

LAPORAN TUGAS UDP SOCKET PROGRAMMING

II2120 JARINGAN KOMPUTER

KELAS K-01/ KELOMPOK NONANANO



Disusun Oleh :

Naura Ayurachmani 18223061

Noeriza Aqila Wibawa 18223095

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2024

DAFTAR ISI

TABEL PENYELESAIAN SPESIFIKASI	1
TABEL PEMBAGIAN TUGAS	3
PROGRAM UDP SOCKET	4
A. CARA KERJA.....	4
A.1. IMPLEMENTASI UDP PADA APLIKASI CHAT	4
A.2 ALUR KERJA	4
B. PENGUJIAN PROGRAM.....	6
B.1 TATA CARA PENGUJIAN PROGRAM	6
B.2 LINGKUNGAN PENGUJIAN PROGRAM	7
B.3 HASIL PENGUJIAN PROGRAM.....	7
SOURCE CODE	12
DAFTAR PUSTAKA	18

TABEL PENYELESAIAN SPESIFIKASI

Berikut merupakan spesifikasi-spesifikasi yang **wajib** dipenuhi:

No	Spesifikasi	Status
1	Server mampu menerima pesan yang dikirim client dan mencetaknya ke layar.	✓
2	Server mampu meneruskan pesan satu client ke client lain.	✓
3	Client mampu mengirimkan pesan ke server dengan IP dan port yang ditentukan pengguna.	✓
4	Client mampu menerima pesan dari client lain (yang diteruskan oleh server), dan mencetaknya ke layar.	✓
5	Client harus memasukkan <i>password</i> untuk dapat bergabung ke chatroom.	✓
6	Client memiliki username yang unik.	✓

Berikut adalah spesifikasi-spesifikasi **opsional** yang dapat dipenuhi:

No	Spesifikasi	Status
1	Aplikasi mengimplementasikan TCP over UDP. Note: asisten sangat menyarankan untuk mengerjakan spesifikasi ini, karena akan memberikan pemahaman kuat terkait TCP.	X
2	Seluruh pesan dienkripsi menggunakan algoritma kriptografi klasik simetris, misal cipher Vigenere atau Caesar.	X
3	Seluruh pesan dienkripsi menggunakan algoritma kriptografi modern simetris, misal cipher RC4.	X
4	Seluruh pesan dienkripsi menggunakan algoritma kriptografi modern asimetris, misal cipher RSA, atau kombinasi algoritma kriptografi modern asimetris dan modern simetris.	✓
5	Seluruh pesan dienkripsi menggunakan algoritma Double-Ratchet atau MLS.	X

6	Aplikasi memiliki GUI.	X
7	Aplikasi mampu digunakan untuk mengirimkan dan menerima pesan bertipe file biner.	X
8	Aplikasi mampu menunjukkan apabila integritas pesan telah rusak, baik dengan memanfaatkan <i>checksum</i> ataupun <i>digital signature</i> .	X
9	Aplikasi mampu menyimpan pesan-pesan lampau meskipun telah ditutup; mekanisme dan tempat penyimpanan bebas, baik di <i>client</i> maupun di <i>server</i> .	X
10	Aplikasi mampu mengotentikasi pengguna.	X
11	Aplikasi diprogram menggunakan paradigma <i>object oriented programming</i> atau pemrograman berorientasi objek	X

TABEL PEMBAGIAN TUGAS

Spesifikasi/tugas	Status	NIM
Server mampu menerima pesan yang dikirim client dan mencetaknya ke layar.	Wajib	18223061 18223095
Server mampu meneruskan pesan satu client ke client lain.	Wajib	18223061 18223095
Client mampu mengirimkan pesan ke server dengan IP dan port yang ditentukan pengguna.	Wajib	18223061 18223095
Client harus memasukkan <i>password</i> untuk dapat bergabung ke chatroom.	Wajib	18223061 18223095
Client memiliki username yang unik.	Wajib	18223061 18223095
Seluruh pesan dienkripsi menggunakan algoritma kriptografi modern asimetris, misal cipher RSA, atau kombinasi algoritma kriptografi modern asimetris dan modern simetris.	Bonus	18223095
Pembuatan laporan	Wajib	18223061 18223095

PROGRAM UDP SOCKET

A. CARA KERJA

UDP (User Datagram Protocol) adalah protokol jaringan di mana tidak ada koneksi khusus antara server dan client (connectionless). Pada implementasinya, UDP hanya akan mengirimkan data dan tidak memeriksa status penerimaan data tersebut pada client. Pada aplikasi chat berbasis UDP, setiap pesan akan dikirimkan dari client ke server, dan server akan meneruskannya ke client lain. Protokol UDP lebih cepat dibandingkan TCP karena tidak ada metode khusus untuk memeriksa penerimaan. Karena kecepatannya tersebut, UDP cocok untuk aplikasi chat yang mengutamakan kecepatan dan latensi rendah daripada keandalan.

A.1. IMPLEMENTASI UDP PADA APLIKASI CHAT

Pada aplikasi chat berbasis UDP, implementasi dasar meliputi elemen-elemen berikut:

1. Server

Server berfungsi sebagai penghubung antar client. Pada aplikasi ini, server melakukan binding dengan alamat IP dan port tertentu untuk menunggu pesan dari client. Jika terdapat lebih dari satu client, server akan meneruskan pesan yang diterimanya ke client lain yang sudah terdaftar. Selagi menunggu request client, server akan tetap aktif tanpa perlu menutup sesi sehingga memungkinkan client untuk bergabung atau keluar kapan saja.

2. Client

Client berperan untuk mengirim pesan ke server melalui socket UDP lalu semua pesan akan dikirim melalui server. Jika client berjumlah lebih dari satu, client tidak membuat koneksi langsung dengan client kemudian setelah mengirim pesan, client juga dapat meneruma pesan dari client lain yang dikirimkan melalui server.

A.2 ALUR KERJA

1. Inisialisasi Server dan Client:

- Server membuka socket UDP dan melakukan binding ke alamat IP dan port tertentu untuk menerima pesan dari client. Server

menghasilkan pasangan kunci publik dan privat menggunakan algoritma RSA.

- Client membuat socket UDP dan siap untuk mengirim pesan ke server. Client menghasilkan pasangan kunci publik dan privat menggunakan algoritma RSA

2. Pertukaran Kunci Publik:

- Client akan terlebih dahulu mengirimkan kunci publiknya ke server dan server kemudian mengirimkan kunci publiknya juga ke client.
- Server akan menyimpan kunci publik klien di daftar kunci publik client. Client juga akan menyimpan kunci publik server untuk melakukan enkripsi pesan yang akan dikirimkan ke server di kemudian hari.

3. Pengiriman Pesan dari Client ke Server:

- Client mengirimkan pesan yang telah terenkripsi ke server. Server menerima pesan ini dan mendekripsinya menggunakan kunci privatnya. Untuk setiap client, server akan mengenkripsi pesan menggunakan kunci publik masing-masing client sebelum diteruskan.

4. Broadcast Pesan ke Client Lain:

- Jika terdapat lebih dari satu pelanggan, setelah menerima pesan, server akan meneruskannya ke client lain dengan mengirimkan pesan tersebut ke alamat client kedua.
- Dalam proses ini, server tidak mengelola status antara client, sehingga memudahkan penambahan atau pengurangan client.

5. Penerimaan Pesan oleh Client Lain:

- Client kedua menerima pesan yang dikirim oleh server lalu mendekripsinya dan menampilkan pesan tersebut kepada client.
- Semua client yang terhubung bisa menerima pesan yang sama, memungkinkan mereka untuk terus berkomunikasi tanpa gangguan.

6. Keluar dari Obrolan:

- Jika client ingin keluar dari obrolan, mereka mengirimkan pesan

B. PENGUJIAN PROGRAM

B.1 TATA CARA PENGUJIAN PROGRAM

1. Persiapan Program

Sebelum menjalankan program, pastikan sudah ada file `server.py` dan `client.py` di folder atau direktori yang sama.

2. Menjalankan Program Server

Untuk menjalankan server, buka terminal dan navigasikan ke folder tempat file `server.py` berada. Kemudian, jalankan server dengan perintah `python server.py`. Server akan berjalan dan mendengarkan di port 9999, serta menunggu koneksi dan autentikasi dari client. Jika sudah ada request dari client, maka server akan meminta password. Default password pada server adalah 12345.

3. Menjalankan Program Client

Untuk menjalankan client, buka terminal dan navigasikan ke folder yang sama. Kemudian, jalankan client dengan perintah `python client.py`. Setelah berhasil dijalankan, program akan meminta untuk dimasukkan port, username yang unik, dan password. Jika password benar, client akan terhubung dengan server.

4. Pengiriman dan Penerimaan Pesan

Setelah terhubung, client bisa mengetik pesan yang ingin dikirim ke server. Server kemudian akan menerima pesan tersebut, mencetaknya di terminal server, serta akan membalas pesan ke client. Selanjutnya, client akan menerima pesan tersebut dan menampilkannya di terminal.

5. Pengujian Kesalahan

Uji skenario kesalahan dapat dilakukan dengan memasukkan password yang salah yang kemudian server akan merespons dengan pesan error. Kemudian, jika client memasukkan username yang sudah ada pada server, maka server akan merespons dengan pesan error dan meminta client memasukkan username lain. Selain itu, jika user memasukkan port selain 9999, server akan merespons dengan mengonfirmasi apakah

client ingin melanjutkan di port yang di-input dan jika memilih tidak server akan mengubah port dari client menjadi default port.

B.2 LINGKUNGAN PENGUJIAN PROGRAM

1. Sistem Operasi

Program ini dapat berjalan pada Linux, macOS, atau Windows dengan Python 3.x.

2. Python 3.x:

Program menggunakan library socket, threading, dan queue.

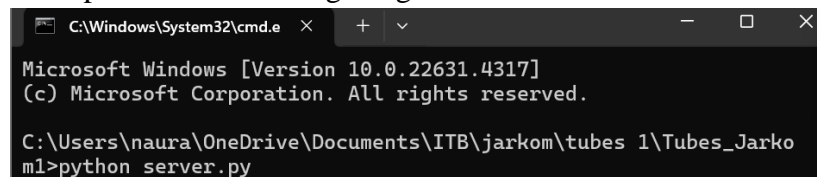
3. Jaringan Lokal dan Port Range Client:

Pengujian dilakukan di jaringan lokal dengan server memakai port 9999 dan client menggunakan port acak antara 8000 dan 9000 untuk proses binding.

B.3 HASIL PENGUJIAN PROGRAM

#SERVER RUNNING

>Tampilan server running dengan lancar:

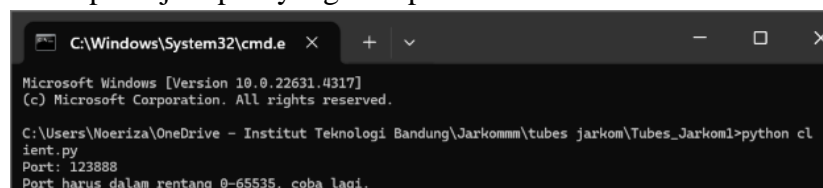


```
C:\Windows\System32\cmd.e  X + v - □ X
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarko
ml>python server.py
```

#PENGUJIAN PORT

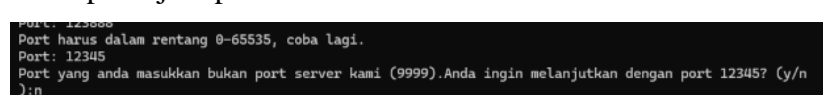
>Tampilan jika port yang di-input oleh client tidak valid:



```
C:\Windows\System32\cmd.e  X + v - □ X
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Noeriza\OneDrive - Institut Teknologi Bandung\Jarkomm\tubes jarkom\Tubes_Jarkomi>python cl
ient.py
Port: 123888
Port harus dalam rentang 0-65535, coba lagi.
```

>Tampilan jika port valid namun bukan 9999:



```
Port: 12345
Port harus dalam rentang 0-65535, coba lagi.
Port: 12345
Port yang anda masukkan bukan port server kami (9999).Anda ingin melanjutkan dengan port 12345? (y/n
):n
```

Jika client memilih n maka port akan otomatis di-setting menjadi 9999.

Sedangkan jika client memilih y maka port akan berjalan di port yang di-input oleh client.

>Tampilan jika port = 9999:

```
C:\Users\Noeriza\OneDrive - Institut Teknologi Bandung\Jarkomm\
tubes_jarkom\Tubes_Jarkom1>python client.py
Port: 9999
IP: 192.168.1.137
```

Jika port valid maka client akan diminta untuk memasukkan IP address server.

#PENGUJIAN IP ADDRESS

>Tampilan jika IP address tidak valid:

```
IP: hakjwi
IP Address tidak valid, coba lagi
IP: 192.168.1.137
```

Jika tidak valid, client akan diminta untuk memasukkan ulang IP address server.

>Tampilan jika IP address valid:

```
Port: 9999
IP: 192.168.1.137
Your IP: 192.168.1.107
```

Setelah valid akan dicetak IP address client.

#PENGUJIAN PASSWORD

>Tampilan jika password salah:

```
Your IP: 192.168.1.107
Password: bbbb
Server response: Password salah! Coba lagi.
Password salah, coba lagi.
Password: hahahaha
Server response: Password salah! Coba lagi.
Password salah, coba lagi.
```

Jika salah, client akan diminta untuk memasukkan ulang password hingga benar.

>Tampilan jika password benar:

```
Password salah, coba lagi.
Password: 12345
Server response: Password benar! Anda berada di obrolan
Username: naura
```

```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarko
m1>python server.py
entered_pass: 12345
```

Jika benar, maka respons client akan dikirim ke server

#PENGUJIAN USERNAME

>Tampilan client dan server jika username yang di-input masih available:

```

C:\Windows\System32\cmd.e  x  +  v  -  □  x
Username: adriana
Checking username availability...
Server response: Username available
Username adriana berhasil diset!
Selected username: adriana

```

```

Broadcasting message: SIGNUP_TAG:adriana
Broadcasting message: SIGNUP_TAG:adriana
Broadcasting message: SIGNUP_TAG:adriana

```

>Tampilan jika username tidak tersedia:

```

Username: sofia
Checking username availability...
Server response: Username unavailable
Username sudah terdaftar, silakan masukkan username yang lain.
Username: hagia
Checking username availability...
Server response: Username available
Username hagia berhasil diset!
Selected username: hagia

```

Client akan diminta untuk memasukkan ulang username.

#CHAT DENGAN 2 DEVICE

>Tampilan chat di dalam client:

```

C:\Windows\System32\cmd.e  x  +  v  -  □  x
Your IP: 192.168.1.137
Password: 12345
Server response: Password benar! Anda berada di obrolan
Username: adriana
Checking username availability...
Server response: Username available
Username adriana berhasil diset!
Selected username: adriana
sofia memasuki obrolan!
sofia memasuki obrolan!
sofia memasuki obrolan!
halo
adriana: halo
sofia: halo adriana
sofia: kamu lagi apa
aku lagi mau makan
adriana: aku lagi mau makan
hagia memasuki obrolan!
hagia memasuki obrolan!
hagia memasuki obrolan!
hagia: HALOO SEMUA DISINI ADA SIAPA
haiiii
adriana: haiiii
sofia: MOY
how
adriana: how
hagia: tes
sofia: tess
sofia: tess
yuhuu memasuki obrolan!
yuhuu memasuki obrolan!
yuhuu memasuki obrolan!
yuhuu: hello yall
hagia: haiiii

```

```

C:\Windows\System32\cmd.e  x  +  v  -  □  x
Microsoft Windows [Version 10.0.22H1.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Noeriza\OneDrive - Institut Teknologi Bandung\Jarkomm\tubes jarkom\Tubes_Jarkom\python client.py
Port: 12345
Port harus dalam rentang 0-65535, coba lagi.
Port: 12345
Port yang anda masukkan bukan port server kami (9999).Anda ingin melanjutkan dengan port 12345? (y/n)
y/n
IP: bajjy
IP Address tidak valid, coba lagi
IP: 192.168.1.137
Your IP: 192.168.1.137
Password: bbb
Server response: Password salah! Coba lagi.
Password: hahahaha
Server response: Password salah! Coba lagi.
Password: hahahaha
Server response: Password salah! Coba lagi.
Password: 12345
Server response: Password benar! Anda berada di obrolan
Username: sofia
Checking username availability...
Server response: Username unavailable
Username sudah terdaftar, silakan masukkan username yang lain.
Username: hagia
Checking username availability...
Server response: Username available
Username hagia berhasil diset!
Selected username: hagia
HALOO SEMUA DISINI ADA SIAPA
hagia: HALOO SEMUA DISINI ADA SIAPA
adriana: haiiii
sofia: MOY
hagia: tes
sofia: tess
sofia: tess
yuhuu memasuki obrolan!
yuhuu memasuki obrolan!
yuhuu memasuki obrolan!
yuhuu: hello yall
hagia: haiiii

C:\Windows\System32\cmd.e  x  +  v  -  □  x
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Noeriza\OneDrive - Institut Teknologi Bandung\Jarkomm\tubes jarkom\Tubes_Jarkom\python client.py
Port: 9999
IP: 192.168.1.137
Your IP: 192.168.1.137
Password: 12345
Server response: Password benar! Anda berada di obrolan
Username: yuhuu
Checking username availability...
Server response: Username available
Username yuhuu berhasil diset!
Selected username: yuhuu
hello yall
yuhuu: hello yall
hagia: haiiii

```

>Tampilan chat di dalam server

```
C:\Windows\System32\cmd.exe X + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarkom1>python server.py
entered_pass: 12345
entered_pass: 12345
Broadcasting message: SIGNUP_TAG:adriana
Broadcasting message: SIGNUP_TAG:adriana
Broadcasting message: SIGNUP_TAG:adriana
Broadcasting message: SIGNUP_TAG:sofia
Broadcasting message: SIGNUP_TAG:sofia
Broadcasting message: SIGNUP_TAG:sofia
Broadcasting message: adriana: halo
Broadcasting message: sofia: halo adriana
Broadcasting message: sofia: kamu lagi apa
Broadcasting message: adriana: aku lagi mau makan
entered_pass: bbbb
entered_pass: hahahaha
entered_pass: 12345
Broadcasting message: SIGNUP_TAG:hagia
Broadcasting message: SIGNUP_TAG:hagia
Broadcasting message: SIGNUP_TAG:hagia
Broadcasting message: hagia: HALOO SEMUA DISINI ADA SIAPA
Broadcasting message: adriana: haiii
Broadcasting message: sofia: WOY
Broadcasting message: adriana: how
Broadcasting message: hagia: tes
Broadcasting message: sofia: tess
Broadcasting message: sofia: tess
entered_pass: 12345
Broadcasting message: SIGNUP_TAG:yuhuu
Broadcasting message: SIGNUP_TAG:yuhuu
Broadcasting message: SIGNUP_TAG:yuhuu
Broadcasting message: yuhuu: hello yall
Broadcasting message: hagia: haiii
```

#CLIENT KELUAR

>Tampilan di dalam client:

```
C:\Windows\System32\cmd.exe X + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarkom1>python client.py
port: 9999
IP: 192.168.1.137
Your IP: 192.168.1.137
Password: 12345
Server response: Password benar! Anda berada di obrolan
Username: lisa
Checking username availability...
Server response: Username available
Username lisa berhasil diset!
Selected username: lisa
hai
lisa: hai
!q
Server: bye bye
Anda telah keluar dari obrolan.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarkom1>
```

>Tampilan di dalam server:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\naura\OneDrive\Documents\ITB\jarkom\tubes 1\Tubes_Jarkom1>python server.py
entered pass: 12345
Broadcasting message: SIGNUP_TAG:lisa
Broadcasting message: SIGNUP_TAG:lisa
Broadcasting message: SIGNUP_TAG:lisa
Broadcasting message: lisa: hai
Broadcasting message: lisa keluar dari obrolan.
```

SOURCE CODE

Pada tugas UDP socket programming ini, kelompok kami memakai VS Code sebagai compiler dan memakai github sebagai media penyimpanan kode kami. Berikut adalah tautan kode program UDP Socket Programming kelompok NonaNano:

https://github.com/naurchmn/Tubes_Jarkom1.git

SERVER

```
1 import socket
2 import threading
3 import queue
4 from rsa_module import generate_keys, encrypt, decrypt
5
6 messages = queue.Queue()
7 clients = []
8 PASSWORD = "12345"
9 auth_clients={}
10 usernames = set()
11 username_map = {}
12
13 server = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
14 server.bind(("0.0.0.0", 9999))
15 public_key, private_key = generate_keys() # generate key buat server
16 client_public_keys = {} #simpen public key client
17
18 def receive():
19     while True:
20         try:
21             message, addr = server.recvfrom(1024)
22
23             if addr not in auth_clients:
24                 if addr not in client_public_keys:
25                     # pesan pertama klien kunci publiknya klien
26                     e, n = map(int, message.decode().split(","))
27                     client_public_keys[addr] = (e, n)
28                     # Kirim kunci publik server ke klien
29                     server.sendto(f"{public_key[0]},{public_key[1]}".encode(), addr)
30                 else:
31                     if message.decode().startswith("PASSWORD:"):
32                         encrypted_pass = message.decode().split(":", 1)[1].strip()
33                         entered_pass = decrypt(encrypted_pass, private_key)
34                         print(f"entered_pass: {entered_pass}")
35
36                     if (entered_pass == PASSWORD):
37                         auth_clients[addr] = True
38                         server.sendto("Password benar! Anda berada di obrolan".encode(), addr)
39                         #messages.put((f"SIGNUP_TAG:{username_map[addr]}".encode(), addr))
40                     else:
41                         server.sendto("Password salah! Coba lagi.".encode(), addr)
42                 else:
43                     server.sendto("Masukkan password dengan format PASSWORD:<password>".encode(), addr)
44
```

```

1  else:
2      #if addr not in username_map:
3      if message.decode().startswith("CHECK_USERNAME:"):
4          username = message.decode().split(":")[1].strip()
5          if username in usernames:
6              server.sendto("Username unavailable".encode(), addr)
7          else:
8              server.sendto("Username available".encode(), addr)
9
10     elif message.decode().startswith("SET_USERNAME:"):
11         username = message.decode().split(":")[1].strip()
12         if username in usernames:
13             server.sendto("Username unavailable".encode(), addr)
14         else:
15             usernames.add(username)
16             username_map[addr] = username
17             server.sendto(f"Username {username} berhasil diset!".encode(), addr)
18             messages.put((f"SIGNUP_TAG:{username}".encode(), addr))
19
20     elif message.decode().startswith("SIGNUP_TAG:"):
21         messages.put((message, addr))
22
23     elif message.decode().startswith("LEAVE_TAG"):
24         username = username_map.get(addr, "Unknown")
25         server.sendto("bye bye".encode(), addr)
26         if addr in clients:
27             clients.remove(addr)
28         if addr in username_map:
29             usernames.remove(username_map[addr])
30             del username_map[addr]
31         if addr in auth_clients:
32             del auth_clients[addr]
33         broadcast_message = f"{username} keluar dari obrolan."
34         messages.put((broadcast_message.encode(), None))
35
36     else:
37         encrypted_message = message.decode()
38         decrypted_message = decrypt(encrypted_message, private_key)
39         messages.put((decrypted_message.encode(), addr))
40
41 except:
42     pass
43
44 def send_message(client, message):
45     # batesin ukuran buat dikirim ke klien
46     max_size = 1000 # maks ukuran tiap bagian pesan
47     for i in range(0, len(message), max_size):
48         chunk = message[i:i + max_size]
49         # tambahkan end buat pesan di akhir
50         if i + max_size >= len(message):
51             chunk += "END"
52         server.sendto(chunk.encode(), client)
53
54 def broadcast():
55     while True:
56         while not messages.empty():
57             message, addr = messages.get()
58             print("Broadcasting message:", message.decode())
59
60             if addr and addr not in clients:
61                 clients.append(addr)
62
63             for client in clients:
64                 try:
65                     if message.decode().startswith("SIGNUP_TAG:"):
66                         username = message.decode().split(":")[1].strip()
67                         if client != addr:
68                             encrypted_message = encrypt(f"{username} memasuki obrolan!", client_public_keys[client])
69                             send_message(client, f"ENCRYPTED:{encrypted_message}")
70                     else:
71                         encrypted_message = encrypt(message.decode(), client_public_keys[client])
72                         send_message(client, f"ENCRYPTED:{encrypted_message}")
73                 except Exception as e:
74                     print(f"Error sending message to client {client}: {e}")
75                     if client in clients:
76                         clients.remove(client)
77
78
79 t1 = threading.Thread(target=receive)
80 t2 = threading.Thread(target=broadcast)
81
82 t1.start()
83 t2.start()

```

CLIENT

```
1 import socket
2 import threading
3 import random
4 from rsa_module import generate_keys, encrypt, decrypt
5
6 client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
7 client.bind(("0.0.0.0", random.randint(8000, 9000)))
8 client.settimeout(3)
9
10 # get hostname sama ip
11 hostname= socket.gethostname()
12 IP = socket.gethostbyname(hostname)
13 # generate public key and private key klien
14 public_key, private_key = generate_keys()
15
16
17 def get_valid_port():
18     while True:
19         try:
20             sendPORT= int(input("Port: "))
21             if 0<= sendPORT<=65535:
22                 if sendPORT !=9999:
23                     while True:
24                         pilih = input(f"Port yang anda masukkan bukan port server kami (9999)."  
f"Anda ingin melanjutkan dengan port {sendPORT}? (y/n):").lower()
25                         if pilih == 'n':
26                             return 9999
27                         elif pilih == 'y':
28                             return sendPORT
29                         else:
30                             print ("pilihan tidak valid, coba lagi.")
31                     else:
32                         return sendPORT
33                 else:
34                     print("Port harus dalam rentang 0-65535, coba lagi.")
35             except ValueError:
36                 print("Port yang dimasukkan tidak valid. Harus berupa angka, coba kembali")
37
38
39 def get_valid_IP():
40     while True:
41         sendIP=input("IP: ")
42         try:
43             socket.inet_aton(sendIP)
44             if sendIP.count('.')==3:
45                 return sendIP
46             else:
47                 print("Format IP address tidak valid, coba lagi")
48         except socket.error:
49             print("IP Address tidak valid, coba lagi")
50
51 sendPORT = get_valid_port()
52 sendIP = get_valid_IP()
53 server_address = (sendIP, sendPORT)
54
55 client.sendto(f"{public_key[0]},{public_key[1]}".encode(), server_address)
56 # Terima kunci publik server
57 message, _ = client.recvfrom(1024)
58 e, n = map(int, message.decode().split(","))
59 server_public_key = (e, n)
60
61 exit_flag = False
```



```

1  def receive_full_message():
2      message_parts = []
3      while True:
4          try:
5              packet, _ = client.recvfrom(1024)
6              if b"END" in packet:
7                  message_parts.append(packet.replace(b"END", b""))
8                  break
9              else:
10                 message_parts.append(packet)
11         except socket.timeout:
12             if exit_flag:
13                 break
14             else:
15                 continue
16         except Exception as e:
17             print(f"Error saat menerima paket: {e}")
18             break
19     full_message = b"".join(message_parts)
20     return full_message.decode()
21
22
23
24 def receive():
25     while not exit_flag:
26         try:
27             full_message = receive_full_message()
28             if full_message == "bye bye":
29                 print("Server:", full_message)
30                 client.close()
31                 break
32             elif full_message.startswith("ENCRYPTED:"):
33                 encrypted_message_hex = full_message.replace("ENCRYPTED:", "").strip()
34                 decrypted_message = decrypt(encrypted_message_hex, private_key)
35                 print(decrypted_message)
36             else:
37                 print(f"Server message: {full_message}")
38         except socket.timeout:
39             continue
40         except Exception as e:
41             print(f"Error saat menerima atau mendekripsi pesan: {e}")
42     if not client._closed:
43         client.close()
44
45     print(f"Your IP: {IP}")
46
47     while True:
48         password = input("Password: ")
49         try:
50             encrypted_password = encrypt(password, server_public_key)
51             client.sendto(f"PASSWORD:{encrypted_password}".encode(), server_address)
52             message, _ = client.recvfrom(1024)
53             response = message.decode()
54             print(f"Server response: {response}")
55             if response == "Password benar! Anda berada di obrolan":
56                 while True:
57                     username = input("Username: ")
58                     client.sendto(f"CHECK_USERNAME:{username}".encode(), server_address)
59
60                     print("Checking username availability...")
61
62                     message, _ = client.recvfrom(1024)
63                     response = message.decode()
64
65                     print(f"Server response: {response}")
66
67                     if response == "Username available":
68                         client.sendto(f"SET_USERNAME:{username}".encode(), server_address)
69                         confirmation, _ = client.recvfrom(1024)
70                         print(confirmation.decode())
71                         break

```

```

1 elif response == "Username unavailable":
2     print("Username sudah terdaftar, silakan masukkan username yang lain.")
3     else:
4         print("Unexpected response from server.")
5         print(f"Selected username: {username}")
6
7         t = threading.Thread(target=receive)
8         t.start()
9
10        client.sendto(f"SIGNUP_TAG:{username}".encode(), server_address)
11        break
12    else:
13        print("Password salah, coba lagi.")
14
15    except Exception as e:
16        print(f"Error saat mengirim password: {e}")
17        exit()
18
19    client.sendto(f"SIGNUP_TAG:{username}".encode(), server_address)
20
21    while True:
22        message = input("")
23        if message == "!q":
24            client.sendto(f"LEAVE_TAG:{username}".encode(), server_address)
25            exit_flag = True
26            break
27        else:
28            encrypted_message = encrypt(f"{username}: {message}", server_public_key)
29            client.sendto(str(encrypted_message).encode(), server_address)
30
31    t.join()
32    print("Anda telah keluar dari obrolan.")

```

RSA MODULE

```
1 import random
2 from math import gcd
3
4 def is_prime(n):
5     if n <= 1:
6         return False
7     for i in range(2, int(n**0.5) + 1):
8         if n % i == 0:
9             return False
10    return True
11
12 def generate_large_prime():
13    while True:
14        num = random.randint(100, 500)
15        if is_prime(num):
16            return num
17
18 def mod_inverse(e, phi):
19    d, x1, x2, y1 = 0, 0, 1, 1
20    temp_phi = phi
21
22    while e > 0:
23        temp1 = temp_phi // e
24        temp2 = temp_phi - temp1 * e
25        temp_phi, e = e, temp2
26        x = x2 - temp1 * x1
27        y = d - temp1 * y1
28        x2, x1 = x1, x
29        d, y1 = y1, y
30
31    if temp_phi == 1:
32        return d + phi
33
34 def generate_keys():
35    p = generate_large_prime()
36    q = generate_large_prime()
37    n = p * q
38    phi = (p - 1) * (q - 1)
39
40    e = random.randrange(1, phi)
41    while gcd(e, phi) != 1:
42        e = random.randrange(1, phi)
43
44    d = mod_inverse(e, phi)
45    return ((e, n), (d, n))
46
47 def encrypt(message, public_key):
48    e, n = public_key
49    cipher = [(ord(char) ** e) % n for char in message]
50    hex_cipher = [format(part, 'x') for part in cipher] # konversi int ke hex
51    cipher_text = ' '.join(hex_cipher) #pake blank sbg separator
52    return cipher_text
53
54 def decrypt(cipher_text, private_key):
55    d, n = private_key
56    hex_cipher = cipher_text.strip().split(' ')
57    cipher = [int(part, 16) for part in hex_cipher]
58    plain = [chr((char ** d) % n) for char in cipher]
59    return ''.join(plain)
```

DAFTAR PUSTAKA

A. Malhotra, V. Sharma, P. Gandhi and N. Purohit, "UDP based chat application," *2010 2nd International Conference on Computer Engineering and Technology*, Chengdu, China, 2010, pp. V6-374-V6-377.