

ĐẠI HỌC QUỐC GIA TP. HCM  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



MẠNG MÁY TÍNH (CO3003)

---

BÀI TẬP LỚN 1

# **STREAMING VIDEO SERVER – CLIENT USE RTSP AND RTP PROTOCOL**

GVHD:      Bùi Xuân Giang

Sinh viên thực hiện:      Nguyễn Quốc Duy – 1811727

Nguyễn Văn Chung – 1811624

Nguyễn Ngọc Chính – 1811622

Trần Đình Dũng - 1811797

Tp. Hồ Chí Minh, Tháng 11/2020

## MỤC LỤC

<b>I. Phân Tích yêu cầu.....</b>	<b>3</b>
<b>II. Đặc tả chức năng .....</b>	<b>3</b>
<b>III. Class Diagram.....</b>	<b>6</b>
<b>IV. Hướng dẫn sử dụng.....</b>	<b>7</b>
<b>IV. Đánh giá kết quả đạt được.....</b>	<b>9</b>
1.The Client.....	9
2. The Server .....	13
3. Extend .....	15
<b>V. Link code.....</b>	<b>17</b>

## I. PHÂN TÍCH YÊU CẦU

- Sinh viên triển khai một server và client để stream video trong đó sử dụng giao thức Real-Time Streaming Protocol (RTSP) và gửi dữ liệu bằng Real-Time Transfer Protocol (RTP).
- Sinh viên được cung cấp code hiện thực giao thức RTSP trong server, RTP de-packetization trong client và hiển thị video đã truyền. Nhiệm vụ của sinh viên là hiện thực giao thức RTSP trong client và RTP packetization trong server.

## II. ĐẶC TẢ CHỨC NĂNG

### 1. RTSP và RTP là gì ?

- Để truyền văn bản/tài liệu đơn giản, ta có thể dùng giao thức HTTP. Tuy nhiên, để truyền dữ liệu hiệu quả, các tầng(layers) của mạng phải được tăng cường khả năng truyền dữ liệu trong khi không làm nghẽn mạng nói chung. Các hệ thống giải trí và truyền thông(streaming media) thường truyền tệp tin đa phương tiện qua giao thức RTSP/RTP.
- RTSP là một giao thức cung cấp khung để truyền dữ liệu phương tiện theo thời gian thực ở tầng ứng dụng. Nó truyền dữ liệu thời gian thực từ đa phương tiện sang thiết bị đầu cuối bằng cách giao tiếp trực tiếp với máy chủ truyền dữ liệu. Giao thức tập trung vào việc kết nối và kiểm soát các phiên phân phối dữ liệu trên các dòng đồng bộ hóa thời gian cho phương tiện liên tục như video và âm thanh.
- RTP là một giao thức mạng để chuyển tệp tin, video, âm thanh qua mạng IP. RTP chạy trên giao thức UDP (User Datagram Protocol), RTP được sử dụng kết hợp với RTP Control Protocol (RTCP). Sử dụng RTSP để kiểm soát việc truyền phương tiện qua RTP.

### 2. Nguyên tắc hoạt động ( ứng với 6 tệp python trong thư mục chương trình ):

- **Client, ClientLauncher:** ClientLauncher khởi động Client và giao diện người dùng mà ta sử dụng để gửi các lệnh RTSP, và cũng để hiển thị video. Trong class Client, ta cần triển khai các hành động được thực hiện khi các nút được nhấn bởi người dùng. Ta không cần sửa đổi module của ClientLauncher
- **Server, ServerWorker:** Hai module này dùng để hiện thực server để phản hồi lại các yêu cầu RTSP và truyền video ngược lại. Tương tác RTSP đã được hoàn thành và ServerWorker gọi các phương thức từ class RtpPacket để điều chỉnh dữ liệu của video. Ta không cần sửa các modules này.
- **RtpPacket:** Class này dùng để xử lý các gói tin của RTP. Nó có các phương thức riêng biệt để xử lý các gói tin nhận được từ client và ta không cần phải sửa đổi chúng. Client cũng đã giải mã dữ liệu và ta không cần sửa phương thức này. Ta cần phải hoàn thành việc triển khai dữ liệu video RTPpacketization(Cái được sử dụng bởi server).
- **VideoStream:** Class này được dùng để đọc dữ liệu video từ tệp trên đĩa. Ta không cần sửa class này.

### 3. Mô tả hàm

#### **Client.sendRtspRequest(self, requestCode)**

- **Chức năng:** Gửi một request RTSP từ Client đến Server tương ứng với requestCode.
- **Input:** requestCode: Loại request cần được gửi từ Client đến Server. Trong hệ thống này, 4 loại request RTSP được sử dụng đã trình bày tại phần 2.1
- **Output:** Một request ở dạng str sẽ được tạo, sau đó được mã hoá (encode) và gửi qua RTSP socket đến Server.

### **Client.parseRtspReply(self, data)**

- **Chức năng:** Phân tích phản hồi (Reply) của Server gửi đến Client sau khi nhận được gói tin RTSP, sau đó thay đổi trạng thái của Client tương ứng với reply đó. – Input: data: Reply RTSP nhận được từ Server
- **Output:** Nếu sessionId của reply nhận được trùng với sessionId hiện tại, thay đổi trạng thái của Client tương ứng với reply đó.

### **Client.openRtpPort(self)**

- **Chức năng:** Tạo một cổng (port) mới tại Client để nhận các gói tin RTP được gửi từ Server đến Client.
- **Input:** Không có.
- **Output:** Một cổng mới với số cổng tương ứng với số cổng mà người dùng yêu cầu sẽ được tạo.

### **RtpPacket.encode(self, version, padding, extension, cc, seqnum, marker, pt, ssrc, payload)**

- **Chức năng:** Mã hoá gói tin RTP theo quy chuẩn.
- **Input:**

**version:** Phiên bản giao thức RTP đang được sử dụng. **padding:** Xác định có sử dụng padding bytes ở cuối gói tin RTP hay không.

**extension:** Xác định có sử dụng thêm extension header giữa header và payload data hay không.

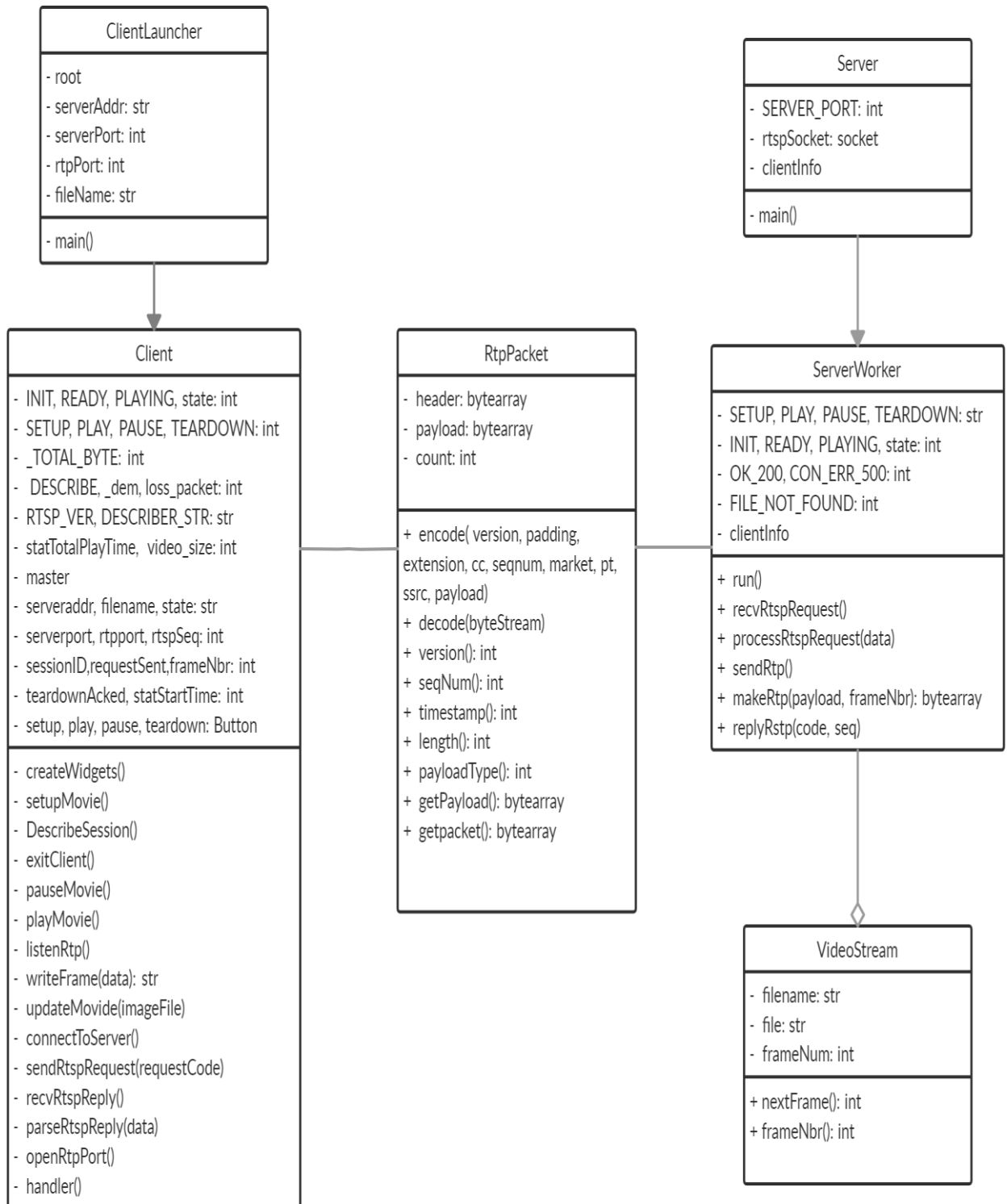
**cc:** Xác định số lượng các Contributing sources. **seqnum:** Số thứ tự của gói tin RTP hiện tại. **marker:** Bit được dùng cho mục đích đánh dấu gói tin.

**pt:** Xác định định dạng của payload data.

**ssrc:** Giá trị định danh Server. **payload:** Phần data được gửi đi trong mỗi gói tin RTP.

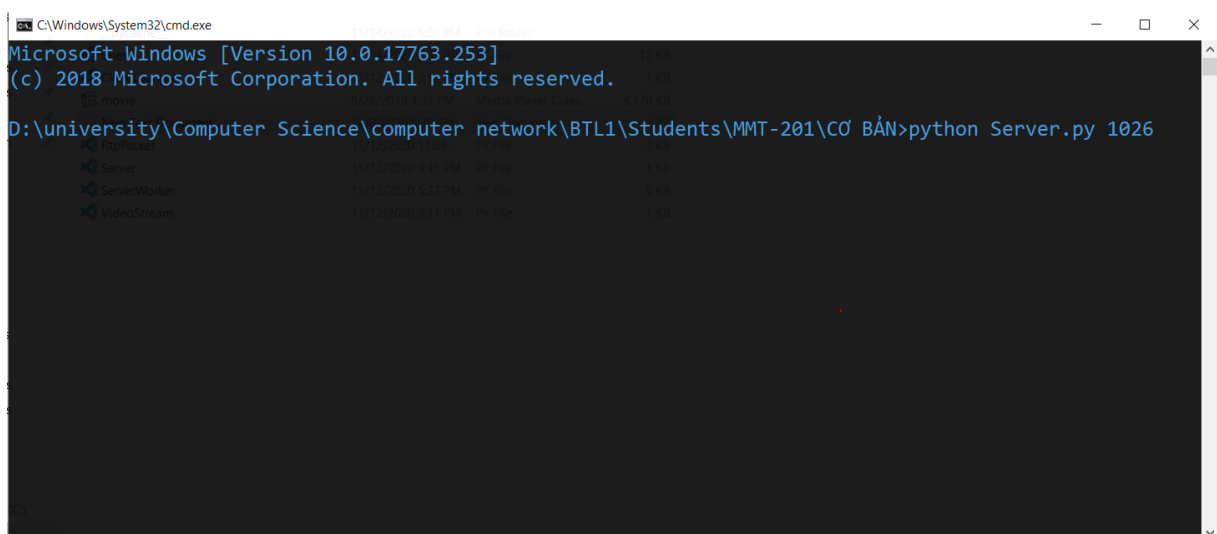
- **Output:** Một gói tin RTP đã được mã hoá theo quy chuẩn.

### III. CLASS DIAGRAM



#### IV. HƯỚNG DẪN SỬ DỤNG

1. Mở 2 terminal cùng truy cập vào thư mục chứa chương trình. Chương trình của chúng ta có 6 file là: Client.py, ClientLauncher.py, ServerWorker.py, Server.py, VideoStream.py, Rtp packet.py, cùng 1 file movie.mpjpeg là file video đã được mã hóa. Ngoài ra còn có thư mục `_pycache_` để compile code.
2. Tạo Server với dòng lệnh sau: `python Server.py server_port` Trong đó: `server_port` là cổng để server của chúng ta nhận được kết nối RTSP. Cổng tiêu chuẩn có giá trị là 554, nhưng chúng ta cần chọn `#port > 1024`. Ví dụ, chúng em chọn giá trị 1026.



Hình 1: : Thiết lập ban đầu với server

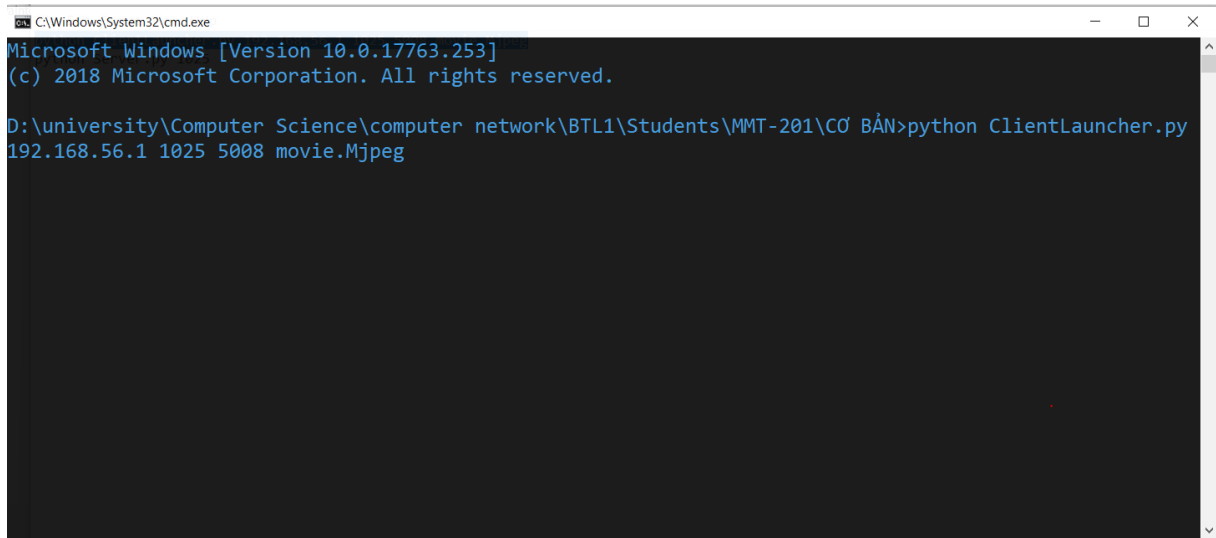
3. Khởi động client bằng lệnh sau:

*python ClientLauncher.py server\_host server\_port RTP\_port video\_file*

Trong đó:

- `server_host`: địa chỉ IP của server đang chạy.
- `server_port`: cổng kết nối với server. Ví dụ ở trên, chúng em chọn `#port 1026`
- `RTP`: cổng nơi RTP packets được nhận, ở đây là 5008

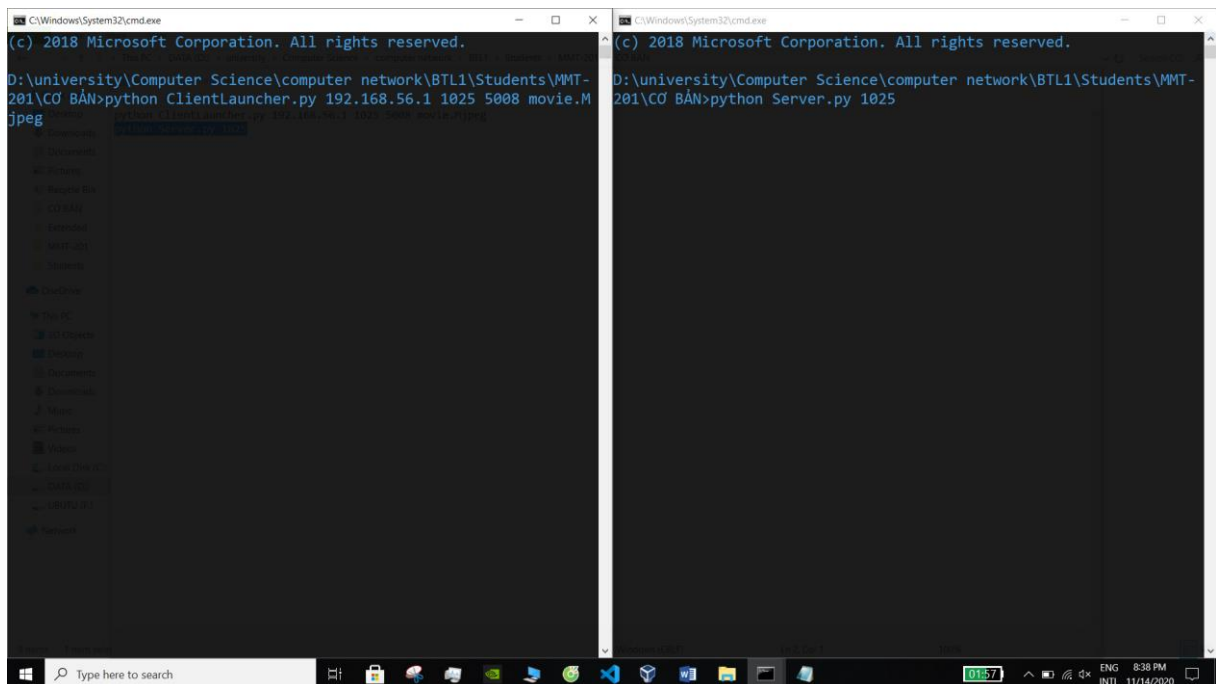
- video\_file: tên của file video ta muốn yêu cầu, ở đây là “movie.mjpeg”



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17763.253]
(c) 2018 Microsoft Corporation. All rights reserved.

D:\university\Computer Science\computer network\BTL1\Students\MMT-201\CƠ BẢN>python ClientLauncher.py
192.168.56.1 1025 5008 movie.Mjpeg
```

Hình 2 : Thiết lập ban đầu với client



```
C:\Windows\System32\cmd.exe
(c) 2018 Microsoft Corporation. All rights reserved.

D:\university\Computer Science\computer network\BTL1\Students\MMT-201\CƠ BẢN>python ClientLauncher.py 192.168.56.1 1025 5008 movie.Mjpeg

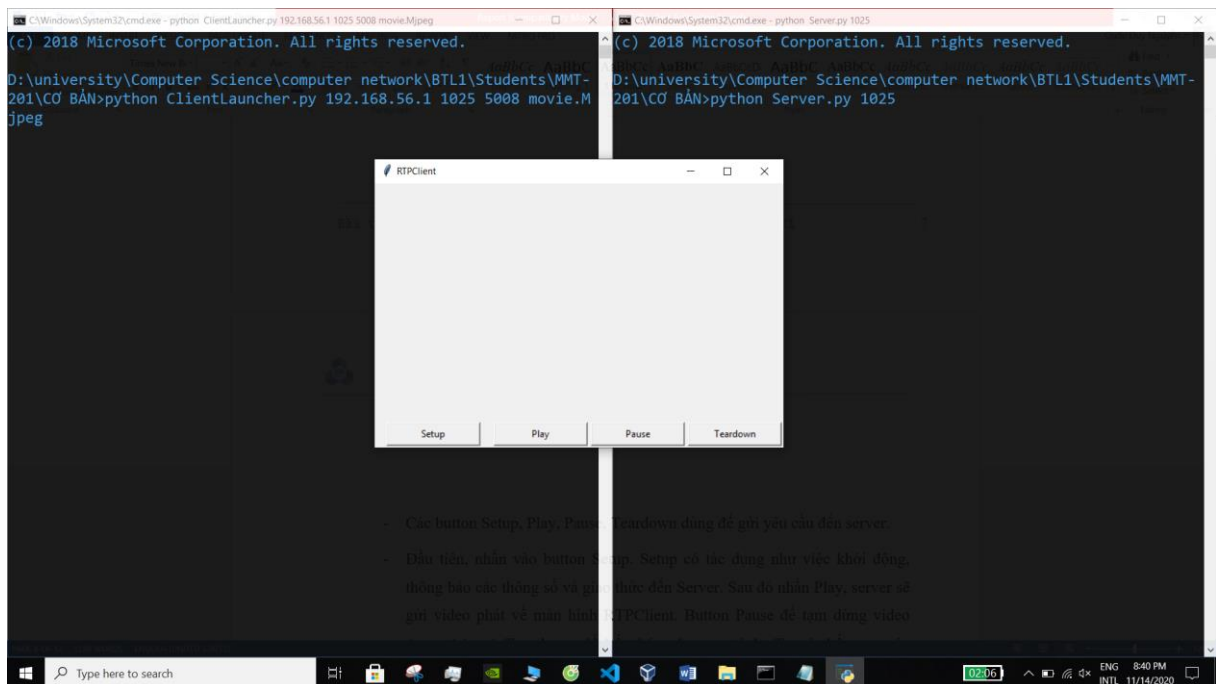
C:\Windows\System32\cmd.exe
(c) 2018 Microsoft Corporation. All rights reserved.

D:\university\Computer Science\computer network\BTL1\Students\MMT-201\CƠ BẢN>python Server.py 1025
```

Hình 3 : Thiết lập ban đầu với client và server

4 . Sau khi khởi động server (khởi động trước) và client, một giao diện xuất hiện như sau:





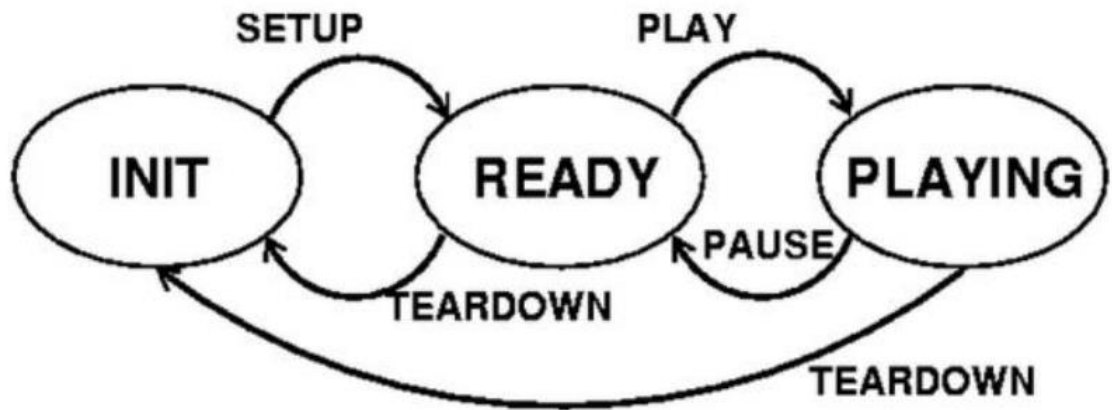
Hình 4 : Hiện giao diện video

- Các button Setup, Play, Pause, Teardown dùng để gửi yêu cầu đến server.
- Đầu tiên, nhấn vào button Setup. Setup có tác dụng như việc khởi động, thông báo các thông số và giao thức đến Server. Sau đó nhấn Play, server sẽ gửi video phát về màn hình RTPClient. Button Pause để tạm dừng video đang phát, và Teardown để kết thúc chương trình. Ta có thể xem các thông số về Session, CSeq, Data gửi đi và nhận được trên 2 terminal đã mở.

## IV. ĐÁNH GIÁ KẾT QUẢ ĐẠT ĐƯỢC

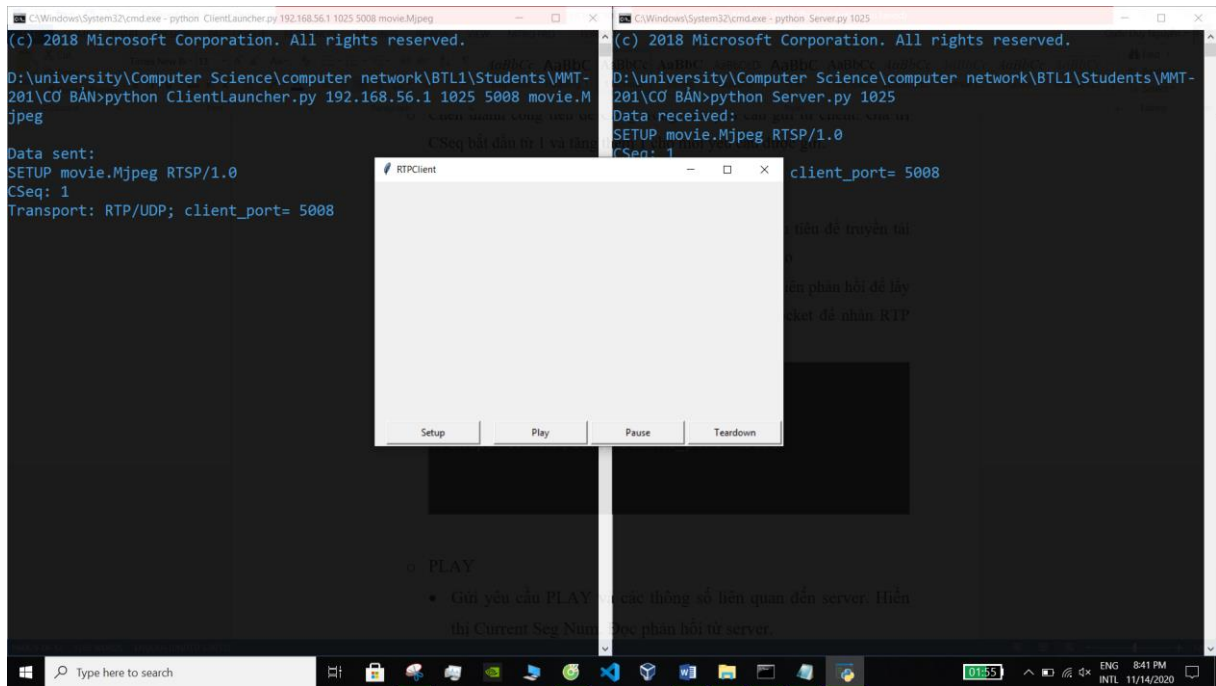
### 1. THE CLIENT

- Hiện thực thành công giao thức RTSP ở client. Hoàn thành các chức năng được gọi khi người dùng click vào các nút trên giao diện ứng dụng.
- Khi khởi động client, nó đồng thời cũng mở socket của RTSP đến server. Chúng em sử dụng thành công socket này để gửi tất cả yêu cầu RTSP
- Trong RTSP, mỗi phiên(session) đều có một trạng thái. Cập nhật thành công trạng thái cho client mỗi khi client nhận được phản hồi từ server theo sơ đồ:



- Đối với các Button:

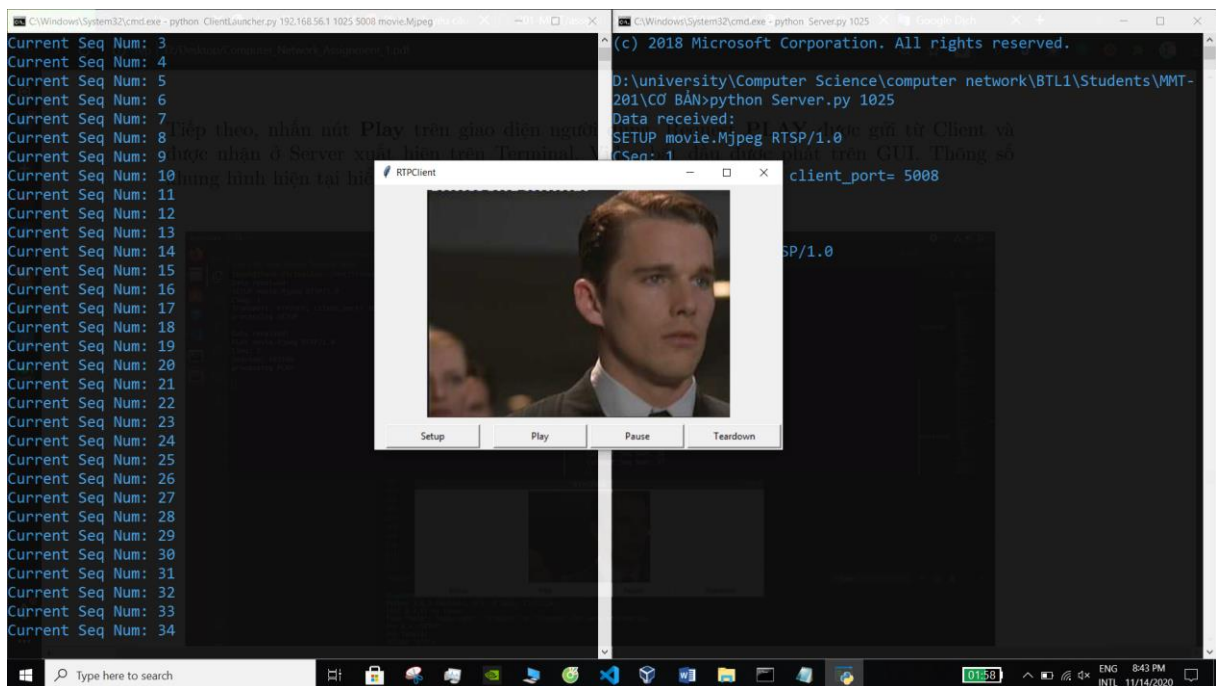
- Chèn thành công tiêu đề CSeq trong mọi yêu cầu gửi từ client. Giá trị CSeq bắt đầu từ 1 và tăng thêm 1 cho mỗi yêu cầu được gửi.
- SETUP
  - Gửi thành công yêu cầu SETUP đến server. Chèn tiêu đề truyền tải trong port được chỉ định của RTP data socket đã tạo
  - Đọc phản hồi của server và phân tích tiêu đề của phiên phản hồi để lấy giá trị phiên RTSP. Tạo thành công datagram socket để nhận RTP data.



Hình 5: Yêu cầu Setup

○ PLAY

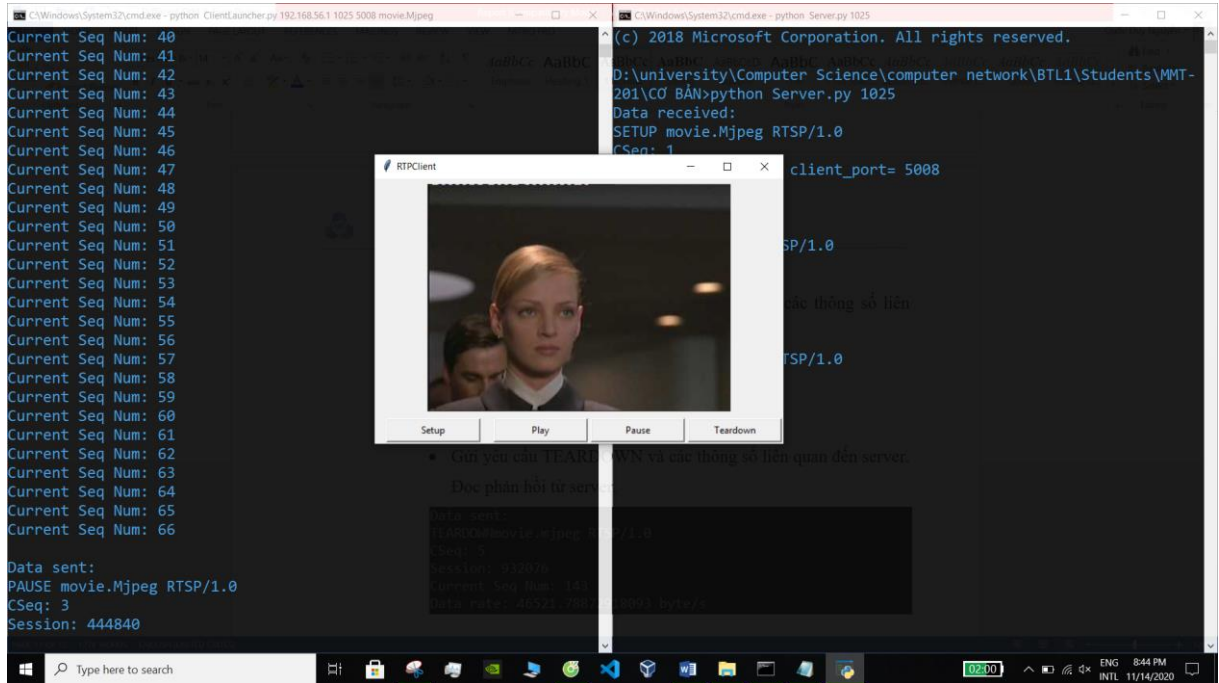
- Gửi yêu cầu PLAY (sau khi nhấn Setup) và các thông số liên quan đến server. Hiển thị Current Seg Num. Đọc phản hồi từ server.



Hình 6: Yêu cầu Play

## ○ PAUSE

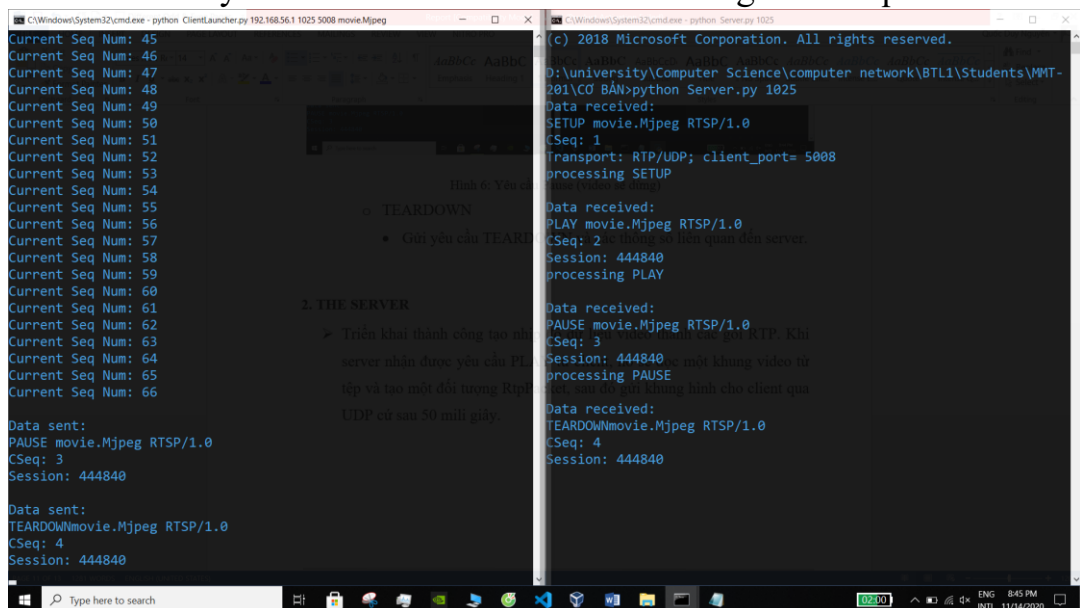
- Gửi yêu cầu PAUSE (sau khi nhấn Play) và các thông số liên quan đến server.



Hình 6: Yêu cầu Pause (video sẽ dừng)

## ○ TEARDOWN

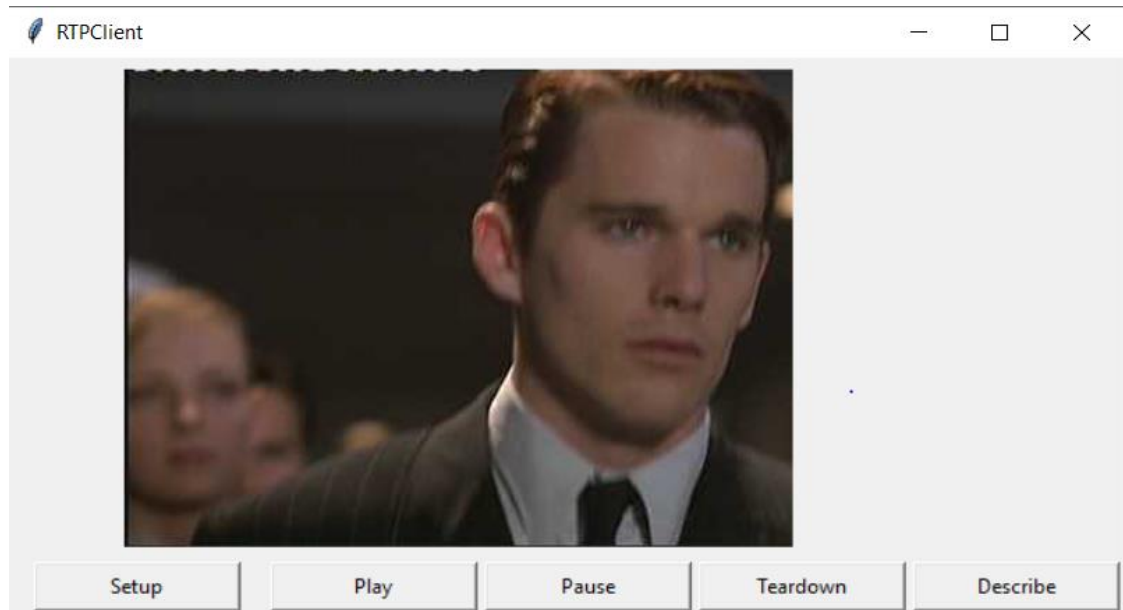
- Gửi yêu cầu TEARDOWN và các thông số liên quan đến server.



Hình 7: Yêu cầu Teardown (GUI sẽ thoát)

## 2. THE SERVER

- Triển khai thành công tạo nhịp độ dữ liệu video thành các gói RTP. Khi server nhận được yêu cầu PLAY từ client, nó sẽ đọc một khung video từ tệp và tạo một đối tượng RtpPacket, sau đó gửi khung hình cho client qua UDP cứ sau 50 mili giây.



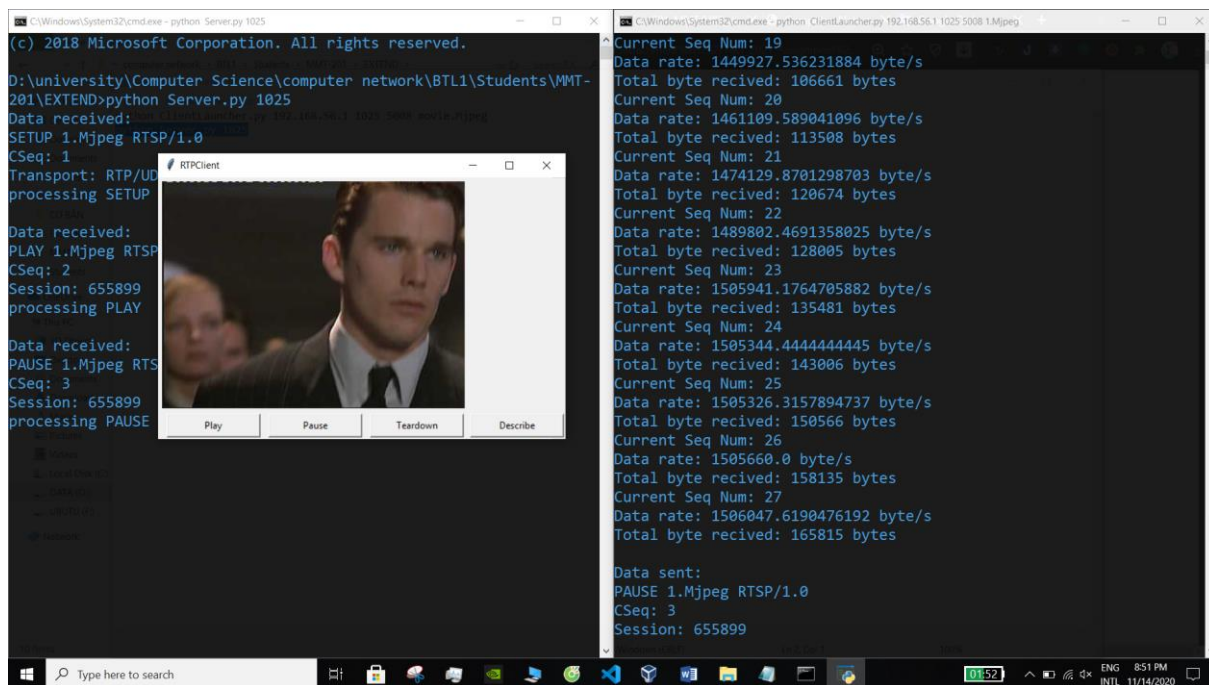
- Đọc các yêu cầu từ client và đưa ra các phản hồi về client. Hiển thị tên Button mà user vừa click xong.

```
Command Prompt - python Server.py 1025  
C:\Users\Cyrus\Máy tính\Mạng máy tính\Test1>python Server.py 1025  
Data received:  
SETUP movie.mjpeg RTSP/1.0  
CSeq: 1  
Transport: RTP/UDP; client_port= 5008  
processing SETUP  
  
Data received:  
PLAY movie.mjpeg RTSP/1.0  
CSeq: 2  
Session: 932076  
processing PLAY  
  
Data received:  
PAUSE movie.mjpeg RTSP/1.0  
CSeq: 3  
Session: 932076  
processing PAUSE  
  
Data received:  
PLAY movie.mjpeg RTSP/1.0  
CSeq: 4  
Session: 932076  
processing PLAY  
  
Data received:  
TEARDOWNmovie.mjpeg RTSP/1.0  
CSeq: 5  
Session: 932076
```

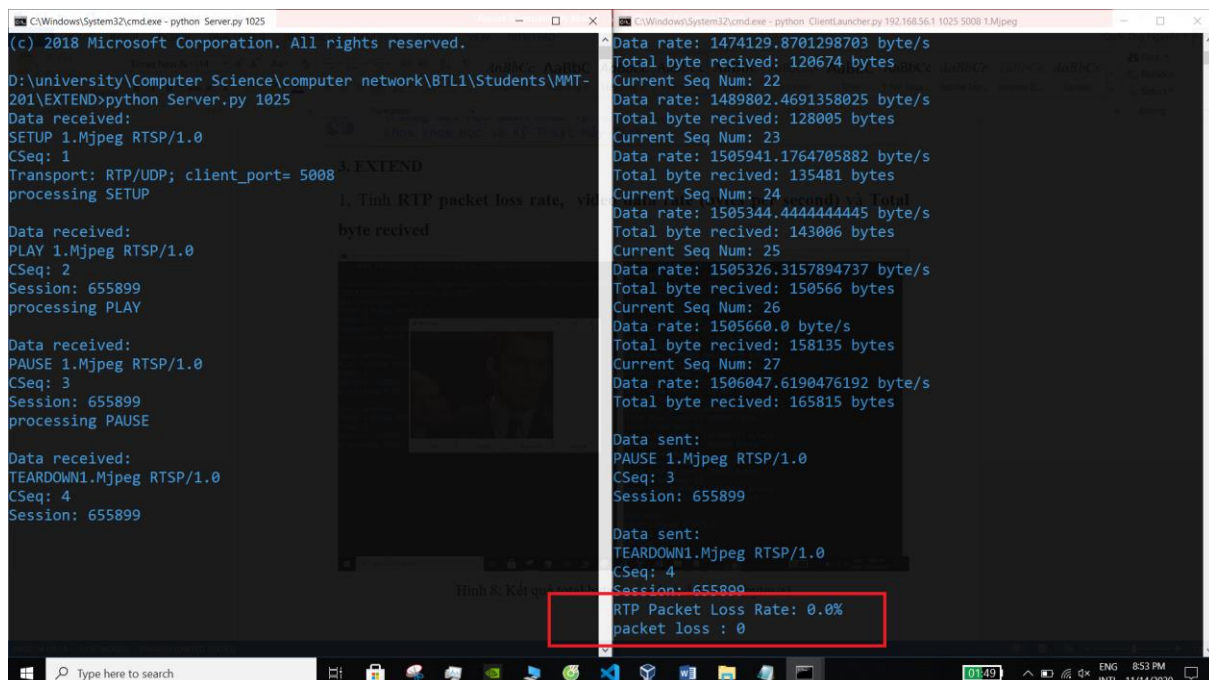


### 3. EXTEND

#### 1. Tính RTP packet loss rate, packet loss, video data rate (bytes per second) và Total byte received

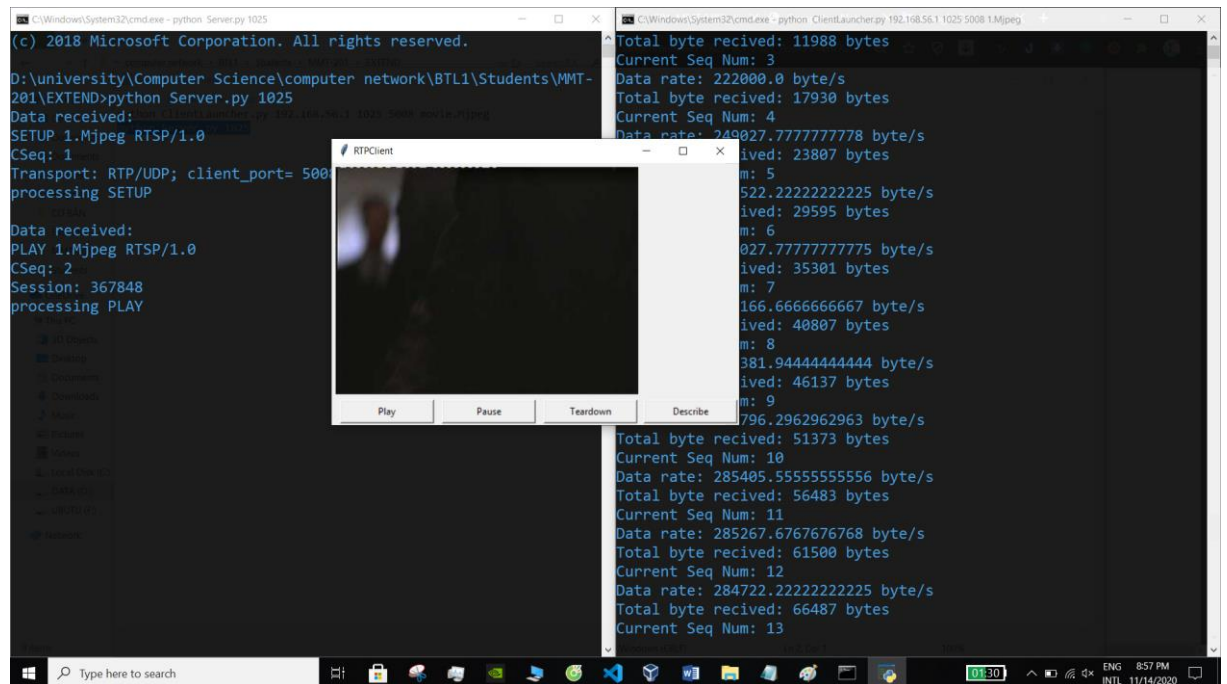


Hình 8: Kết quả total byte received và data rate (bytes/s)



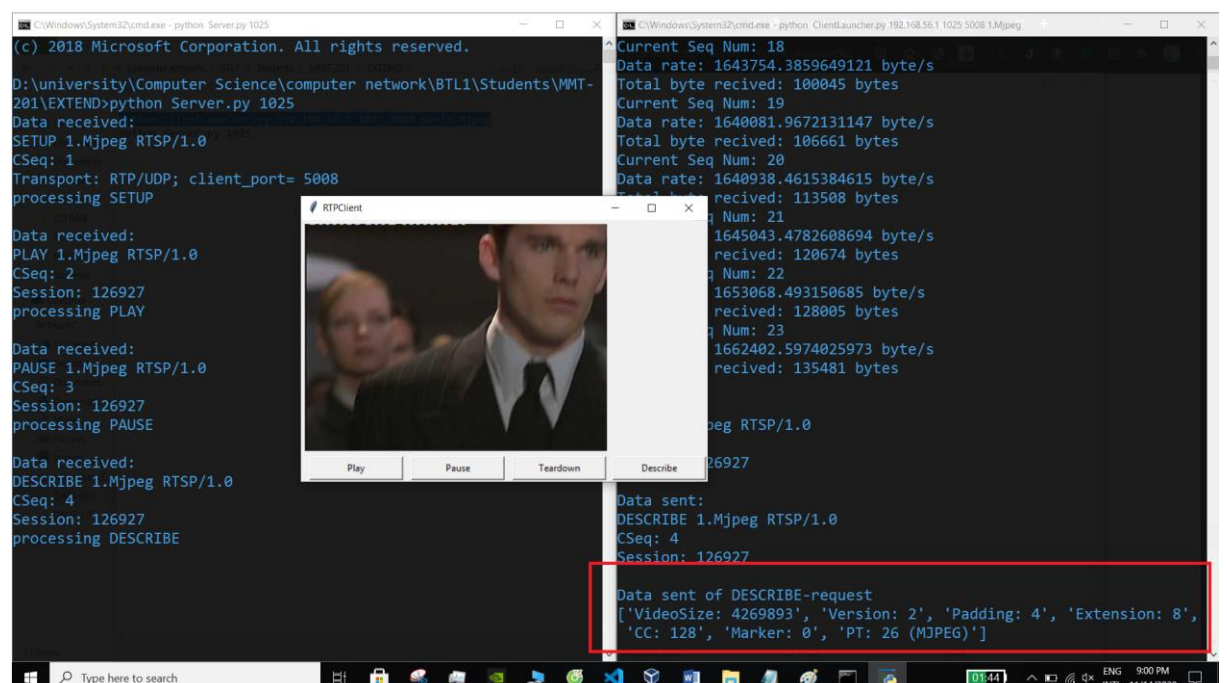
Hình 9: Kết quả packet loss và RTP Packet Loss Rate

2. Chúng ta sẽ cài mặc định Setup luôn có sẵn khi kết nối đối với windows media player. Sau đây là kết quả



Hình 10: Kết quả không cần Setup, video sẽ chạy như windows media player

3. Chúng ta thiết kế button DESCRIBE và Server sẽ gửi thông tin encodings mà sử dụng.



Hình 11: Kết quả khi nhấn button DESCRIBE





## V. Link code:

Code được lưu vào github: <https://github.com/nqqduy/MMT-201/tree/master>

Link code cho bài cơ bản:

<https://github.com/nqqduy/MMT-201/tree/master/C%C6%A0%20B%E1%BA%A2N>

Link code cho bài mở rộng:

<https://github.com/nqqduy/MMT-201/tree/master/EXTEND>