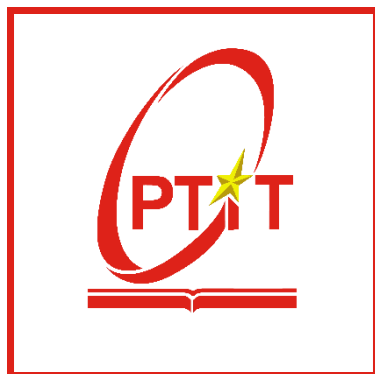


HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN



THỰC TẬP CƠ SỞ
Bài 15: Lập trình client/server để
trao đổi thông tin an toàn

Sinh viên	Nguyễn Duy Đạt
MSV	B21DCAT056
Giảng viên	Vũ Minh Mạnh

Hà Nội – 2024

Môn học Thực tập cơ sở

Bài 15: Lập trình client/server để trao đổi thông tin an toàn

I. Lý thuyết

1. Tìm hiểu về các khái niệm liên quan tới lập trình socket với TCP

Socket là gì? Đây chính là điểm cuối end-point tại liên kết truyền thông 2 chiều (two-way communication) và biểu diễn kết nối giữa Server – Client. Những lớp Socket hiện đang ràng buộc với 1 cổng port (thể hiện là 1 con số cụ thể) để những tầng TCP (hay TCP Layer) hoàn toàn có thể định danh được ứng dụng mà dữ liệu gửi đến. Vậy cụ thể cơ chế hoạt động của Socket là gì?

- Cơ chế hoạt động của Socket là gì? Hiện tại, chức năng của socket chính là kết nối giữa server và client thông qua UDP, TCP/IP để có thể truyền cũng như nhận nhận dữ liệu thông qua internet.

- Hiện tại giao diện của lập trình ứng dụng mạng chỉ có thể hoạt động nếu như đã có những thông tin liên quan tới thông số IP cũng như số hiệu cổng của hai ứng dụng cần phải trao đổi dữ liệu.

- Như vậy hai ứng dụng đang cần truyền thông tin bắt buộc phải đáp ứng được những điều kiện cơ bản sau đây thì socket mới hoạt động, cụ thể:

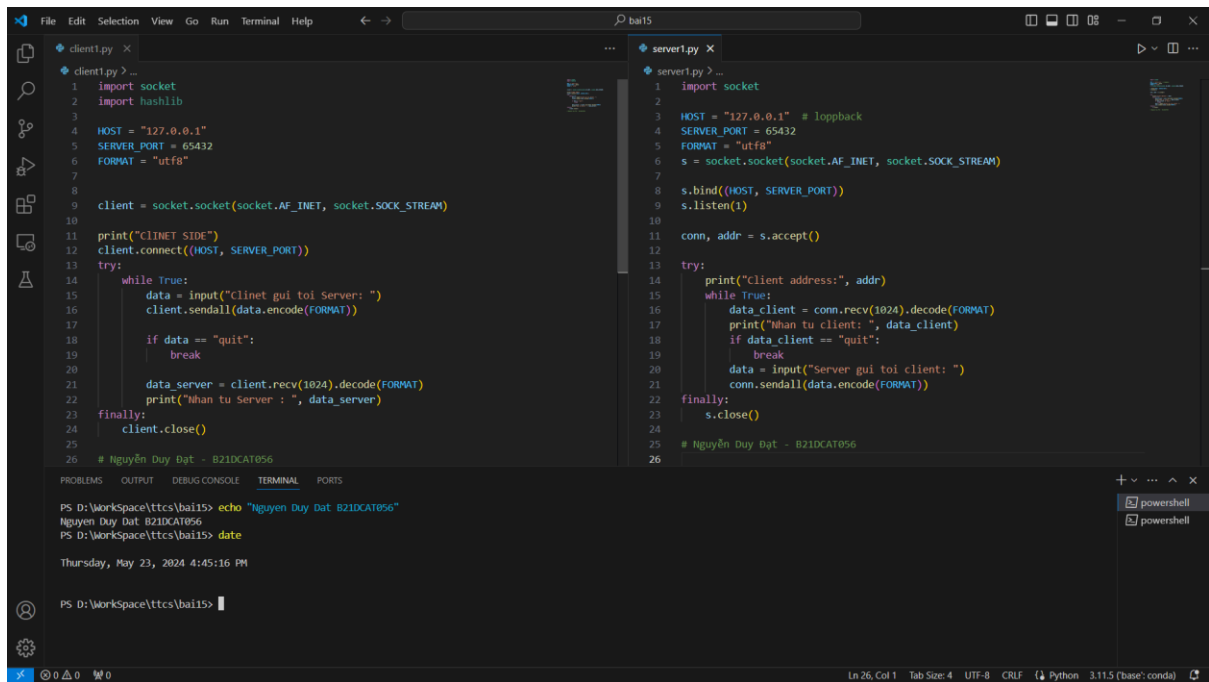
+ Hai ứng dụng hoàn toàn có thể nằm cùng trên một máy hay hai máy khác nhau.

+ Đối với trường hợp nếu như hai ứng dụng cùng trên một máy thì hiệu số cổng bắt buộc không được trùng với nhau.

II. Thực hành

1. Lập trình client và server với TCP socket

- Code Client và Server

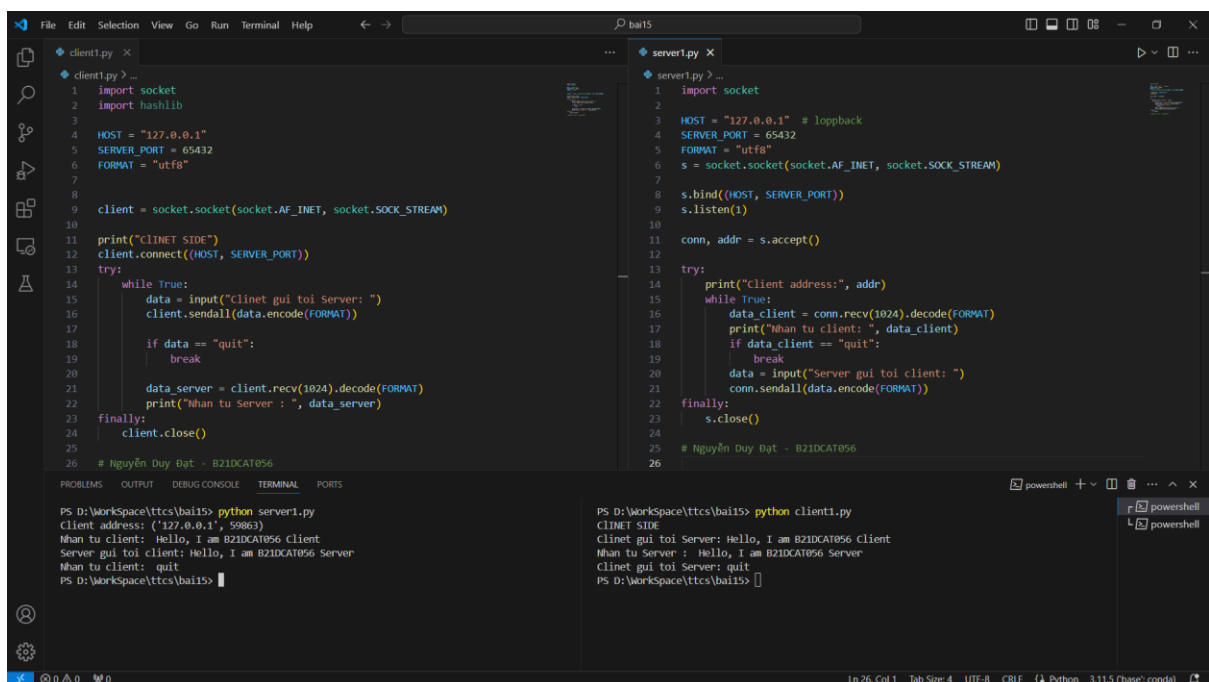


```
client.py
1 import socket
2 import hashlib
3
4 HOST = "127.0.0.1"
5 SERVER_PORT = 65432
6 FORMAT = "utf8"
7
8
9 client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10
11 print("CLIENT SIDE")
12 client.connect((HOST, SERVER_PORT))
13 try:
14     while True:
15         data = input("Client gui toi Server: ")
16         client.sendall(data.encode(FORMAT))
17
18         if data == "quit":
19             break
20
21         data_server = client.recv(1024).decode(FORMAT)
22         print("Nhan tu Server : ", data_server)
23 finally:
24     client.close()
25
26 # Nguyễn Duy Đạt - B21DCAT056
```

```
server1.py
1 import socket
2
3 HOST = "127.0.0.1" # localhost
4 SERVER_PORT = 65432
5 FORMAT = "utf8"
6 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7
8 s.bind((HOST, SERVER_PORT))
9 s.listen(1)
10
11 conn, addr = s.accept()
12
13 try:
14     print("Client address:", addr)
15     while True:
16         data_client = conn.recv(1024).decode(FORMAT)
17         print("Nhan tu client: ", data_client)
18         if data_client == "quit":
19             break
20         data = input("Server gui toi client: ")
21         conn.sendall(data.encode(FORMAT))
22 finally:
23     s.close()
24
25 # Nguyễn Duy Đạt - B21DCAT056
```

```
PS D:\Workspace\ttcs\bai15> echo "Nguyễn Duy Đạt B21DCAT056"
Nguyễn Duy Đạt B21DCAT056
PS D:\Workspace\ttcs\bai15> date
Thursday, May 23, 2024 4:45:16 PM
PS D:\Workspace\ttcs\bai15>
```

- Client gửi thông điệp cá nhân hóa cho server: “Hello, I am B21DCAT056 Client”. Server nhận được hiển thị thông điệp nhận được và gửi lại client thông điệp: server gửi lại “Hello, I am B21DCAT056 Server”



```
client.py
1 import socket
2 import hashlib
3
4 HOST = "127.0.0.1"
5 SERVER_PORT = 65432
6 FORMAT = "utf8"
7
8
9 client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10
11 print("CLIENT SIDE")
12 client.connect((HOST, SERVER_PORT))
13 try:
14     while True:
15         data = input("Client gui toi Server: ")
16         client.sendall(data.encode(FORMAT))
17
18         if data == "quit":
19             break
20
21         data_server = client.recv(1024).decode(FORMAT)
22         print("Nhan tu Server : ", data_server)
23 finally:
24     client.close()
25
26 # Nguyễn Duy Đạt - B21DCAT056
```

```
server1.py
1 import socket
2
3 HOST = "127.0.0.1" # localhost
4 SERVER_PORT = 65432
5 FORMAT = "utf8"
6 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7
8 s.bind((HOST, SERVER_PORT))
9 s.listen(1)
10
11 conn, addr = s.accept()
12
13 try:
14     print("Client address:", addr)
15     while True:
16         data_client = conn.recv(1024).decode(FORMAT)
17         print("Nhan tu client: ", data_client)
18         if data_client == "quit":
19             break
20         data = input("Server gui toi client: ")
21         conn.sendall(data.encode(FORMAT))
22 finally:
23     s.close()
24
25 # Nguyễn Duy Đạt - B21DCAT056
```

```
PS D:\Workspace\ttcs\bai15> python server1.py
Client address: ('127.0.0.1', 59863)
Nhan tu client: Hello, I am B21DCAT056 Client
Server gui toi client: Hello, I am B21DCAT056 Server
Nhan tu client: quit
PS D:\Workspace\ttcs\bai15>

PS D:\Workspace\ttcs\bai15> python client.py
CLIENT SIDE
Client gui toi Server: Hello, I am B21DCAT056 Client
Nhan tu Server : Hello, I am B21DCAT056 Server
Client gui toi Server: quit
PS D:\Workspace\ttcs\bai15>
```

- Sử dụng Wireshark để bắt các thông tin đã gửi từ client đến server và ngược lại

The screenshot shows the Wireshark interface with a packet capture on the interface \Device\NPF_{...}. The packet list shows a series of TCP and TLSv1.2 packets. The packet details pane shows the selected packet (No. 116) with the following information:

- Frame 116: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF_{...}
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- Transmission Control Protocol, Src Port: 59863, Dst Port: 65432, Seq: 1, Ack: 1, Len: 29
- Data (29 bytes)

The packet bytes pane shows the raw data of the selected packet, which is a TLSv1.2 packet. The data is displayed in hexadecimal and ASCII. The ASCII part shows the text "o, I am B21DCAT056 Client".

A Command Prompt window is open, showing the following commands and output:

```
C:\Users\datnd>echo "Nguyen Duy Dat B21DCAT056"
"Nguyen Duy Dat B21DCAT056"

C:\Users\datnd>date
The current date is: Thu 05/23/2024
Enter the new date: (mm-dd-yy)
```

The screenshot shows the Wireshark interface with a packet capture on the interface \Device\NPF_{...}. The packet list shows a series of TCP and TLSv1.2 packets. The packet details pane shows the selected packet (No. 158) with the following information:

- Frame 158: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF_{...}
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- Transmission Control Protocol, Src Port: 59863, Dst Port: 65432, Seq: 1, Ack: 30, Len: 29
- Data (29 bytes)

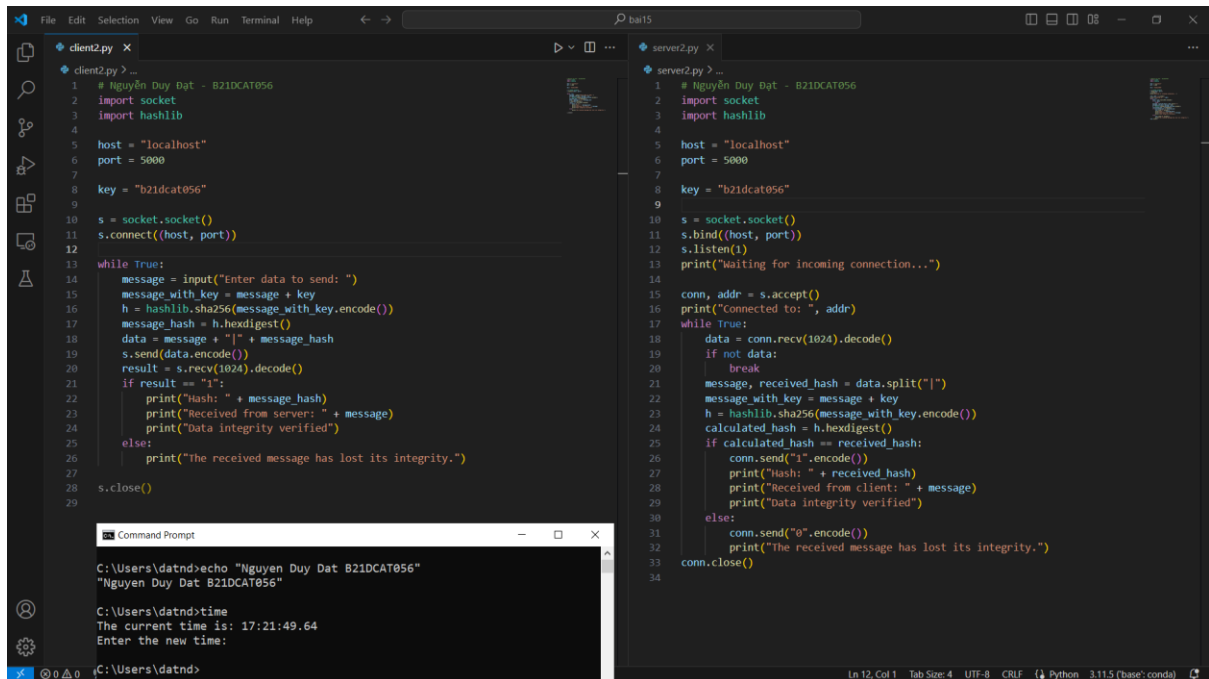
The packet bytes pane shows the raw data of the selected packet, which is a TLSv1.2 packet. The data is displayed in hexadecimal and ASCII. The ASCII part shows the text "o, I am B21DCAT056 Server".

A Command Prompt window is open, showing the following commands and output:

```
C:\Users\datnd>echo "Nguyen Duy Dat B21DCAT056"
"Nguyen Duy Dat B21DCAT056"

C:\Users\datnd>date
The current date is: Thu 05/23/2024
Enter the new date: (mm-dd-yy)
```

2. Trao đổi thông điệp giữa client và server và đảm bảo tính toàn vẹn của thông điệp khi trao đổi
- Code của Client và Server



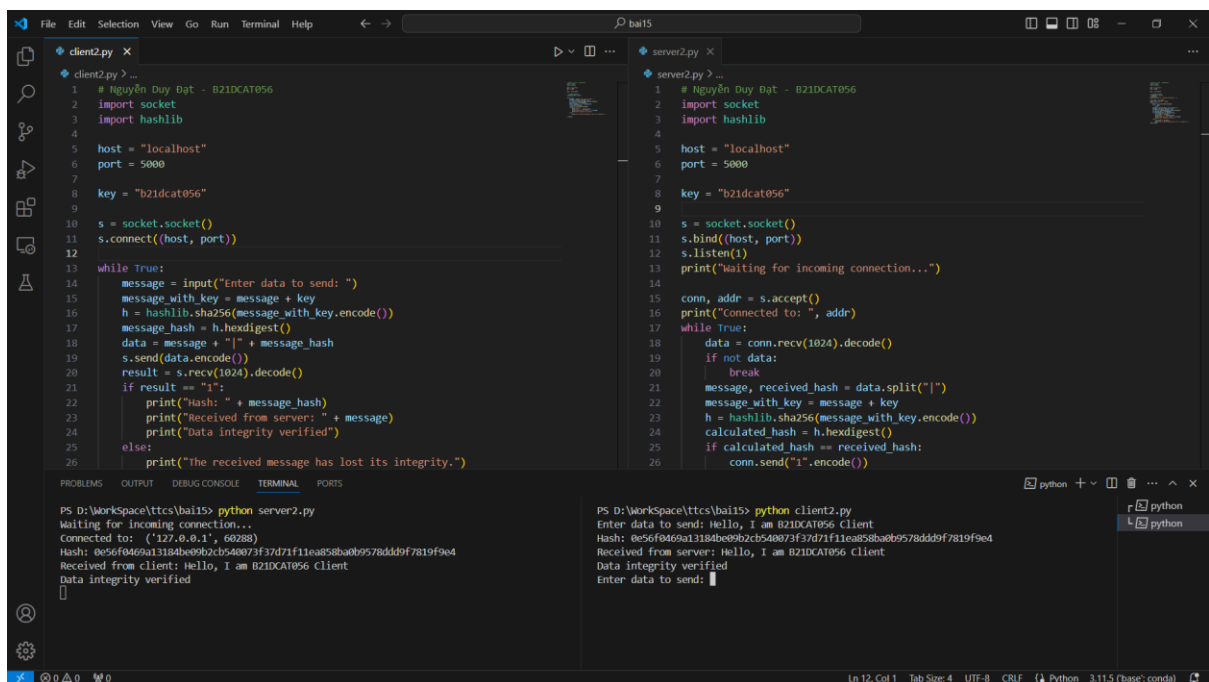
```
client2.py
1 # Nguyễn Duy Đạt - B21DCAT056
2 import socket
3 import hashlib
4
5 host = "localhost"
6 port = 5000
7
8 key = "b21dcat056"
9
10 s = socket.socket()
11 s.connect((host, port))
12
13 while True:
14     message = input("Enter data to send: ")
15     message_with_key = message + key
16     h = hashlib.sha256(message_with_key.encode())
17     message_hash = h.hexdigest()
18     data = message + "|" + message_hash
19     s.send(data.encode())
20     result = s.recv(1024).decode()
21     if result == "1":
22         print("Hash: " + message_hash)
23         print("Received from server: " + message)
24         print("Data integrity verified")
25     else:
26         print("The received message has lost its integrity.")
27
28 s.close()
29
```

```
server2.py
1 # Nguyễn Duy Đạt - B21DCAT056
2 import socket
3 import hashlib
4
5 host = "localhost"
6 port = 5000
7
8 key = "b21dcat056"
9
10 s = socket.socket()
11 s.bind((host, port))
12 s.listen(1)
13 print("Waiting for incoming connection...")
14
15 conn, addr = s.accept()
16 print("Connected to: ", addr)
17 while True:
18     data = conn.recv(1024).decode()
19     if not data:
20         break
21     message, received_hash = data.split("|")
22     message_with_key = message + key
23     h = hashlib.sha256(message_with_key.encode())
24     calculated_hash = h.hexdigest()
25     if calculated_hash == received_hash:
26         conn.send("1".encode())
27         print("Hash: " + received_hash)
28         print("Received from client: " + message)
29         print("Data integrity verified")
30     else:
31         conn.send("0".encode())
32         print("The received message has lost its integrity.")
33
34 conn.close()
35
```

```
Command Prompt
C:\Users\datnd>echo "Nguyễn Duy Đạt B21DCAT056"
"Nguyễn Duy Đạt B21DCAT056"

C:\Users\datnd>time
The current time is: 17:21:49.64
Enter the new time:
```

- phía Client sẽ gửi thông điệp "Hello, I am B21DCAT056 Client" dạng plain text + giá trị băm của(thông điệp +key).Key được Client và Server thống nhất từ trước. Phía Server sẽ nhận và đối chiếu, nếu đảm bảo tính toàn vẹn thì sẽ in ra "Data integrity verified" và đồng thời gửi "1" cho Client để báo rằng tính toàn vẹn được xác thực



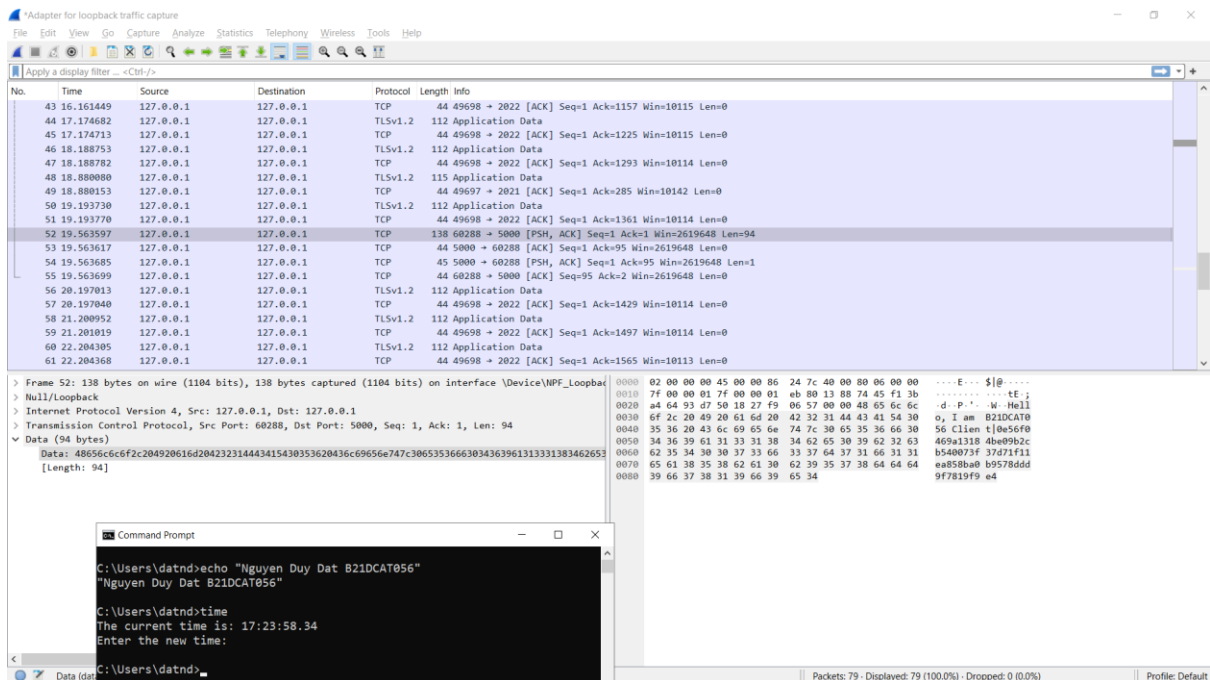
```
client2.py
1 # Nguyễn Duy Đạt - B21DCAT056
2 import socket
3 import hashlib
4
5 host = "localhost"
6 port = 5000
7
8 key = "b21dcat056"
9
10 s = socket.socket()
11 s.connect((host, port))
12
13 while True:
14     message = input("Enter data to send: ")
15     message_with_key = message + key
16     h = hashlib.sha256(message_with_key.encode())
17     message_hash = h.hexdigest()
18     data = message + "|" + message_hash
19     s.send(data.encode())
20     result = s.recv(1024).decode()
21     if result == "1":
22         print("Hash: " + message_hash)
23         print("Received from server: " + message)
24         print("Data integrity verified")
25     else:
26         print("The received message has lost its integrity.")
27
28 s.close()
29
```

```
server2.py
1 # Nguyễn Duy Đạt - B21DCAT056
2 import socket
3 import hashlib
4
5 host = "localhost"
6 port = 5000
7
8 key = "b21dcat056"
9
10 s = socket.socket()
11 s.bind((host, port))
12 s.listen(1)
13 print("Waiting for incoming connection...")
14
15 conn, addr = s.accept()
16 print("Connected to: ", addr)
17 while True:
18     data = conn.recv(1024).decode()
19     if not data:
20         break
21     message, received_hash = data.split("|")
22     message_with_key = message + key
23     h = hashlib.sha256(message_with_key.encode())
24     calculated_hash = h.hexdigest()
25     if calculated_hash == received_hash:
26         conn.send("1".encode())
27         print("Hash: " + received_hash)
28         print("Received from client: " + message)
29         print("Data integrity verified")
30     else:
31         conn.send("0".encode())
32         print("The received message has lost its integrity.")
33
34 conn.close()
35
```

```
PS D:\Workspace\ttcs\bai15> python server2.py
Waiting for incoming connection...
Connected to: ('127.0.0.1', 60288)
Hash: 0e56f0469a13184be09b2cb540073f37d71f1ea858ba0b9578dd9f7819f9e4
Received from client: Hello, I am B21DCAT056 Client
Data integrity verified
1
```

```
PS D:\Workspace\ttcs\bai15> python client2.py
Enter data to send: Hello, I am B21DCAT056 Client
Hash: 0e56f0469a13184be09b2cb540073f37d71f1ea858ba0b9578dd9f7819f9e4
Received from server: Hello, I am B21DCAT056 Client
Data integrity verified
Enter data to send:
```

- Bắt được các bản tin trao đổi giữa client và server trong Wireshark



- Bây giờ ta sẽ thay đổi Key ở phía Client và sau đó gửi lại thông điệp "Hello, I am B21DCAT056 Client" + giá trị băm. Server nhận được và đối chiếu giá trị băm thấy thay đổi và in ra "The received message has lost its integrity." và gửi "0" lại cho Client để báo rằng thông điệp bị mất đi tính toàn vẹn.

