EAST ASIA UNIVERSITY OF TECHNOLOGY

LẬP TRÌNH MẠNG
(Network Programming)
SOCKET – UTP/TPC

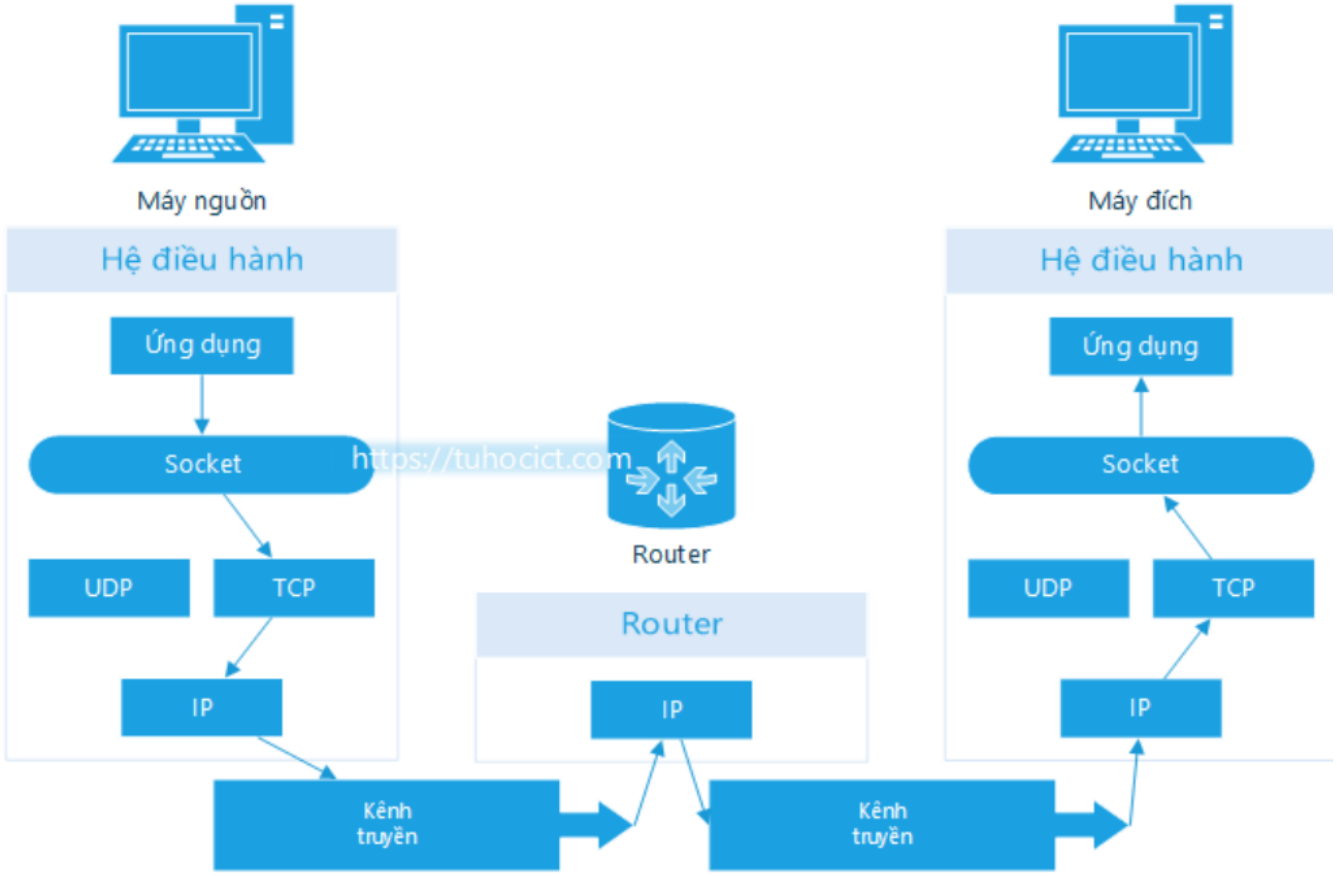
Nguyễn Anh Thơ
natho5578@gmail.com

Nội dung

Tuần	Nội dung	
01	Tổng quan lập trình Socket	
02	Lập trình Socket UDP	
03	Lập trình Socket TCP	
04	Lập trình Socket IP	

1. Tổng quan lập trình Socket

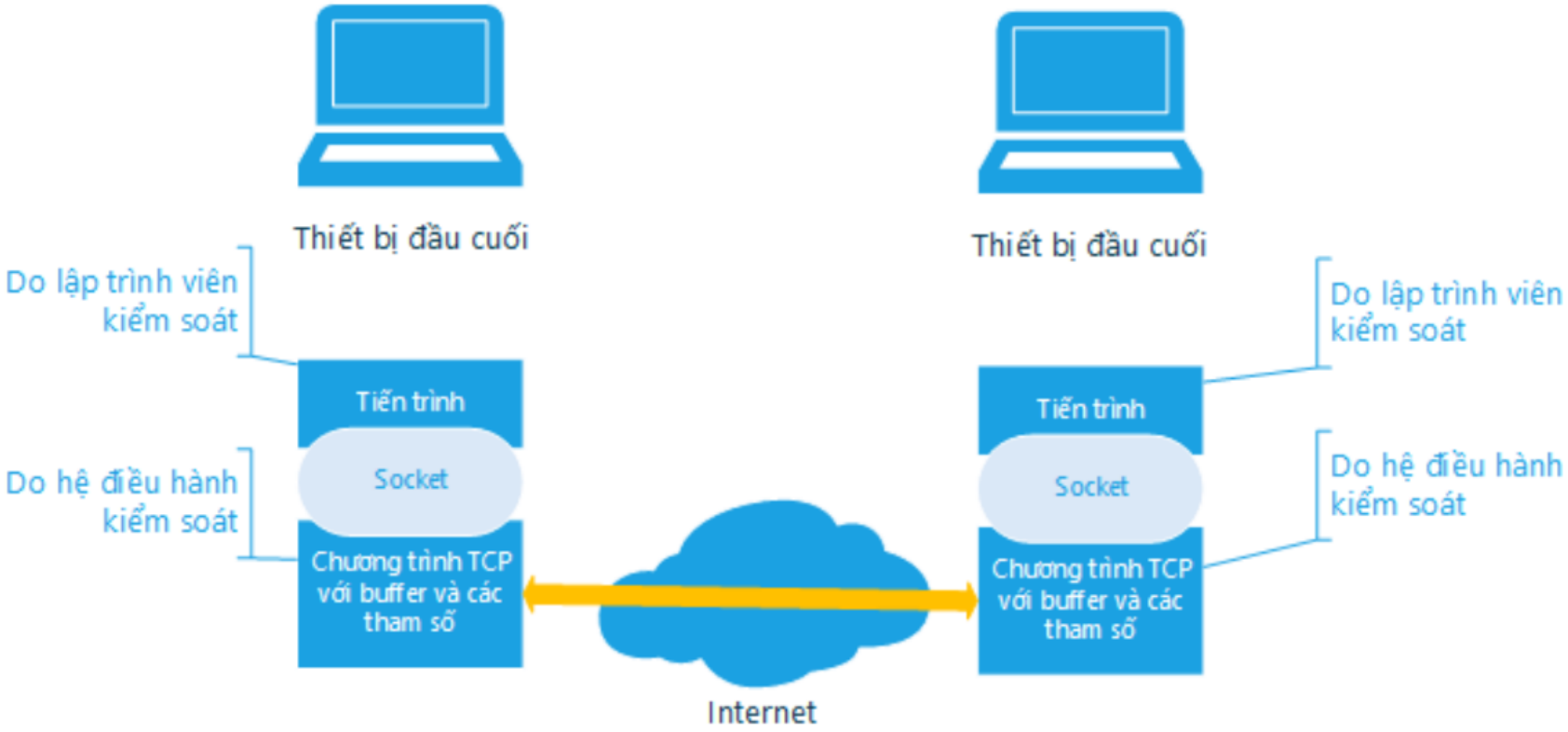
Góc nhìn truyền thông



Đường đi của dữ liệu từ ứng dụng qua socket tới mạng

1. Tổng quan lập trình Socket

- **Góc nhìn mô hình mạng:** Socket là giao diện tầng ứng dụng và dịch vụ tầng giao vận trên mỗi host



1. Tổng quan lập trình Socket

- **Góc nhìn lập trình**
  - Socket là một giao diện lập trình ứng dụng (API – Applycation Programing Interface) để gọi tới các chương trình con của hệ điều hành
  - Windows: Socket API gọi Windows Socket (Winsock)

1. Tổng quan lập trình Socket

- **Phân loại Socket**
  - **TCP Socket:**
    - Cung cấp dịch vụ truyền dữ liệu theo một liên kết ảo giữa hai tiến trình,
    - Socket hướng kết nối (Connectio – oriented socket) ,
    - Truyền dữ liệu như một chuỗi byte liên tục (stream socket)
  - **UDP Socket**
    - Không tạo liên kết (Socket phi liên kết – connectionless socket)
    - Truyền dữ liệu theo các gói (datagram) độc lập
  - **Raw Socket**
    - Dữ liệu truyền thẳng đến IP bỏ quan TCP/UDP

2. Lập trình Socket UDP

Cấu trúc header gói UDP

Offset		Octet#	0								1								2								3							
Octet	Bit	Bit#	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0		Source port																Destination port															
4	32		Length																Checksum															

- Source port number (2 byte): Số cổng tiến trình nguồn
- Destination port number (2 byte): Số cổng tiến trình đích
- Length = Header (8 byte)+ data (65527 byte)
- Checksum: kiểm tra lỗi (IP6 bắt buộc)

Giao thức sử dụng

- Domain Name System (DNS), Network Time Protocal (NTP), Network File System(NFS), DHCP, Trivial FTP(TFTP), Remote Procedure Call (RPC), Streaming media (IPTV)

2. Lập trình Socket UDP

Bước 1. Khởi tạo Socket UDP

```
Socket socket = new Socket(SocketType.Dgram, ProtocolType.Udp);
Socket socket = new Socket(AddressFamily.InterNetwork,
                           SocketType.Dgram, ProtocolType.Udp);
```

Bước 3. Xác định địa chỉ cổng tiến trình

```
string localIp = IPAddress.Any;
int localPort = 1308;
IPEndPoint localEndPoint = new IPEndPoint(localIp, localPort);
Socket socket = new Socket(AddressFamily.InterNetwork,
                           SocketType.Dgram, ProtocolType.Udp);
socket.Bind(localEndPoint);
```

2. Lập trình Socket UDP

Bước 3. Truyền dữ liệu qua Socket

- ***3.1. Chuyển đổi dữ liệu người dùng thành chuỗi byte***
  - Data Serialization
  - Net
    - Dữ liệu văn bản - Lớp Encoding với phương thức GetBytes
    - Số , logic – Lớp BitConverter
- ***3.2. Truyền dữ liệu***
  - Phương thức **SendTo**

```
// gửi mảng byte trên đến tiến trình server
socket.SendTo(sendBuffer, serverEndpoint);
// gửi kết quả lại cho client
socket.SendTo(sendBuffer, remoteEndpoint);
```

Nhận dữ liệu qua Socket UDP

```
Tại máy Server
int size = 1024;
byte receiveBuffer = new byte[size];
// biến này về sau sẽ chứa địa chỉ của tiến trình client nào gửi gói tin tới
EndPoint remoteEndpoint = new IPEndPoint(IPAddress.Any, 0);
// khi nhận được gói tin nào sẽ lưu lại địa chỉ của tiến trình client
int length = socket.ReceiveFrom(receiveBuffer, ref remoteEndpoint);
string text = Encoding.ASCII.GetString(receiveBuffer, 0, length);
```

```
Tại máy Client
int size = 1024; // kích thước của bộ đệm
byte receiveBuffer = new byte[size]; // mảng byte làm bộ đệm
// endpoint này chỉ dùng khi nhận dữ liệu
EndPoint dummyEndpoint = new IPEndPoint(IPAddress.Any, 0);
// nhận mảng byte từ dịch vụ Udp và lưu vào bộ đệm
// biến dummyEndpoint có nhiệm vụ lưu lại địa chỉ của tiến trình nguồn
// tuy nhiên, ở đây chúng ta đã biết tiến trình nguồn là Server
// do đó dummyEndpoint không có giá trị sử dụng
int length = socket.ReceiveFrom(receiveBuffer, ref dummyEndpoint);
// chuyển đổi mảng byte về chuỗi
string result = Encoding.ASCII.GetString(receiveBuffer, 0, length);
```

2. Lập trình Socket UDP

- **Bước 4. Đóng Socket**

```
// đóng socket và giải phóng tài nguyên
socket.Close();
```

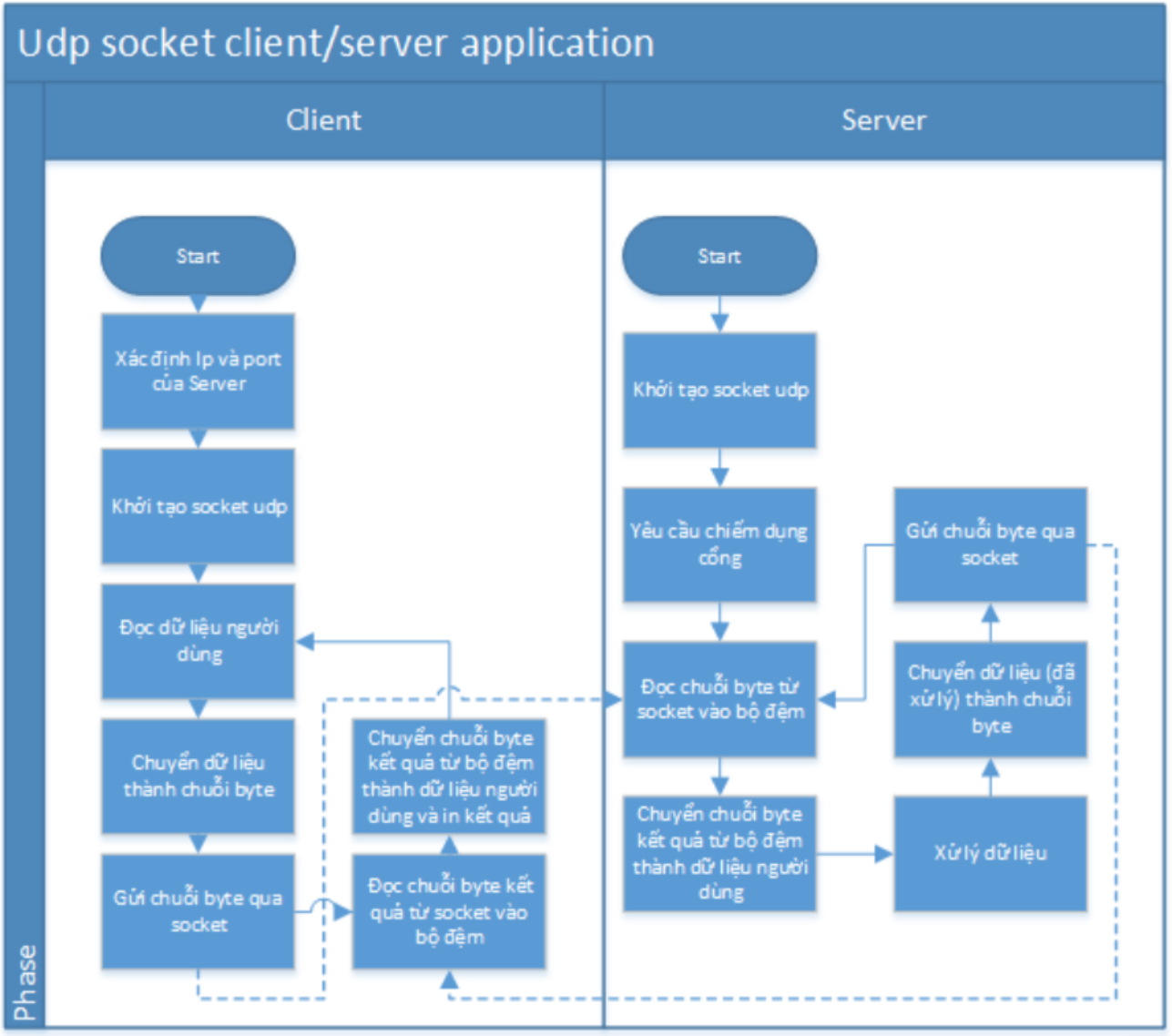
Bài tập

Bài toán

Viết một ứng dụng dòng lệnh đơn giản (console application) theo mô hình client/ server đáp ứng các yêu cầu:

1. Client cho phép người dùng nhập một chuỗi ký tự từ bàn phím và gửi chuỗi ký tự cho server;
2. Server nhận chuỗi ký tự, chuyển đổi tất cả ký tự thành dạng in hoa và gửi chuỗi kết quả lại cho client;
3. Client nhận kết quả và hiển thị lại cho người dùng.

Sơ đồ khối bài toán



Sơ đồ khối hoạt động của chương trình cơ bản sử dụng socket Udp

Tại máy Client

```
Console.Title = "Udp Client";
// yêu cầu người dùng nhập ip của server
Console.Write("Server IP address: ");
var serverIpStr = Console.ReadLine();
// chuyển đổi chuỗi ký tự thành object thuộc kiểu IPAddress
var serverIp = IPAddress.Parse(serverIpStr);
// yêu cầu người dùng nhập cổng của server
Console.Write("Server port: ");
var serverPortStr = Console.ReadLine();
// chuyển chuỗi ký tự thành biến kiểu int
var serverPort = int.Parse(serverPortStr);
// đây là "địa chỉ" của tiến trình server trên mạng
// mỗi endpoint chứa ip của host và port của tiến trình
var serverEndpoint = new IPEndPoint(serverIp, serverPort);
var size = 1024; // kích thước của bộ đệm
var receiveBuffer = new byte[size]; // mảng byte làm bộ đệm
```

Tại máy Client

```
var receiveBuffer = new byte[size]; // mảng byte làm bộ đệm
while (true)
{
    // yêu cầu người dùng nhập một chuỗi
    Console.ForegroundColor = ConsoleColor.Green;
    Console.Write("# Text >>> ");
    Console.ResetColor();
    var text = Console.ReadLine();
    // khởi tạo object của lớp socket để sử dụng dịch vụ Udp
    // lưu ý SocketType của Udp là Dgram (datagram)
    var socket = new Socket(SocketType.Dgram, ProtocolType.Udp);
    // biến đổi chuỗi thành mảng byte
    var sendBuffer = Encoding.ASCII.GetBytes(text);
    // gửi mảng byte trên đến tiến trình server
    socket.SendTo(sendBuffer, serverEndpoint);
    // endpoint này chỉ dùng khi nhận dữ liệu
    EndPoint dummyEndpoint = new IPEndPoint(IPAddress.Any, 0);
    // nhận mảng byte từ dịch vụ Udp và lưu vào bộ đệm
    // biến dummyEndpoint có nhiệm vụ lưu lại địa chỉ của tiến trình nguồn
    // tuy nhiên, ở đây chúng ta đã biết tiến trình nguồn là Server
    // do đó dummyEndpoint không có giá trị sử dụng
    var length = socket.ReceiveFrom(receiveBuffer, ref dummyEndpoint);
    // chuyển đổi mảng byte về chuỗi
    var result = Encoding.ASCII.GetString(receiveBuffer, 0, length);
    // xóa bộ đệm (để lần sau sử dụng cho yên tâm)
    Array.Clear(receiveBuffer, 0, size);
    // đóng socket và giải phóng tài nguyên
    socket.Close();
    // in kết quả ra màn hình
    Console.WriteLine($">>> {result}");
}
```

Tại Server

```
Console.Title = "Udp Server";
// giá trị Any của IPAddress tương ứng với Ip của tất cả các giao diện mạng trên máy
var localIp = IPAddress.Any;
// tiến trình server sẽ sử dụng cổng 1308
var localPort = 1308;
// biến này sẽ chứa "địa chỉ" của tiến trình server trên mạng
var localEndPoint = new IPEndPoint(localIp, localPort);
// yêu cầu hệ điều hành cho phép chiếm dụng cổng 1308
// server sẽ nghe trên tất cả các mạng mà máy tính này kết nối tới
// chỉ cần gói tin udp đến cổng 1308, tiến trình server sẽ nhận được
// một overload khác của hàm tạo Socket
// InterNetwork là họ địa chỉ dành cho IPv4
var socket = new Socket(AddressFamily.InterNetwork, SocketType.Dgram, ProtocolType.Udp);
socket.Bind(localEndPoint);
Console.WriteLine($"Local socket bind to {localEndPoint}. Waiting for request ...");
```

Tại Server

```
var size = 1024;
var receiveBuffer = new byte[size];
while (true)
{
    // biến này về sau sẽ chứa địa chỉ của tiến trình client nào gửi gói tin tới
    EndPoint remoteEndpoint = new IPEndPoint(IPAddress.Any, 0);
    // khi nhận được gói tin nào sẽ lưu lại địa chỉ của tiến trình client
    var length = socket.ReceiveFrom(receiveBuffer, ref remoteEndpoint);
    var text = Encoding.ASCII.GetString(receiveBuffer, 0, length);
    Console.WriteLine($"Received from {remoteEndpoint}: {text}");
    // chuyển chuỗi thành dạng in hoa
    var result = text.ToUpper();
    var sendBuffer = Encoding.ASCII.GetBytes(result);
    // gửi kết quả lại cho client
    socket.SendTo(sendBuffer, remoteEndpoint);
    Array.Clear(receiveBuffer, 0, size);
}
```

Bài tập 01

- Viết chương trình theo mô hình client- server sử dụng UDP Socket thực hiện yêu cầu sau:
  - Tại Client chọn chức năng sau
    - 1. Tạo dãy số gồm n phần tử ngẫu nhiên từ 1→100
    - 2. In dãy số
    - 3. Sắp xếp dãy số
    - 4. In ra các phần tử là số nguyên tố
    - 5. Thoát chương trình
  - Tại Server
    - Khi nhận được số 1→5 thì xử lý thông tin và gửi lại kết quả cho Client theo chức năng tương ứng.
  - Cliet nhận kết quả xử lý từ Server và hiện thị Kết quả + Thời gian hiện tại (Tính theo thời gian của hệ thống)

Bài tập 02

- Viết chương trình theo mô hình client- server sử dụng UDP Socket thực hiện yêu cầu sau:
  - Tại Cliet nhập vào danh sách Điểm học phần của sinh viên gồm thông tin: DiemSV(ID, Hovaten, DiemHP1, DiemHP2). Việc nhập danh sách chỉ dừng khi nhập ID=0;
  - Tại Server Nhận được thông tin DiemSV thực hiện tạo và cập nhật vào danh sách theo thông tin sau:

    ID, HovaTen, DiemGPA

    Biết rằng: DiemGPA được xác định như sau:

    Nếu 0<DTBHP<5 → DiemGPA="D"

    Nếu 5<DTBHP<7 → DiemGPA="C"

    Nếu 7<DTBHP<9 → DiemGPA="B"

    Nếu 9<DTBHP<10 → DiemGPA="A"

    DTBHP = (HP1 + 2*HP2)/3;
  - Server gửi lại Cliet dữ liệu đã xử lý + Thời gian xử lý (Thời gian hệ thống)
  - Client in lại kết quả nhận lại từ Server

QUESTION ?