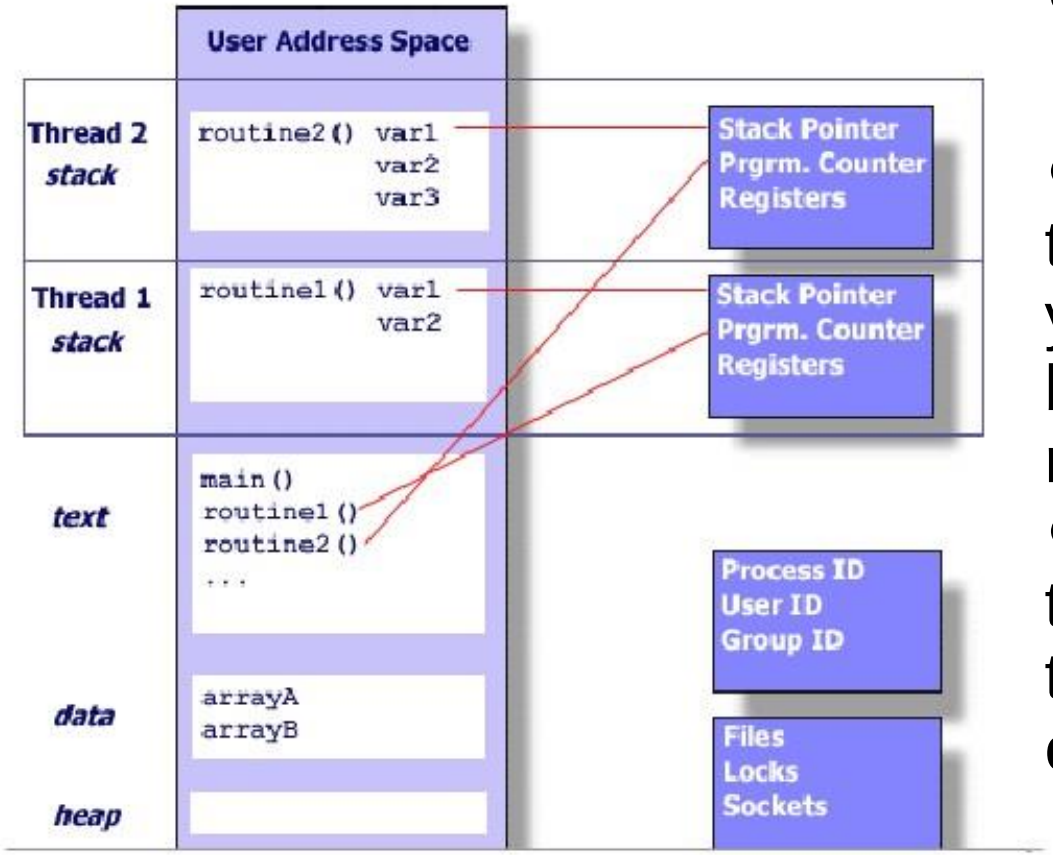
Project hy335b
Computer Networks
Threads examples
by Alexmil
8/3/2017

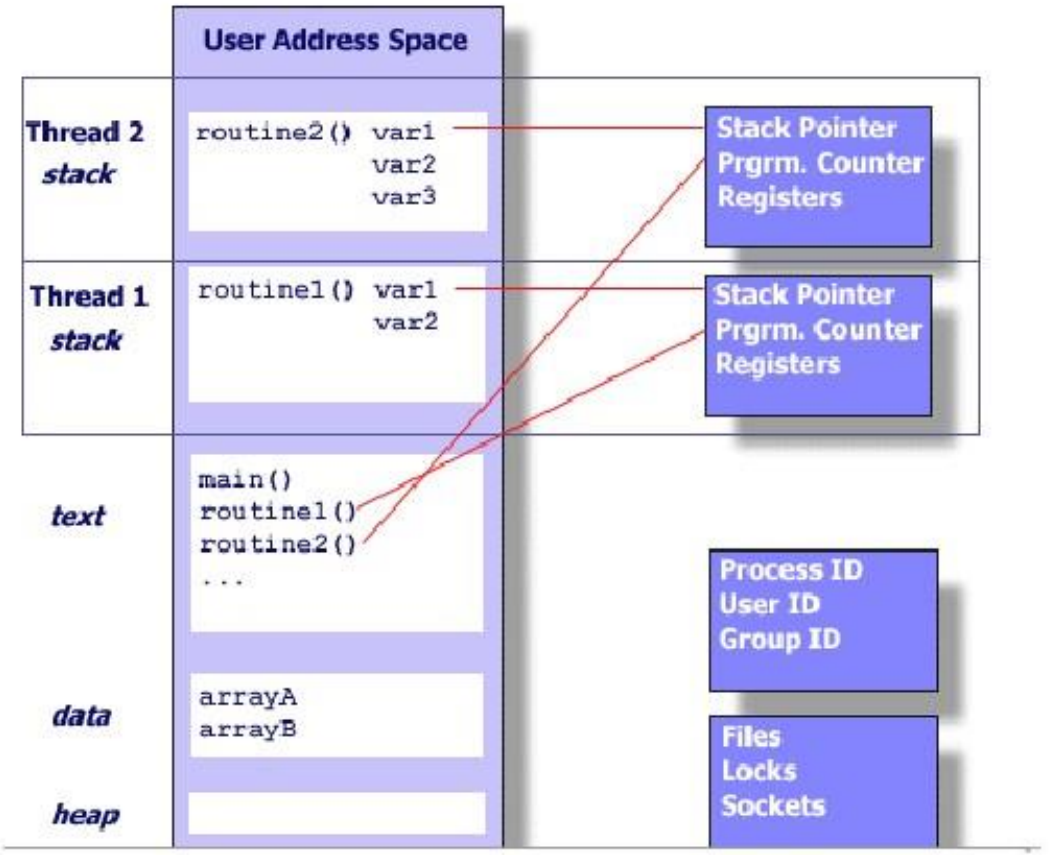
What is a thread(1/3)

- A "procedure" that runs independently from its main program



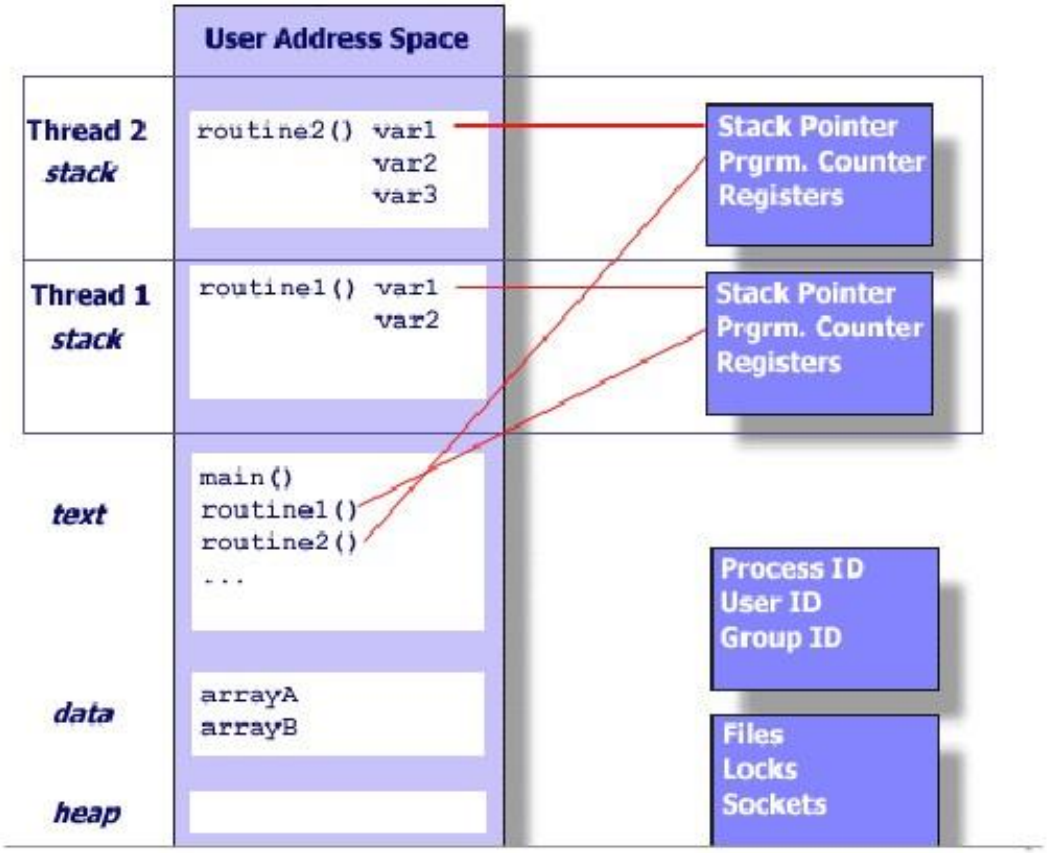
- Can be considered as lightweight processes
- Threads use and exist within these process resources, yet are able to be scheduled by the operating system and run as independent entities
- They duplicate only the bare essential resources that enable them to exist as executable code.

What is a thread(2/3)



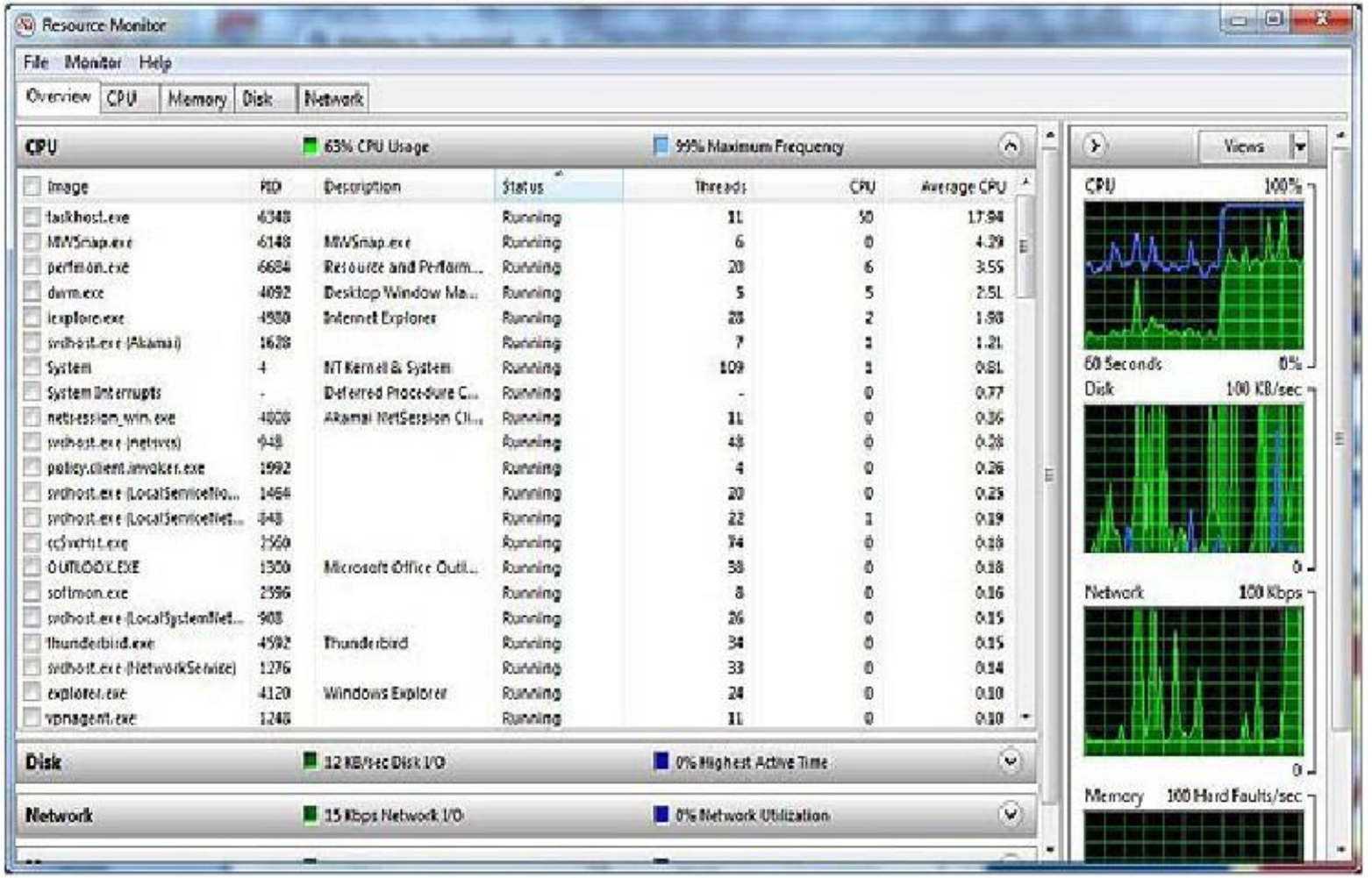
- a thread maintains its own:
  - Stack pointer
  - Registers
  - Scheduling properties (such as policy or priority)
  - Set of pending and blocked signals
  - Thread specific data.

What is a thread(3/3)

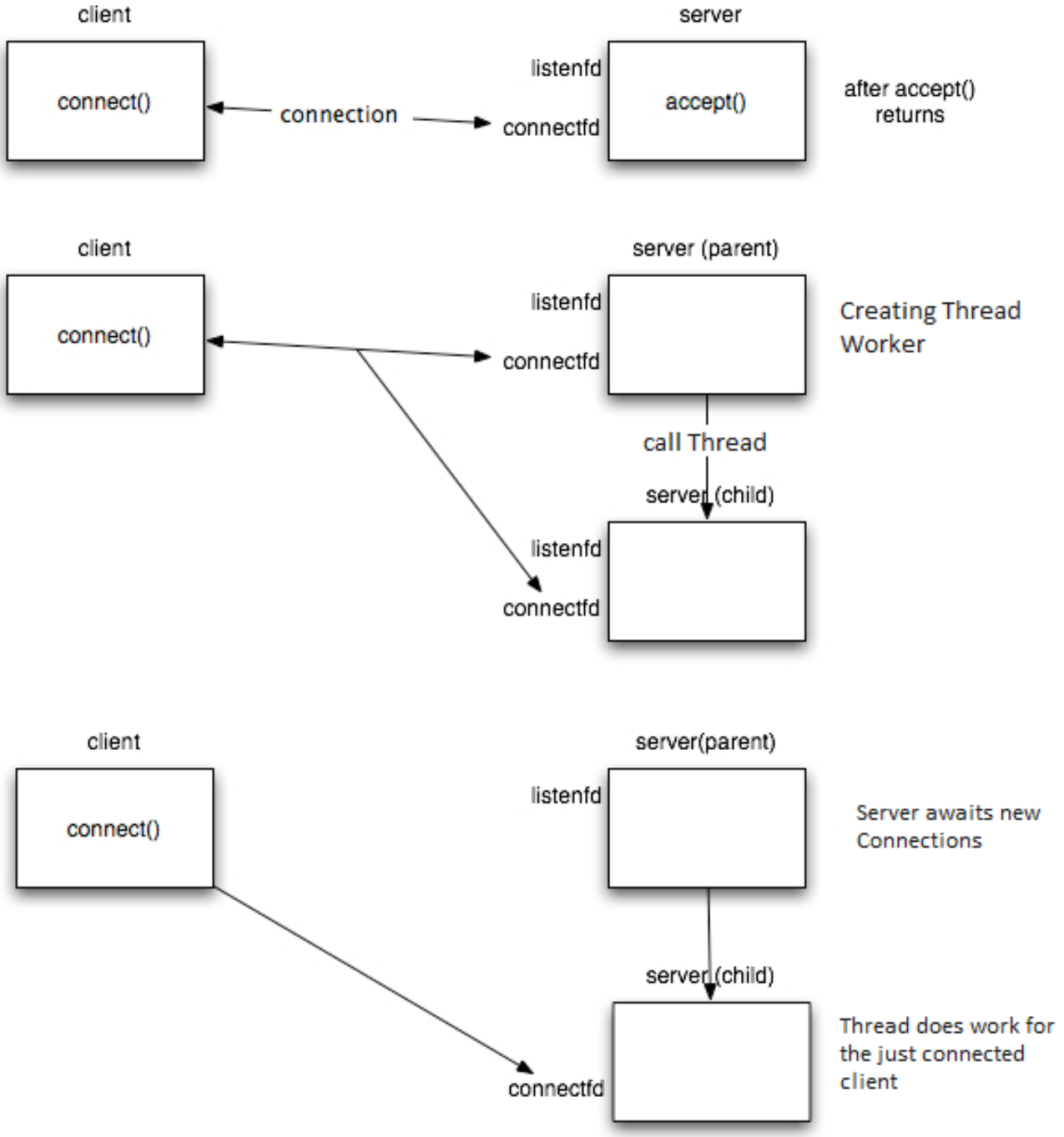


- in summary, in the UNIX environment a thread:
  - Exists within a process and uses the process resources
  - Has its own independent flow of control as long as its parent process exists and the OS supports it
  - Duplicates only the essential resources it needs to be independently schedulable
  - May share the process resources with other threads that act equally independently (and dependently)
  - Dies if the parent process dies - or something similar
  - Is "lightweight" because most of the overhead has already been accomplished through the creation of its process.

MS windows using threads



Why use Client – Server use threads



Server Thread example.c

```
2  void *conn_handler(void *sock){
3      char  sendBuff[100], client_message[2000];
4      while((n=recv(sock,client_message,2000,0))>0){
5          send(sock,client_message,n,0);
6      }
7  }
```

```
9      int listenfd;
10     socklen_t clilen;
11     struct sockaddr_in cliaddr;
12     
13     clilen = sizeof(cliaddr);
14     connfd = accept (listenfd, (struct sockaddr *) &cliaddr, &clilen);
15     
16     if( pthread_create( &thread , NULL , conn_handler,(void*) connfd) <0){
17         perror("could not create thread");
18         return 0;
19     
```

- Due to Recv been a blocking function in order to keep the server functioning and accepting new connections an independent thread is created to serve the client which just connected.

- More thread examples and various problems on the .c and python files. Look both.

Server thread python example

```
2  while 1:
3      # accept connections from outside
4      ( clientsocket,address ) = serversocket.accept()
5      # now do something with the clientsocket
6      ct = thread(connection_handler)
7      ct.run()
8
9
10 def connection_handler:
11     .
12     .
13     .
```