An Empirical Study of Video Messaging Services on Smartphones

Yao Liu
SUNY Binghamton
yaoliu@cs.binghamton.edu

Lei Guo
Ohio State University
lguo@cse.ohio-state.edu

ABSTRACT

With the advancements in wireless networks and the pervasive adoptions of smartphones with camera capabilities, hundreds of millions mobile users are attracted to video messaging services on their smartphones. Unlike text messaging, transmitting large video messages (with a size of $\sim$ 10 MBytes vs. a 140 Bytes limit for text messages) demands effective network resource provisioning.

In this work, we investigate current practices of video messaging on smartphones. Focusing on the two most popular video messaging services, WhatsApp and WeChat, we conduct extensive experiments with commodity smartphones based on the three mainstream mobile OSes, namely, iOS, Android, and Windows Phone, from both the USA and China. We find that WhatsApp and WeChat have different resource provisioning strategies when serving video messaging clients, leading to a degraded service experience for WeChat users compared to WhatsApp users. Neither WhatsApp nor WeChat provides location aware service that can significantly improve the user experience. We further evaluate a few enhancements that can help reduce end-to-end delay in video messaging services. Our results provide new insights for both service providers and users for this type of newly emerging services.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Experimentation, Measurement

Keywords

Video Messaging, Throughput, Latency, Transcoding

1. INTRODUCTION

Today, mobile applications are developed to allow users to send/receive messages using wireless network connections (e.g., cellular or WiFi). Two such applications are WhatsApp [7] and WeChat (WeiXin in Chinese) [5], and they are rapidly gaining popularity among mobile users. It is reported that WhatsApp now delivers more than 1 billion messages per day [4], and that WeChat has attracted more than 300 million users within 2 years of its initial launch [6]. In addition to providing the traditional text messaging service, these applications also allow users to exchange audio and video messages.

Compared to text messaging that has been used for chatting in many countries, effectively delivering video messages in real-time is challenging. Video messages are often much larger than text messages. Thus, if improperly implemented, delivery of video messages may take much longer than text messaging, degrading the user experience for timely interaction.

In this study, we investigate the current status and challenges of the existing Internet video messaging services. We focus on WhatsApp and WeChat, as these are the most popular video messaging services in the USA and China, respectively, and they are available on multiple platforms. To examine each application's performance, we use commodity iOS, Android, and Window Phone smartphones to send and receive video messages to/from each other in both the USA and China. We find that the time to deliver and receive video messages varies significantly. A major factor that contributes to this lengthened time is the limited placements of WhatsApp and WeChat servers. Our study reveals that each application locates its servers in a single geographical area: the WeChat servers are located in Shanghai, China while the WhatsApp servers are located in Texas, USA. Thus, delivering a 20-second message from a client located in the USA to a client located in China takes more than 300 seconds while it takes less than 20 seconds between two USA clients. Our further investigation shows that the servers take similar amounts of time for delivering notifications, but they have different resource provisioning policies on allocating resources for uploading video messages. In particular, the WeChat server only advertises a 2,856 Bytes TCP window for receiving uploading traffic from clients. Depending on the round trip time, our findings show that the uploading throughput can be as low as 6 KBytes/s, resulting in long uploading latency.

These measurement results suggest the following improvements for the existing video messaging services: (1) client-side transcoding can be leveraged to reduce the upload traffic; (2) server-side transcoding can be used to deliver a lower quality version to the recipient; and (3) service providers should provide location aware services to enable fast uploading and downloading of video messages. We evaluate some of these improvements and show that although transcoding at client or server-side takes time, it can effectively reduce the transmission time for senders and recipients with slow connection speeds. These results provide helpful insights for both the service providers and end users.

NOSSDAV'14, March 19 - 21 2014, Singapore, Singapore
Copyright 2014 ACM 978-1-4503-2706-0/14/03 $15.00.
http://dx.doi.org/10.1145/2578260.2578271.

Table 1: Video Messaging Applications Used

Name	iOS App	Android App	Windows Phone App
WhatsApp	2.8.6	2.8.7326	2.8.2.0
WeChat	4.3.2.6	4.2	3.4.0.0

The rest of the paper is organized as follows. We present our measurement methodology in Section 2. We analyze the network support for video messaging services in Section 3. Based on our measurement findings, we suggest some improvements in Section 4. Some related work is described in Section 5, and we make concluding remarks in Section 6.

2. METHODOLOGY

To investigate mobile video messaging services, we conduct experiments at two different locations: Fairfax, VA, USA at George Mason University campus and Beijing, China with household network connections. Six smartphones running three different mobile operating systems (iOS, Android and Windows Phone) are used in experiments: iPhone 4S (iOS 6.0.1), iPhone 3GS (iOS 5.0.1), Galaxy S III (Android 4.0.4), Nexus One (Android 2.3.4), Nokia Lumia 920 (Windows Phone 8), and Meizu MX (Android 4.0). Our study focuses on the two most popular mobile messaging applications on smartphones: WhatsApp and WeChat. Table 1 shows the versions of these applications that we installed on our testing devices.

In this study, we focus on the network performance of messaging. That is, we ask how long it takes for the sender to upload a video message to the server, how long it takes the receiver to download the message, and how much network traffic is generated by both upstream and downstream transmissions.

We examine the amount of time that our testing devices use to upload/send and download/receive video messages at each location. In all experiments, testing devices are connected to a dedicated WiFi network, and we monitor network traffic while the applications are in use. We capture raw packets using Wireshark [8] and analyze the captured traffic file offline to determine how each application uses network resources.

3. NETWORK SUPPORT STUDY

3.1 Service Architectures

Both WhatsApp and WeChat adopt a store-and-forward message delivery mechanism. To send a message to either a single recipient or multiple recipients, the message is first uploaded to a server. The server pushes a notification message (a text message) to the receiver. Once the receiver is notified, it downloads the message from the server.

3.1.1 WhatsApp

The WhatsApp mobile application (app) establishes a TCP connection with `c.whatsapp.net`. This connection is responsible for status updates and is kept open as long as the mobile app is running. When the user sends a text message via WhatsApp, this connection is also used by the mobile app to send the text message to the WhatsApp server.

If the user wants to send a message with multimedia content such as a photo, a voice message, or a video message, the mobile app establishes a new TCP connection with a new host: `mms.whatsapp.net`. This host name resolves to several IP addresses, and the mobile app chooses one of them, then uploads the multimedia message content to this server. After uploading finishes, the application closes the connection.

Table 2: Average Round Trip Time to Servers

Name	RTT (ms)	IP Hops	Location
Client in Fairfax, Virgina, USA			
WhatsApp	80	19	Dallas, Texas, USA
WeChat	341	27	Shanghai, China
Client in Beijing, China			
WhatsApp	1021	13	Dallas, Texas, USA
WeChat	33	12	Shanghai, China

Table 3: Amazon EC2 Instances at Different Regions

Location	mms.whatsapp.net			long.weixin.qq.com		
	IP address	RTT	Hops	IP address	RTT	Hops
Virginia	50.23.142.174	71	18	112.64.237.196	334	23
Oregon	50.22.210.134	88	15	112.64.200.218	249	22
California	50.22.210.134	79	11	112.64.237.201	154	18
Ireland	50.22.210.131	97	12	112.64.237.188	450	21
Tokyo	50.23.142.174	130	12	112.64.237.201	246	20
Sydney	50.22.210.135	234	19	112.64.234.229	317	22
S. Africa	50.23.142.174	189	17	112.64.234.229	485	22

As the message is received at the WhatsApp server, it processes the message and pushes a notice to the desired recipients. If the message is a text message, the recipient contacts `c.whatsapp.net` to receive it. Otherwise, if the message is a multimedia message, the recipient sets up a new TCP connection with another host: `mms***.whatsapp.net`, where `***` stands for a 3-digit number. For example, in one of our experiments, `mms931.whatsapp.net` is the server that stores video messages. These multimedia hostnames (`mms***.whatsapp.net`) resolve to only a single IP address, and the IP address used to download a video message is the same address where the original video message is uploaded to.

3.1.2 WeChat

When the WeChat mobile app is activated, it establishes two TCP connections with `short.weixin.qq.com` and `long.weixin.qq.com`. The `short` connection is used for transmitting periodic heart beat messages and client-side statistics to the WeChat server. Separate TCP connections are established for each of these status messages. The `long` connection, however, is always kept open. The mobile app uses this `long` connection to send both text messages and multimedia messages to the WeChat server. Note that this behavior is different from WhatsApp, where text messages and multimedia messages are sent to different servers.

When the message is successfully uploaded, the WeChat server pushes a notification to all recipients. The recipient uses its `long` connection to download the text or the multimedia message.

3.2 Server Locations

We first identify the physical locations of the servers that our testing clients in Fairfax, VA, USA and Beijing, China connect to during the experiments using IP Locator [3]. This step ensures that we take into account the server location. Because video messages are often much larger than text messages, proximity to a messaging server can have a significant impact on uploading and downloading speeds.

Table 2 shows that WhatsApp users from China and the USA are directed to servers at the same location in Dallas, TX, USA. Similarly, WeChat users are all directed to servers at the same location in Shanghai, China. This indicates neither of these services is location aware because neither places servers at edge locations that are

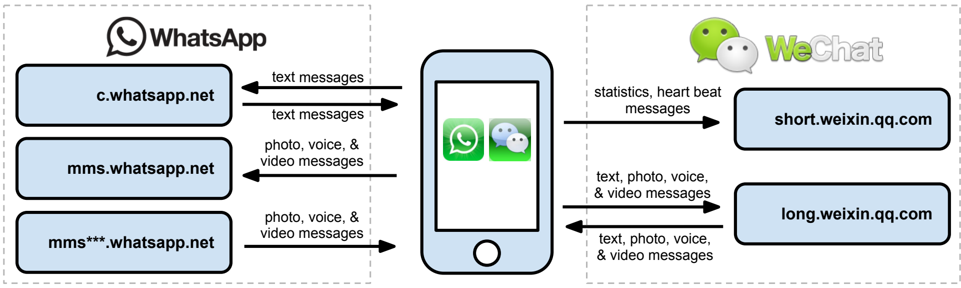
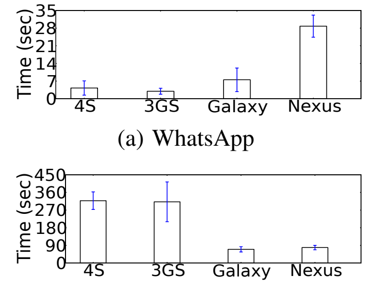


Figure 1: Connections made by WhatsApp (Left) and WeChat (Right) Clients



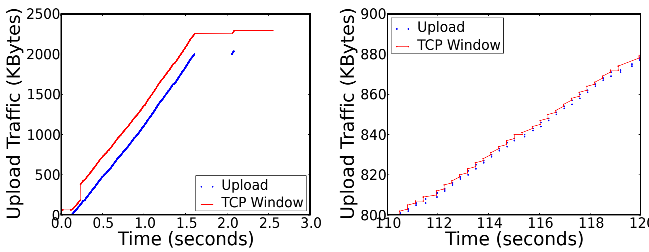
(a) WhatsApp

(b) WeChat

Figure 2: Sending/Uploading 20-second Video Messages

Table 4: Average Uploading Time Comparison

App Name	Sender Location	File Size (Bytes)	Avg. Time (sec)	Avg. Throughput (KBytes/s)
WhatsApp	USA	6,502,199	7.53	863.51
	China	6,264,878	189.30	33.09
WeChat	USA	415,602	70.10	5.93
	China	470,856	15.42	30.54



(a) WhatsApp

(b) WeChat (10-sec snapshot)

Figure 3: iPhone 4S Uploads 20-second Video Messages

close to the physical location of the user. Such transoceanic server connections result in long round trip times (RTT). For example, the RTT between our testing client in China and the WhatsApp server is more than 1 second.

Besides clients in the USA and China, we further examine if users from other parts of the world may receive location aware services. We start one EC2 instance at each Amazon AWS region, and examine what IP addresses the WhatsApp and WeChat servers would resolve to. We further examine the round trip time and IP hops of between our testing EC2 instances and these IP addresses. Table 3 shows the results. Note that all RTT values shown in the table are in milliseconds. We find that WhatsApp resolves EC2 instances from all over the world to a same IP range, which is a data center located in Texas, USA. For WeChat, all instances are resolved to the same IP range, which is a data center in Shanghai, China.

3.3 Uploading Time

To accurately measure the time used for uploading, we record the time difference between when the first packet is transmitted and when the last TCP ACK from the server arrives. Given that users at different locations connect to a single server, we first compare the uploading time and the average throughput observed in the two testing locations. For comparisons, we use Android 4.0.x devices at both locations, e.g., Galaxy S III in the USA and MX in China.

For WhatsApp, Table 4 shows the average uploading time observed in our tests. We find that while the size of a video message is similar (6.2 MBytes), the mobile device in the USA uses much less time to upload (8 seconds) than the mobile device in China (190 seconds). We caculate the average uploading throughput based on average uploading times and find that the throughput can vary by a factor of 26 or more. While the throughput may be affected by several factors, we can infer that connecting to a transoceanic server does significantly increase uploading time.

For WeChat, we find that the average throughput is much smaller at both locations compared to WhatsApp. This is surprising because our testing client, located in Beijing, China, is physically close to the server's location and has a very small RTT time of 33 ms.

Intuitively, a slow uploading speed is caused by TCP congestion control due to limited bandwidth availability. We try to confirm this hypothesis by letting multiple mobile devices upload at the same time to the same server. If the bandwidth is really the bottleneck, then the throughput would decrease and the uploading time would increase. However, after repeating experiments at different times, we find that the throughput and uploading time are almost constant across different tests, indicating the bandwidth is not the bottleneck. Instead, after repeated tests, we find that the TCP window size at the server-side is the source of the throughput problem. The server advertises a window size of only up to 2,856 bytes. Therefore, the uploading client can only upload 2,856 bytes a time and must pause to wait for the window to be re-opened on the server side. For our testing client in China, even though the RTT is only 33 ms, the resulting uploading throughput would still be smaller than 100 KBytes/s. For our testing client in the USA, given the high RTT of more than 300 ms, it is not surprising that the uploading throughput is only around 6 KBytes/s.

The difference is illustrated by Figure 3. The figure shows the accumulated traffic pattern for our iPhone 4S located in the USA. With WhatsApp, the TCP window opened at the server side is consistently around 150 KBytes, while the uploading with WeChat is constrained by the small TCP window size.

We conjecture that the reason WeChat uses such a small TCP window size is to allow its server to support more client connections. Recall that when the WeChat mobile app starts, it will keep a `long` connection with the server. This `long` connection is responsible for all messages, including text, photo, audio and video messages. Setting a small TCP window size for each such connection can reduce the amount of memory for maintaining each connection and therefore increases the number of concurrent connections at the server. Such a design is beneficial for supporting text messages or photo messages where the message size is relatively small. However, for video messages, this design results in the slow uploading speeds observed by our clients. These slow uploading speeds can significantly deteriorate user experience.

Compared to WeChat, which uses a single connection with a single server for all messages, WhatsApp uses dedicated MMS servers. Separating video messages from text messages give the

Table 5: WhatsApp: Sending/Uploading a Video Message (A Typical Experiment)

Sender	Audio (Kbps)	Video (Kbps)	Length (sec)	File Size (Bytes)	Time (sec)
iPhone 4S	62	706	20.75	2,002,178	3.07
iPhone 3GS	63	726	20.96	2,080,612	3.96
Galaxy S III	123	2302	21.56	6,502,199	7.39
Nexus One	96	2366	20.86	6,426,996	31.11

Table 6: WeChat: Sending/Uploading a Video Message (A Typical Experiment)

Sender	Audio (Kbps)	Video (Kbps)	Length (sec)	File Size (Bytes)	Time (sec)
iPhone 4S	62	716	21.10	2,079,366	261.57
iPhone 3GS	63	723	20.87	2,077,745	261.16
Galaxy S III	19	131	22.15	415,602	58.75
Nexus One	18	143	21.93	446,443	69.28

WhatsApp server more chances for optimization without sacrificing the user experience.

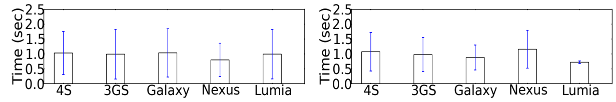
Besides throughput, the file size of video messages also affects uploading time. Table 5 shows the amount of time that our testing devices in the USA used for uploading a 20-second video message to WhatsApp. Video messages originating from iOS devices are encoded at similar bit rates, around 800 Kbps, while videos originating from Android devices are encoded at much higher bit rates of more than 2,400 Kbps. As a result, iOS messages have smaller sizes (around 2 MB) compared to Android messages (around 6 MB). iOS device finishes uploading within 4 seconds, while Galaxy S III uses 8 seconds. Nexus One uses more than 30 seconds for uploading, and other experiments with Nexus One also show similar long uploading times. This is due to a problem WhatsApp has reading video files from external SD card. While Table 5 shows the results of one typical experiment, Figure 2(a) shows the average and standard deviation of uploading times from 10 WhatsApp experiments.

Table 6 shows the amount of time that our testing devices in the USA used for uploading a 20-second video message to WeChat. Video messages originating from iOS devices are encoded at similar bit rates, around 800 Kbps, while video messages originating from Android devices are encoded at much lower bit rates around 160 Kbps. As a result, iOS messages have larger sizes (around 2 MB) compared to Android messages (around 400 KB). Because the uploading throughput of WeChat is constrained by the server, the throughput is relatively stable. As a result, transmitting a larger video message recorded by iOS devices takes about 5x the amount of time compared to a smaller Android-recorded video message. The average results from 10 experiments are shown in Figure 2(b). These results show that iOS devices consistently need more time to upload video messages on WeChat than Android devices.

3.4 Notification Latency

After uploading is finished and the video messaging server successfully receives the full video message, the sever will send a notification message to the recipient. Note that the video message is not included in the notification message. Instead, notification messages are usually only of a few bytes. Notifications should arrive promptly to ensure the interactivity of video messaging service.

To accurately estimate the notification latency, we let the sending and receiving devices use the same WiFi network to access video messaging services. By placing them in the WiFi network, we are able to calculate the time difference from when the last uploading packet is sent out by the sending device to when the notification message arrives at the receiving device. Figure 4(a) shows the amount of time used for WhatsApp notification messages to arrive at our five testing devices in the USA as well as the standard deviation. We find that for both WhatsApp and WeChat, the average notification latency for all devices, regardless of their underlying mobile operating systems, is around 1 second. For WeChat, shown in Figure 4(b), although the server is placed in China, we found that the notification latency is also around 1 second, which is very similar to that of WhatsApp. This low latency indicates the text message delivery is quite mature.



(a) WhatsApp (b) WeChat

Figure 4: Time Elapsed Before Receiving Notification

Table 7: Average Downloading Time Comparison

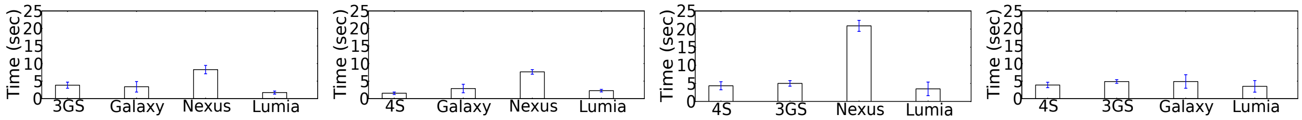
App Name	Receiver Location	File Size (Bytes)	Avg. Time (sec)	Avg. Throughput (KBytes/s)
WhatsApp	USA	2,002,178	3.40	588.86
	China	1,984,071	262.27	7.56
WeChat	USA	2,079,366	60.93	34.13
	China	2,079,366	12.44	167.15

3.5 Downloading Time

Since video messaging services use server-based delivery, we next examine if recipients at different locations experience different downloading times from these servers. We first use Galaxy S III in the USA and MX in China for comparisons. Both testing devices receive 20-second video messages sent by iOS devices via both WeChat and WhatApp in separate experiments. The downloaded file sizes are quite similar with a size around 2 MB across applications.

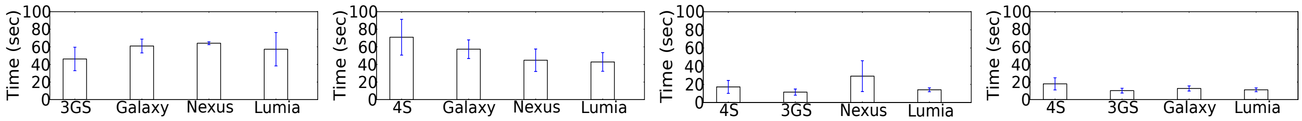
Table 7 shows that for the client in the USA, downloading a video message from the server located in Texas, USA is very fast. The average throughput reach more than 500 KBytes/s, and the video message is fully downloaded within 4 seconds. For the client in China, the throughput is much smaller, and the downloading takes more than 260 seconds to finish on average. This slower speed may be due to the limited bandwidth and the long RTT between the client in China and the server in the USA. For WeChat, because its server is located in China, our testing client in China is able to download the video message in 12 seconds on average. Our testing client in the USA, however, takes 5x time of 60 seconds. The results above confirm that clients in different locations can experience significantly different downloading throughput.

Besides location, the size of video message to download also plays a role in the downloading time. We conduct experiments with our five testing devices in the USA. Figure 5 shows the amount of time used by each device to download video messages shown in Table 5 that were sent from their corresponding originating devices and services. Note that although the WhatsApp server conducts server-side transcoding, the file size is only reduced by less than 3%. That is, receiving video messages recorded by Android devices still generates about 3x traffic compared to receiving video messages recorded by iOS devices. Comparing Figures 5(a)(b) with Figures 5(c)(d), we find that receiving Android messages takes more time than receiving iOS messages, which are smaller. Figure 6 shows the amount of time used by each device to download video messages shown in Table 6 that were sent from their corresponding originating devices and services. The iOS messages are about 2 MB in size and take 40 to 80 seconds to download. On the other



(a) Msgs Sent From iPhone 4S (b) Msgs Sent From iPhone 3GS (c) Msgs Sent From Galaxy S III (d) Msgs Sent From Nexus One

Figure 5: WhatsApp: Time Used for Receiving 20-second Video Messages



(a) Msgs Sent From iPhone 4S (b) Msgs Sent From iPhone 3GS (c) Msgs Sent From Galaxy S III (d) Msgs Sent From Nexus One

Figure 6: WeChat: Time Used for Receiving 20-second Video Messages

hand, Android messages with an average size around 420 KB can be downloaded in less than 20 seconds.

3.6 Summary

Our study shows that while both WhatsApp and WeChat can provide comparable performance when notifying users about their pending video messages, they have allocated resources differently for message uploading and downloading. WhatsApp allocates dedicated servers to video messaging services and does not constrain the uploading speed of mobile users. In contrast, WeChat consistently applies an uploading limit through the TCP window size in order to support a greater number of concurrent sessions.

Nevertheless, neither of these popular services provides location-awareness. As shown in our experiments, proximity plays an important role in the service response time, and improving this aspect of the service could lead to significant improvements in the user experience.

4. IMPROVEMENTS

Our measurement results in the previous section show that current video messaging services face the following two problems:

Network Traffic Amount: Sending and receiving video messages may generate a large amount of network traffic. For example, a 20-second video recorded by Android devices with WhatsApp can be as large as 7 MB. The large file size causes a large amount of traffic at both the sender side and the receiver side. Mobile users may be using cellular data connections, which are often billed based on the monthly total traffic amount and pricing tiers. Therefore, transmitting such a large amount of video messaging traffic may increase the monetary cost for users and discourage users from using video messaging services.

Uploading and Downloading Latency: To improve the user experience, it is desirable that users can send and receive messages promptly. However, according to our measurement results, it takes longer than 300 seconds on average for a client in the USA to send a 20-second video message to WeChat, and it takes longer than 300 seconds on average for a client in China to receive a 20-second message from WhatsApp. Such a long uploading/downloading time can significantly deteriorate the user experience.

To improve the user experience and encourage video messaging, we propose the following enhancements that can be quickly deployed in video messaging services based on the current practice.

4.1 Client-side Transcoding

It is preferable that users be given greater flexibility in the choice of both recording and transmission quality. More choices could allow a user to record at high quality but transmit a lower quality message if only limited bandwidth is available. The original high quality video could then be saved in the device's gallery for future use.

With client-side transcoding, the video message size can be effectively reduced because the original video is transcoded to a lower resolution. With a reduced video size, the amount of traffic to upload by the sender is reduced, saving users' network traffic cost. Because less data is uploaded, latency can also be reduced, which can improve interactivity in messaging applications.

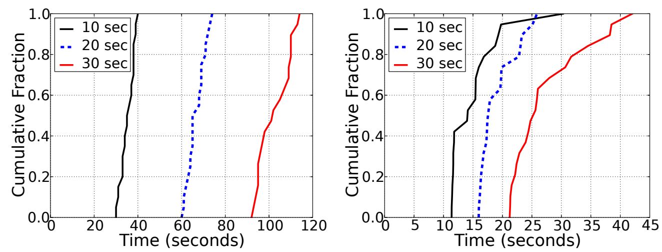
A natural concern toward client-side transcoding is if mobile devices are powerful enough to support the computation intensive transcoding tasks. To study the cost of client-side transcoding, we build ffmpeg [2] on an Android phone, Nexus 4. We use ffmpeg to repeatedly transcode 3 different videos. These videos are 10 seconds, 20 seconds, and 30 seconds long. All are captured by Nexus 4 at the resolution of 1920×1080, and the bit rate is around 12,000 Kbps. We transcode these videos to 480×270 at the bit rate around 430 Kbps, thus achieving 96% reduction in video file size. Figure 7(a) shows the distribution of the amount of time Nexus 4 performs the transcoding. We find that even on a device like Nexus 4 with powerful CPU, transcoding is very slow. It takes at least 30 seconds to transcode a 10-second video.

While client-side transcoding could be slow, it can benefit the user by effectively reducing the video file size. For senders with poor connection speeds such as uploading to WeChat and senders who want to deliver urgent messages, transcoding at the client side is an viable option as it can reduce the file size and thus significantly reduce the sending time.

4.2 Server-side Transcoding

While client-side transcoding can be used to reduce sender's traffic and uploading latency, server-side transcoding can be used to ensure compatibility and reduce recipient's traffic and downloading latency. For example, we believe that server-side transcoding can be used to provide a lower quality version of the same video. This type of transcoding gives recipients with a slow network connection or cellular data connection more choices, such as receiving the message faster and without much cellular data cost.

One way to efficiently and quickly conduct server-side transcoding is to leverage the cloud resources. Recently, Amazon Web Service released Amazon Elastic Transcoder [1], which allows users to utilize cloud resources including Amazon EC2 and Amazon S3 in an easy way for transcoding video content between different codecs, formats, and sizes. Pricing is based on the duration of transcoded videos. For example, transcoding a one-minute 1920×1080 video clip into 480×270, the user pays for one-minute which is 0.015 dollars.



(a) Android Nexus 4 (b) Amazon Elastic Transcoder

Figure 7: Time Used for Transcoding 3 Videos (10, 20, and 30sec)

To study how fast the cloud can transcode compared to mobile devices, we submit multiple jobs to the Elastic Transcoder to repeatedly transcode the same 3 videos we transcoded on Nexus 4. The transcoded videos have a resolution of 480×270 at the bit rate around 720 Kbps, thus achieving 94% reduction in video file size. For a recipient with slow network connection speeds, reduction in video file size can effectively reduce the transmission time, allowing the user to watch the video message sooner. Figure 7(b) shows the distribution of time used by Amazon Elastic Transcoder to transcode these 3 videos. Compared to transcoding at the client-side shown in Figure 7(a), cloud-based transcoding is faster. The median transcoding time is around 14 seconds for the 10-sec video, 17 seconds for the 20-sec video, and 25 seconds for the 30-sec video.

Although being faster than client-side transcoding, we find that the speed of cloud-based transcoding varies: a 10-sec video may take up to 30 seconds to finish transcoding. Such speed variation is not desirable for a real-time video messaging system. We plan to further investigate this issue in our future work.

4.3 Cloud-assisted Video Message Delivery

As shown in Table 3, neither WhatsApp nor WeChat today is location-aware. WhatsApp has servers located only in Dallas, TX, USA, and WeChat has servers located only in China. For each application, users from all over the world must connect to the same server cluster. The potential transoceanic connections over public networks can cause large round trip times and are more susceptible to bandwidth bottlenecks.

One way to increase transoceanic throughput is to set up servers at different data centers in the cloud all over the world and leverage the inter-datacenter bandwidth to transmit video messages among servers using the fast private network link. Existing research such as Airlift [9] has studied how to route video flows through inter-datacenter network under end-to-end delay constraints, and they can be beneficial to video messaging systems as well.

5. RELATED WORK

Short message service (SMS) is one of the most popular services on the mobile web, and has attracted much attention. Meng et al. analyzed the reliability of a SMS service, and found that bulk message delivery may cause network congestion [12]. Naor studied the signaling load for delivering SMS messages [13]. Most recently, mobile messaging applications such as WhatsApp and WeChat are gaining increased population among mobile users. Research has also been conducted to examine security aspects of these applications. For example, Schrittwieser et al. evaluated the security models and authentication mechanisms of nine mobile messaging applications including WhatsApp [14].

In parallel, mobile devices today are widely used for watching videos. Users are allowed to watch videos stored on mobile devices, or watch streaming videos with an Internet connection. However, watching videos on smartphones is complicated by the heterogeneity problem among mobile devices. Previous studies [10, 11] have investigated how to address this heterogeneity problem by leveraging server-side resource and cloud resource for transcoding. However, it is not clear how fast can transcoding be conducted at the server-side, and if that is fast enough for real-time video messaging services. In this study, we investigate the current practice of video messaging services, reveal a few challenges in serving mobile users, and propose some effective improvements accordingly.

6. CONCLUSION

The advancements in wireless and mobile networks have significantly improved the mobile transmission speed, which has enabled new video messaging services on smartphones. These video messaging services have already attracted hundreds of millions of users. The large number of users, however, leads to challenges that must be overcome due to the vast flow of messages that must be supported. In this study, we have conducted extensive measurements with two most popular video messaging services from different locations on commodity smartphones running three different mobile operating systems. Our studies reveal that existing video messaging services lack adequate network support, providing undesirable service in some cases. We propose a few enhancements that can be easily applied based on the readily-available techniques. We plan to further investigate these improvements with implementations in future work.

7. REFERENCES

[1] Amazon Elastic Transcoder. `http://aws.amazon.com/elastictranscoder/`.
[2] FFmpeg. http://www.ffmpeg.org/.
[3] IP Tracing. `http://www.ip-adress.com/ip_tracer/`.
[4] One Billion Messages. `http://blog.whatsapp.com/index.php/2011/10/one-billion-messages/`.
[5] WeChat. `http://www.wechat.com/`.
[6] WeChat attracts 300m users in less than 2 years. `http://www.chinadaily.com.cn/cndy/2013-01/17/content_16128915.htm`.
[7] WhatsApp. `http://www.whatsapp.com/`.
[8] Wireshark. http://www.wireshark.org.
[9] Y. Feng, B. Li, and B. Li. Airlift: Video Conferencing as a Cloud Service using Inter-Datacenter Networks. In *Proc. of IEEE ICNP*, 2012.
[10] Z. Li, Y. Huang, G. Liu, F. Wang, Z.-Li. Zhang, and Y. Dai. Cloud Transcoder: Bridging the Format and Resolutioin Gap between Internet Videos and Mobile Devices. In *Proc. of ACM NOSSDAV*, 2012.
[11] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen. A Server's Perspective of Internet Streaming Delivery to Mobile Devices. In *Proc. of IEEE INFOCOM*, 2012.
[12] X. Meng, P. Zerfos, V. Samanta, S. H.Y. Wong, and S. Lu. Analysis of the Reliability of a Nationwide Short Message Service. In *Proc. of IEEE INFOCOM*, 2007.
[13] Z. Naor. An Efficient Short Messages Transmission in Cellular Networks. In *Proc. of IEEE INFOCOM*, 2004.
[14] S. Schrittwieser, F. Fruhwirt, P. Kieseberg, M. Leithner, M. Mulazzani, M. Huber, and E. Weippl. Guess Who's Texting You? Evaluating the Security of Smartphone Messaging Applications. In *Proc. of NDSS*, 2012.