytes are present in every command part beginning with \x05 and the fields have the following meaning:

- *Begin Flag (BF, 1 byte):* This field (\x05) identifies the flag of QQ multimedia packets, Which includes many applications, such as file transfer, audio and video etc.
- *Multimedia Sequence Number (MS, 2 bytes):* The MS is increased by a random number for each packet in one flow and helps to control UDP stream orderly.
- *Video Flag (VF, 3 bytes):* This field defines the flag of video application which is "\x00 \x2d \x64".
- *Video Sequence Number (VS, 5 bytes):* The VS is increased by one for each packet in one video stream and used for detecting video packet loss and restoring the packet sequence.
- *Frame Flag (FF, 4 bytes):* FF is the same number if the packets are fragmented. One decoded video frame may be too big to be divided into several packets to be transmitted. The FF is an important field to mark one video frame.
- *End Flag (FF, 1 byte):* This field(\x03) is used as a terminator.

2) *Case 2*

The fields of command part beginning with \x04 are shown in Table III. The first 47 bytes are present in every command part beginning with \x04, including the 31 bytes part and 16 bytes part beginning with \x05 (PBW\x05, same

as above). The fields of the 31 bytes part have the following meaning:

- *Media Flag (MF, 1 byte)*: This field also identifies the flag of QQ multimedia packets, which is \x04.
- *Version (V, 2 bytes):* This field identifies the version of QQ.
- *Length (L, 2 bytes):* This field identifies the length of UDP payload.
- *Sequence Number (SN, 4 bytes):* The SN is increased by a random number for each packet in one flow.
- *Number (N, 4 bytes):* this field records the QQ number.
- *Unknown (UN, 16 bytes):* The role of this field isn't clear and most of the UN block is filled with \x00.
- *Payload length (PL, 2 bytes):* This field is different from the L field and identifies the length of PBW\x05.

Table III Command Part Fields Based on UDP

Command part beginning with \x04							
MF	V	L	SN	N	UN	PL	PBW\x05
1B	2B	2B	4B	4B	16B	2B	16B

B. Based on TCP

The command part based on TCP is relatively simple , it has the following fields:

Table IV Command Part Fields Based on TCP

Command part based TCP			
FF	FI	MS	TVF
2B	4B	2B	2B

The first 10 bytes are present in every command part. The Frame Flag (FF) and Multimedia Sequence number (MS) is the same to the two fields of command part based on UDP.

- *Fill-in (FI, 2 bytes):* This field is filled with \x00 \x00 \x00 \x2d.
- *TCP Video Flag (TVF, 4 bytes):* This field is "\x03 \x03", which represents the video application.

V. QQ VIDEO CODEC

QQ video codec is negotiated in call setup which is encrypted transmitted. The default codec for QQ video is VP8 [13], which is a new video compression format of Google. The other choices for QQ video codec are H.264 and H.263. However, only VP8 codec is used in the latest version of QQ. Therefore, we conjecture that H.264 and H.263 codecs are for compatibility with the older versions of QQ.

The process of VP8 decoding is shown in the following:

- Obtaining the pure compressed video data based on the format of the QQ video transport protocol.
- Gathering fragment video data to a video frame.
- Packaging the VP8 header for a complete compressed video data based on VP8 header format.
- Decoding the compressed video data to the video frame of YUV format through VP8 codec algorithm.
- Playing every video frame based on its appropriate parameters

The QQ video data can be decoded through the above process successfully. In addition, it is impossible to play the decoded video frame if you don't know appropriate parameters. And we verify that the default video frame width is 320 and the default height is 240.

VI. CONCLUSION

In this paper, we first revealed the architecture of QQ video communication, including the call setup and video data transport. QQ encrypts the call setup and transports video data with proprietary protocol. Moreover, this paper also uncovers the proprietary protocol structure of video transport and video codec. In a word, we make a breakthrough for understanding QQ video communication and provide a complete, internet-based, real-time video communication mechanism.

ACKNOWLEDGMENT

This work was supported by the National Key Technology R&D Program of China under Grant No.2012BAH46B04. We would like to thank the reviewers for their insightful comments.

REFERENCES

[1] QQ official website, [Online]. Availabe: http://im.qq.com/index.htm..

[2] S. Soltani , K. Misra and H. Radha "Delay constraint error control protocol for real-time video communication", IEEE Trans. Multimedia, vol. 11, no. 4, pp.742 -751 2009.

[3] A. Keromytis, "A Comprehensive Survey of Voice over IP Security Research", IEEE Communications Surveys & Tutorials,vol. 99, pp.1-24, April 2011.

[4] T. Zourzouvillys and E. Rescorla, "An Introduction to Standards-Based VoIP: SIP, RTP, and Friends, " March/April 2010.

[5] Jennings C and Hardie T, "Real-time communications for the web, " IEEE, Communications Magazine, vo l51 pp 20-26 April 2013.

[6] RFC SIP, [Online], "http://www.ietf.org/rfc/rfc3261.txt".

[7] RFC RTP, [Online], "http://www.ietf.org/rfc/rfc3550.txt".

[8] Rui Lu and Jia Mi, "Design and Implementation of Instant Messenger Security Monitoring System Based on Protocol Analysis, " Control and Decision Conference (CCDC), 2010 Chinese, pp 4290-4293.

[9] D. Bonfiglio, M. Mellia, D. Rossi, and P. Tofanelli, "Revealing Skype Traffic: when randomness plays with you," ACM SIGCOMM '07, Kyoto, Japan, August, 2007.

[10] Pan, Wang. "Model and algorithm of tencent's voice service identification based on SAT: session association technology. " WiCom'09. 5th International Conference on. IEEE, 2009.

[11] S. A. Baset, "An analysis of the skype peer-to-peer internet telephony protocol," in Proceedings of IEEE INFOCOM, 2006:

[12] RFC NAT, [Online], "http://www.ietf.org/rfc/rfc1631.txt".

[13] VP8 Data Format and Decoding Guide RFC 6386, [Online]. Availabe: http://datatracker.ietf.org/doc/rfc6386/