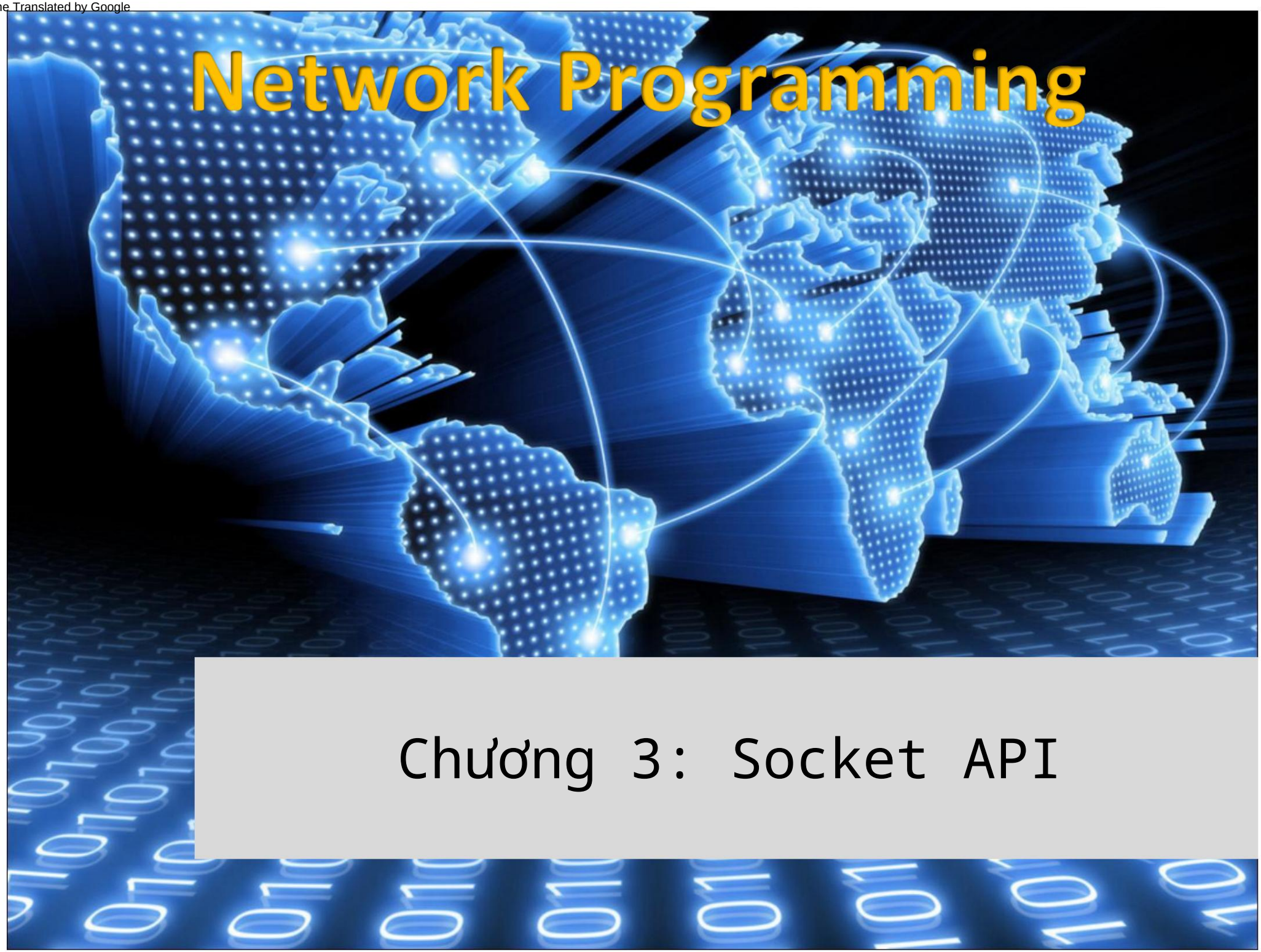


Chương 3: Socket API

Nội dung

- Cơ bản về socket
- API ổ cắm

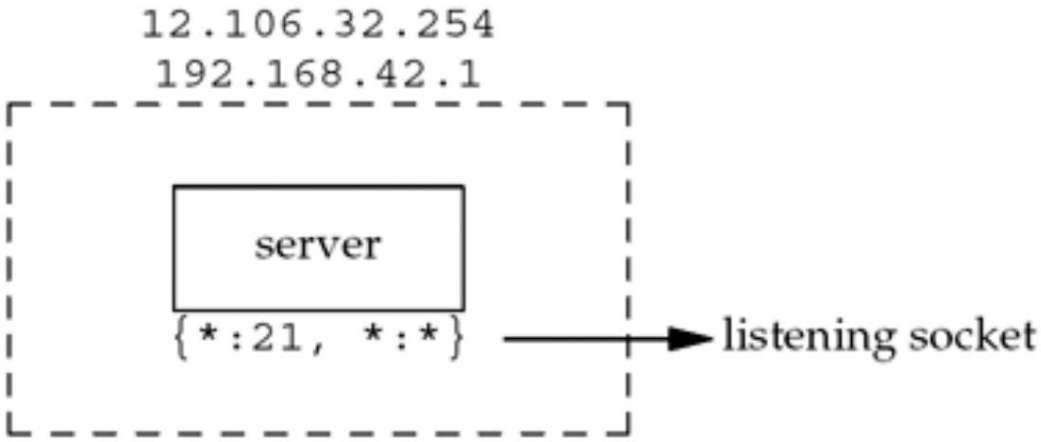
CƠ BẢN VỀ SOCKET

Cặp ổ cắm

- bốn-tuple xác định hai điểm cuối của sự kết nối
  - địa chỉ IP cục bộ
  - cảng địa phương
  - địa chỉ IP nước ngoài
  - cảng nước ngoài
- 2 giá trị xác định mỗi điểm cuối, địa chỉ IP và số cổng, thường được gọi là ổ cắm.

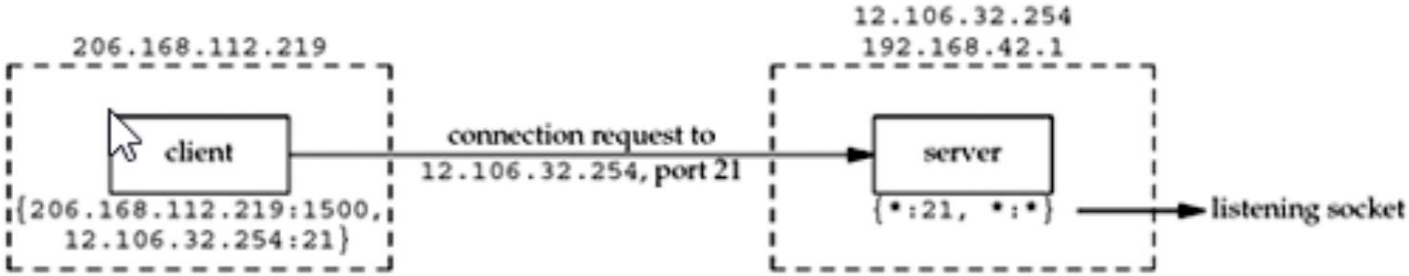
Số cổng TCP và đồng thời

Máy chủ (1)



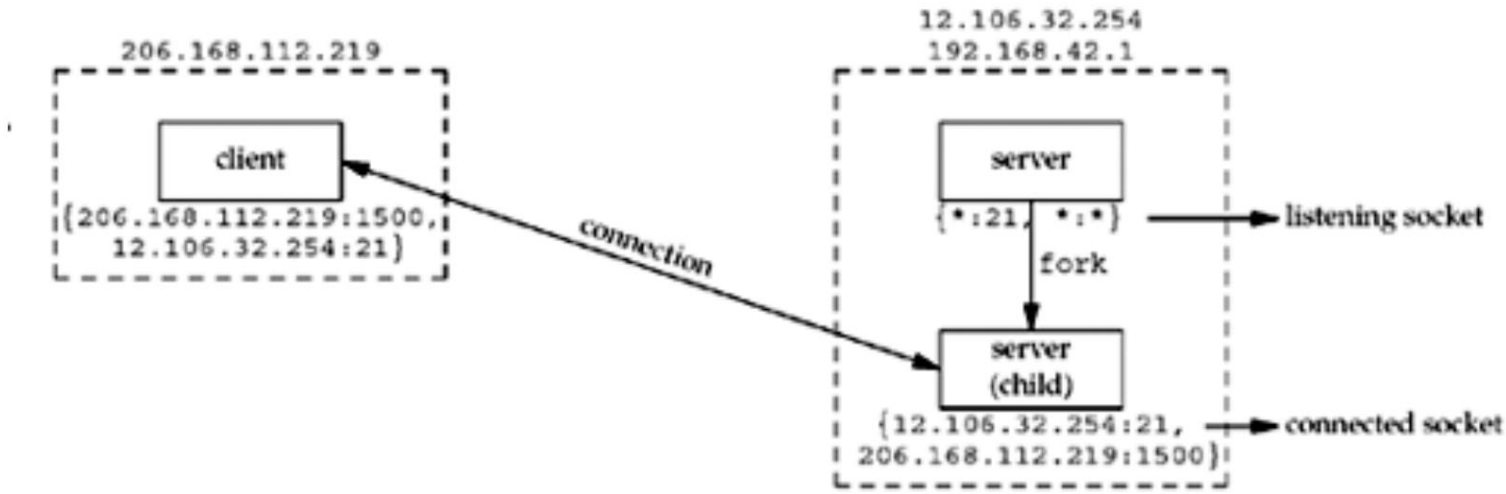
Số cổng TCP và đồng thời

Máy chủ (2)



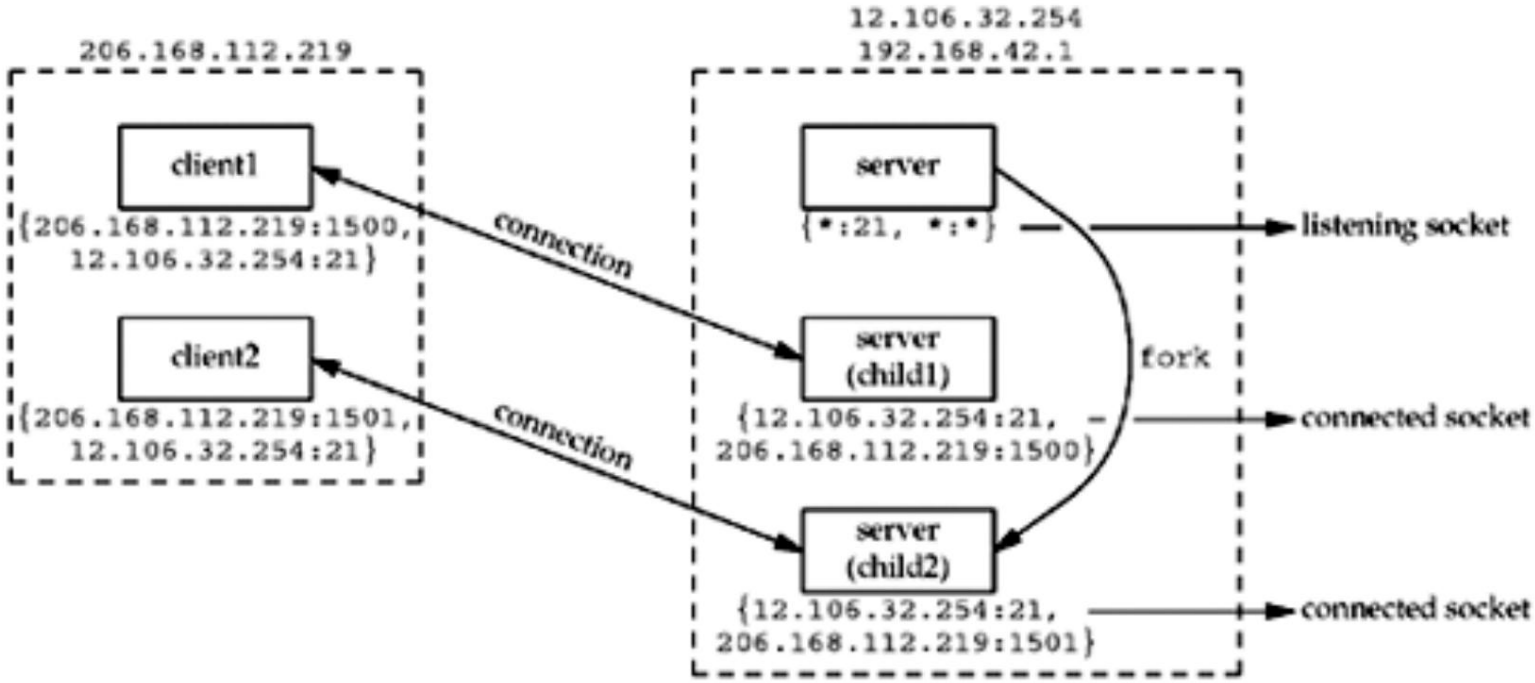
Số cổng TCP và đồng thời

Máy chủ (3)



Số cổng TCP và đồng thời

Máy chủ (4)



Kích thước và giới hạn của bộ đệm

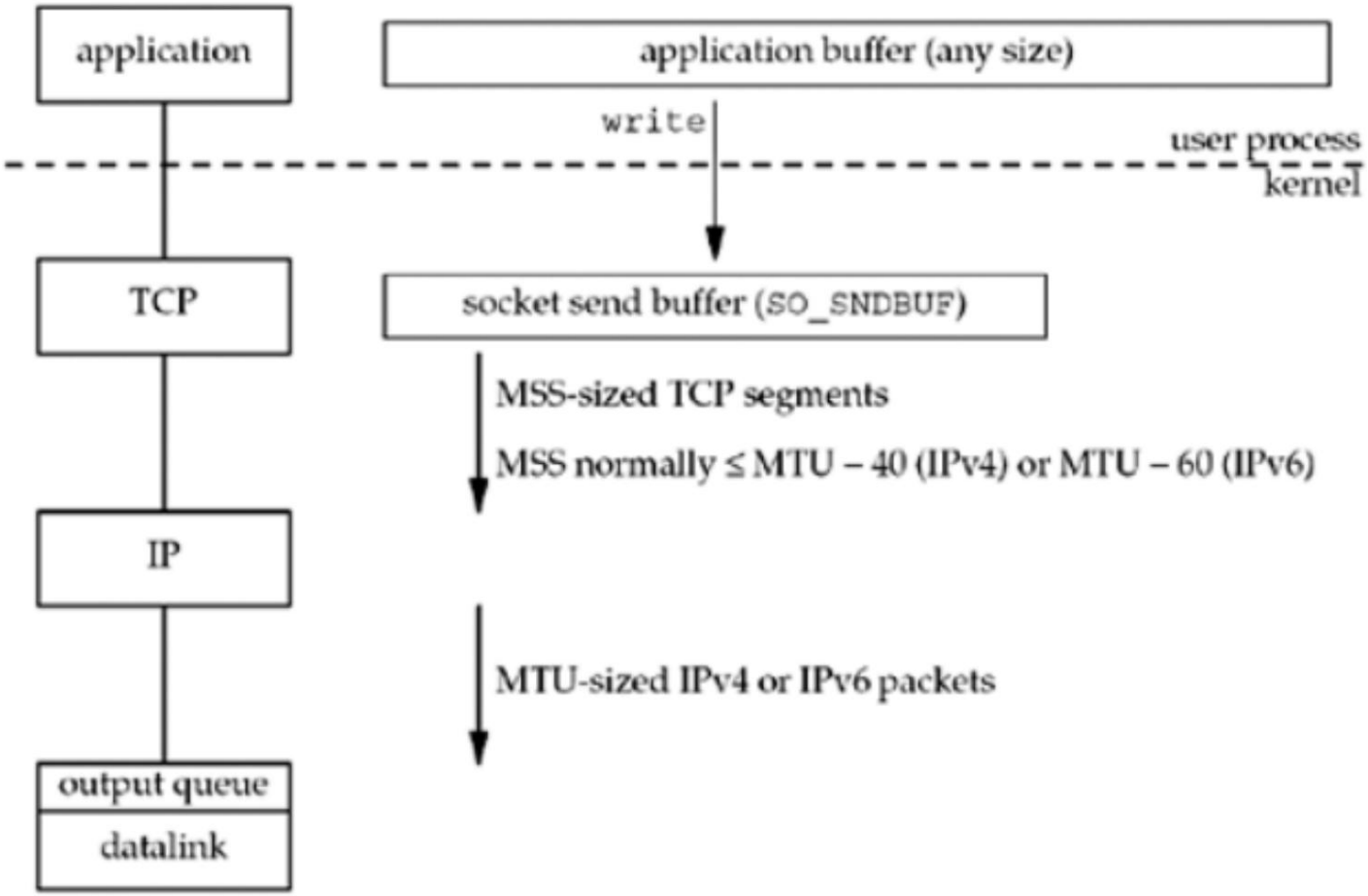
• Kích thước tối đa của một datagram IPv4: 65.538 byte • MTU (Đơn vị truyền tối đa) • Phân mảnh khi kích thước của datagram vượt quá MTU của liên kết.

  – Bit DF (không phân mảnh)

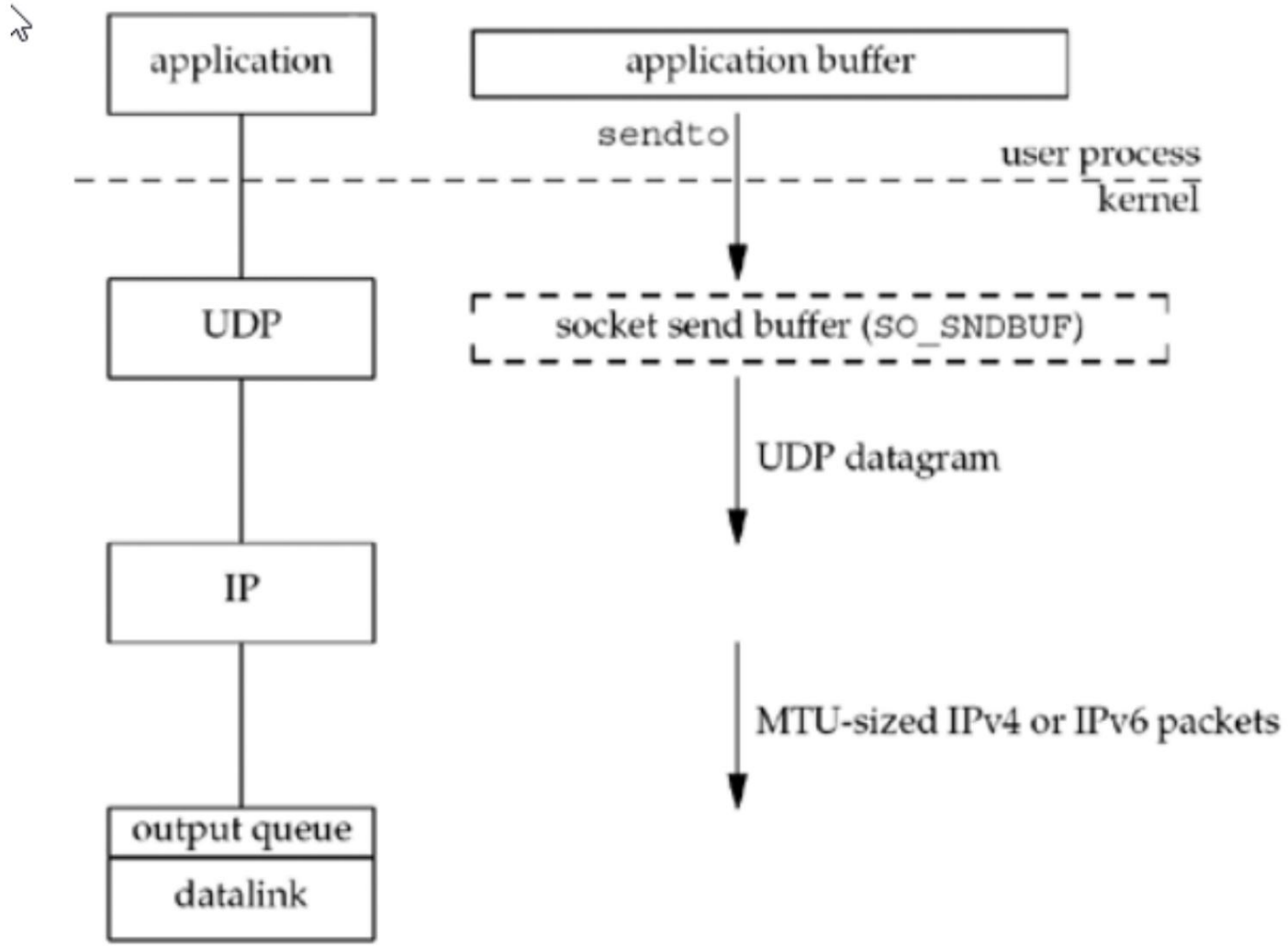
• MSS (kích thước phân đoạn tối đa): thông báo cho TCP ngang hàng về lượng dữ liệu TCP tối đa mà ngang hàng có thể gửi trên mỗi phân đoạn.

• MSS = MTU – kích thước cố định của header IP và TCP

Đầu ra TCP



Đầu ra UDP



Sử dụng giao thức theo Common Ứng dụng Internet

Application	IP	ICMP	UDP	TCP	SCTP
ping		•			
traceroute		•	•		
OSPF (routing protocol)	•				
RIP (routing protocol)			•		
BGP (routing protocol)				•	
BOOTP (bootstrap protocol)			•		
DHCP (bootstrap protocol)			•		
NTP (time protocol)			•		
TFTP			•		
SNMP (network management)			•		
SMTP (electronic mail)				•	
Telnet (remote login)				•	
SSH (secure remote login)				•	
FTP				•	
HTTP (the Web)				•	
NNTP (network news)				•	
LPR (remote printing)				•	
DNS			•	•	
NFS (network filesystem)			•	•	
Sun RPC			•	•	
DCE RPC			•	•	
IUA (ISDN over IP)					•
M2UA,M3UA (SS7 telephony signaling)					•
H.248 (media gateway control)			•	•	•
H.323 (IP telephony)			•	•	•
SIP (IP telephony)			•	•	•

API Ổ CẮM

Cấu trúc địa chỉ ổ cắm

- Cấu trúc địa chỉ ổ cắm IPv4: sockaddr_in (bao gồm <netinet/in.h>)

```
struct in_addr {
  in_addr_t    s_addr;          /* 32-bit IPv4 address */
                                /* network byte ordered */
};

struct sockaddr_in {
  uint8_t         sin_len;      /* length of structure (16) */
  sa_family_t     sin_family;   /* AF_INET */
  in_port_t       sin_port;     /* 16-bit TCP or UDP port number */
                                /* network byte ordered */
  struct in_addr  sin_addr;     /* 32-bit IPv4 address */
                                /* network byte ordered */
  char            sin_zero[8];  /* unused */
};
```

Ví dụ chương trình: init_sockaddr_in.c

Các kiểu dữ liệu được yêu cầu bởi đặc tả POSIX

Datatype	Description	Header
int8_t	Signed 8-bit integer	<sys/types.h>
uint8_t	Unsigned 8-bit integer	<sys/types.h>
int16_t	Signed 16-bit integer	<sys/types.h>
uint16_t	Unsigned 16-bit integer	<sys/types.h>
int32_t	Signed 32-bit integer	<sys/types.h>
uint32_t	Unsigned 32-bit integer	<sys/types.h>
sa_family_t	Address family of socket address structure	<sys/socket.h>
socklen_t	Length of socket address structure, normally uint32_t	<sys/socket.h>
in_addr_t	IPv4 address, normally uint32_t	<netinet/in.h>
in_port_t	TCP or UDP port, normally uint16_t	<netinet/in.h>

Địa chỉ các họ trong sys/socket.h

```
/*
 * Address families.
 */
#define AF_UNSPEC       0               /* unspecified */
#define AF_LOCAL        1               /* local to host (pipes, portals) */
#define AF_UNIX         AF_LOCAL        /* backward compatibility */
#define AF_INET         2               /* internetwork: UDP, TCP, etc. */
#define AF_IMPLINK      3               /* arpanet imp addresses */
#define AF_PUP          4               /* pup protocols: e.g. BSP */
#define AF_CHAOS        5               /* mit CHAOS protocols */
#define AF_NS           6               /* XEROX NS protocols */
#define AF_ISO          7               /* ISO protocols */
#define AF_OSI          AF_ISO
#define AF_ECMA         8               /* European computer manufacturers */
```

Đối số giá trị-kết quả

- khi cấu trúc địa chỉ socket được truyền tới bất kỳ hàm socket nào, nó luôn được truyền theo tham chiếu.

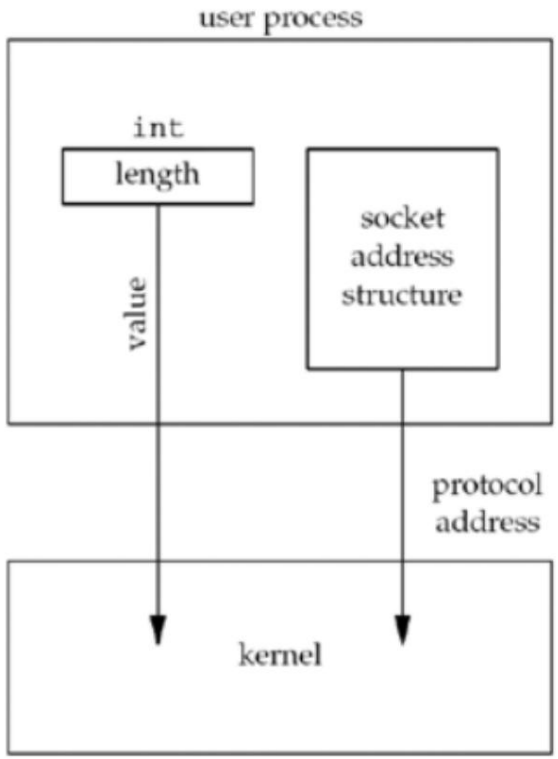
- Chiều dài của cấu trúc cũng được truyền vào như một đối số.
- 2 hướng: xử lý à kernel hoặc kernel à quá trình

Đối số giá trị-kết quả

- 3 chức năng: bind, connect & sendto: xử lý đến kernel

```
struct sockaddr_in serv;

/* fill in serv{} */
connect (sockfd, (SA *) &serv, sizeof(serv));
```



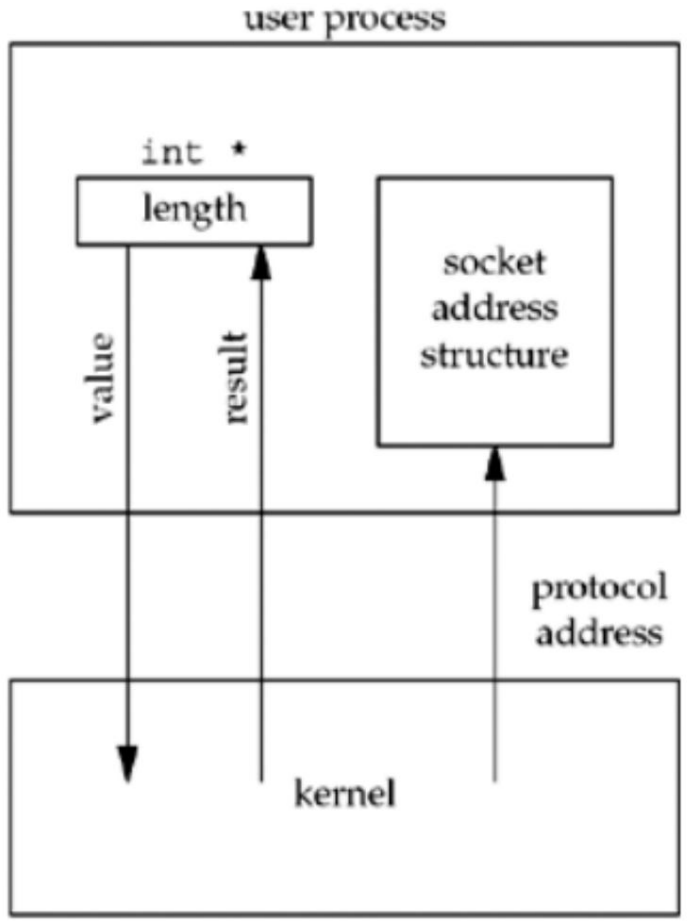
Ví dụ chương trình: ex_connect.c

Đối số giá trị-kết quả

- 4 chức năng: accept, recvfrom, getsockname và getpeername: kernel đến process

```
struct sockaddr_un  cli;    /* Unix domain */
socklen_t  len;

len = sizeof(cli);          /* len is a value */
getpeername(unixfd, (SA *) &cli, &len);
/* len may have changed */
```



Ví dụ chương trình: ex_getpeername.c

Chức năng chuyển đổi địa chỉ

- Họ chuyển đổi địa chỉ Internet giữa các chuỗi ASCII (cái gì con người thích sử dụng) và các giá trị nhị phân được sắp xếp theo byte của mạng (các giá trị được lưu trữ trong cấu trúc địa chỉ ổ cắm).

```
#include <arpa/inet.h> int
inet_aton(const char *strptr, struct in_addr *addrptr); Trả
về: 1 nếu chuỗi hợp lệ, 0
nếu có lỗi

in_addr_t inet_addr(const char *strptr);
Trả về: Địa chỉ IPv4 được sắp xếp theo byte mạng nhị phân 32 bit;
INADDR_NONE nếu lỗi

char *inet_ntoa(cấu trúc in_addr inaddr);
Trả về: con trỏ đến chuỗi thập phân chấm
```

Ví dụ chương trình: Convert_IP.c

Chức năng chuyển đổi địa chỉ (2)

```
#include <arpa/inet.h>

int inet_pton(int gia đình, const char *strptr,
void *addrptr);
```

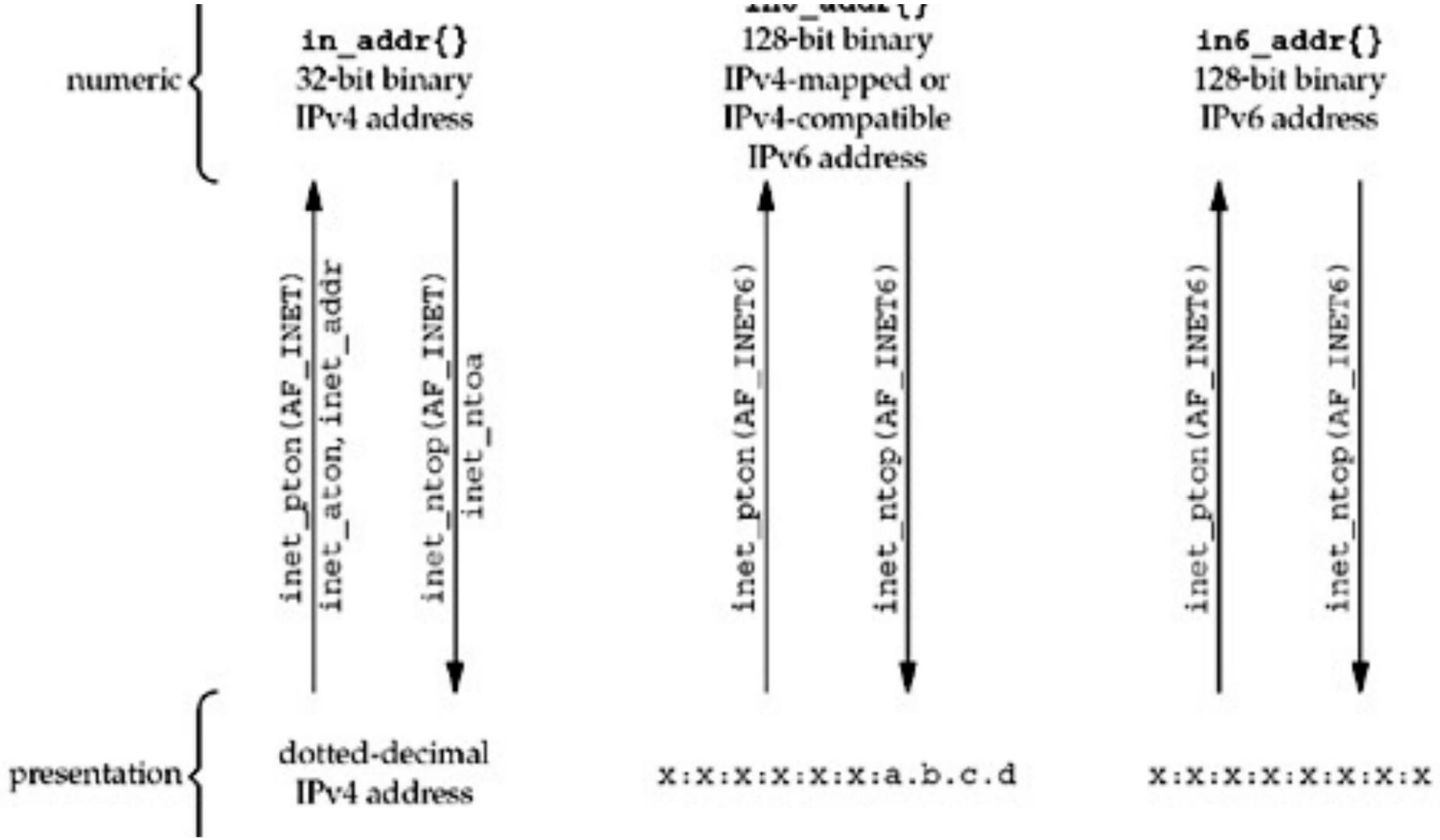
Trả về: 1 nếu OK, 0 nếu đầu vào không phải là định dạng trình bày hợp lệ, -1 nếu lỗi

```
const char *inet_ntop(int gia đình, const void *addrptr,
char *strptr, size_t len);
```

Trả về: con trỏ tới kết quả nếu OK, NULL nếu có lỗi

Ví dụ chương trình: Convert_IP_v4-6.c

Tóm tắt các hàm chuyển đổi địa chỉ



hàm sock_ntop

- Vấn đề của inet_ntop: nó yêu cầu người gọi phải truyền một con trỏ tới một địa chỉ nhị phân.

```
struct sockaddr_in địa chỉ;
inet_ntop(AF_INET,&addr.sin_addr,
  chuỗi, kích thước của(chuỗi));
```

- hàm sock_ntop

```
#include "unp.h"
char *sock_ntop(const struct sockaddr
  *sockaddr, socklen_t addrlen);
```

Ví dụ chương trình: example_of_sock_ntop.c

So sánh inet_ntop và sock_ntop

Aspect	`inet_ntop`	`sock_ntop`
Input	IP address in binary form (`in_addr`, `in6_addr`).	Socket address (`struct sockaddr`).
Output	IP address in string format.	Both IP address and port in string format.
Scope	Focuses on IP address only (IPv4 or IPv6).	Can handle both IP address and port (more general).
Transport Layer	Does not handle port numbers or protocol information.	Includes port numbers and supports transport layers.
Flexibility	Requires the user to separately manage socket info.	Handles entire socket info in one step.

đọc()

- Hàm read():

- Hàm read() được sử dụng để đọc dữ liệu từ một mô tả tệp, có thể bao gồm các mô tả tệp socket. Đối với luồng socket, read() đọc dữ liệu được gửi bởi một đối tác qua mạng.
- ssize_t đọc(int sockfd, void *buf, size_t đếm);
- Các thông số:
  - sockfd: Mô tả tệp của socket (được tạo bằng socket()).
  - buf: Con trỏ tới bộ đệm nơi dữ liệu được đọc từ ổ cắm sẽ được lưu trữ.
  - count: Số byte tối đa có thể đọc (kích thước của bộ đệm).

đọc()

- hàm read() trả về:
  - Nếu thành công, số byte thự c sự được đọc sẽ được trả về (có thể ít hơn count nếu số byte khả dụng ít hơn).
  - Khi xảy ra lỗi, trả về -1 và errno được thiết lập một cách thích hợp.
  - Nếu đối tác đã đóng kết nối, 0 là được trả về (cuối tệp).

đọc()

- Ví dụ:

```
char buffer[1024]; int
n = read(sockfd, buffer, sizeof(buffer)); if (n > 0)
{ printf("Đã
     nhận tin nhắn: %s\n", buffer); } else if (n == 0)
{ printf("Kết nối đã đóng
     bởi đối tác\n"); } else { perror("lỗi đọc");



}
```

viết()

- Hàm write():
- Hàm write() được sử dụng để gửi dữ liệu đến một mô tả tệp, có thể bao gồm các mô tả tệp socket. Đối với luồng socket, write() gửi dữ liệu đến đối tác được kết nối.

- ssize_t ghi(int sockfd, const void *buf, size_t đếm);

viết()

- Các thông số:
  - sockfd: Mô tả tệp của socket (được tạo bằng socket()).
  - buf: Một con trỏ tới bộ đệm chứa dữ liệu cần lưu trữ đã gửi.
  - đếm: Số byte để ghi (gửi) từ bộ đệm.
- Giá trị trả về:
  - Khi thành công, số byte thực sự được ghi là đã trả lại.
  - Khi xảy ra lỗi, -1 được trả về và errno được thiết lập phù hợp.

viết()

- Ví dụ:

```
char *message = "Xin chào, máy chủ!";
int n = write(sockfd, tin nhắn,
strlen(tin nhắn));
nếu (n > 0) {
    printf("Đã gửi tin nhắn: %s\n",
tin nhắn);
} khác {
    perror("lỗi ghi");
}
```

Chương trình ví dụ:

đọc-ghi-máy chủ.c

đọc-ghi-client.c