# Bruteforce\_client

\*\*Bruteforce - client.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:  
 client\_socket.connect((HOST, PORT))  
 print(f'Connected to {HOST}:{PORT}. Type "bye" to exit.')  
   
 while True:  
 message = input("You: ")  
 client\_socket.sendall(message.encode())  
 data = client\_socket.recv(1024)  
 print(f'Server: {data.decode()}')  
 if message.lower() == 'bye':  
 break

# Bruteforce\_server

\*\*Bruteforce - server.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
import itertools  
import string  
  
def bruteforce\_attack(password):  
 chars = string.printable.strip()  
 attempts = 0  
 for length in range(1, len(password) + 1):  
 for guess in itertools.product(chars, repeat=length,):  
 attempts += 1  
 guess = ''.join(guess)  
 if attempts>=2000000:  
 break  
 if guess == password:  
 return (attempts, guess)  
 return (attempts, None)  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as server\_socket:  
 server\_socket.bind((HOST, PORT))  
 server\_socket.listen()  
 print(f'Server started. Listening on {HOST}:{PORT}')  
   
 while True:  
 conn, addr = server\_socket.accept()  
 with conn:  
 print(f'Connected by {addr}')  
 while True:  
 data = conn.recv(1024)  
 if not data:  
 break  
 print(f'Client: {data.decode()}')  
 pt=data.decode()  
 attempts, guess = bruteforce\_attack(pt)  
 if attempts<2000000:  
 if guess:  
 response=f"Password cracked in {attempts} attempts. The password is {guess}."  
 else:  
 respone=f"Password not cracked after {attempts} attempts."  
 else:  
 response=f"Password not cracked within 2000000 attempts."  
 conn.sendall(response.encode())  
 if data.decode().lower() == 'bye':  
 break  
 print(f'Client {addr} disconnected')

# caesar\_client

\*\*caesar - client.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:  
 client\_socket.connect((HOST, PORT))  
 print(f'Connected to {HOST}:{PORT}. Type "bye" to exit.')  
   
 while True:  
 message = input("You: ")  
 client\_socket.sendall(message.encode())  
 data = client\_socket.recv(1024)  
 print(f'Server: {data.decode()}')  
 if message.lower() == 'bye':  
 break

# caesar\_server

\*\*caesar - server.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as server\_socket:  
 server\_socket.bind((HOST, PORT))  
 server\_socket.listen()  
 print(f'Server started. Listening on {HOST}:{PORT}')  
   
 while True:  
 conn, addr = server\_socket.accept()  
 with conn:  
 print(f'Connected by {addr}')  
 while True:  
 data = conn.recv(1024)  
 if not data:  
 break  
 print(f'Client: {data.decode()}')  
 pt=data.decode()  
 conn.sendall(("".join([chr((ord(pt[i])-ord('A')+3)%26 + ord('A')) for i in range(len(pt))])).encode())  
 if data.decode().lower() == 'bye':  
 break  
 print(f'Client {addr} disconnected')

# cap\_to\_lower\_client

\*\*cap\_to\_lower - client.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:  
 client\_socket.connect((HOST, PORT))  
 print(f'Connected to {HOST}:{PORT}. Type "bye" to exit.')  
   
 while True:  
 message = input("You: ")  
 client\_socket.sendall(message.encode())  
 data = client\_socket.recv(1024)  
 print(f'Server: {data.decode()}')  
 if message.lower() == 'bye':  
 break

# cap\_to\_lower\_server

\*\*cap\_to\_lower - server.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as server\_socket:  
 server\_socket.bind((HOST, PORT))  
 server\_socket.listen()  
 print(f'Server started. Listening on {HOST}:{PORT}')  
   
 while True:  
 conn, addr = server\_socket.accept()  
 with conn:  
 print(f'Connected by {addr}')  
 while True:  
 data = conn.recv(1024)  
 if not data:  
 break  
 print(f'Client: {data.decode()}')  
 conn.sendall(data.decode().lower().encode())  
 if data.decode().lower() == 'bye':  
 break  
 print(f'Client {addr} disconnected')

# DES\_client

\*\*DES - client.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:  
 client\_socket.connect((HOST, PORT))  
 print(f'Connected to {HOST}:{PORT}. Type "bye" to exit.')  
   
 while True:  
 message = input("You: ")  
 client\_socket.sendall(message.encode())  
 data = client\_socket.recv(1024)  
 print(f'Server: {data.decode()}')  
 if message.lower() == 'bye':  
 break

# DES\_server

\*\*DES - server.txt\*\*  
  
#Functions to perform DES  
from Crypto.Cipher import DES  
from Crypto.Random import get\_random\_bytes  
from base64 import b64encode, b64decode  
  
def pad(text):  
 while len(text) % 8 != 0:  
 text += b' '  
 return text  
  
def des\_encrypt(key, plaintext):  
 cipher = DES.new(key, DES.MODE\_ECB)  
 padded\_plaintext = pad(plaintext)  
 ciphertext = cipher.encrypt(padded\_plaintext)  
 return b64encode(ciphertext).decode()  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as server\_socket:  
 server\_socket.bind((HOST, PORT))  
 server\_socket.listen()  
 print(f'Server started. Listening on {HOST}:{PORT}')  
   
 while True:  
 conn, addr = server\_socket.accept()  
 with conn:  
 print(f'Connected by {addr}')   
 while True:  
 data = conn.recv(1024)  
 if not data:  
 break  
 print(f'Client: {data.decode()}')  
 #Code to encrypt  
 key = get\_random\_bytes(8)  
 bytes=des\_encrypt(key, data)  
 conn.sendall(bytes.encode())  
 if data.decode().lower() == 'bye':  
 break  
 print(f'Client {addr} disconnected')

# MD5\_client

\*\*MD5 - client.txt\*\*  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:  
 client\_socket.connect((HOST, PORT))  
 print(f'Connected to {HOST}:{PORT}. Type "bye" to exit.')  
   
 while True:  
 message = input("You: ")  
 client\_socket.sendall(message.encode())  
 data = client\_socket.recv(1024)  
 print(f'Server: {data.decode()}')  
 if message.lower() == 'bye':  
 break

# MD5\_server

\*\*MD5 - server.txt\*\*  
  
import hashlib  
  
def calculate\_md5(input\_string):  
 # Encode the input string to bytes  
 input\_bytes = input\_string.encode('utf-8')  
  
 # Create an MD5 hash object  
 md5\_hash = hashlib.md5()  
  
 # Update hash object with the input bytes  
 md5\_hash.update(input\_bytes)  
  
 # Get the hexadecimal representation of the hash  
 hashed\_str = md5\_hash.hexdigest()  
  
 return hashed\_str  
  
import socket  
  
HOST = 'localhost'  
PORT = 12345  
  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as server\_socket:  
 server\_socket.bind((HOST, PORT))  
 server\_socket.listen()  
 print(f'Server started. Listening on {HOST}:{PORT}')  
   
 while True:  
 conn, addr = server\_socket.accept()  
 with conn:  
 print(f'Connected by {addr}')  
 while True:  
 data = conn.recv(1024)  
 if not data:  
 break  
 print(f'Client: {data.decode()}')  
 pt=data.decode()  
 conn.sendall((calculate\_md5(pt)).encode())  
 if data.decode().lower() == 'bye':  
 break  
 print(f'Client {addr} disconnected')