OS 2017

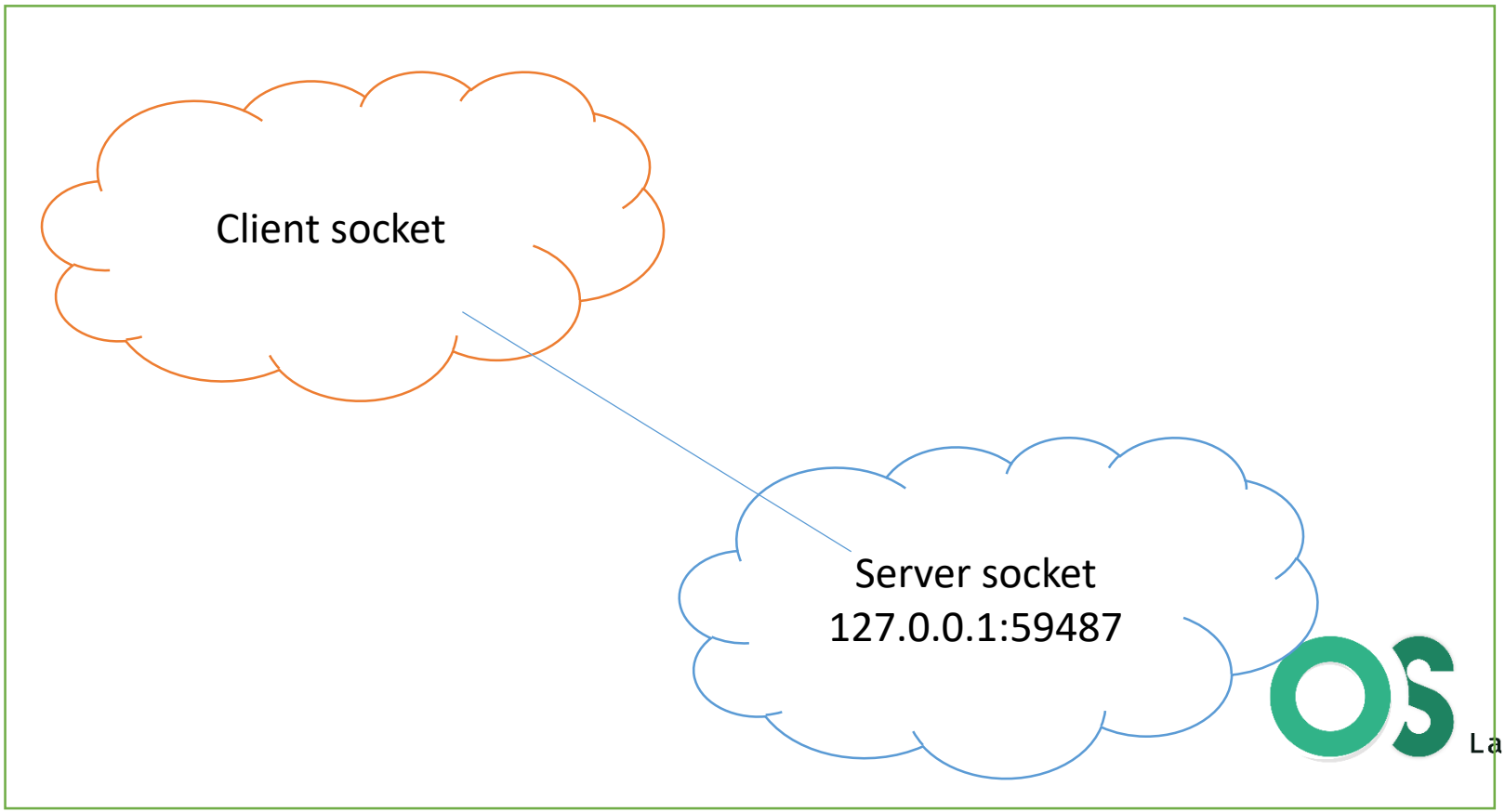
Homework1: machine monitor

(Due day 11/09 23:59:59)

Lab @NCKU

Socket communication example

- Concatenation of IP address and port
  - The socket 127.0.0.1:59487 refers to port 59487 on host 127.0.0.1
- The communication consists between a pairs of sockets.



Socket programming requirements:

- Write one server & one client
  - Client can ask process information like *pid* from server.
  - Server ***MUST*** read process information under /proc.
  - Please refer page 4 for the detail.
- Multi-threads server
  - Server should have a thread to listen connection request of clients.
  - When server receive connection request, server create one thread to handle the request. Then each thread do the task client requested.
  - Server can be connected by multi-clients at the same time
- Implement querying interface like page 4 or GUI.
- Query items include-
  - All processes id, process name, command line, process status, parent's PID, child's PID, thread's IDs, all ancients of PIDs, virtual memory size, physical memory size, exit.
- Coding style-(when push to server)
  - Follow astyle in Gitlab server.
  - Ex: astyle –style=linux –indent=tab –max-code-length=80 ${file}

Client example

```
miyavi:hw1$ ./client
============================================================
(a)list all process ids
(b)thread's IDs
(c)child's PIDs
(d)process name
(e)state of process(D,R,S,T,t,W,X,Z)
(f)command line of excuting process(cmdline)
(g)parent's PID
(h)all ancients of PIDs
(i)virtual memory size(VmSize)
(j)physical memory size(VmRSS)
(k)exit
which? b
pid? 1

[tid] 1

============================================================
(a)list all process ids
(b)thread's IDs
(c)child's PIDs
(d)process name
(e)state of process(D,R,S,T,t,W,X,Z)
(f)command line of excuting process(cmdline)
(g)parent's PID
(h)all ancients of PIDs
(i)virtual memory size(VmSize)
(j)physical memory size(VmRSS)
(k)exit
which? i
pid? 1

[physical memory] 3416 KB

============================================================
```

(a)No input. Just list all processes ids.
(b)~(j)Input is pid.

1 return:
b.d.e.f.g.i.j

multi return:
a.c.h

References

- Socket programming-
  - Linux Socket Tutorial
- Pthread-
  - Pthread tutorial
  - Thread pool
- Proc filesystem-
  - Proc introduction
- Coding style-
  - Astyle