

## LAB 6

### Concurrent Server - High-level Network programming



Họ tên và MSSV: Trần Quốc Phú B2205899

Nhóm học phần: CT29301

- *Các sinh viên bị phát hiện sao chép bài của nhau sẽ nhận 0đ cho tất cả bài thực hành của môn này.*
- *Bài nộp phải ở dạng PDF, hình minh họa phải rõ ràng chi tiết. Hình minh họa chỉ cần chụp ở nội dung thực hiện, không chụp toàn màn hình.*

#### 1. Bài 1

Cập nhật phần Server của Bài 2-Lab05 và Bài 4-Lab05 để có thể phục vụ song song nhiều Client cùng một lúc (sử dụng kỹ thuật Multi-Threading).

*(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)*

*Bài 2\_lab5:*

*Code: (phía Server):*

```
1 import socket
2 import threading
3
4 # Bảng ánh xạ số sang chữ
5 num_to_word = {
6     '0': "khong",
7     '1': "mot",
8     '2': "hai",
9     '3': "ba",
10    '4': "bon",
11    '5': "nam",
12    '6': "sau",
13    '7': "bay",
14    '8': "tam",
15    '9': "chin"
16 }
17
18 HOST = '0.0.0.0' # Lắng nghe trên tất cả các địa chỉ IP
19 PORT = 8888
20
21
22 # Hàm xử lý mỗi client
23 def handle_client(conn, addr):
24     print(f"[+] Client kết nối từ {addr}")
25     try:
26         data = conn.recv(1024).decode().strip()
27         print(f"[{addr}] Nhận: '{data}'")
28
29         if data in num_to_word:
30             response = num_to_word[data]
31         else:
32             response = "Không phải số nguyên"
33
34         conn.send(response.encode())
35     except Exception as e:
36         print(f"[{addr}] Lỗi: {e}")
37     finally:
38         conn.close()
39         print(f"[-] Đóng kết nối từ {addr}")
40
41 def main():
42     server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
43     server_socket.bind((HOST, PORT))
44     server_socket.listen(5)
45     print(f"Server đang lắng nghe tại cổng {PORT}...")
46
47     while True:
48         conn, addr = server_socket.accept()
49         thread = threading.Thread(target=handle_client, args=(conn, addr))
50         thread.start()
51         print(f"[=] Đang phục vụ {threading.active_count() - 1} client(s)")
52
53 if __name__ == "__main__":
54     main()
```

Code (phía client):

```
1  import socket
2
3
4  HOST = 'localhost' # hoặc IP của server
5  PORT = 8888
6
7  def main():
8      # Nhập 1 ký tự
9      ch = input("Nhập một ký tự số (0-9): ").strip()
10
11     # Tạo socket TCP
12     client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
13     client_socket.connect((HOST, PORT))
14
15     client_socket.send(ch.encode())
16     response = client_socket.recv(1024).decode()
17
18     print("Phản hồi từ Server:", response)
19
20     client_socket.close()
21
22 if __name__ == "__main__":
23     main()
```

Kết quả hiển thị:

```
D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5>python tcp_server.py
Server đang lắng nghe tại cổng 8888...
[+] Client kết nối từ ('127.0.0.1', 59865)
[=] Đang phục vụ 1 client(s)
[('127.0.0.1', 59865)] Nhận: '7'
[-] Đóng kết nối từ ('127.0.0.1', 59865)
█
```

```
D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5>python tcp_client.py
Nhập một ký tự số (0-9): 7
Phản hồi từ Server: bay
```

<pre>D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5&gt;python tcp_server.py Server đang lắng nghe tại cổng 8888... [+] Client kết nối từ ('127.0.0.1', 59865) [=] Đang phục vụ 1 client(s) [['127.0.0.1', 59865]] Nhận: '7' [-] Đóng kết nối từ ('127.0.0.1', 59865)</pre>	<pre>D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5&gt;python tcp_client.py Nhập một ký tự số (0-9): 7 Phản hồi từ Server: bay  D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5&gt; D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai2_lab5&gt;</pre>
---	--

#### Bài 4\_lab5:

Code (Phía server):

```
1  import socket
2  import threading
3  import os
4
5  HOST = '0.0.0.0'
6  CMD_PORT = 8000
7  DATA_PORT = 8001
8
9  pending_data_tasks = {}
10 pending_lock = threading.Lock()
11
12 def handle_data_connection(conn_data, addr):
13     client_ip = addr[0]
14     try:
15         with pending_lock:
16             if client_ip in pending_data_tasks:
17                 action, target = pending_data_tasks.pop(client_ip)
18             else:
19                 conn_data.close()
20                 return
21
22         if action == "GET":
23             with open(target, "rb") as f:
24                 while True:
25                     chunk = f.read(1024)
26                     if not chunk:
27                         break
28                     conn_data.sendall(chunk)
29
30         elif action == "LIST":
31             files = "\n".join(os.listdir(target))
32             conn_data.sendall(files.encode())
33
34     except Exception as e:
35         print(f"[DATA] Lỗi xử lý dữ liệu từ {addr}: {e}")
36     finally:
37         conn_data.close()
38
39 def data_server():
40     with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as data_socket:
41         data_socket.bind((HOST, DATA_PORT))
42         data_socket.listen(10)
43         print(f"[DATA SERVER] Đang lắng nghe dữ liệu tại cổng {DATA_PORT}...")
44
45         while True:
46             conn_data, addr = data_socket.accept()
47             threading.Thread(target=handle_data_connection, args=(conn_data, addr)).start()
```

```
1 def handle_command(conn_cmd, addr):
2     client_ip = addr[0]
3     try:
4         while True:
5             data = conn_cmd.recv(1024).decode().strip()
6             if not data:
7                 break
8             print(f"[COMMAND] Nhận từ {addr}: {data}")
9
10            parts = data.split()
11            if len(parts) != 2:
12                conn_cmd.sendall("ERROR\n".encode())
13                continue
14
15            cmd, target = parts
16
17            if cmd == "GET":
18                if os.path.isfile(target):
19                    conn_cmd.sendall("OK\n".encode())
20                    with pending_lock:
21                        pending_data_tasks[client_ip] = ("GET", target)
22                else:
23                    conn_cmd.sendall("ERROR\n".encode())
24
25            elif cmd == "DELETE":
26                if os.path.isfile(target):
27                    os.remove(target)
28                    conn_cmd.sendall("OK\n".encode())
29                else:
30                    conn_cmd.sendall("ERROR\n".encode())
31
32            elif cmd == "LIST":
33                if os.path.isdir(target):
34                    conn_cmd.sendall("OK\n".encode())
35                    with pending_lock:
36                        pending_data_tasks[client_ip] = ("LIST", target)
37                else:
38                    conn_cmd.sendall("ERROR\n".encode())
39
40            else:
41                conn_cmd.sendall("ERROR\n".encode())
42        except Exception as e:
43            print(f"[COMMAND] Lỗi xử lý lệnh từ {addr}: {e}")
44        finally:
45            print(f"[COMMAND] Đóng kết nối với {addr}")
46            conn_cmd.close()
```

```
1 def main():
2     threading.Thread(target=data_server, daemon=True).start()
3
4     with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_cmd_socket:
5         server_cmd_socket.bind((HOST, CMD_PORT))
6         server_cmd_socket.listen(10)
7         print(f"[SERVER] Đang lắng nghe lệnh tại cổng {CMD_PORT}...")
8
9         while True:
10             conn_cmd, addr = server_cmd_socket.accept()
11             print(f"[SERVER] Kết nối mới từ {addr}")
12             threading.Thread(target=handle_command, args=(conn_cmd, addr)).start()
13
14 if __name__ == "__main__":
15     main()
```

Code (Phía client):

```

1  import socket
2  import os
3
4  SERVER_IP = 'localhost'
5  CMD_PORT = 8000
6  DATA_PORT = 8001
7
8  def receive_data_to_file(filename):
9      with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
10         s.connect((SERVER_IP, DATA_PORT))
11         # with open("client_" + filename, "wb") as f:
12         with open("client_" + os.path.basename(filename), "wb") as f:
13             while True:
14                 data = s.recv(1024)
15                 if not data:
16                     break
17                 f.write(data)
18             print(f"[CLIENT] Đã lưu vào client_{filename}")
19
20 def receive_data_to_screen():
21     with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
22         s.connect((SERVER_IP, DATA_PORT))
23         data = s.recv(4096)
24         print("[CLIENT] Nội dung thư mục:")
25         print(data.decode())
26
27 def main():
28     with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as cmd_socket:
29         cmd_socket.connect((SERVER_IP, CMD_PORT))
30         while True:
31             command = input(
32                 "Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: "
33             ).strip()
34             if command.upper() == "EXIT":
35                 break
36
37             # Kiểm tra định dạng lệnh trước khi gửi
38             parts = command.strip().split()
39             if len(parts) != 2 or parts[0].upper() not in [
40                 "GET", "DELETE", "LIST"
41             ]:
42
43                 print("[CLIENT] Lệnh không hợp lệ. Vui lòng nhập đúng định dạng.")
44                 continue
45
46             cmd_socket.sendall((command + "\n").encode())
47             response = cmd_socket.recv(1024).decode().strip()
48             print(f"[CLIENT] Phản hồi: {response}")
49
50             if response == "OK":
51                 if parts[0].upper() == "GET":
52                     filename = parts[1]
53                     receive_data_to_file(filename)
54                 elif parts[0].upper() == "LIST":
55                     receive_data_to_screen()
56
57
58 if __name__ == "__main__":
59     main()

```

*Kết quả thực thi: (LIST, GET, DELETE)*

LIST:

```
D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai4_lab5>python tcp_client_dual.py
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: LIST Biba
[CLIENT] Phản hồi: OK
[CLIENT] Nội dung thư mục:
abc.txt
phutv.txt
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: █
```

```
D:\MangTTDL_CT293\NopLab\Lab6\C01\Bai4_lab5>python tcp_server_dual.py
[SERVER] Đang lắng nghe lệnh tại cổng 8000...
[DATA SERVER] Đang lắng nghe dữ liệu tại cổng 8001...
[SERVER] Kết nối mới từ ('127.0.0.1', 61929)
[COMMAND] Nhận từ ('127.0.0.1', 61929): LIST Biba
█
```

GET:

```
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: GET Biba/abc.txt
[CLIENT] Phản hồi: OK
[CLIENT] Đã lưu vào client_Biba/abc.txt
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: █
[COMMAND] Nhận từ ('127.0.0.1', 61929): GET Biba/abc.txt
█
```

DELETE:

```
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: DELETE
Biba/phutv04.txt
[CLIENT] Phản hồi: OK
Nhập lệnh (GET/DELETE/LIST <tên file hoặc thư mục>) hoặc EXIT: █
[COMMAND] Nhận từ ('127.0.0.1', 61929): DELETE Biba/phutv04
.txt
█
```

## 2. Bài 2

Tham khảo ví dụ MessageServer.py và MessageClient.py, viết chương trình Chat đơn giản sử dụng UDP socket cho phép hai người trên hai máy tính trò chuyện với nhau. Lưu ý: tạo 2 thread (1 dùng để gửi, 1 để nhận thông điệp) để chương trình cho phép người dùng nhận và gửi thông điệp song song.

(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)



Code: (Phía server)

```
1 # MessageServer.py
2 import socket
3 import threading
4
5 def receive_messages(sock):
6     while True:
7         try:
8             data, addr = sock.recvfrom(1024)
9             print(f"\n Từ {addr}: {data.decode()}")
10        except Exception as e:
11            print(f"Lỗi khi nhận: {e}")
12            break
13
14 def send_messages(sock, client_ip, client_port):
15     while True:
16         try:
17             message = input("Bạn: ")
18             sock.sendto(message.encode(), (client_ip, client_port))
19        except Exception as e:
20            print(f"Lỗi khi gửi: {e}")
21            break
22
23 def main():
24     server_ip = "0.0.0.0"
25     server_port = int(input("✎ Nhập cổng để lắng nghe (VD: 20001): "))
26     client_ip = input("✎ Nhập IP client (VD: 192.168.1.2): ").strip()
27     client_port = int(input("✎ Nhập PORT client (VD: 20002): "))
28
29     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
30     sock.bind((server_ip, server_port))
31
32     print(f"👉 Server đang lắng nghe tại {server_ip}:{server_port}...")
33
34     # Thread nhận và gửi
35     threading.Thread(target=receive_messages, args=(sock,), daemon=True).start()
36     threading.Thread(target=send_messages, args=(sock, client_ip, client_port), daemon=True).start()
37
38     # Giữ chương trình chạy
39     while True:
40         pass
41
42 if __name__ == "__main__":
43     main()
```

Phía client:

```
1 # MessageClient.py
2 import socket
3 import threading
4
5 def receive_messages(sock):
6     while True:
7         try:
8             data, addr = sock.recvfrom(1024)
9             print(f"\n 📄 Từ {addr}: {data.decode()}")
10        except Exception as e:
11            print(f"Lỗi khi nhận: {e}")
12            break
13
14 def send_messages(sock, server_ip, server_port):
15     while True:
16         try:
17             message = input("Bạn: ")
18             sock.sendto(message.encode(), (server_ip, server_port))
19        except Exception as e:
20            print(f"Lỗi khi gửi: {e}")
21            break
22
23 def main():
24     client_ip = "0.0.0.0"
25     client_port = int(input(" 📄 Nhập cổng để lắng nghe (VD: 20002): "))
26     server_ip = input("Nhập IP server (VD: 192.168.1.1): ").strip()
27     server_port = int(input("Nhập PORT server (VD: 20001): "))
28
29     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
30     sock.bind((client_ip, client_port))
31
32     print(f" 📄 Client đang lắng nghe tại {client_ip}:{client_port}...")
33
34     # Thread nhận và gửi
35     threading.Thread(target=receive_messages, args=(sock,), daemon=True).start()
36     threading.Thread(target=send_messages, args=(sock, server_ip, server_port), daemon=True).start()
37
38     while True:
39         pass
40
41 if __name__ == "__main__":
42     main()
```

Kết quả thực thi:

```
D:\MangTTDL_CT293\NopLab\Lab6\C02>python MessageServer.py
```

```
👉 Nhập cổng để lắng nghe (VD: 20001): 20001
```

```
Nhập IP client (VD: 192.168.1.2): 127.0.0.1
```

```
👉 Nhập cổng để lắng nghe (VD: 20001): 20001
```

```
Nhập IP client (VD: 192.168.1.2): 127.0.0.1
```

```
Nhập IP client (VD: 192.168.1.2): 127.0.0.1
```

```
Nhập PORT client (VD: 20002): 20002
```

```
Nhập PORT client (VD: 20002): 20002
```

```
📡 Server đang lắng nghe tại 0.0.0.0:20001...
```

```
📡 Server đang lắng nghe tại 0.0.0.0:20001...
```

```
Bạn: Hello
```

```
Bạn: Hello
```

```
Bạn:
```

```
    Từ ('127.0.0.1', 20002): Hi
```

```
Hi
```

```
Bạn: What is your name ?
```

```
Bạn:
```

```
    Từ ('127.0.0.1', 20002): My name is Phv
```

```
█
```

```
D:\MangTTDL_CT293\NopLab\Lab6\C02>python MessageClient.py
👉 Nhập cổng để lắng nghe (VD: 20002): 20002
Nhập IP server (VD: 192.168.1.1): 127.0.0.1
👉 Nhập cổng để lắng nghe (VD: 20002): 20002
Nhập IP server (VD: 192.168.1.1): 127.0.0.1
Nhập IP server (VD: 192.168.1.1): 127.0.0.1
Nhập PORT server (VD: 20001): 20001
Nhập PORT server (VD: 20001): 20001
👉 Client đang lắng nghe tại 0.0.0.0:20002...
👉 Client đang lắng nghe tại 0.0.0.0:20002...
Bạn:
Bạn:
👉 Từ ('127.0.0.1', 20001): Hello
👉 Từ ('127.0.0.1', 20001): Hello
Hi
Bạn:
👉 Từ ('127.0.0.1', 20001): Hi

👉 Từ ('127.0.0.1', 20001): What is your name ?
My name is Phv
Bạn: 
```

### 3. Bài 3


Sử dụng các thư viện hỗ trợ lập trình mạng ở mức độ High-level của Python để viết một chương trình EmailCrawler cho phép tìm các địa chỉ email trên một website.

Gợi ý:

- (1) Sử dụng hàm `urllib.request.urlopen` để load trang chủ của 1 website.
- (2) Tìm trong nội dung trang web tải xuống các email và URL. Đưa email và URL vào danh sách.
- (3) Tiếp tục tải các trang web có trong website sử dụng các URL tìm được trong danh sách. Lập lại đến khi tất cả hết các URL.
- Có thể sử dụng hàm `re.findall` để tìm các email và URL các trang web. Ví dụ:  
+ `emails = re.findall(r'[\w.+-]+@[\w-]+\.[\w.-]+', text)`  
+ `urls = re.findall(r'(http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|[*\(\),]|[\?:%[0-9a-fA-F][0-9a-fA-F]))+', text)`

(Chụp hình minh họa code bài làm và kết quả thực thi; đính kèm tập tin code khi nộp bài.)

Code:



```
1  import urllib.request
2  import re
3  from urllib.parse import urljoin, urlparse
4  from collections import deque
5
6  visited_urls = set()
7  found_emails = set()
8
9  def is_same_domain(base_url, target_url):
10     return urlparse(base_url).netloc == urlparse(target_url).netloc
```

```
1 def crawl(start_url):
2     queue = deque([start_url])
3
4     while queue:
5         url = queue.popleft()
6         if url in visited_urls:
7             continue
8
9         print(f"\n Đang kiểm tra: {url}")
10        visited_urls.add(url)
11
12        try:
13            with urllib.request.urlopen(url, timeout=5) as response:
14                content_type = response.headers.get("Content-Type", "")
15                if "text/html" not in content_type:
16                    continue
17
18                html = response.read().decode(errors='ignore')
19
20
21                print("\n--- Nội dung HTML ---")
22                print(html[:500]) # In 500 ký tự đầu tiên
23                print("--- Kết thúc trích HTML ---\n")
24
25                # Tìm email
26                emails = re.findall(r'[\w.+-]+@[ \w-]+\.[\w.-]+', html)
27                found_emails.update(emails)
28
29                # Tìm URL
30                urls = re.findall(
31                    r'https?://(?:[a-zA-Z0-9]|[$-_@.&+]|[*() ,]|(?:%[0-9a-fA-F]{2}))+',
32                    html
33                )
34
35                for link in urls:
36                    full_url = urljoin(url, link)
37                    if is_same_domain(start_url, full_url) and full_url not in visited_urls:
38                        queue.append(full_url)
39
40        except Exception as e:
41            print(f"Lỗi khi truy cập {url}: {e}")
```

```
1 def main():
2     start_url = input("Nhập URL trang web (VD: https://example.com): ").strip()
3     print("Bắt đầu tìm kiếm email...\n")
4     crawl(start_url)
5
6     print("\nHoàn tất. Các email tìm thấy:")
7     if found_emails:
8         for email in sorted(found_emails):
9             print("✉", email)
10    else:
11        print("Không tìm thấy email nào.")
12
13 if __name__ == "__main__":
14     main()
```

Kết quả thực thi:

```
D:\MangTTDL_CT293\NopLab\Lab6\C03>python email_crawler.py
Nhập URL trang web (VD: https://example.com): https://cs.stanford.edu/people/eroberts/
Bắt đầu tìm kiếm email...
```

```
Đang kiểm tra: https://cs.stanford.edu/people/eroberts/
```

```
--- Nội dung HTML ---
```

```
<html>
<head>
<title>Eric Roberts: Home Page</title>
</head>
<body>


<h1>Eric Roberts</h1>

Charles Simonyi Professor of Computer Science, <i>emeritus</i> <br />
<h4>Links:</h4>
<ul>
<li><a href="courses/index.html">Courses</a></li>
<li><a href="books/index.html">Books</a></li>
<li><a href="papers/index.html">Papers</a></li>
<li><a href="talks/index.html">Talks</a></li>
<li><a href="bio.html">Biographical sketch</a></li>
<li><a href="cv.html"
```

```
--- Kết thúc trích HTML ---
```

```
Hoàn tất. Các email tìm thấy:
```

```
✉ eroberts@cs.stanford.edu
```

--- Hết ---