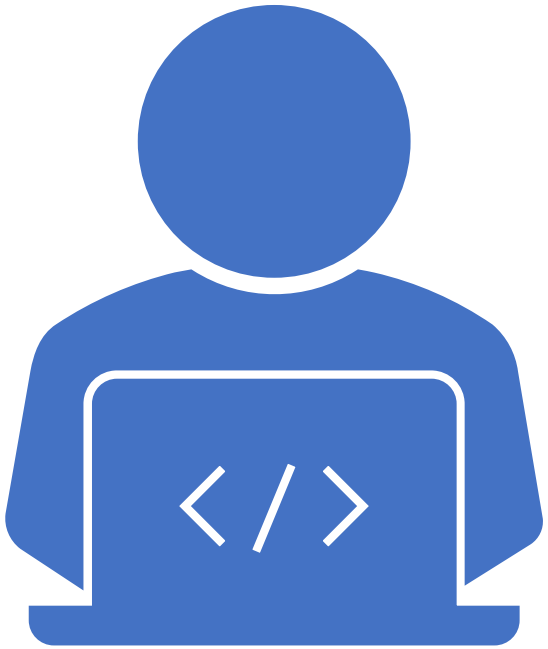
CAN 201

Dr. Fei Cheng & Dr. Gordon Boateng

Lab 5
UDP/TCP programming
+ WireShark
&
File operation (python)



Demo for Practice of last week

https://box.xjtlu.edu.cn/f/45142cdef93641238ef0/

UDP + WireShark:

a long video, you should watch it!
Not only look at the PPT!!!

TCP + WireShark:

a long video, you should watch it!
Not only look at the PPT!!!

You need to show TA what you did using wireshark to monitor your UDP and TCP code.

File operation

"***open***" method is exploit to operate files:

- `f = open(file='filename.xxx', mode='r')`
  - "open" method open a file and return a stream (a file object)
  - `file` is a path of filename:
    - if the file is in the current working directory, relative path: eg. record.txt, or folder_a/record.txt
    - Otherwise, the full path (absolute path) : eg. (Win)D:\your_dir\record.txt or (Mac)/Users/your_home/record.txt
  - `mode` specifies the mode in which the file is opened.

current working directory: default is the folder that has the current executing .py file. You can also change the CWD.

File open modes

Character	Meaning
‘r’	open for reading (default)
‘w’	open for writing, truncating the file first
‘x’	create a new file and open it for writing
‘a’	open for writing, appending to the end of the file if it exists
‘b’	binary mode, used for media files
't'	text mode (default)
'+'	open a disk file for updating (reading and writing)
'U'	universal newline mode (deprecated)

* We used the red modes normally

Text file write

- The following methods can be used:
  - `f.write(Str)  # Write a str to a file`
  - `f.writelines(StrList)  # Write a list of str to a file`

```
cont = ['Hello, Li Lei.\n', 'Hello, Han Meimei.\n',
        'How are you?\n',
        'Fine, thank you. And you?\n',
        'I am fine too.\n']
f = open('/Users/fei/lesson1.txt', 'w')
# Write all the list once
f.writelines(cont)
# Write line by line
for l in cont:
    f.write(l)

f.close()
```

Text file read

- The following methods can be used:
  - `f.read(n) # read n chars(including \n), n=-1 means read all`
  - `f.readline()  # read one line (including \n)`
  - `f.readlines()  # return a list of all lines`

```
f = open('/Users/fei/lesson1.txt', 'r')
print(f.read())   # Read all contents

f.seek(0)  # Set the file pointer to the beginning
print(f.read(10))  # Read 10 chars

f.close()
```

```
# Use readline
f = open('/Users/fei/lesson1.txt', 'r')

while True:
    line = f.readline()   # Read one line
    if line:  # if the line is not empty, print, otherwise break
        print(line)
    else:
        break

f.close()

# Use readlines
f = open('/Users/fei/lesson1.txt', 'r')

cont = f.readlines()
# Iterate all the lines
for line in cont:
    print(line)

f.close()
```

Text file appending

- Open a file with ‘a’ mode. Then, write something. The new content will append to the end of that file. If there is no that file, it will create a new one. It is very useful to record logs.

```
f = open('/Users/fei/lesson1.txt', 'a')

f.write('Bye.\n')

f.close()
```

Binary file read

- For binary files, still we have those 3 methods to read them:
  - `f.read(n) # read n bytes(including \n), n=-1 means read all`
  - `f.readline()  # read one line (including \n)`
  - `f.readlines()  # return a list of all lines`

```
# Open with text read mode
f = open('/Users/fei/lesson1.txt', 'r')
print(type(f.read()))
f.close()

# Open with binary read mode, what's the difference?
f = open('/Users/fei/lesson1.txt', 'rb')
print(type(f.read()))
f.close()
```

Binary file write

- The following methods can be used:
  - `f.write(bytes)  # Write some bytes to a file`
- Binary file operation is very powerful, but we don’t go deep today.
  - We just open a binary file (jpeg image), read all the data, ant then write the same data to another jpg file.

```
f = open('/Users/fei/a.jpg', 'rb')
# Read all the binary bytes from a.jpg
image_data = f.read()

f.close()

# Write to a new jpg file
f = open('/Users/fei/aa.jpg', 'wb')
f.write(image_data)
f.close()
```

File operation demo

https://box.xjtlu.edu.cn/f/591c52bf8ab9464f8032/

Lab practice 2

- Design a pair of UDP based server and client programs:
  - Client sends a small jpeg image, named xjtlu.jpg to server side;
  - Server receives the image data, and save it as xjtlu1.jpg on server;
  - Server sends back a small jpeg image, named xjtlu1.jpg to client;
  - Client receives the image data, and save it as xjtlu2.jpg on client.
- Design a pair of TCP based server and client program, the requirements are the same as above.
- xjtlu.jpg: https://box.xjtlu.edu.cn/f/9cea9c68568942a5b970/?dl=1
- No special module(lib) is need. (opencv is not needed!)
- Your codes should be executed on different machines (one real PC and one virtual machine, or two virtual machines).
- No submission

Thanks