Báo cáo bài thực hành số 15

Môn học

Thực tập cơ sở

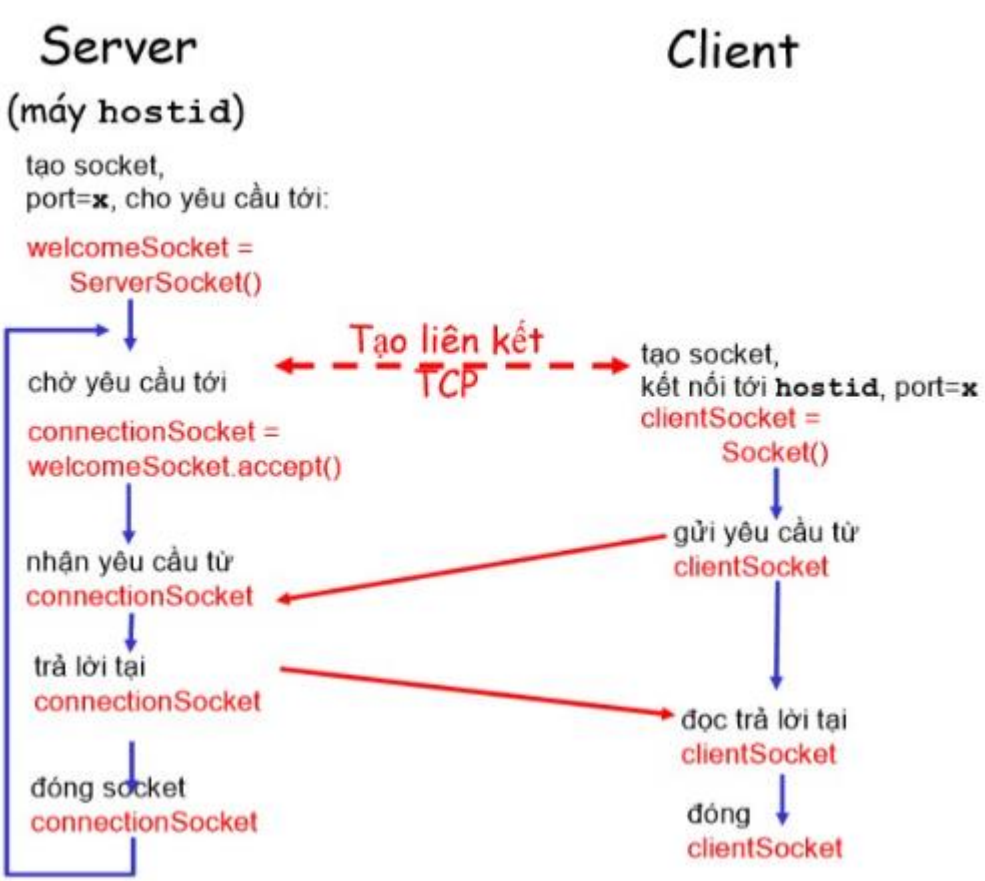
Giảng viên : Hoàng Xuân Dậu

Họ tên : Nguyễn Minh Phương

Mã SV: B19DCAT141

I. Lý thuyết:

- Socket
  + Là điểm cuối (end-point) trong liên kết truyền thông hai chiều (two-way communication) biểu diễn kết nối giữa Client – Server
  + Phân loại Socket:
    - Stream Socket: Dựa trên giao thức TCP, thiết lập giao tiếp 2 chiều; đảm bảo dữ liệu được truyền đến nơi nhận một cách đáng tin cậy, đúng tuần tự.
    - Datagram Socket: Dựa trên giao thức UDP, không yêu cầu có sự thiết lập kết nối giữa 2 process; ưu điểm là tốc độ giao thức nhanh.
- Lập trình socket với TCP



II. Thực hành:

- Lập trình client

client.py

Socket-Programing > client.py > ...

```python
import socket
import threading
import time
import hash

# IP Address and Port
HOST = '127.0.0.1'
PORT = 8080

# Key to ensure integrity msg
SECRET_KEY = "5D2E44719232EA78CD2B32"

# Create socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_address = (HOST, PORT)
print('connecting to %s port ' %str(server_address))
s.connect(server_address)

# Function sent message to server
def sent_msg():
    while True:
        msg = input('\nSent to server: ')
        if len(msg) == 0:
            continue

        time.sleep(0.2)

        # Hash msg and send msg + hash code
        hash_code = hash.get_hash_code(msg, SECRET_KEY)
        data_sent = msg + '|' + hash_code

        # Send data with TCP
        s.sendall(bytes(data_sent, "utf8"))

# Function receive data to server
def rev_msg():
    while True:
        # Receive data
        data = s.recv(1024)
        msg_rev, hash_code_rev = data.decode("utf8").split('|')

        # Check integrity of mesage
        if hash.check_integrity_msg(msg_rev, SECRET_KEY, hash_code_rev):
            print('\nReceive from server: ', msg_rev)
        else:
            print("\nThe received message has lost its integrity")

        time.sleep(0.2)
```

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

client.py

Socket-Programing > client.py > ...

```python
try:
    # Thread sent message
    th1 = threading.Thread(target=sent_msg, name='t1')

    # Thread receive data
    th2 = threading.Thread(target=rev_msg, name='t2')

    # Set two thread is Deamon
    th1.daemon = True
    th2.daemon = True

    # Start thread
    th1.start()
    th2.start()

    time.sleep(1)

    # End thread
    th1.join()
    th2.join()

finally:
    s.close()
```

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

- Lập trình server

server.py M

Socket-Programing > server.py > ...

```python
import socket
import sys
import threading
import time
import hash

# IP Address and Port
HOST = '127.0.0.1'
PORT = 8080

# Key to ensure integrity msg
SECRET_KEY = "5D2E44719232EA78CD2B32"

# Create socket
# Listen max 2 socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(2)

# Function sent message to client
def sent_msg(client):
    while True:
        msg = input('\nSent to client: ')
        if len(msg) == 0:
            continue

        time.sleep(0.2)

        # Hash msg and send msg + hash code
        hash_code = hash.get_hash_code(msg, SECRET_KEY)
        data_sent = msg + '|' + hash_code

        # Send data with TCP
        client.sendall(bytes(data_sent, "utf8"))


# Function receive data to client
def rev_msg(client):
    while True:
        # Receive data
        data = client.recv(1024)
        msg_rev, hash_code_rev = data.decode("utf8").split('|')

        # Check integrity of mesage
        if hash.check_integrity_msg(msg_rev, SECRET_KEY, hash_code_rev):
            print("\nReceive from client: " + msg_rev)
        else:
            print("\nThe received message has lost its integrity")
        time.sleep(0.2)
```

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

server.py M

Socket-Programing > server.py > rev_msg

```
while True:
    # Accept client connect to server
    # Create client socket
    client, addr = s.accept()

    try:
        print('Connected by', addr)

        # Thread sent message
        th1 = threading.Thread(target=sent_msg, args=(client,), name='t1')

        # Thread receive data
        th2 = threading.Thread(target=rev_msg, args=(client,), name='t2')

        # Set two thread is Deamon
        th1.daemon = True
        th2.daemon = True

        # Start thread
        th1.start()
        th2.start()

        time.sleep(1)

        # End thread
        th1.join()
        th2.join()

    finally:
        client.close()

s.close()
```

C:\WINDOWS\system32\cmd.exe

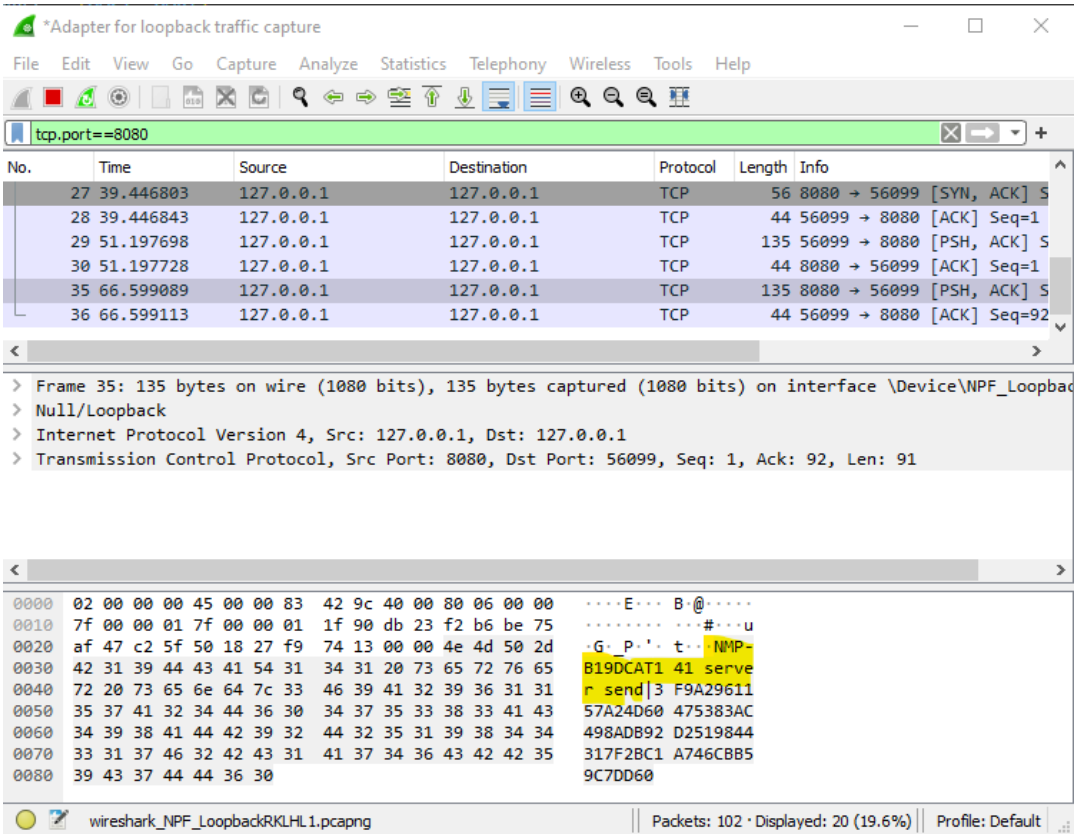
```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

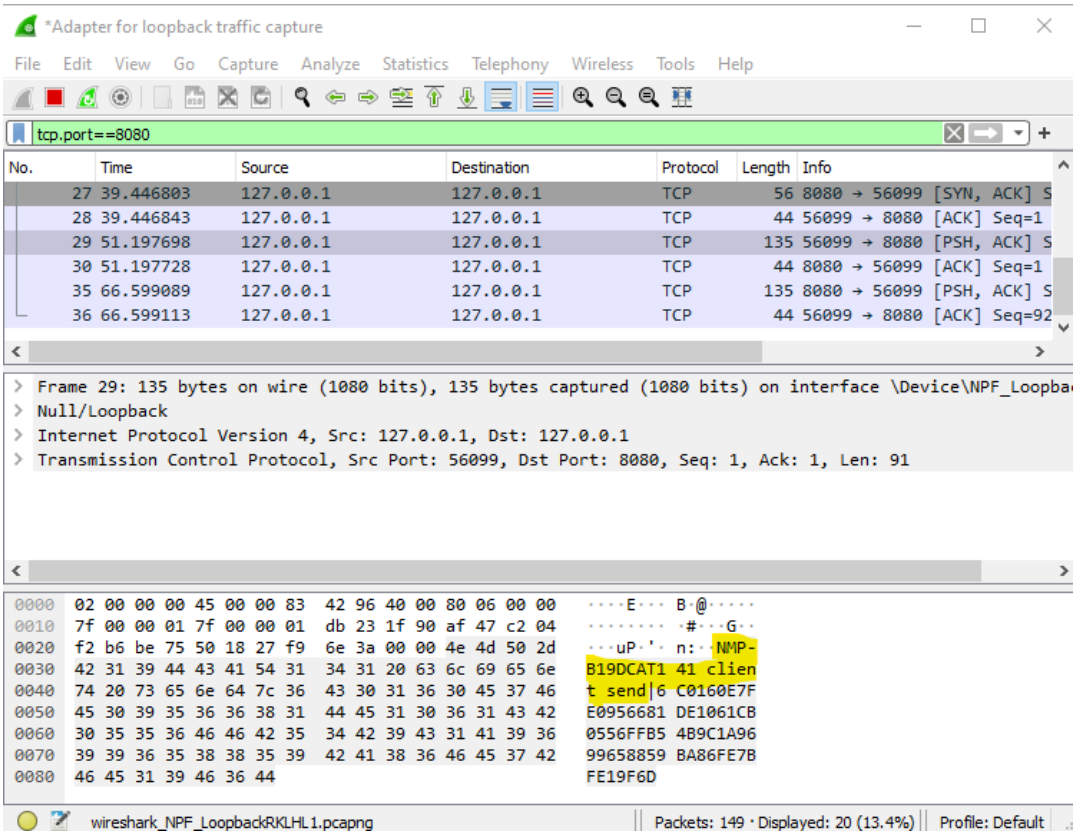
- Chạy server sau đó chạy client

```
PS C:\Users\phuon\OneDrive\Desktop\TTCS\Socket-Programing> python server.py
Connected by ('127.0.0.1', 56099)

Sent to client:
Receive from client: NMP-B19DCAT141 client send


Sent to client: NMP-B19DCAT141 server send

Sent to client:
```

```
PS C:\Users\phuon\OneDrive\Desktop\TTCS\Socket-Programing> python client.py
connecting to ('127.0.0.1', 8080) port

Sent to server: NMP-B19DCAT141 client send

Sent to server:
Receive from server:  NMP-B19DCAT141 server send


Sent to server:
```

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


*Adapter for loopback traffic capture
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port==8080
No. Time Source Destination Protocol Length Info
27 39.446803 127.0.0.1 127.0.0.1 TCP 56 8080 → 56099 [SYN, ACK] S
28 39.446843 127.0.0.1 127.0.0.1 TCP 44 56099 → 8080 [ACK] Seq=1
29 51.197698 127.0.0.1 127.0.0.1 TCP 135 56099 → 8080 [PSH, ACK] S
30 51.197728 127.0.0.1 127.0.0.1 TCP 44 8080 → 56099 [ACK] Seq=1
35 66.599089 127.0.0.1 127.0.0.1 TCP 135 8080 → 56099 [PSH, ACK] S
36 66.599113 127.0.0.1 127.0.0.1 TCP 44 56099 → 8080 [ACK] Seq=92
Frame 35: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on interface \Device\NPF_Loopbac
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 8080, Dst Port: 56099, Seq: 1, Ack: 92, Len: 91
0000 02 00 00 00 45 00 00 83 42 9c 40 00 80 06 00 00 ····E··· B·@·····
0010 7f 00 00 01 7f 00 00 01 1f 90 db 23 f2 b6 be 75 ···· ···#···u
0020 af 47 c2 5f 50 18 27 f9 74 13 00 00 4e 4d 50 2d ·G·_P·'· t···NMP-
0030 42 31 39 44 43 41 54 31 34 31 20 73 65 72 76 65 B19DCAT1 41 serve
0040 72 20 73 65 6e 64 7c 33 46 39 41 32 39 36 31 31 r send|3 F9A29611
0050 35 37 41 32 34 44 36 30 34 37 35 33 38 33 41 43 57A24D60 475383AC
0060 34 39 38 41 44 42 39 32 44 32 35 31 39 38 34 34 498ADB92 D2519844
0070 33 31 37 46 32 42 43 31 41 37 34 36 43 42 42 35 317F2BC1 A746CBB5
0080 39 43 37 44 44 36 30 9C7DD60
wireshark_NPF_LoopbackRKLHL1.pcapng
Packets: 102 · Displayed: 20 (19.6%)
Profile: Default


*Adapter for loopback traffic capture
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.port==8080
No. Time Source Destination Protocol Length Info
27 39.446803 127.0.0.1 127.0.0.1 TCP 56 8080 → 56099 [SYN, ACK] S
28 39.446843 127.0.0.1 127.0.0.1 TCP 44 56099 → 8080 [ACK] Seq=1
29 51.197698 127.0.0.1 127.0.0.1 TCP 135 56099 → 8080 [PSH, ACK] S
30 51.197728 127.0.0.1 127.0.0.1 TCP 44 8080 → 56099 [ACK] Seq=1
35 66.599089 127.0.0.1 127.0.0.1 TCP 135 8080 → 56099 [PSH, ACK] S
36 66.599113 127.0.0.1 127.0.0.1 TCP 44 56099 → 8080 [ACK] Seq=92
Frame 29: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on interface \Device\NPF_Loopba
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 56099, Dst Port: 8080, Seq: 1, Ack: 1, Len: 91
0000 02 00 00 00 45 00 00 83 42 96 40 00 80 06 00 00 ····E··· B·@·····
0010 7f 00 00 01 7f 00 00 01 db 23 1f 90 af 47 c2 04 ···· ·#···G··
0020 f2 b6 be 75 50 18 27 f9 6e 3a 00 00 4e 4d 50 2d ···uP·'· n:··NMP-
0030 42 31 39 44 43 41 54 31 34 31 20 63 6c 69 65 6e B19DCAT1 41 clien
0040 74 20 73 65 6e 64 7c 36 43 30 31 36 30 45 37 46 t send|6 C0160E7F
0050 45 30 39 35 36 36 38 31 44 45 31 30 36 31 43 42 E0956681 DE1061CB
0060 30 35 35 36 46 46 42 35 34 42 39 43 31 41 39 36 0556FFB5 4B9C1A96
0070 39 39 36 35 38 38 35 39 42 41 38 36 46 45 37 42 99658859 BA86FE7B
0080 46 45 31 39 46 36 44 FE19F6D
wireshark_NPF_LoopbackRKLHL1.pcapng
Packets: 149 · Displayed: 20 (13.4%)
Profile: Default

- Từ client và server, sửa đổi để sao cho: khi gửi thông điệp sẽ gửi kèm theo giá trị băm của (thông điệp+key) để phía bên kia kiểm tra xác minh tính toàn vẹn. Hai bên có thể thống nhất một giá trị key trước đó.
    + Tạo hash module trong Python:

hash.py

Socket-Programing > hash.py > ...

```python
import hmac
import hashlib
import binascii

def get_hash_code(msg, key):
    # Convert hex to bin
    # Encode string with utf8
    # Hash msg with SHA256, convert to hex
    key = binascii.unhexlify(key)
    msg = msg.encode()
    return hmac.new(key, msg, hashlib.sha256).hexdigest().upper()

def check_integrity_msg(msg, key, code_rev):
    # Hash msg with key and compare
    if code_rev == get_hash_code(msg, key):
        return True
    else:
        return False

if __name__ == "__main__":
    print(get_hash_code("PYTHON | language", "5D2A7139232AD22B23"))
```

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

```python
# Function sent message to client
def sent_msg(client):
    while True:
        msg = input('\nSent to client: ')
        if len(msg) == 0:
            continue

        time.sleep(0.2)

        # Hash msg and send msg + hash code
        hash_code = hash.get_hash_code(msg, SECRET_KEY)
        data_sent = msg + '|' + hash_code

        # Send data with TCP
        client.sendall(bytes(data_sent, "utf8"))
```

C:\WINDOWS\system32\cmd.exe

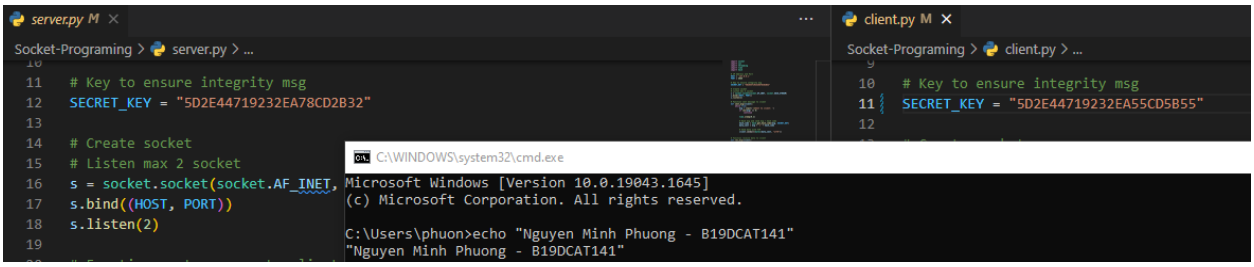
```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

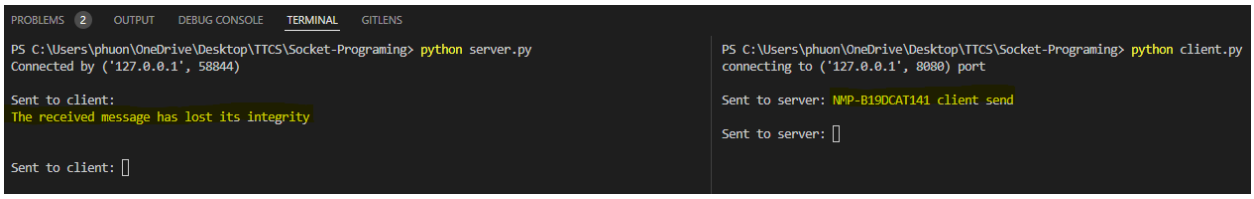
```python
# Function receive data to client
def rev_msg(client):
    while True:
        # Receive data
        data = client.recv(1024)
        msg_rev, hash_code_rev = data.decode("utf8").split('|')

        # Check integrity of mesage
        if hash.check_integrity_msg(msg_rev, SECRET_KEY, hash_code_rev):
            print("\nReceive from client: " + msg_rev)
        else:
            print("\nThe received message has lost its integrity")
        time.sleep(0.2)
```

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\phuon>echo "Nguyen Minh Phuong - B19DCAT141"
"Nguyen Minh Phuong - B19DCAT141"

C:\Users\phuon>
```

- Thay đổi giá trị key tại client và thực hiện gửi lại, nếu không đáp ứng tính toàn vẹn cần thông báo: “The received message has lost its integrity.”





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