High

Performance

Distributed

System

KUAS – High Performance Distributed System

Linux Programming – Socket #1

Reporter: Po-Sen Wang

HPDS

Socket Function (1/7)

- int socket(int *domain*, int *type*, int *protocol*)
  //**Create a socket.**
- int bind(int *socket*, const struct sockaddr **address*, size_t *address_len*);
  //**Name a socket.**
- int listen(int *socket*, int *backlog*);
  //**Create a queue of socket.**
- int accept(int *socket*, struct sockaddr **address*, size_t **address_len*);
  //**Accept connetion.**
- int connect(int socket, const struct sockaddr *address, size_t address_len);
  //**Request connection.**

HPDS

Socket Function (2/7)

- #include <sys/types.h>
- #include <sys/socket.h>
- int socket(int *domain*, int *type*, int *protocol*)
  - *domain*:

AF_UNIX	UNIX internal (file system sockets)
AF_INET	ARPA internet protocols (UNIX network sockets)
AF_ISO	ISO standard protocols
AF_NS	Xerox network systems protocols
AF_IPX	Novell IPX protocol
AF_APPLETALK	Apple talk DDS

  - *type*: SOCK_STREAM (TCP) 、 SOCK_DGRAM (UDP)
  - *protocol*: 0 代表使用預設的協定。

HPDS

Socket Function (3/7)

- #include <sys/socket.h>
- int bind(int *socket*, const struct sockaddr **addre ss*, size_t *address_len*);
  - *socket*: File descriptor.
  - *address*: Socket address.
  - *address_len*: Address length.

HPDS

Socket Function (4/7)

- AF_UNIX:

```
struct sockaddr_un {
    sa_family_t sun_family;
    char sun_path[];
};
```

- AF_INET:

```
struct sockaddr_in {
    short int sin_family;
    unsigned short int sin_port;
    struct in_addr sin_addr;
};
```

```
struct in_addr {
    unsigned long int s_addr;
};
```

HPDS

Socket Function (5/7)

- #include <sys/socket.h>
- int listen(int *socket*, int *backlog*);
  - *socket*: File descriptor.
  - *backlog*: Maximum of queue for listen.

HPDS

Socket Function (6/7)

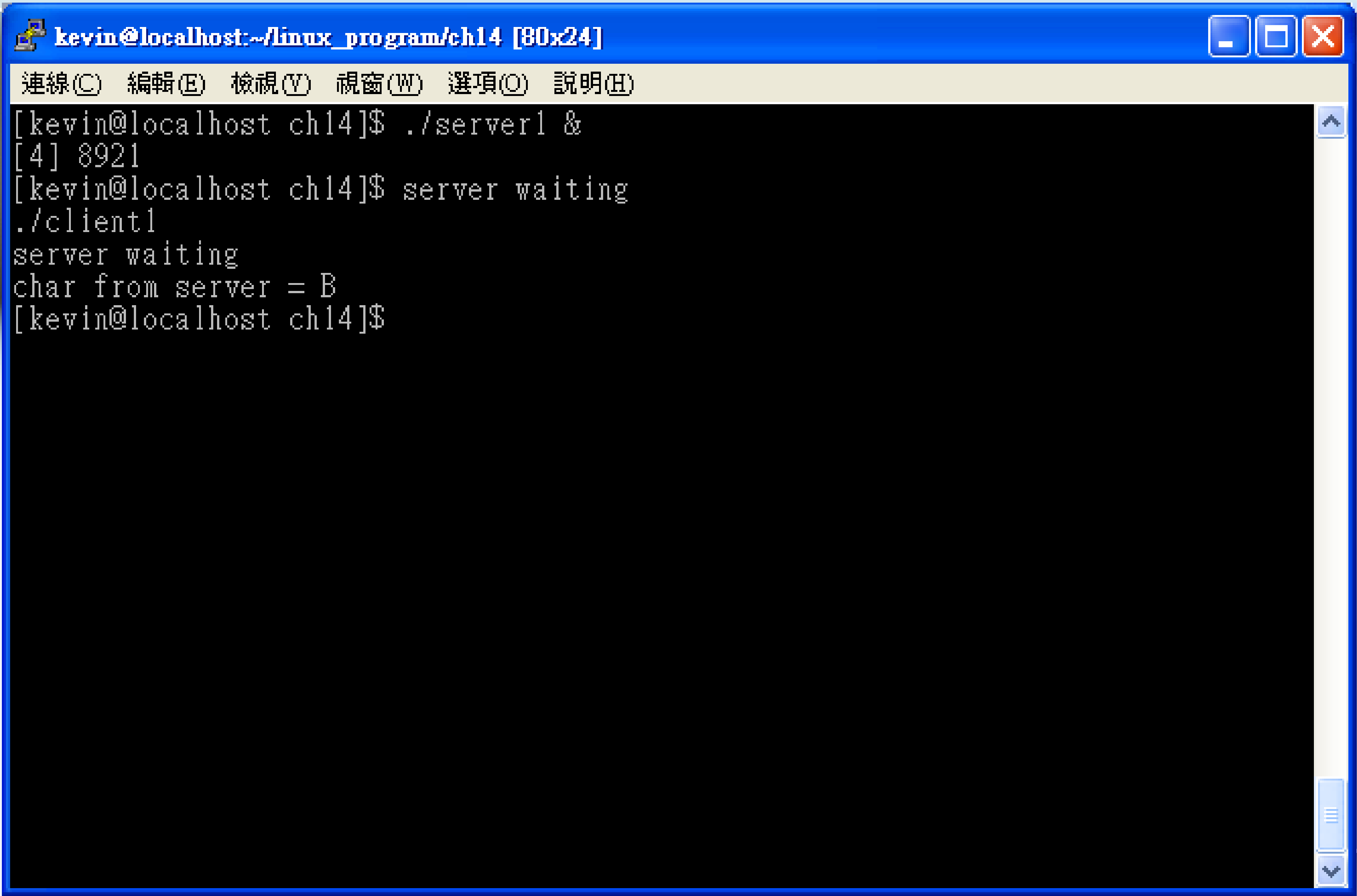
- #include <sys/socket.h>
- int accept(int *socket*, struct sockaddr **address*, size_t **address_len*);
  - *socket*: File descriptor.
  - *address*: Socket address.
  - *address_len*: Address length.

Socket Function (7/7)

- #include <sys/socket.h>
- int connect(int *socket*, const struct sockaddr **address*, size_t *address_len*);
    - *socket*: File descriptor.
    - *address*: Socket address.
    - *address_len*: Address length.

HPDS

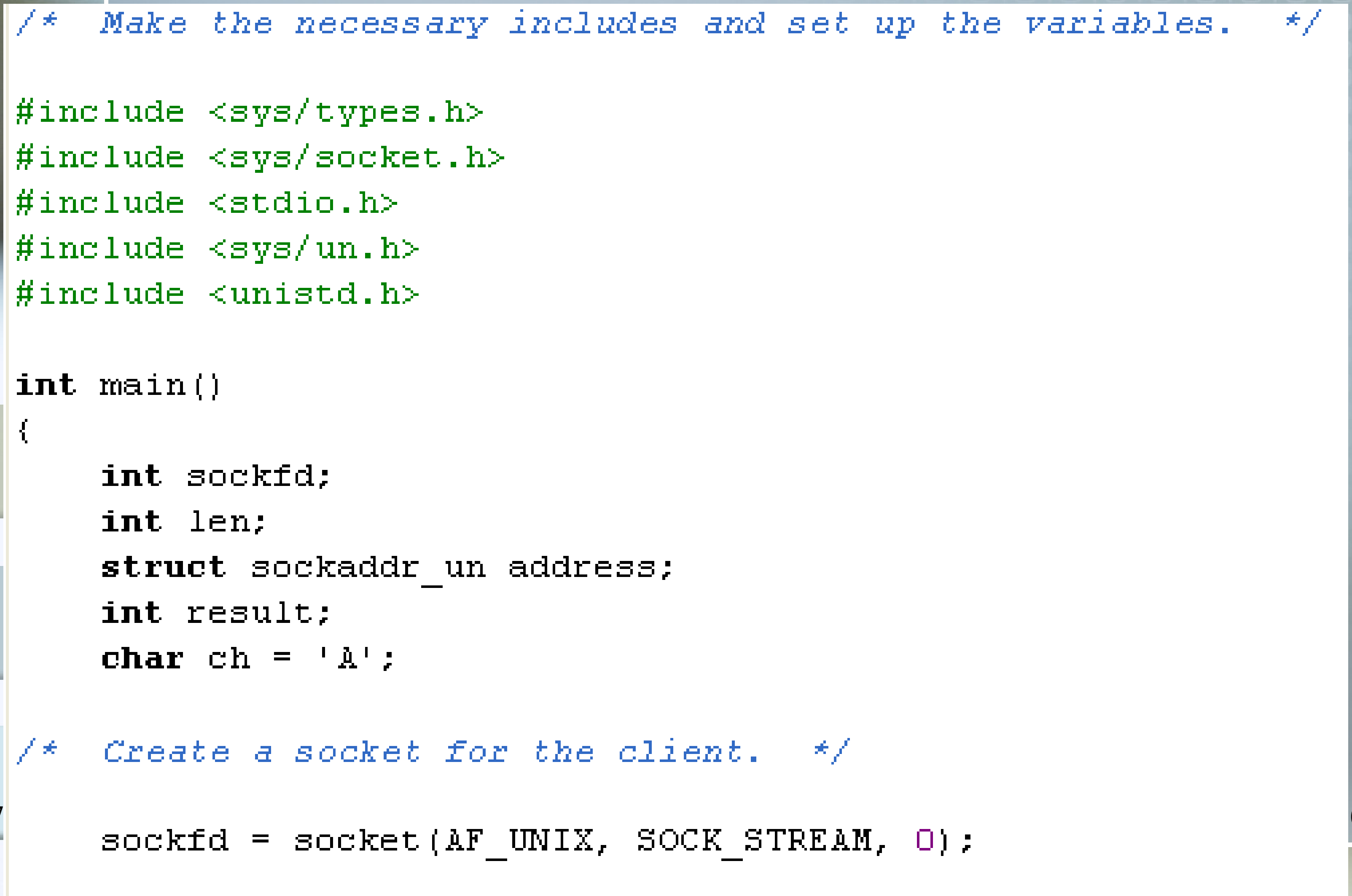
Socket Example 1 – Use AF_UNIX (1/6)


kevin@localhost:~/linux_program/ch14 [80x24]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)

```
[kevin@localhost ch14]$ ./server1 &
[4] 8921
[kevin@localhost ch14]$ server waiting
./client1
server waiting
char from server = B
[kevin@localhost ch14]$
```

Socket Example 1 – Use AF_UNIX (2/6)

- client1.c

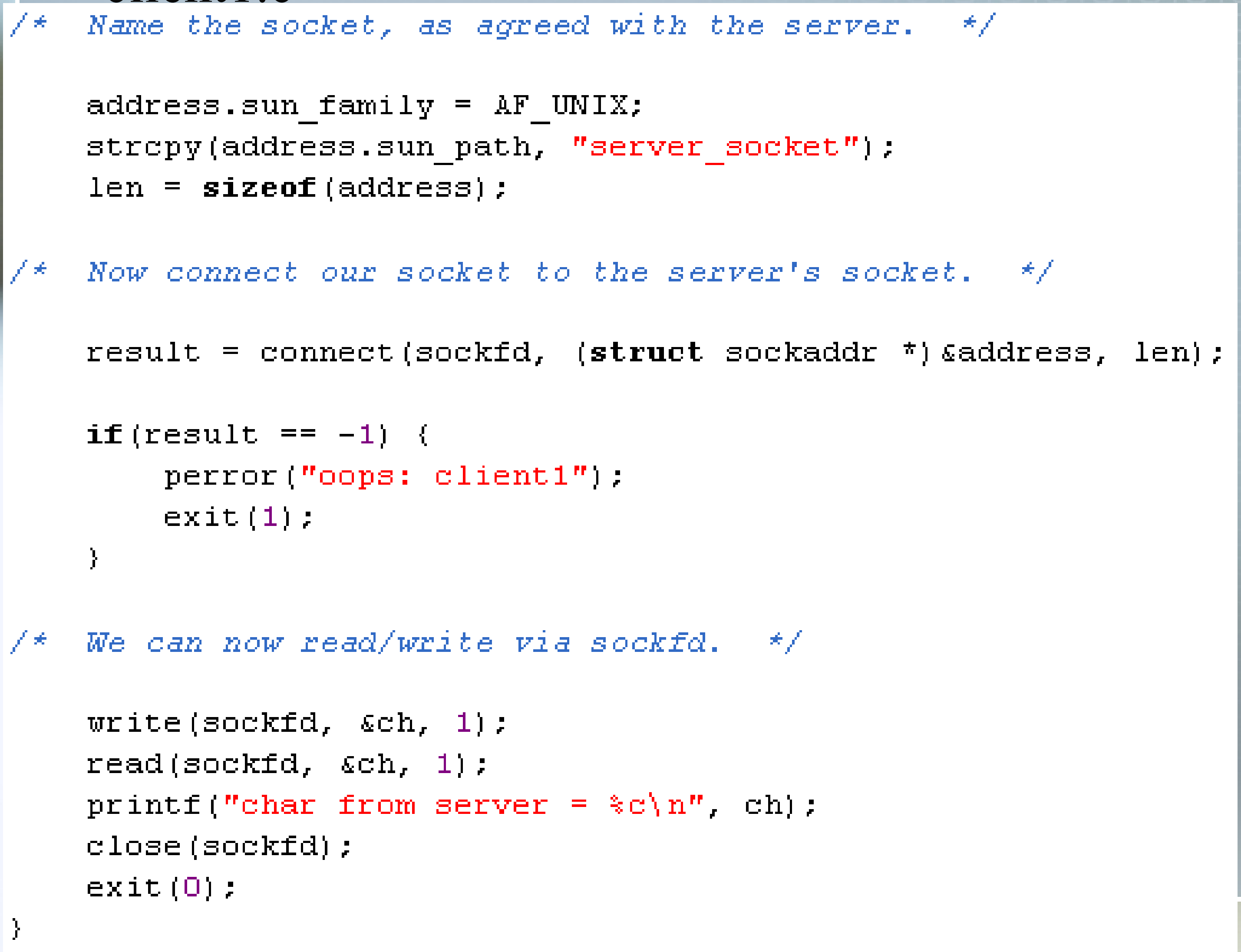


```
/*  Make the necessary includes and set up the variables.  */

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <sys/un.h>
#include <unistd.h>

int main()
{
    int sockfd;
    int len;
    struct sockaddr_un address;
    int result;
    char ch = 'A';

/*  Create a socket for the client.  */

    sockfd = socket(AF_UNIX, SOCK_STREAM, 0);
```

HPDS

Socket Example 1 – Use AF_UNIX (3/6)

- client1.c



```
/*  Name the socket, as agreed with the server.  */

    address.sun_family = AF_UNIX;
    strcpy(address.sun_path, "server_socket");
    len = sizeof(address);

/*  Now connect our socket to the server's socket.  */

    result = connect(sockfd, (struct sockaddr *)&address, len);

    if(result == -1) {
        perror("oops: client1");
        exit(1);
    }

/*  We can now read/write via sockfd.  */

    write(sockfd, &ch, 1);
    read(sockfd, &ch, 1);
    printf("char from server = %c\n", ch);
    close(sockfd);
    exit(0);
}
```

HPDS

Socket Example 1 – Use AF_UNIX (4/6)

- server1.c

```c
/*  Make the necessary includes and set up the variables.  */

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <sys/un.h>
#include <unistd.h>

int main()
{
    int server_sockfd, client_sockfd;
    int server_len, client_len;
    struct sockaddr_un server_address;
    struct sockaddr_un client_address;

/*  Remove any old socket and create an unnamed socket for the server.  */

    unlink("server_socket");
    server_sockfd = socket(AF_UNIX, SOCK_STREAM, 0);
```

HPDS

Socket Example 1 – Use AF_UNIX (5/6)

- server1.c

```
/*  Name the socket.  */

    server_address.sun_family = AF_UNIX;
    strcpy(server_address.sun_path, "server_socket");
    server_len = sizeof(server_address);
    bind(server_sockfd, (struct sockaddr *)&server_address, server_len);

/*  Create a connection queue and wait for clients.  */

    listen(server_sockfd, 5);
    while(1) {
        char ch;

        printf("server waiting\n");
```

HPDS

Socket Example 1 – Use AF_UNIX (6/6)

- server1.c

```
/*  Accept a connection.  */

        client_len = sizeof(client_address);
        client_sockfd = accept(server_sockfd,
            (struct sockaddr *)&client_address, &client_len);

/*  We can now read/write to client on client_sockfd.  */

        read(client_sockfd, &ch, 1);
        ch++;
        write(client_sockfd, &ch, 1);
        close(client_sockfd);
    }
}
```

HPDS

Socket Example 2 – Use AF_INET (1/5)

- client2.c

```
/*  Make the necessary includes and set up the variables.  */

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>

int main()
{
    int sockfd;
    int len;
    struct sockaddr_in address;
    int result;
    char ch = 'A';

/*  Create a socket for the client.  */

    sockfd = socket(AF_INET, SOCK_STREAM, 0);
```

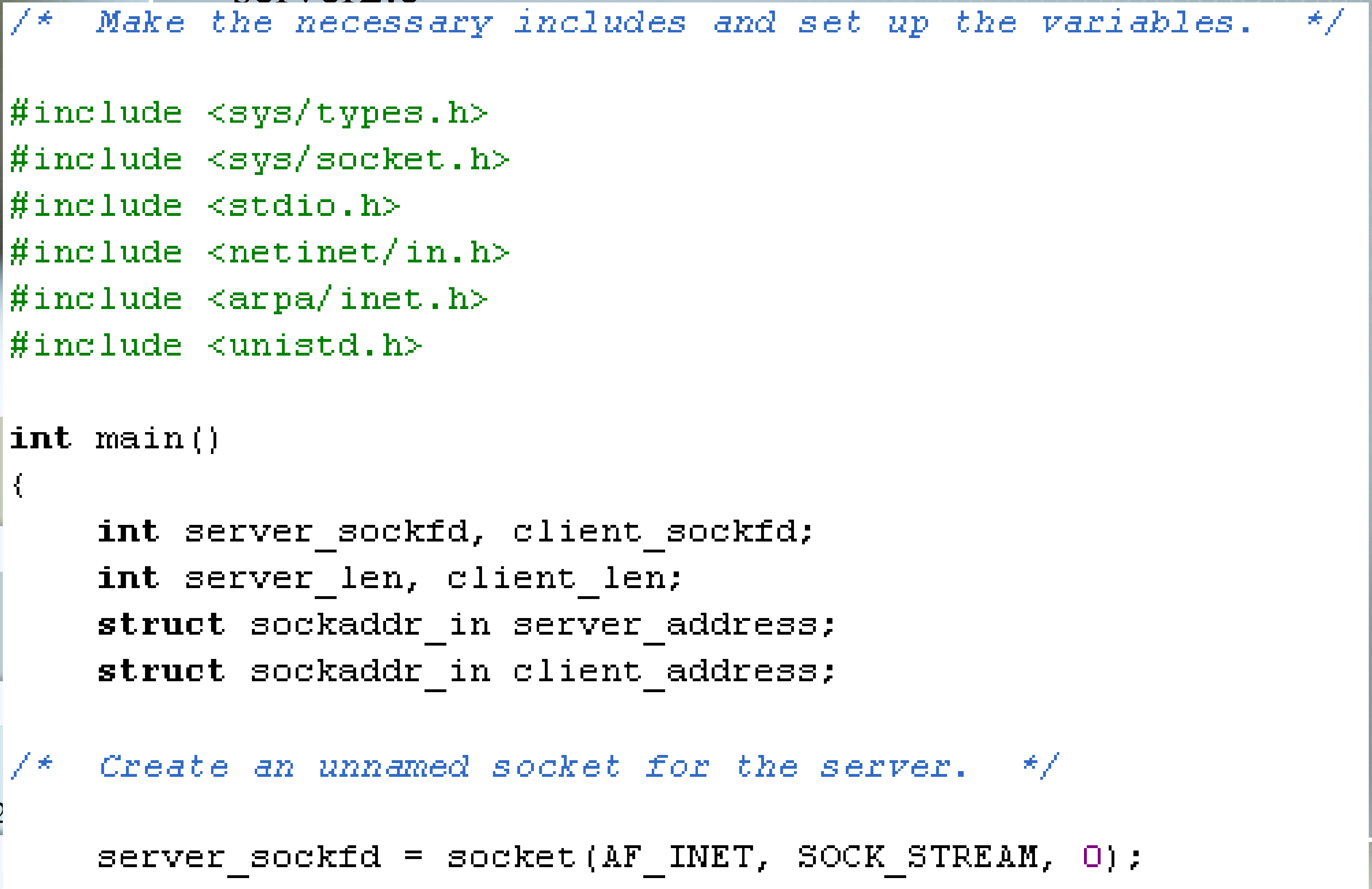
Socket Example 2 – Use AF_INET (2/5)

- client2.c

```
/*  Name the socket, as agreed with the server.  */

    address.sin_family = AF_INET;
    address.sin_addr.s_addr = inet_addr("127.0.0.1");
    address.sin_port = 9734;
    len = sizeof(address);

/*  Now connect our socket to the server's socket.  */

    result = connect(sockfd, (struct sockaddr *)&address, len);

    if(result == -1) {
        perror("oops: client2");
        exit(1);
    }

/*  We can now read/write via sockfd.  */

    write(sockfd, &ch, 1);
    read(sockfd, &ch, 1);
    printf("char from server = %c\n", ch);
    close(sockfd);
    exit(0);
}
```

Socket Example 2 – Use AF_INET (3/5)

- server2.c



```
/*  Make the necessary includes and set up the variables.  */

#include <sys/types.h>
#include <sys/socket.h>
#include <stdio.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>

int main()
{
    int server_sockfd, client_sockfd;
    int server_len, client_len;
    struct sockaddr_in server_address;
    struct sockaddr_in client_address;

/*  Create an unnamed socket for the server.  */

    server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
```

HPDS

Socket Example 2 – Use AF_INET (4/5)

- server2.c

```
/*  Name the socket.  */

    server_address.sin_family = AF_INET;
    server_address.sin_addr.s_addr = inet_addr("127.0.0.1");
    server_address.sin_port = 9734;
    server_len = sizeof(server_address);
    bind(server_sockfd, (struct sockaddr *)&server_address, server_len);

/*  Create a connection queue and wait for clients.  */

    listen(server_sockfd, 5);
    while(1) {
        char ch;

        printf("server waiting\n");
```

HPDS

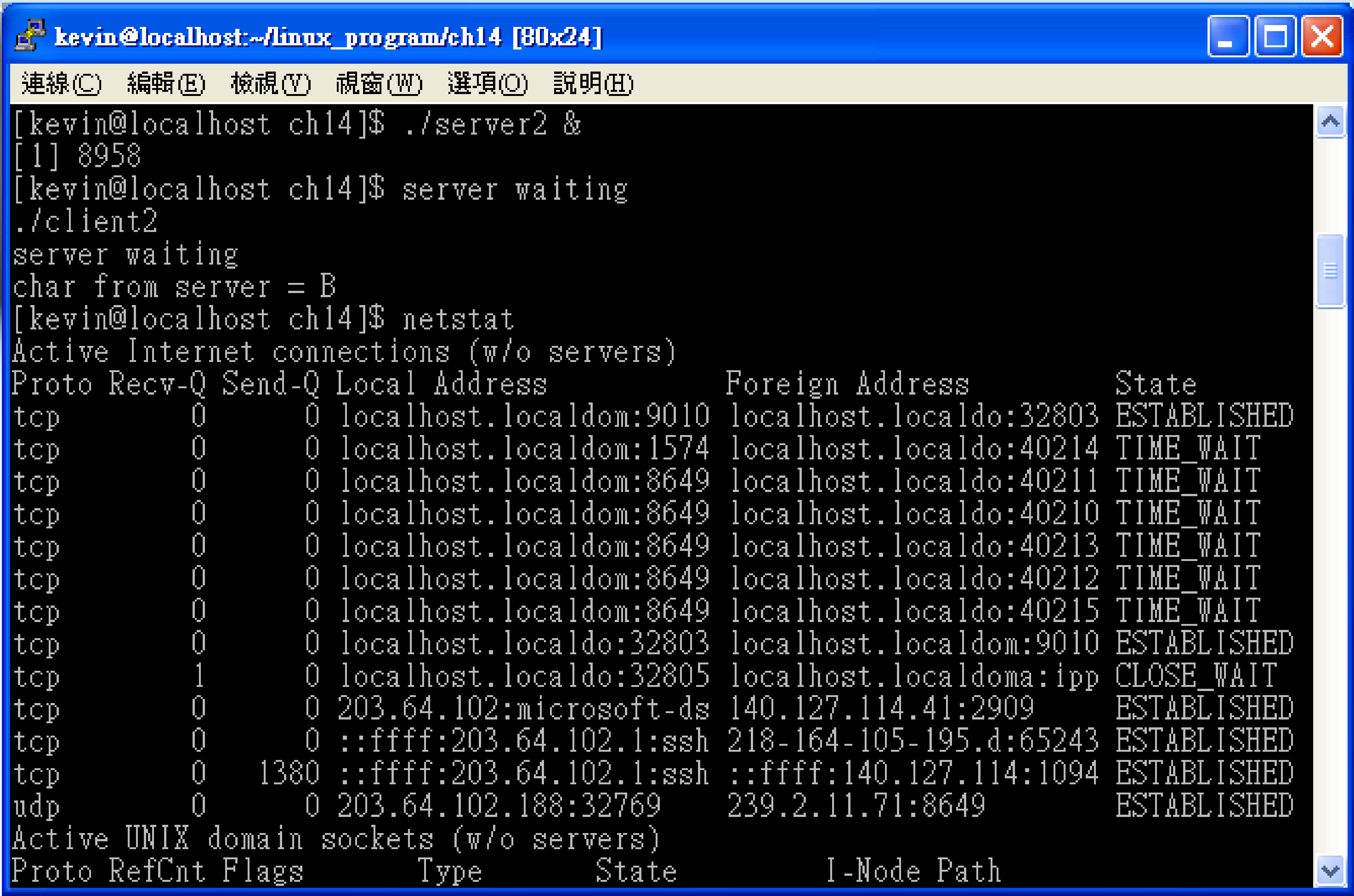
Socket Example 2 – Use AF_INET (5/5)

- server2.c

```
/*  Accept a connection.  */

        client_len = sizeof(client_address);
        client_sockfd = accept(server_sockfd,
            (struct sockaddr *)&client_address, &client_len);

/*  We can now read/write to client on client_sockfd.  */

        read(client_sockfd, &ch, 1);
        ch++;
        write(client_sockfd, &ch, 1);
        close(client_sockfd);
    }
}
```

HPDS

網路位元組排序 (1/4)


kevin@localhost:~/linux_program/ch14 [80x24]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)

```
[kevin@localhost ch14]$ ./server2 &
[1] 8958
[kevin@localhost ch14]$ server waiting
./client2
server waiting
char from server = B
[kevin@localhost ch14]$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 localhost.localdom:9010 localhost.localdo:32803 ESTABLISHED
tcp        0      0 localhost.localdom:1574 localhost.localdo:40214 TIME_WAIT
tcp        0      0 localhost.localdom:8649 localhost.localdo:40211 TIME_WAIT
tcp        0      0 localhost.localdom:8649 localhost.localdo:40210 TIME_WAIT
tcp        0      0 localhost.localdom:8649 localhost.localdo:40213 TIME_WAIT
tcp        0      0 localhost.localdom:8649 localhost.localdo:40212 TIME_WAIT
tcp        0      0 localhost.localdom:8649 localhost.localdo:40215 TIME_WAIT
tcp        0      0 localhost.localdo:32803 localhost.localdom:9010 ESTABLISHED
tcp        1      0 localhost.localdo:32805 localhost.localdoma:ipp CLOSE_WAIT
tcp        0      0 203.64.102:microsoft-ds 140.127.114.41:2909     ESTABLISHED
tcp        0      0 ::ffff:203.64.102.1:ssh 218-164-105-195.d:65243 ESTABLISHED
tcp        0   1380 ::ffff:203.64.102.1:ssh ::ffff:140.127.114:1094 ESTABLISHED
udp        0      0 203.64.102.188:32769    239.2.11.71:8649        ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node Path
```

網路位元組排序 (2/4)

- #include <netinet/in.h>
- unsigned long int htonl(unsigned long int *hostlong*);
- unsigned short int htons(unsigned short int *hostshort*);
- unsigned long int ntohl(unsigned long int *hostlong*);
- unsigned short int ntohs(unsigned short int *hostshort*);

網路位元組排序 (3/4)

- server3.c:

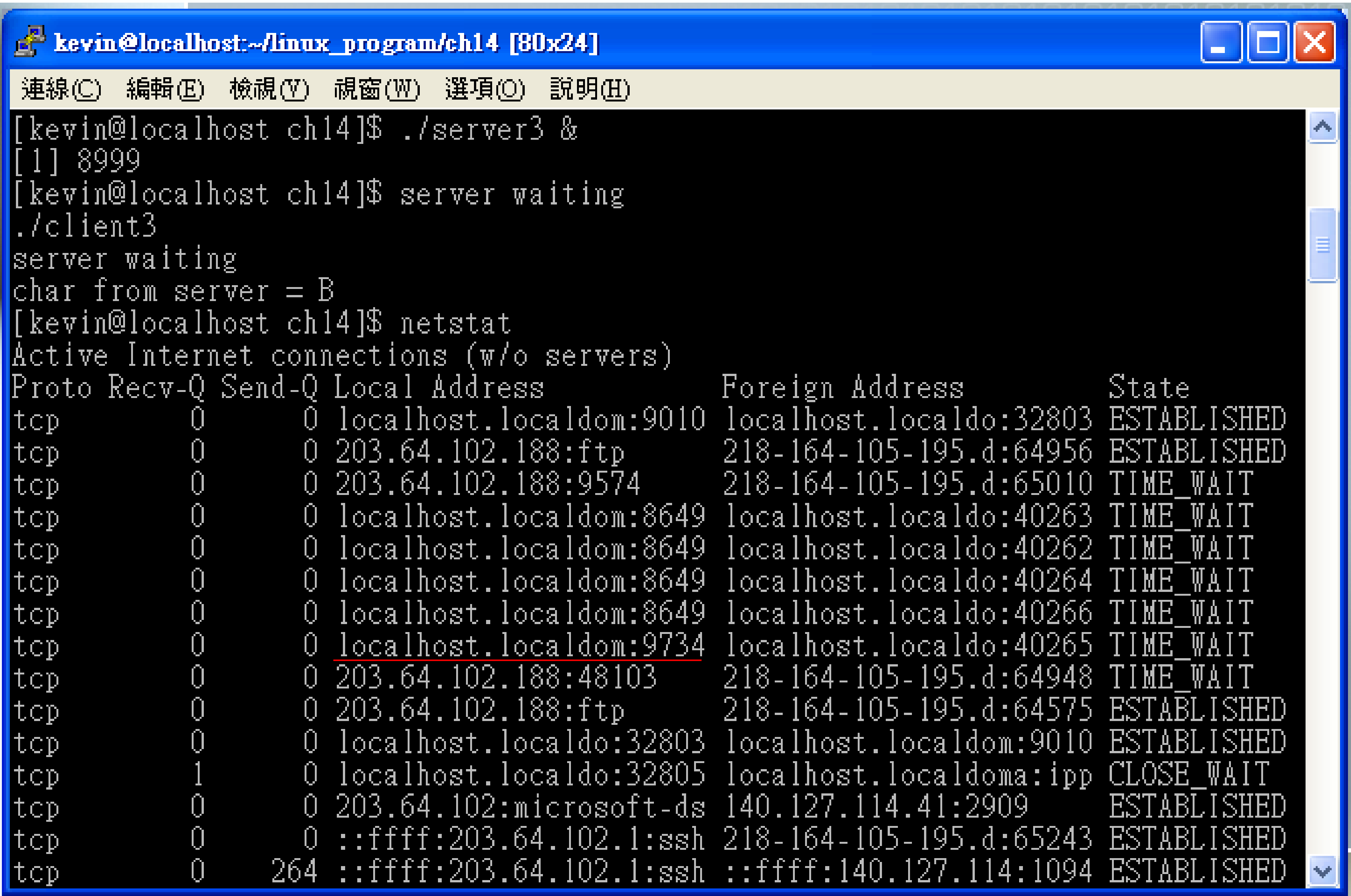
```
server_address.sin_addr.s_addr = htonl(INADDR_ANY);
server_address.sin_port = htons(9734);
```

- client3.c:

```
address.sin_port = htons(9734);
```

HPDS

網路位元組排序 (4/4)


kevin@localhost:~/linux_program/ch14 [80x24]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)

```
[kevin@localhost ch14]$ ./server3 &
[1] 8999
[kevin@localhost ch14]$ server waiting
./client3
server waiting
char from server = B
[kevin@localhost ch14]$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 localhost.localdom:9010     localhost.localdo:32803     ESTABLISHED
tcp        0      0 203.64.102.188:ftp          218-164-105-195.d:64956     ESTABLISHED
tcp        0      0 203.64.102.188:9574         218-164-105-195.d:65010     TIME_WAIT
tcp        0      0 localhost.localdom:8649     localhost.localdo:40263     TIME_WAIT
tcp        0      0 localhost.localdom:8649     localhost.localdo:40262     TIME_WAIT
tcp        0      0 localhost.localdom:8649     localhost.localdo:40264     TIME_WAIT
tcp        0      0 localhost.localdom:8649     localhost.localdo:40266     TIME_WAIT
tcp        0      0 localhost.localdom:9734     localhost.localdo:40265     TIME_WAIT
tcp        0      0 203.64.102.188:48103        218-164-105-195.d:64948     TIME_WAIT
tcp        0      0 203.64.102.188:ftp          218-164-105-195.d:64575     ESTABLISHED
tcp        0      0 localhost.localdo:32803     localhost.localdom:9010     ESTABLISHED
tcp        1      0 localhost.localdo:32805     localhost.localdoma:ipp     CLOSE_WAIT
tcp        0      0 203.64.102:microsoft-ds     140.127.114.41:2909         ESTABLISHED
tcp        0      0 ::ffff:203.64.102.1:ssh     218-164-105-195.d:65243     ESTABLISHED
tcp        0    264 ::ffff:203.64.102.1:ssh     ::ffff:140.127.114:1094     ESTABLISHED
```