How to Transfer Files in the Network using Sockets in Python

Writing a server and client Python scripts that receives and sends files in the network using sockets module in Python.

Abdou Rockikz · 8 min read · Updated Feb 2021 · 51.6K · Python Standard Library

Tags

Disclosure: This post may contain affiliate links, meaning when you click the links and make a purchase, we receive a commission.

File transfer is the process of copying or moving a file from a computer to another over a network or Internet connection. In this tutorial we'll go step by step on how you can write client/server Python scripts that handles that.

The basic idea is to create a server that listens on a particular port, this server will be responsible for receiving files (you can make the server sends files as well). On the other hand, the client will try to connect to the server and send a file of any type.

We are going to use `socket` module which comes built-in with Python and provides us with socket operations that are widely used on the Internet, as they are behind of any connection to any network.

Related: *How to Send Emails in Python*.

First, we gonna need to install `tqdm` which will enable us to print fancy progress bars:

```
pip3 install tqdm
```

Client Code

Let's start with the client, the sender:

```
import socket
import tqdm
import os

SEPARATOR = "<SEPARATOR>"
BUFFER_SIZE = 4096 # send 4096 bytes each time step
```

We need to specify the IP address and the port of the server we want to connect to, and also the name of the file we want to send.

```
# the ip address or hostname of the server, the receiver
host = "192.168.1.101"
# the port, let's use 5001
port = 5001
# the name of file we want to send, make sure it exists
filename = "data.csv"
# get the file size
filesize = os.path.getsize(filename)
```

The `filename` needs to exist in the current directory, or you can use an absolute path to that file somewhere in your computer. This is the file you want to send.

`os.path.getsize(filename)` gets the size of that file in bytes, that's great, as we need it for printing progress bars in the client and the server.

Let's create the TCP socket:

```
# create the client socket
s = socket.socket()
```

Connecting to the server:

```
print(f"[+] Connecting to {host}:{port}")
s.connect((host, port))
print("[+] Connected.")
```

`connect()` method expects an address of the pair `(host, port)` to connect the socket to that remote address. Once the connection is established, we need to send the name and size of the file:

```
# send the filename and filesize
s.send(f"{filename}{SEPARATOR}{filesize}".encode())
```

I've used `SEPARATOR` here just to separate the data fields, it is just a junk message, we can just use

Machine Learning
Ethical Hacking
General Python Tutorials
Web Scraping
Computer Vision
Python Standard Library
Application Programming Interfaces
Database
Finance
Packet Manipulation Using Scapy
Natural Language Processing
Healthcare
Web Programming

New Tutorials

Webhooks in Python with Flask
How to Create a Watchdog in Python
How to Change Text Color in Python
Conversational AI Chatbot with Transformers in Python
Asynchronous Tasks with Celery in Python

Popular Tutorials

How to Encrypt and Decrypt Files in Python
How to Convert Speech to Text in Python
How to Read Emails in Python
How to Transfer Files in the Network using Sockets in Python
How to Get Hardware and System Information in Python

`send()` twice, but we may don't wanna do that anyways. `encode()` function encodes the string we passed to `'utf-8'` encoding (that's necessary).

Now we need to send the file, and as we are sending the file, we'll print nice progress bars using `tqdm` library:

```
# start sending the file
progress = tqdm.tqdm(range(filesize), f"Sending {filename}", unit="B", unit_sc
with open(filename, "rb") as f:
    while True:
        # read the bytes from the file
        bytes_read = f.read(BUFFER_SIZE)
        if not bytes_read:
            # file transmitting is done
            break
        # we use sendall to assure transimission in
        # busy networks
        s.sendall(bytes_read)
        # update the progress bar
        progress.update(len(bytes_read))
# close the socket
s.close()
```

Basically what we are doing here is opening the file as read in binary, read chunks from the file (in this case, `4096` bytes or 4KB) and send them to the socket using `sendall()` function, and then we update the progress bar each time, once that's finished, we close that socket.

Related: *How to Make a Chat Application in Python*.

Server Code

Alright, so we are done with the client. Let's dive into the server, so open up a new empty Python file and:

```
import socket
import tqdm
import os
# device's IP address
SERVER_HOST = "0.0.0.0"
SERVER_PORT = 5001
# receive 4096 bytes each time
BUFFER_SIZE = 4096
SEPARATOR = "<SEPARATOR>"
```

I've initialized some parameters we gonna use, notice that I've used `"0.0.0.0"` as the server IP address, this means all IPv4 addresses on the local machine. You may wonder, why we don't just use our local IP address or `"localhost"` or `"127.0.0.1"` ? Well, if the server has two IP addresses,

let's say `"192.168.1.101"` on a network, and `"10.0.1.1"` on another, and the server listens on `"0.0.0.0"`, it will be reachable at both of those IPs.

Alternatively, you can use either your public or private IP address, depending on your clients. If the connected clients are in your local network, you should use your private IP (you can check it using `ipconfig` command in Windows or `ifconfig` command in Mac OS/Linux), but if you're expecting clients from the Internet, you definitely should use your public address.

Also, Make sure you use the same port in the server as in the client.

Let's create our TCP socket:

```
# create the server socket
# TCP socket
s = socket.socket()
```

Now this is different from the client, we need to bind the socket we just created to our `SERVER_HOST` and `SERVER_PORT`:

```
# bind the socket to our local address
s.bind((SERVER_HOST, SERVER_PORT))
```

After that, we gonna listen for connections:

```
# enabling our server to accept connections
# 5 here is the number of unaccepted connections that
# the system will allow before refusing new connections
s.listen(5)
print(f"[*] Listening as {SERVER_HOST}:{SERVER_PORT}")
```

Once the client connects to our server, we need to accept that connection:

```
# accept connection if there is any
client_socket, address = s.accept()
# if below code is executed, that means the sender is connected
print(f"[+] {address} is connected.")
```

Remember that when the client is connected, it'll send the name and size of file, let's receive them:

```
# receive the file infos
# receive using client socket, not server socket
received = client_socket.recv(BUFFER_SIZE).decode()
filename, filesize = received.split(SEPARATOR)
# remove absolute path if there is
filename = os.path.basename(filename)
# convert to integer
filesize = int(filesize)
```

As mentioned earlier, the received data is combined of the `filename` and and the `filesize`, we can easily extract them by splitting by `SEPARATOR` string.

After that, we need to remove the absolute path of the file, that's because the sender sent the file with his own file path, which may differ from ours, `os.path.basename()` returns the final component of a path name.

Now we need to receive the file:

```
# start receiving the file from the socket
# and writing to the file stream
progress = tqdm.tqdm(range(filesize), f"Receiving {filename}", unit="B", unit_
with open(filename, "wb") as f:
    while True:
        # read 1024 bytes from the socket (receive)
        bytes_read = client_socket.recv(BUFFER_SIZE)
        if not bytes_read:
            # nothing is received
            # file transmitting is done
            break
        # write to the file the bytes we just received
        f.write(bytes_read)
        # update the progress bar
        progress.update(len(bytes_read))

# close the client socket
client_socket.close()
# close the server socket
s.close()
```

Not quite different from the client code. However, we are opening the file as write in binary here, and using `recv(BUFFER_SIZE)` to receive `BUFFER_SIZE` bytes from the client socket and write it to the file. Once that's finished, we close both the client and server sockets.

Alright, let me try it on my own private network:

```
C:\> python receiver.py

[*] Listening as 0.0.0.0:5001
```

I need to go to my Linux box and send some example file:

```
root@rockikz:~/tools# python3 sender.py
[+] Connecting to 192.168.1.101:5001
[+] Connected.
Sending data.npy:   9%|
```

Let's see the server now:

```
[+] ('192.168.1.101', 47618) is connected.
Receiving data.npy:  33%|
```

Great, we are done!

You can extend this code for your own needs now, here are some examples you can implement:

- Enabling the server to receive multiple files from multiple clients in the same time using threads.
- Compressing the files before sending them.
- Encrypting the file before sending it, to ensure that no one has the ability to intercept and read that file, this tutorial will help.
- Ensuring the file is sent properly by checking the checksums of both files (the original file of the sender and the sent file in the receiver). In this case, you need secure hashing algorithms to do it.
- Adding a chat room so you can both chat and transfer files.

Finally, if you're a beginner and want to learn Python, I suggest you take Master Python in 5 Online Courses from University of Michigan, in which you'll learn a lot about Python, good luck!

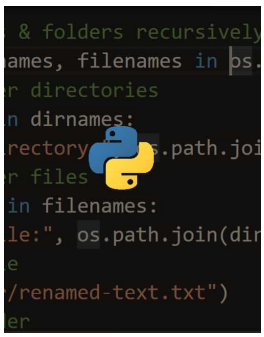
Read Also: *How to Manipulate IP Addresses in Python using ipaddress Module.*

Happy Coding ♥

VIEW FULL CODE

Sharing is caring!

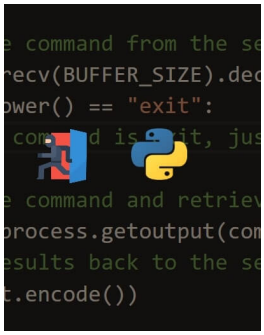
Read Also



How to Handle Files in Python

Learn how to work with files in Python using os and shutil modules including creating, renaming, moving, removing files and directories, listing all current files and directories and more.

VISIT →



How to Create a Reverse Shell in Python

Building a reverse shell in Python using sockets that can execute remote shell commands and send the results back to the server.

VISIT →

How to Download Files from URL in Python

Learn how to use requests and tqdm libraries to build a powerful file downloader with progress bar using Python.

VISIT →

Follow @ThePythonCode

Comment panel

MAB a year ago

Hi,May I know how to find my ip address in kali linux 2020.1b?

REPLY

Abdou Rockikz 7 months ago

Hi there, Yes, you can via "ifconfig" command, look at your interface and use that IP!

REPLY

Edward Arrow a year ago

Can I use this to implement files transfer from linux server to windows desktop

REPLY

Abdou Rockikz 7 months ago

Hello Edward, Of course you can, they just need to install Python !

REPLY

Ali a year ago

hi

REPLY

does the tqdm progress bar work fine here,
in my case it move back and forth till the progress bar end

Abdou Rockikz 7 months ago
Hello Ali, I'm not sure why it resets in the end, I will look into that, thanks for pointing out!
REPLY

Abdou Rockikz 5 months ago
Hi Ali,
I've fixed the progress bar reset, check it out!
REPLY

cemre 10 months ago
Hi! Can I send a file from my ubuntu virtual box to host, windows ?
REPLY

Abdou Rockikz 7 months ago
Hello Cemre,
Of course you can do that, just make sure you have network access in the virtual machine and then you get the IP address of both machines and do the job.
However, for your case, there might be more quick and easy solution, which is installing VirtualBox guest additions and then you can only drag and drop files.
REPLY

Riley N 9 months ago
Hello! I tried to send a .mp3 but it sent it as 0 bytes Any help? Thanks!
REPLY

Abdou Rockikz 7 months ago
Hey Riley, I need some more information, is the connection established normally ?
If so, how was the output in both the server and client.
REPLY

Vincent 9 months ago
Hi, nice post, but i have a cuestion, is it posibble to use this code to pass information from my PC divice to a Server host (cPanel), i've already have Python on both, but i'm having troubles when i want to connect them.
REPLY

Abdou Rockikz 7 months ago
Hello Vincent, it is worth to point out that you need open/allow the port you're using in the server side in the firewall, otherwise, the client won't be able to connect.
REPLY

Abdou Rockikz 7 months ago
You only do that when the server and the client are in different networks, like in your case.
REPLY

Sebastian 9 months ago
Hi! Can you help me by telling me why do I get: UnicodeDecodeError: 'utf-8' codec can't decode byte 0xff in position 0: invalid start byte on server?
Thank you!
REPLY

Abdou Rockikz 7 months ago
Hello Sebastian,
REPLY

I'm not sure why, can you send the whole output along with the line?
If the comment section doesn't fit, consider contacting here:
https://www.thepythoncode.com/contact
And I'll get back to you soon!

Ryan 9 months ago REPLY
Hello!
I'm getting a ValueError on the server side saying "ValueError: not enough values to unpack (expected 2, got 1)"

If you can help that'd be awesome!

Abdou Rockikz 7 months ago REPLY
Hello Ryan,
I would like to know which line this occured ? Can you send the full error traceback?
If it doesn't fit here in the comments section, please contact me here:
https://www.thepythoncode.com/contact

Shank 9 months ago REPLY
Hey, how can I request files from the client side?

Abdou Rockikz 7 months ago REPLY
Hello Shank, sorry for the late response. But, you can always reverse the order, making the receiver as the client and the server as the sender, and that will be okey!
Good luck on that.

gonelastvirus 8 months ago REPLY
Hey can you help to send text and image like you have send filename and file size? please help me i need to complete my project. I am stuck with that problem

Abdou Rockikz 7 months ago REPLY
Hey there,
We have a tutorial where we create a chat application, which includes sending plain text, check it out: https://www.thepythoncode.com/article/make-a-chat-room-application-in-python
You can use the code of this tutorial to send images normally, I hope you figure it out!

basak 8 months ago REPLY
in this process, we have used TCP-IP am i right?

Abdou Rockikz 7 months ago REPLY
Hey Basak,
Yes, you're absolutely right, we're creating a TCP socket using the port we want and sending files across it!

Mihai Visan 7 months ago

REPLY

Hi I was wondering what I would need to do if I wanted to use this code to transfer files from one raspberry pi device to another when they are connected to the same network via WiFi. Would I need to change the Server Host IP address and if yes then what would I change it to?

Abdou Rockikz 7 months ago

REPLY

Hello Mihai,
Yes you can absolutely transfer files in that case, in the server code, server IP address can stay either 0.0.0.0 or the actual private IP address of the server.
But the most important, is you need to change the `host` in client code to the private IP address of the server.
Hope this helps!

Panam Shah 7 months ago

REPLY

My Progress Bar is getting reset after 100% completion of file transfer. Until then its progressing as it should.

Abdou Rockikz 7 months ago

REPLY

Hello Panam, I'm not sure why it resets in the end, I will look into that, thanks for pointing out!

Abdou Rockikz 5 months ago

REPLY

Hi Panam,
I've fixed the progress bar reset bug, by simply changing in the loop "for _ in progress" to "while True:"
Hope this helps,

shawn 7 months ago

REPLY

Thank you, this works fine. I wish to add a GUI to the client code such that the interface will ask to select the desired file and send them, instead of mentioning the filename in the code. How can I do that?

Abdou Rockikz 6 months ago

REPLY

Hello Shawn,
You can easily use tkinter's filedialog:

```
import tkinter as tk
from tkinter import filedialog
root = tk.Tk()
root.withdraw()
file_path = filedialog.askopenfilename()
```

This will open your file explorer to the file you want to specify, and then file_path will include the file absolute path you specified, good luck!

Eric Hom 6 months ago

REPLY

Hi! I wonder if it is possible to adjust the code to make it transfer files between virtual machines? Also if it is possible to make a shared folder, which is shared by 3 virtual machines and when a file is uploaded by one of the vms then the other two will automatically download it?THX!

Abdou Rockikz 6 months ago

REPLY

Hey, once you're able to make connection between the virtual machines, you can easily run the code to transfer files, just make sure you use reachable IP addresses.

For your second question, yes you can make such server using Python that is able to do what you asked for, good luck!

JimT 5 months ago

REPLY

How to change the client/server code to remove the progress bar?

for _ in progress: changes to ? Thanks

Abdou Rockikz 5 months ago

REPLY

Hi Jim,

All you have to do is:

1 - remove the progress = ... line

2 - instead of:

for _ in progress:

you do:

while True:

3 - And you must remove the line progress.update... inside the loop.

After you do these step, the progress bar should disappear, make sure to remove tqdm imports as well.

Also, If you wanted to remove the progress bar because of the reset in the end, I just fixed it!

user614 5 months ago

REPLY

Hello.... But I can't transfer big .mp4 files (like 200Mb or more) with this. The transfer stops at some 24% and after that it shows 0%. Hence only a small portion of the file is available at the receiver system. Is there any limit to the data file size here? How do I fix this?

Thank you.

Abdou Rockikz 5 months ago

REPLY

Hey there,

Not sure why that happened, it should work for any size, just make sure the connection is stable on both ends, and also the disk storage is enough to store that in the receiver.

akanjigbolahan 5 months ago

REPLY

Hello! How can I send multiple files at the same time from client to server? For example, i want to send 4 files at a time (in batch) from a folder which contains many files.

Abdou Rockikz 3 months ago

REPLY

Hi there,

In this case, you have two ways:

1. This is the easiest one, you should zip the 3 files into one zip file and then send it.

2. You need to include a loop in both client and server codes, and this will require code changing with your Python programming skills :)

NISHIKANT 4 months ago

REPLY

i want to put inside a command ,how can i do that?

Abdou Rockikz 3 months ago

REPLY

Hi there,

I didn't quite understand what you want to achieve, where inside ? and what command ?

Thanks.

lionnoir35 3 months ago

REPLY

hello I'm trying to download a 25mb file May at 96% nothing happens what to do please

Abdou Rockikz 2 months ago

REPLY

Hi there,

Please send detailed information about your problem here:

https://www.thepythoncode.com/contact

I may help you.

Thanks,

user51 29 days ago

REPLY

Hi, thanks for the tutorial. I found the code works when the server and client are in the same network, i.e., connect to the same router. However, when they are on a different network and I change the host name to the server's public IP address, it's not working. Do you have any suggestions?

Abdou Rockikz 6 days ago

REPLY

Hi there,

An important step before trying to connect across Internet is that you need to allow the port number on your firewall, on a hosted box, you can use the ufw command, such as:

$ ufw allow 5001

If you're on home setup, then you need to enable the port 5001 (or whatever you used in the code) on your router and map it to your private local IP, you usually find it in Port Forwarding tab on the router web page.

ver starts).

Hope this helps!

Your email address will not be published.

Enter your name

Enter Email

Write a comment...

I'm not a robot

reCAPTCHA
Privacy - Terms

Subscribe for our newsletter

COMMENT