VIDNOW: SECURE AND PRIVATE LIVE VIDEO BROADCAST

By

LINGALA ARJUN REDDY

Bachelor of Technology in Computer Science and Engineering

Kakatiya University

Warangal, Telangana, India

2011

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2015

VIDNOW: SECURE AND PRIVATE LIVE VIDEO BROADCAST

Thesis Approved:

Dr. Eric Chan Tin

Thesis Adviser

Dr. K. M. George

Dr. Nohpill Park

ACKNOWLEDGEMENTS

I am very thankful to my parents who supported me throughout my life.

I would like to thank my advisor Dr. Eric Chan Tin, who made this app development easy to me. I am very grateful for the assistance and support he provided me.

I would like to thank Dr. K. M. George and Dr. Nohpill Park, who added some ideas to this app.

I would like to thank Dheeraj Jami, Jai Hari Rajendran, Daniel Contreras Chavez and Kody, who helped in building the user interface and releasing the app to App Store.

I would like to thank Oklahoma State University Technology Business Development Program (TBDP) and the Oklahoma State University App Center who partially supported this app.

Finally, I would like to thank all people, who supported me directly or indirectly.

Name: LINGALA ARJUN REDDY

Date of Degree: JULY, 2015

Title of Study: SECURE AND PRIVATE LIVE VIDEO BROADCAST

Major Field: COMPUTER SCIENCE

Abstract: Smartphones are becoming increasingly popular and are being used by over 5 billion people worldwide. It is expected that smartphones will replace digital devices like laptops and personal computers in the near future. Smartphones are being continuously enhanced using various technologies such as GPS, front and rear facing cameras, microphone, and always-on Internet connection.

With the increase in the usage of smartphones, exchange of information among people has become more convenient. Multimedia is an important type of content, which includes sound and video. Multimedia applications allow a user to share videos, which is an extremely powerful way to convey information as it incorporates a personal touch that other types of media lack. The advantage of integrating video into a multimedia application makes the user effectively convey a great deal of information in a short amount of time.

With the introduction of newer and more powerful generation of handheld devices, along with always-on Internet, developing mobile applications that make use of these advancements is one area that has immense potential with very limited research done in it. In this paper, we propose a model to develop an easy to use mobile app for users to broadcast live video.

Our mobile app, VidNow, provides privacy to the end user by encrypting the live video end-to-end and also allows the sender of the video to decide whether the receiver can save the received video to disk. A thin lightweight server is used to validate users and to send push notifications to users. As VidNow is mobile-to-mobile, the lightweight server only acts as a repeater to broadcast the live video to potentially millions of receivers. The latency for live video is 650ms on average. VidNow has been submitted to the Apple app store for public release.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

Smartphones are becoming increasingly popular and are being used by over 5 billion people. It is expected that smartphones can replace digital devices like laptops and personal computers in the near future. Smartphones are being enhanced day-by-day using various technologies such as GPS, front and rear facing cameras, microphone, and always-on Internet connection.

With the increase in the usage of smartphones, exchange of information among people has become more convenient. Multimedia is an important type of content, which includes sound and video. Multimedia applications allow a user to share videos, which is an extremely powerful way to convey information as it incorporates a personal touch that other types of media lack. The advantage of integrating video into a multimedia application makes the user effectively convey a great deal of information in a short amount of time.

With the introduction of newer and more powerful versions of handheld devices, along with always-on Internet connection, developing mobile applications that make use of these advancements is one area that has immense potential with very limited research done in it. In this paper, we design and implement an easy to use mobile app for users to broadcast live video. The model provides *encryption* of video to ensure secure communication and also provides *privacy* by giving control to the user to determine whether the video is to be saved in her friends' device(s). Also, as part of this research, we will determine the latency between recording a video

and receiving it and the *scalability* of the system. Based on the model proposed in this paper, we expect that our app will enable users to communicate efficiently with each other.

The latency of the live streaming is tested by conducting different types of experiments. The *latency* is calculated as the average of all delays of the live stream, in each experiment. Each experiment is done for 30 minutes and many times and a cumulative distribution function curve drawn with all the delays. The *scalability* is tested by increasing the number of devices and calculating the delay. The delay stays less than 1 second even as the number of devices receiving or broadcasting increases. VidNow provides end-to-end security by encrypting all the communications between the *smartphone* and *server*. VidNow will not save any information on the server other than username and device token (a unique ID to identify each device). VidNow has been submitted to the Apple app store for public release.

CHAPTER II

REVIEW OF LITERATURE

There are different ways to communicate with family and friends. Not all the live streaming apps have *smartphone* versions. If a person is in the middle of an event and wants to show the event to her friends, then the person can record the video using her smartphone or any recording device and upload the video to YouTube [1] (or other similar video service) and send the link to her family and friends. Though YouTube [1] provides the feature of uploading videos, the videos are saved on a central server and the recording is not live. Other users can only watch the video when the recording is completed.

Producer [2] is live streaming software that has apps for smartphones and computers. Producer will also live stream all the types of events like sports and music concerts etc. Producer will have all the live streaming done through a server where the server see all the video and Producer lacks the feature of allowing the receivers to save the video to their devices. Meerkat [3] is also an app which is used for live streaming which will not allow the receivers to save videos to their device for later.

Ustream [4] is another live streaming software which has both desktop and mobile versions. Ustream saves the data to the server, that is server can see the videos. Vine [5], Whatsapp [6] and snapchat [7] allows the users to share small length videos which is like an alternative to chatting. However the sharing is not live as the small length videos are recorded and

then sent to the receiver and the server can see the videos because they are saved to the server. Periscope [12] which is also a live streaming app saves all the data on the server. VidNow provides the live streaming feature using the same small length videos and the video is not saved on any central server as Vine, Whatsapp does. VidNow also allows the sender to give permissions to receivers to save the video to her device.

FaceTime [8] is the one of most famous video calling apps which provide HD quality. One thing FaceTime is lack of is conference i.e., talking or sending the video to multiple people at same time. Skype [9] is a peer-to-peer model which provides security to the user as peer-to-peer will not save any video on the server. But, Skype need to maintain a server for the login validation and a peer must always be running. VidNow uses a thin server, which is a commodity hardware working as server.

Google Hangout [10] and Facebook video [11] are the video calling apps which are used by almost all the people today. These apps are used for video calling but the apps will not allow the receivers to save the video to her device. VidNow allows to save the video if the recorder allows it.

CHAPTER III

DESIGN

Our design mainly consists of two components: *smartphone* and *server*.

3.1 Smartphone

Smartphone is used for broadcasting videos and receiving the videos. The user can record the video and watch the video using a smartphone. Using her smartphone the user will be able to login into VidNow through a unique username.

3.2 Server

A thin server is used to validate the login of the user and send push notifications to the users who are receiving the video. The server will not save any videos but will transfer the video bytes to the intended users.

The functionalities of the app are divided into two steps

1) Login
2) Broadcast Video

Login:

The user will enter the unique username or phone number and this username is sent to the server for validation as shown in fig. 3.1. Each device will have a unique device token assigned by Apple. This device token is used to send push notifications. Push notifications are to notify the user when the user is not active on the app. Unique device token is also sent to the server along with username (or phone number). The server will check whether the username is unique or already taken. If the username is unique the server will return 'GOOD' to the user smartphone as

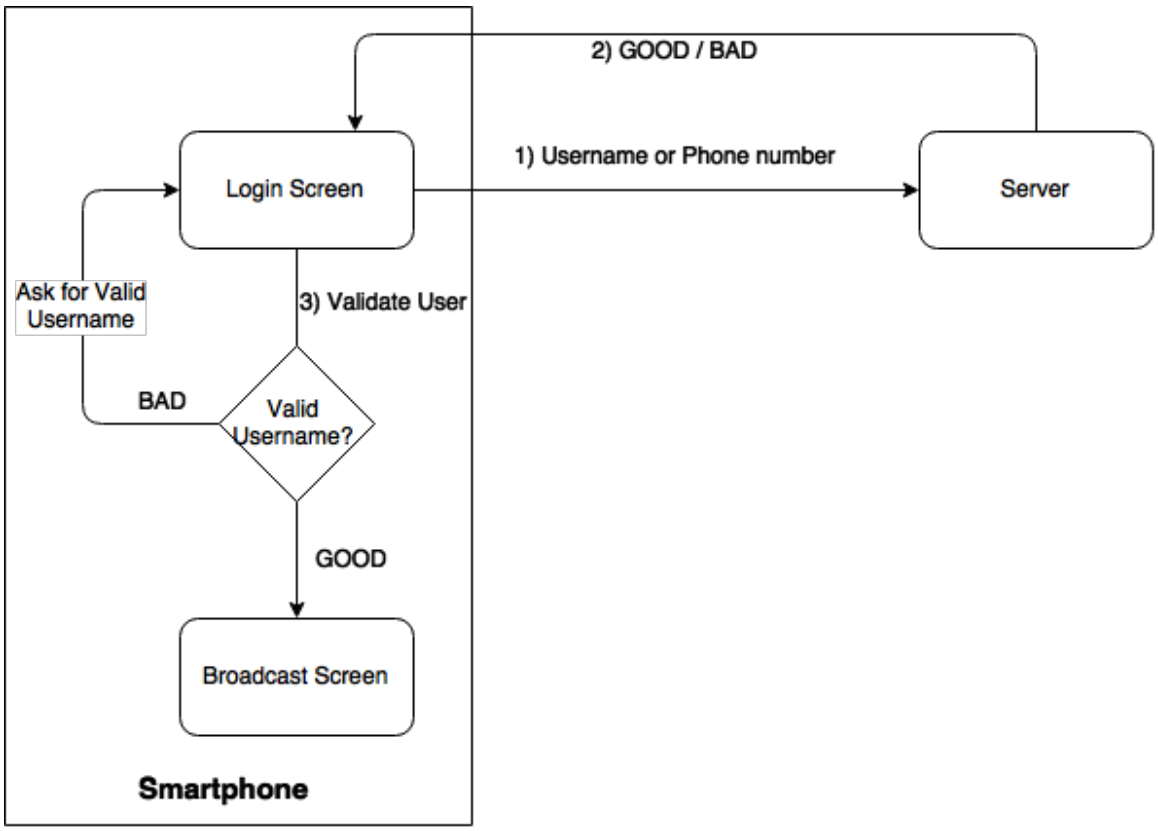


Fig. 3.1 Login Procedure

shown in fig. 3.1 and will send 'BAD' to the user smartphone if the username is already taken (A username like "John" may not be unique as it is very common). If the server returns 'BAD' then

the user will be asked to enter a valid user name as shown in fig. 3.1. If the server returns 'GOOD' then the user will be logged into the app and will be shown Broadcast Screen.

Broadcast Video:

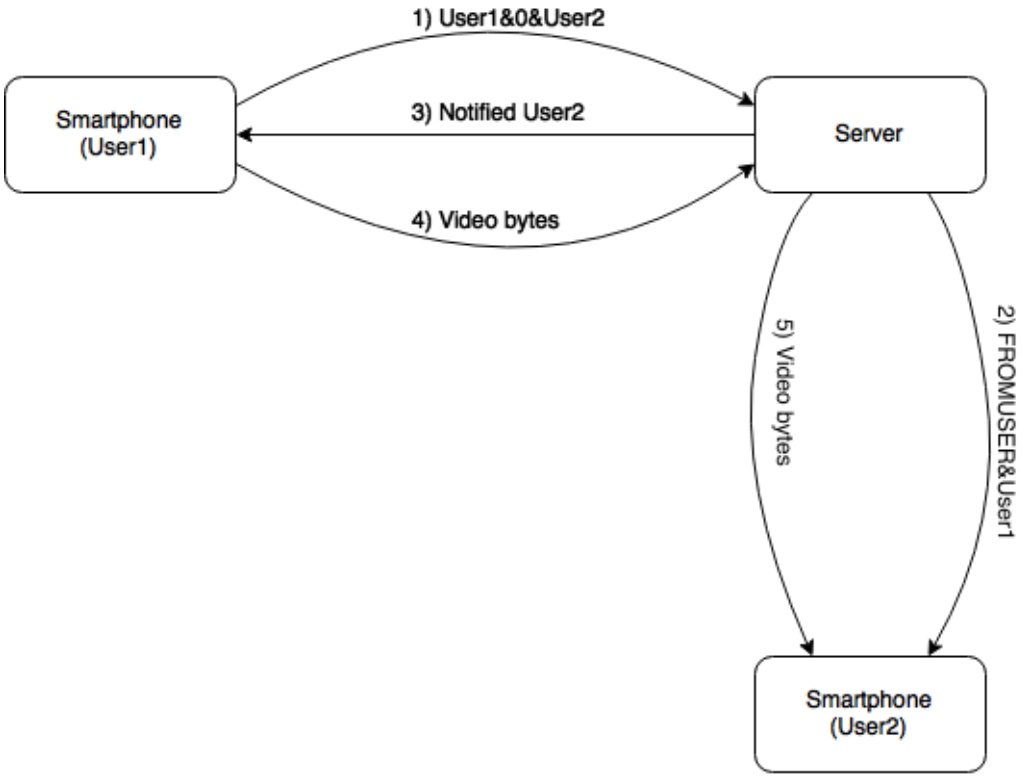


Fig. 3.2 Broadcast Video

To broadcast the video to friends the user will select the friends to whom she wants to send to. When the user selects the friends and clicks a button to broadcast the video, the following contents are sent to the server.

User1&0&User2,User3,User4,……Userk

The above string is divided into three parts.

1) User1 → The user who is sending the video
2) '0' or '1' → '0' means the user is not allowing the receiver to save the video

   '1' means the user is allowing the receiver to save the video
3) User3,User4,….Userk -> These are usernames of the users to whom the 'User1' want to send the video to.

Once the server receives the above information the server will check whether the users who are intended to receive the users are valid users (the users currently exists) or not. The server will send notification to all the valid users among User3,User4,….Userk that User1 is sending a video to them. The receivers can either watch the video or cancel it. If the receiver cancels the notification from the server, then a message is sent to the server that a particular user has rejected the video. The server will confirm to 'User1' that it sent notifications to all the users. Once 'User1' gets the confirmation from the server, it will start sending the encrypted video bytes to the server. The server will transfer the bytes to all the users who are intended to be. The receivers will decrypt the video and play the video. The live stream is done in the form of short length videos each of 6 seconds. When the length of the video is less than 6 seconds experimental delay is found which causes the missing of videos on the receiving side.

When 'User1' stops the broadcasting the video a stop message is sent to the server from 'User1'. The server will forward this stop message to all the receivers and an alert is shown to the receivers that the video broadcasting is stopped.

CHAPTER IV

IMPLEMENTATION

We implemented our proposed application as a smartphone app using the programming language Objective-C. The proposed design of our app contains two main components: *Smartphone* and *Server*. Smartphone is used for recording the video to be presented by the user and also to display received videos. The load on the smartphone app increases exponentially while the smartphone starts recording the video to be sent to other users. So, a thin lightweight server that is responsible for sending data from one smartphone to multiple smartphones is used to reduce the load. Server will also validate the user login so as to prevent users with duplicate usernames.

4.1 Smartphone

Smartphones are used for broadcasting the video and watching the receiving video. Each smartphone when logged in will register with the server. Each smartphone will have a unique device token. The device token and the username are stored in the server. Server searches for the user and sends the push notification to the user using her device's unique device token. Push notifications are to alert the user if the VidNow app is running in the background or closed. The push notification is sent to the receiver(s).

4.2 Server

A thin lightweight server is used to reduce the load on the smartphone. Server stores the information about all the users who are using the app. The server will store the user name and the device token of the device. Server will ensure that no two existing user names are same. Server will assign a socket to each user each time the user logs into the app. All the communications between smartphone and server will be made through this socket. Server is also responsible for transferring the video bytes to the intended users. The video bytes are encrypted using *RNCryptor* [13] by the smartphone before sending to the server. The server will just transfer the bytes to the users and will not save any information about video bytes.

4.3 Login Procedure

The first screen of the app, as shown in fig. 4.1, has a simple text field named "Username or Phone Number" which asks the user to enter the individual's unique username or phone number. The user is also allowed to login with her Facebook using "Login with Facebook" button. In this case the Facebook name of the user is taken as login name of username for the app.

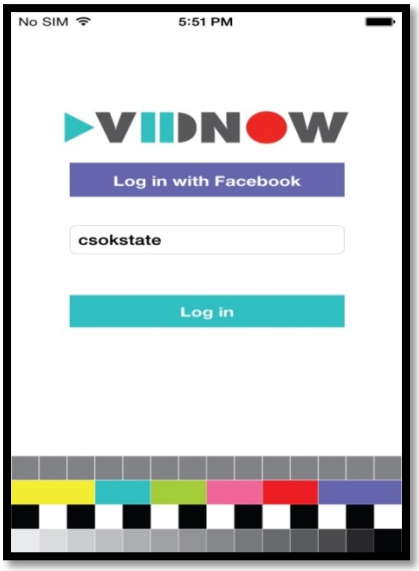


Fig. 4.1 Login Screen

Once the user clicks the login button with the username entered in the username text field, the username or phone number is sent to the server for validating the uniqueness of the username. If the username already exists in the server the user will be asked to enter a valid username. If the username is valid then the user will be redirected to another screen called Broadcast Screen, as shown in fig. 4.2.

The user login name will be displayed in the Broadcast Screen and the user can change her login details by clicking the "Change Login" button, which will redirect to Login Screen.

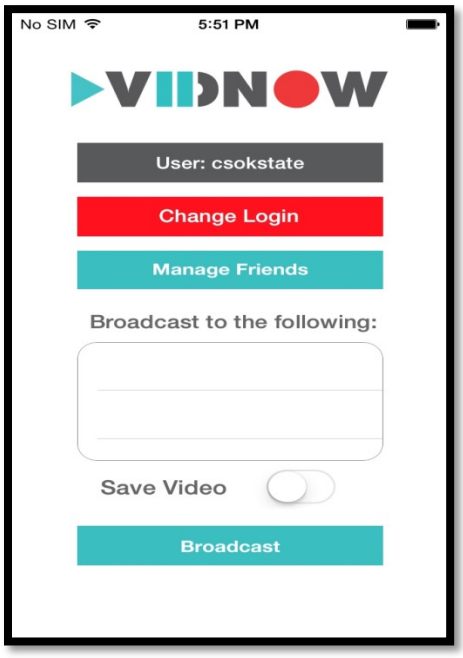


Fig. 4.2 Broadcast Screen

Broadcast Screen contains "Manage Friends" button, which will redirect the user to the Friends Screen as shown in fig. 4.3, which lists all the friends added by the user.

4.4 Selecting Friends

Friends Screen displays all the friends of the user. This list will be empty when the user visits this screen for the first time. The user can add her friends by clicking the "Add" button at the right top corner of the Friends Screen. The user is shown a dialog box asking for friend's name. Once the user clicks "OK" in the box after entering the friend's name, the friend will be added to the friends list.

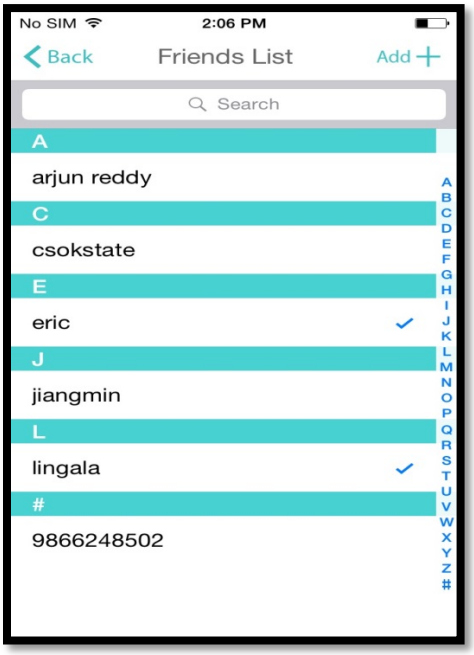


Fig. 4.3 Friends Screen

This friends' list is local to the smartphone and not stored on the server. The user can add any number of friends to her list. Server will not have any check about the friends added by the user. The friends who are added by other users will not get any notification. This new updated friends list will be shown from the next time the user comes to the Friends Screen. The friend's name added to the list should also be unique as this is the name the user will be used to broadcast the videos. If the friend's name doesn't match with any name on the server, the video will not be broadcasted. The user can delete the friends from the list by clicking on the delete button, which

shows up by swiping the column of friend's name from the right side. The user can select the friends whom she wants to broadcast to by tapping the friend's name. A checkmark is shown to the right side of the friend's name as shown is fig. 4.3. The user can deselect the friend by tapping on the friend's name again. The checkmark that appeared before will disappear. If the user cannot see the friend she want to broadcast to, she can either scroll down the list to find the friend or search the friend by typing the friend name in the search box of the Friends Screen. When the friend's name is found the user can select the friend. Once the user done with the selection of friends she wants to broadcast to, she can click "Back" button at the top left. The "Back" button will redirect the user to the Broadcast Screen with friends selected before displaying in the table view of the Broadcast Screen. When the user clicks "Manage Friends" button and goes to the Friends Screen again, checkmarks will be shown beside the names of the friends selected by the user before.

4.5 Broadcast Video

When the user clicks the "Broadcast" button in the Broadcast Screen the video that the user is recording will be broadcasted to her friends she selected. Broadcast Screen also contains a switch called "Save Video". If the save video is set to "On" then the users receiving the video will be prompted with an alert asking whether they want to save the video to their device or not. If the receiver accepts to save the video to her device then the video will be saved to the receiver's device. If the receiver does not accept to save the video to her device or the save video is set to "Off" by the recorder, then the video is just played on the friends (receiving) devices and will not be stored. Smartphones will be responsible for recording the video, encrypting the video and sending the encrypted video bytes to the server. Server will be responsible for transferring the encrypted video bytes to all the users the recorder intended to. The receiver will decrypt the video bytes after receiving from the server and playing the video. Server will make sure that the users

who are already broadcasting a video will not get any notification about receiving the video from other user.

4.6 Receive Video

The users to whom the video is sent will be notified that the receiver is broadcasting the video. The server sends push notifications to the receivers of the video using JavaPNS [14].

> JavaPNS is Java Push Notification Service that creates a payload and sends the notification using Apple Push Notification Service.

When the push notification is opened the app will show an alert asking the receiver whether she want to watch the video from sender or not. When the receiver accepts by clicking "Yes" in the alert the app starts asking the encrypted video bytes from the server. The app on the receiver side will decrypt the video bytes and play the video. If the receiver wants to stop receiving (watching) the video she can close the app. If the sender stops broadcasting the video the receiver will be prompted with an alert saying that sender has stopped broadcasting.

If the user receiving the video wants to stop receiving the video, then the user can close the app by double clicking the home button of iPhone and sliding up the app. If the user comes back to the app the user will not receive video from the user who she was receiving before. The user will be notified if any other user is sending the video.

CHAPTER V

VALIDATIONS

1) In the login screen if the user tries to click “Login” button without entering any name or phone number, then the user will be prompted with an alert to enter a name.

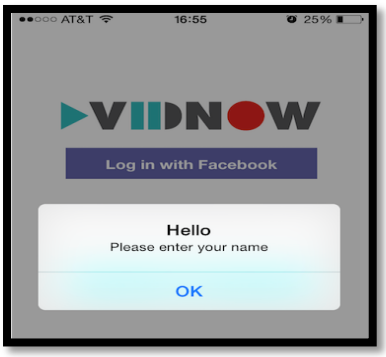


Fig. 5.1 Login Validation

2) In the login screen if the user tries to login with username which already exists in the server, then the user will be prompted with an alert saying that the “Username already exists”.

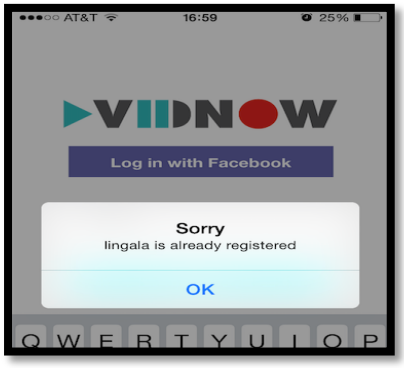


Fig. 5.2 Duplicate User Validation

3) In the friends screen if the user tries to add a friend who is already exists in the friends' list, then the user will be prompted with an alert saying that the friend already exists.

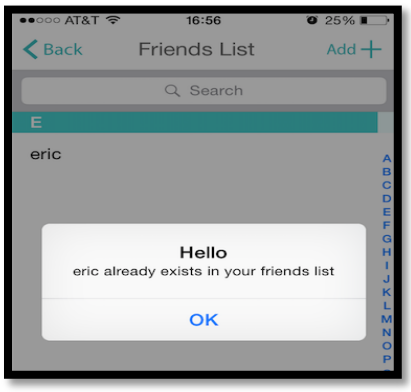


Fig. 5.3 Friends List Validation

4) If the user tries to broadcast the video without selecting friends, then user is prompted with an alert asking the user to select at least one friend to broadcast.

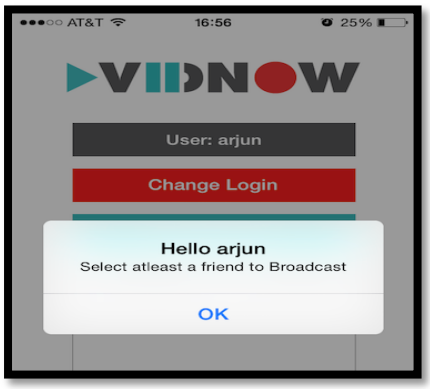


Fig. 5.4 Broadcast Validation

CHAPTER VI

EVALUATION

The live video streaming between devices is done using short length videos each of length 6 seconds and sent to the server. The server will transfer these videos to the users. In order to analyze the performance of our approach proposed in this paper, we calculated the delay between the user sending the video and the user receiving the video. To test *scalability* we used multiple broadcast devices and multiple receiving devices.

6.1 One Broadcaster, Multiple receivers

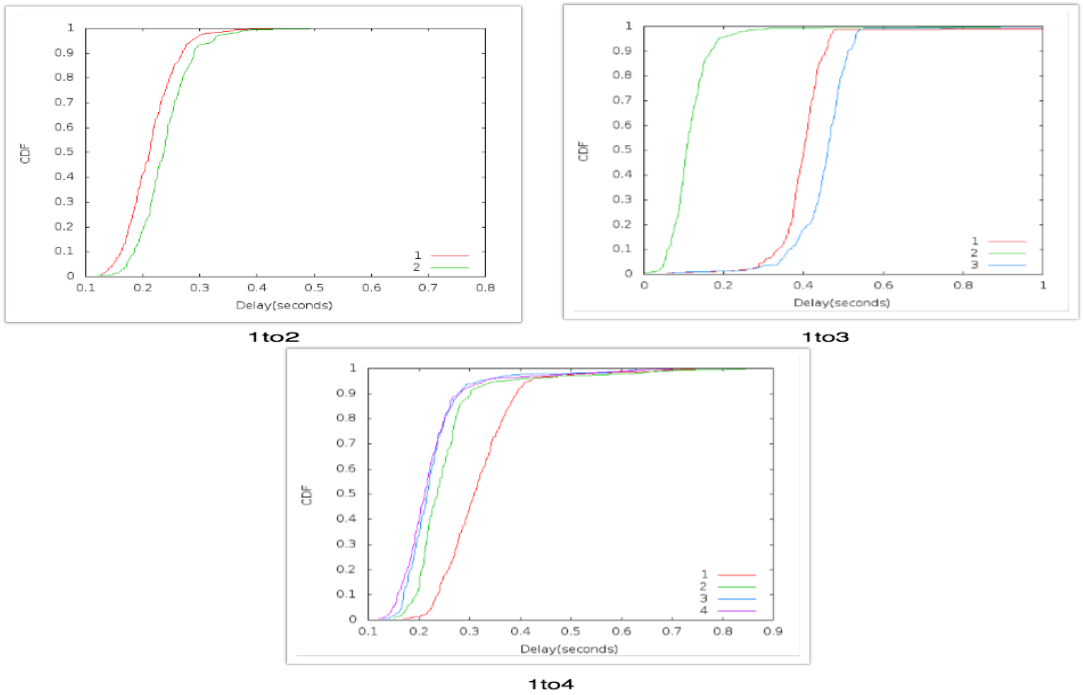


Fig. 6.1 One Broadcaster, Multiple receivers

In fig. 6.1 three graphs represents the delays in broadcasting the video from one device to many devices in different ways. First graph (1to2) represents broadcasting from one device to two devices all are on the same Wi-Fi network. The two lines represent CDF drawn using the delays of the two devices. From the graph we can see that the average delay of 0.3 seconds or less. Second graph (1to3) represents broadcasting from one device to three devices all are on the same Wi-Fi network. From the graph we can see that the average delay is 0.45 seconds or less. The third graph (1to4) represents broadcasting from one device to four devices all are on the same Wi-Fi network. From the graph we can see that the average delay is 0.35 seconds. From the above graphs we conclude that the delay is not increased when the number of receivers increases. Most of the delays in fig. 1to2 and 1to4 range between 200ms-400ms where as the delays in fig. 1to3 range between 200ms-500ms. Due to different bandwidths and multiple hops the delays may differ by 100ms.

6.2 Multiple Broadcasters, Multiple Receivers

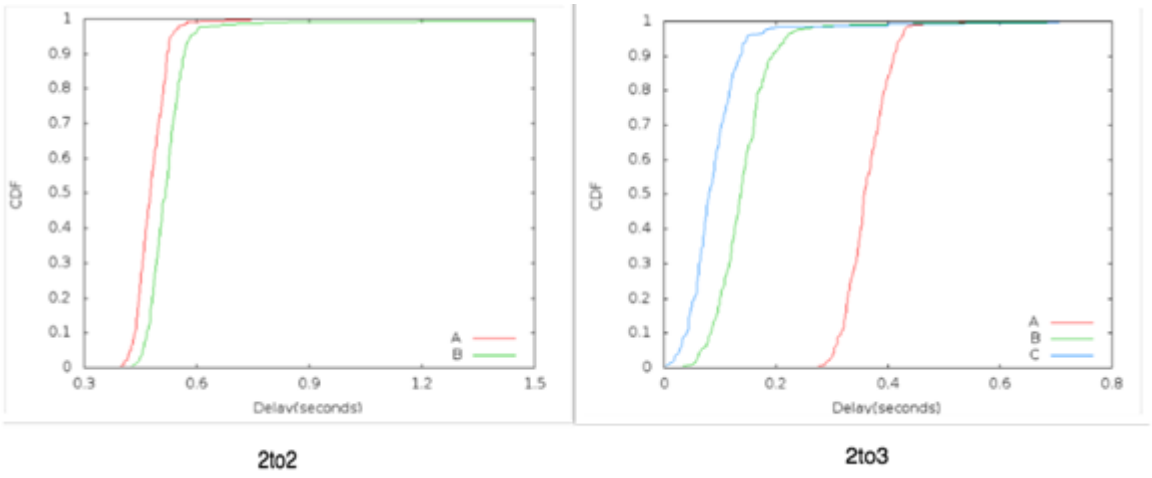
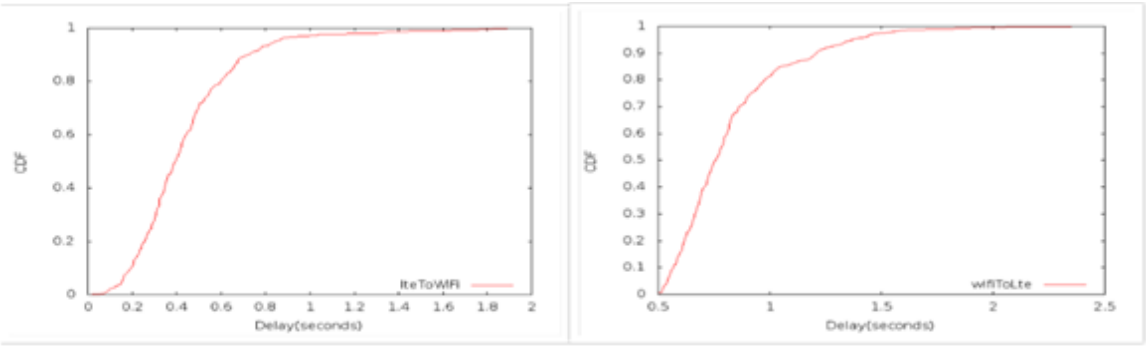


Fig. 6.2 Multiple Broadcasters, Multiple Receivers

In the fig. 6.2 two graphs represents the delays in broadcasting the video from two devices to many devices in different ways. First graph (2to2) represents broadcasting from two devices to two devices at the same time all are on the same Wi-Fi network. The two lines represent CDF drawn using the delays of two devices. In the 2to2 graph the plotted line 'A' represents the delay

of first device and ‘B’ represents the delay of second device. From the graph we can see that the major part of the streaming has delay of 0.5 seconds or less. Second graph (2to3) represents broadcasting from two devices to three devices at the same time all are on the same Wi-Fi network. The three lines represent CDF drawn using the delays of three receiving devices. In 2to3 model there are 2 types of broadcasting, one device to one device and one device to two devices. In the 2to3 graph the plotted line ‘A’ represents one-to-one broadcasting, ‘B’ and ‘C’ represents the first and second of one-to-two broadcasting respectively. From the graph we can see that the major part of the streaming has delay of 0.4 seconds or less. So, from the above graphs we can conclude that the delay is not increased with the increase in the number of devices receiving the video.

6.3 LTE-WiFi:



LTE to WiFi WiFi to LTE

Fig. 6.3 LTE-WiFi

In the fig. 6.3 two graphs represents the delays in broadcasting the video from lte-to-wifi and wifi-to-lte. In the first graph (lte-wifi) the plotted line represents the CDF drawn using the delay of wifi (receiving) device. From the graph we can see that the major part of the streaming has delay of 0.8 seconds or less. The delay in this case is more than the delay in the broadcasting using all the WiFi devices. This is because the LTE takes more time to send video bytes than WiFi. In the second graph (wifi-lte) the plotted line represents the CDF drawn using the delay of lte (receiving) device. From the graph we can see that the major part of the streaming has delay of

1.2 seconds. The delay in this case is bit more because the LTE takes more time to send video bytes than WiFi.

All the above graphs experiments were done at the same place i.e., the recording and receiving devices are at the same place. For the below experiments recording and receiving devices are at different places.

6.4 Different Places, Same WiFi Network

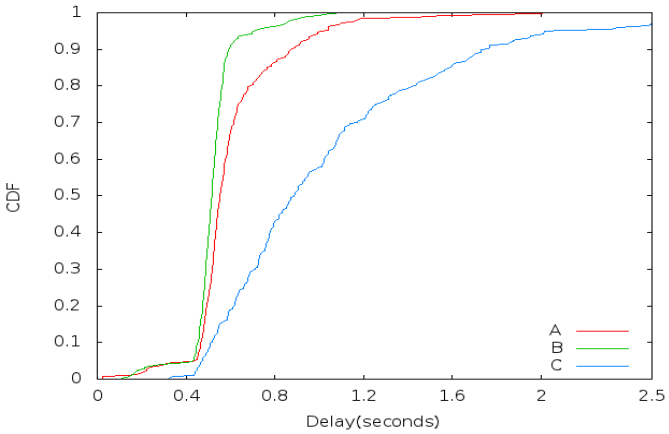


Fig. 6.4 Different Places, Same WiFi Network

The graph in the fig. 6.4 represents the CDF of delays in broadcasting the video to different places. In the graph the plotted line 'A' represents the CDF of delay of the device that is about 400 meters from the recording device and using WiFi. From the graph we can see that the major part of the streaming has a delay of 0.5 seconds or less. In the graph the plotted line 'B' represents the CDF of delay of the device that is about 100 meters from the recording device using WiFi and the major part of the streaming has a delay of 0.4 seconds or less. In the graph the plotted line 'C' represents the CDF of delay of the device that is at the same place where the recording device is, but using LTE and having a delay of 1 second or less. From the above graph we can conclude that the devices using WiFi having less delay and the devices using LTE are receiving the video with a delay bit greater than that of devices using WiFi.

6.5 Different WiFi Networks:

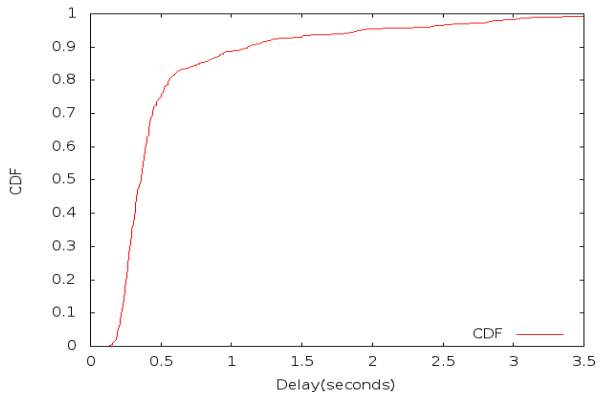


Fig. 6.5 Different WiFi Networks

This experiment is done between two devices, one recording device using a home WiFi and a receiving device using a different home WiFi. In the graph the plotted line is the CDF of the delays of the receiving device. From the graph we can see that the major part of the streaming has a delay of 0.6 seconds or less.

6.6 Summary

From the above evaluations we can conclude that the average delay is less than 1 second. The average delay on the Internet is about 100-200 millisecond. Considering the multiple hops involved, the delay in using this app is not significant. The video is thus **broadcast live**. Based on the experiments above, the *scalability* of the app is tested i.e., there is a difference of 100ms-200ms in delay with the increase in the number of devices which can be considered as live. The delay is sometimes reduced depending on the bandwidth.

CONCLUSION AND FUTURE WORK

Until now there have been apps that provide live streaming feature but all of them rely on some central server that can see all the videos. VidNow provides privacy by not saving the video data to the server. VidNow only requires a unique username or phone number to login. VidNow can add any number of friends and send live streaming to any number of friends who are using the app.

VidNow also provides a new feature to the recorder. If the recorder allows to save the video then the receivers are prompted with an alert asking whether they want to save the video. All the communications between the smartphones and server are encrypted. VidNow is submitted to Apple app store for review and will be available on the app store once the review is done.

Live streaming allows many features to provide to the user. This opens further work on different parts of VidNow. A new feature can be included into VidNow i.e., pausing the video when the receiver is watching the video and when the receiver pauses the video, a new button 'Go to Live' is enabled which takes the user to live or the receiver can play to continue the live streaming.

VidNow is submitted to the Apple app store for review. VidNow provides easy-to-use environment for the end user and also provides security by encrypting all the communications between smartphone and thin server.

REFERENCES

[1] YouTube. https://www.youtube.com/yt/about/

[2] Producer (Live stream). https://livestream.com/producer

[3] Meerkat. https://meerkatapp.co/

[4] Ustream. http://www.ustream.tv/

[5] Vine. https://vine.co/

[6] Whatsapp. https://www.whatsapp.com/

[7] Snapchat. https://www.snapchat.com/

[8] FaceTime. https://www.apple.com/ios/facetime/

[9] Skype. http://www.skype.com/en/about/

[10] Google Hangouts. http://www.google.com/+/learnmore/hangouts/

[11] Facebook Video. https://www.facebook.com/help/439078162792430/

[12] Periscope. https://www.periscope.tv/

[13] RNCrytor. https://github.com/RNCryptor/RNCryptor

[14] JavaPNS. https://code.google.com/p/javapns/wiki/PushNotificationBasic

[15] Carlsten Ullrich, Ruimin Shen, Ren Tong, Xiaohong Tan, "A Mobile Live Video Learning System for Large Scale Learning" 2010

VITA

LINGALA ARJUN REDDY

Candidate for the Degree of

MASTER OF SCIENCE

Thesis: SECURE AND PRIVATE LIVE VIDEO BROADCAST

Major Field: COMPUTER SCIENCE

Biographical:

Education:

Completed the requirements for the Master of Science in Computer Science at Oklahoma State University, Stillwater, Oklahoma in July, 2015.