Network Programming (Part 1)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - Network programming key concepts
  - Client/server computing
  - Client/server computing in COS 333
  - Network programming in Python
    - How to compose a client
    - How to compose a server

Agenda

- **Key concepts**
- Client/server computing
- Client/server computing in COS 333
- Network programming: daytime example
- Network programming: echo example

Key Concepts

- Network Address
  - ***Medium Access Control (MAC) address***
    - Example: 90:1b:0e:6a:32:26
  - ***Internet Protocol (IP) address***
    - Example: 128.112.136.61
    - Example: 127.0.0.1

What is the IP address of your computer?
Mac or Linux: `ifconfig`
MS Windows: `ipconfig`

Key Concepts

- Network address (cont.)
  - ***Domain name***
    - ***Domain Name System (DNS)*** converts to IP address
    - Example: cs.princeton.edu
      - Same as 128.112.136.61
    - Example: localhost
      - Same as 127.0.0.1

Key Concepts

- See **ipaddress.py**

```
$ python ipaddress.py cs.princeton.edu
Host name: cs.princeton.edu
IP address: 128.112.136.61
$
```

What is the IP address of the main Princeton domain name (princeton.edu)?

What is the IP address of localhost?

Key Concepts

- ***Port***
  - A software abstraction
  - 16-bit integer (0 - 65535)

Key Concepts

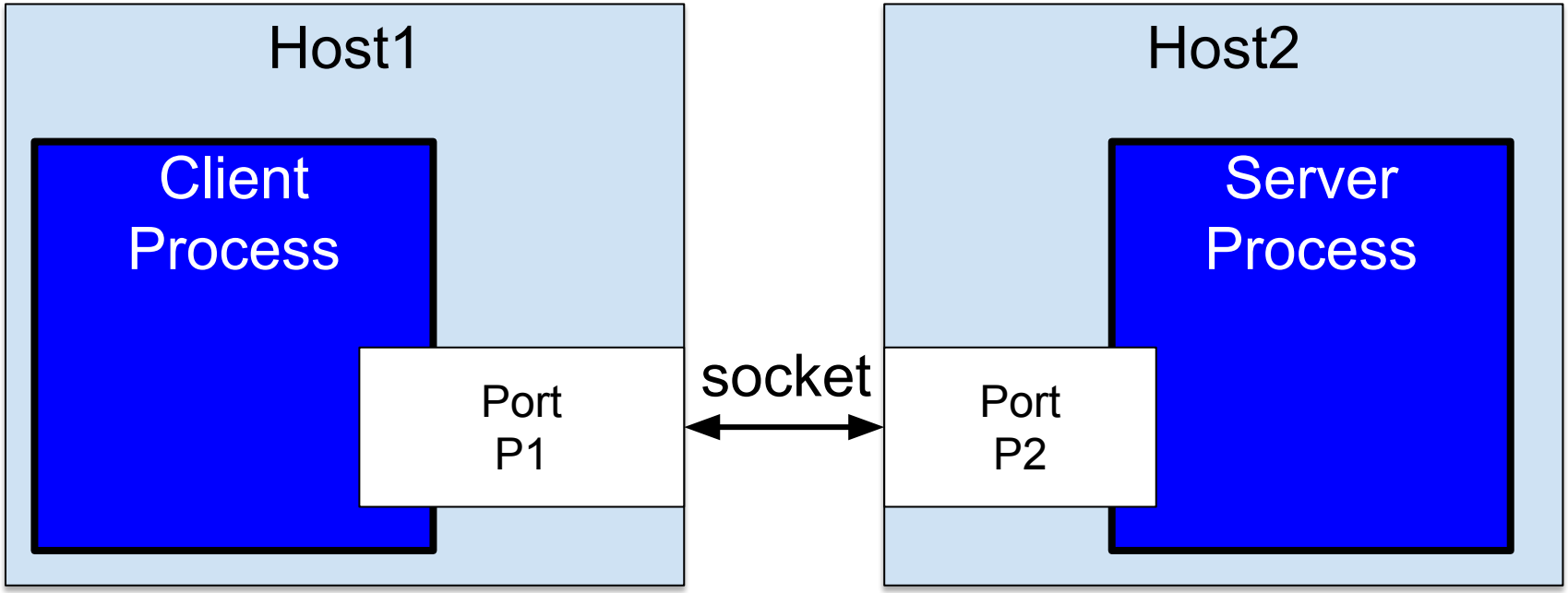
- ***Socket***
  - IP address + port
  - Used to implement…

Agenda

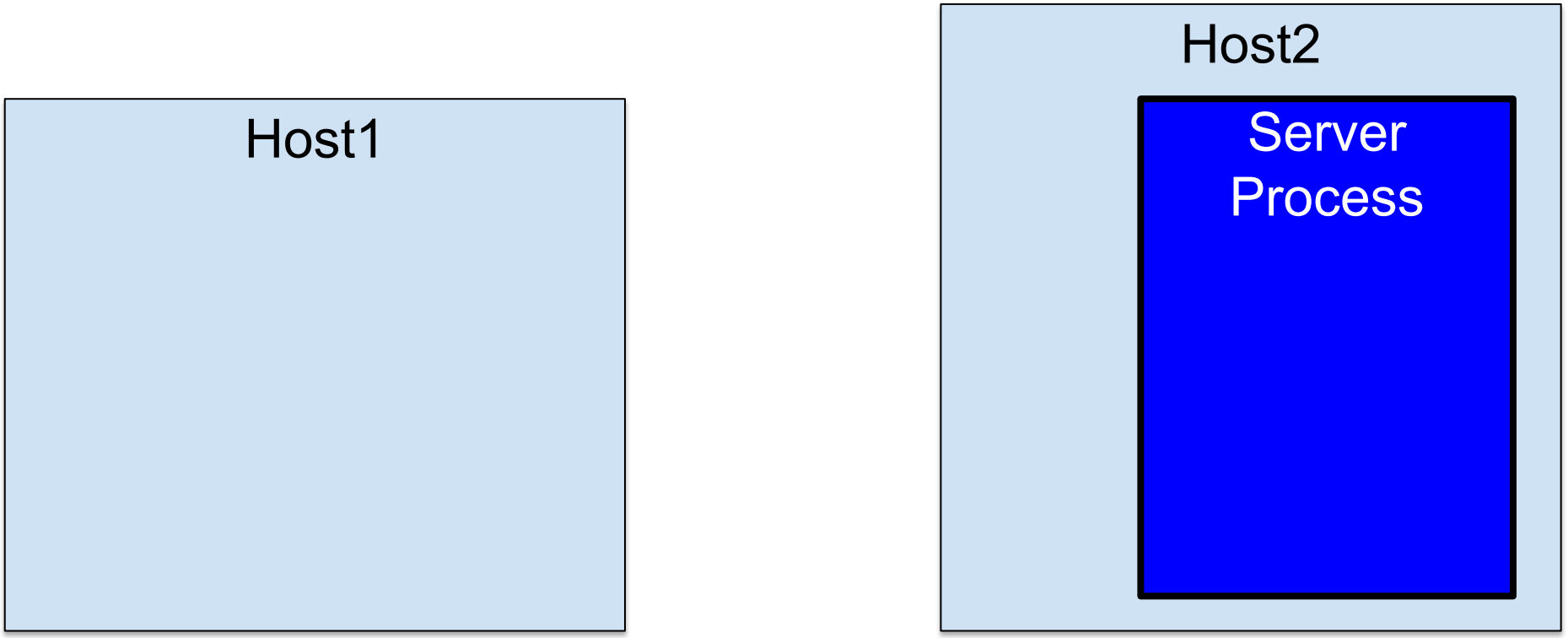
- Key concepts
- **Client/server computing**
- Client/server computing in COS 333
- Network programming: daytime example
- Network programming: echo example

Client/Server Computing

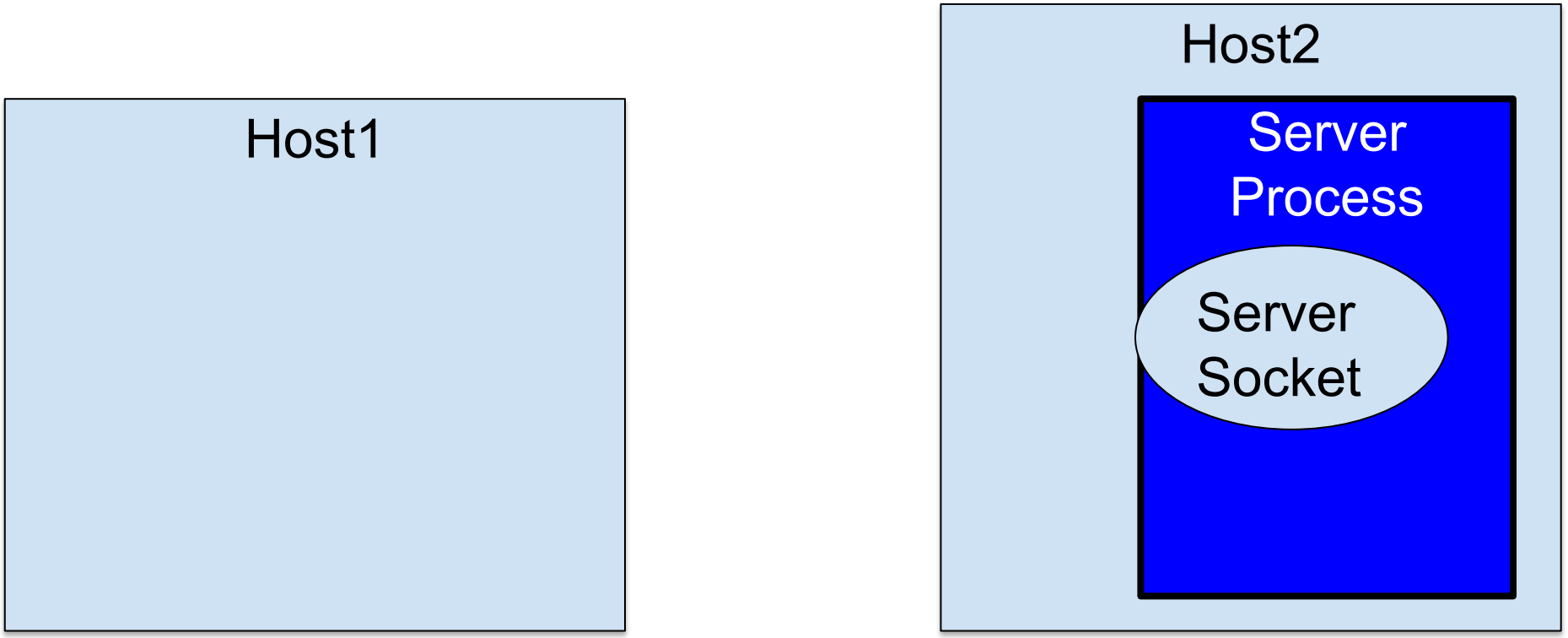
The big picture



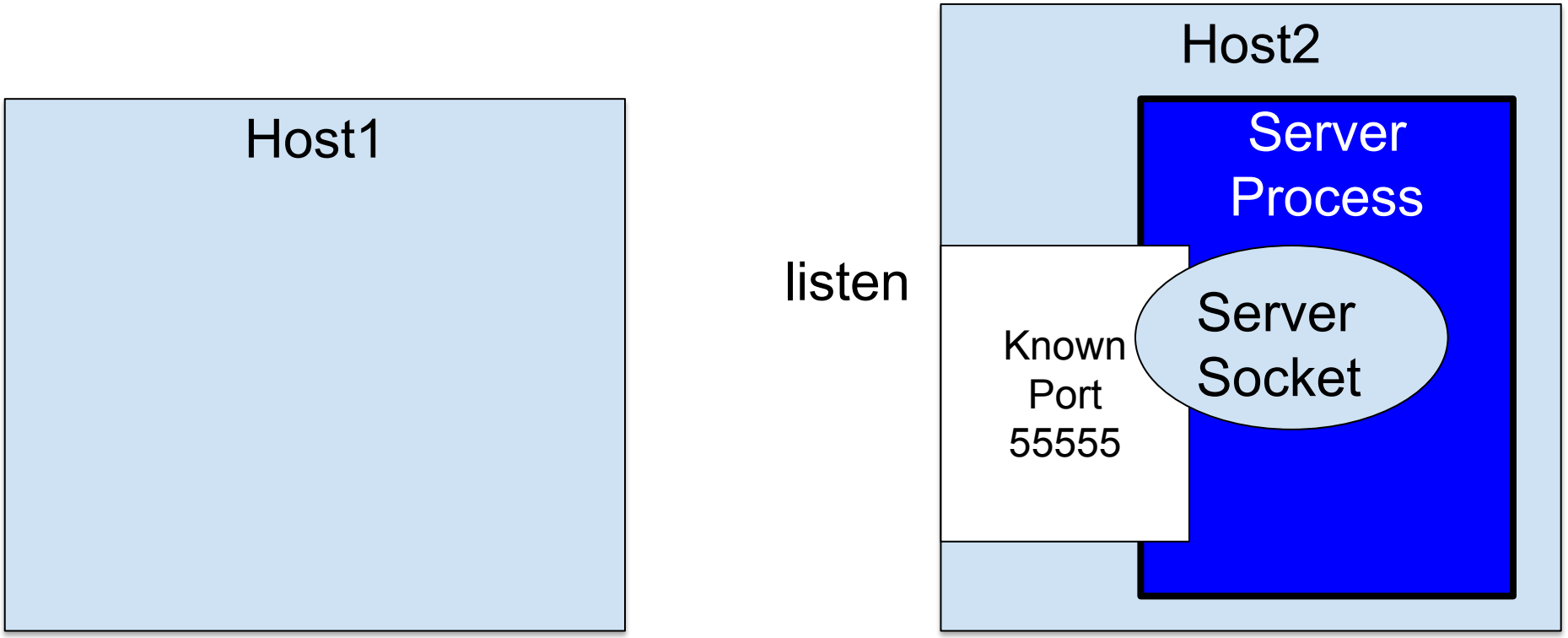
Client/Server Computing



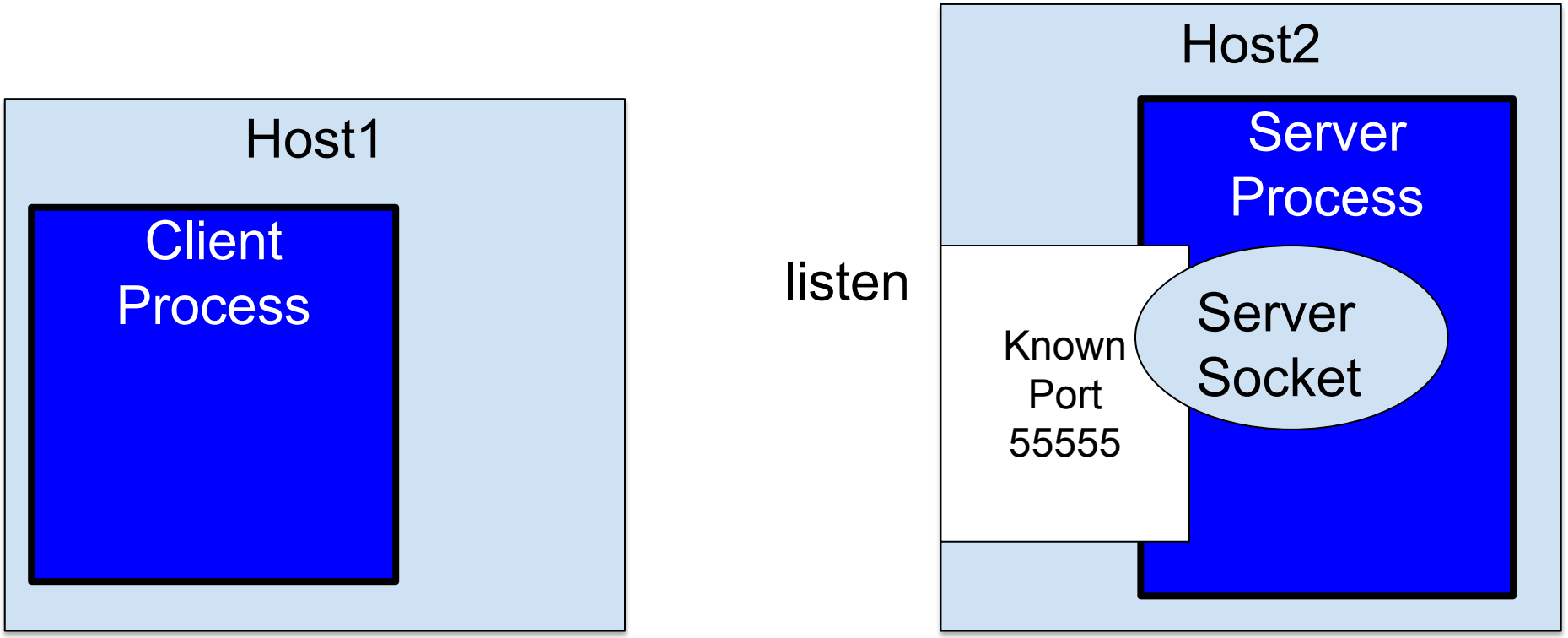
Client/Server Computing



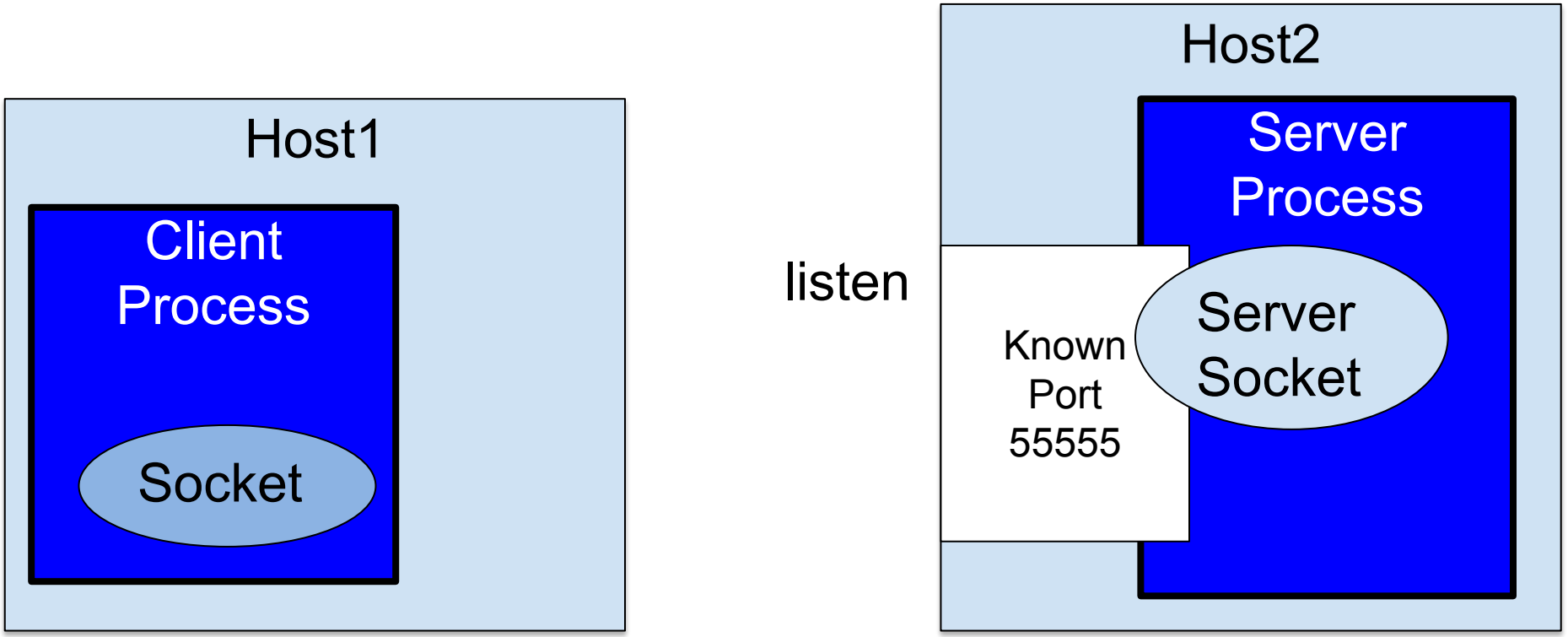
Client/Server Computing



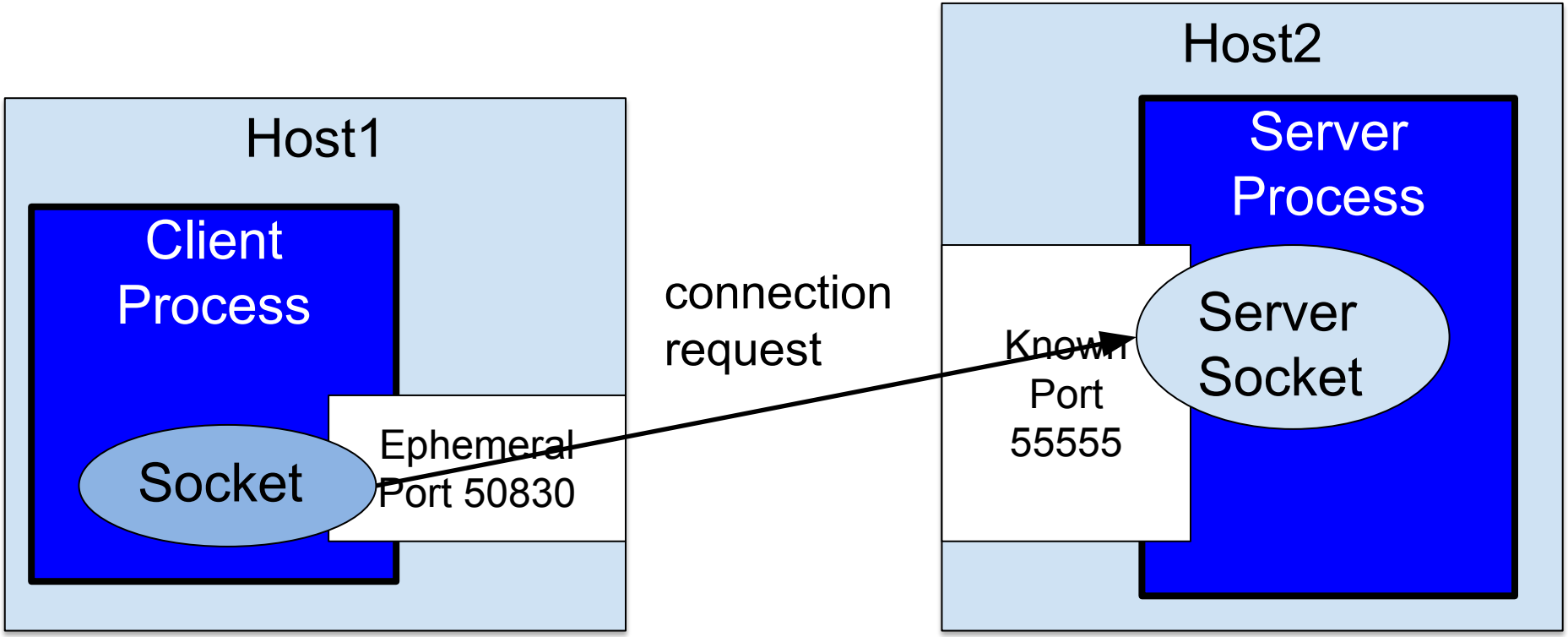
Client/Server Computing



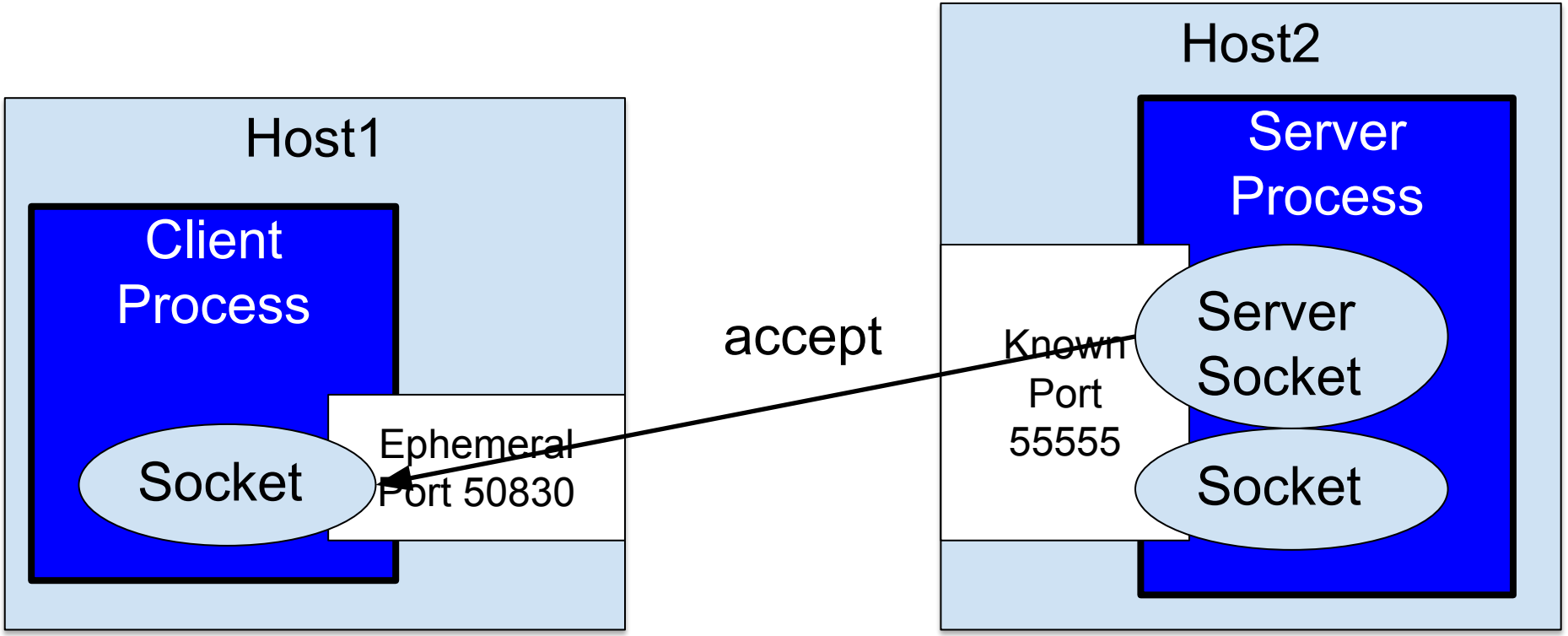
Client/Server Computing



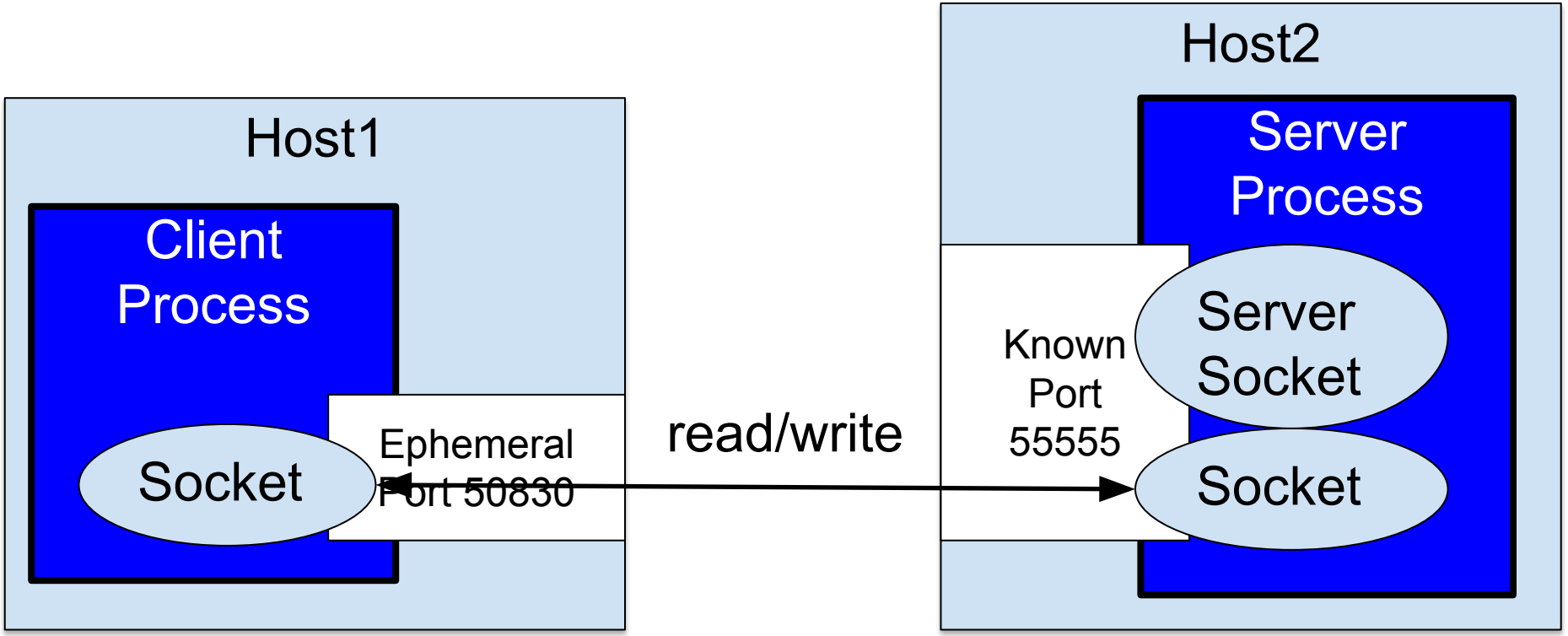
Client/Server Computing



Client/Server Computing



Client/Server Computing

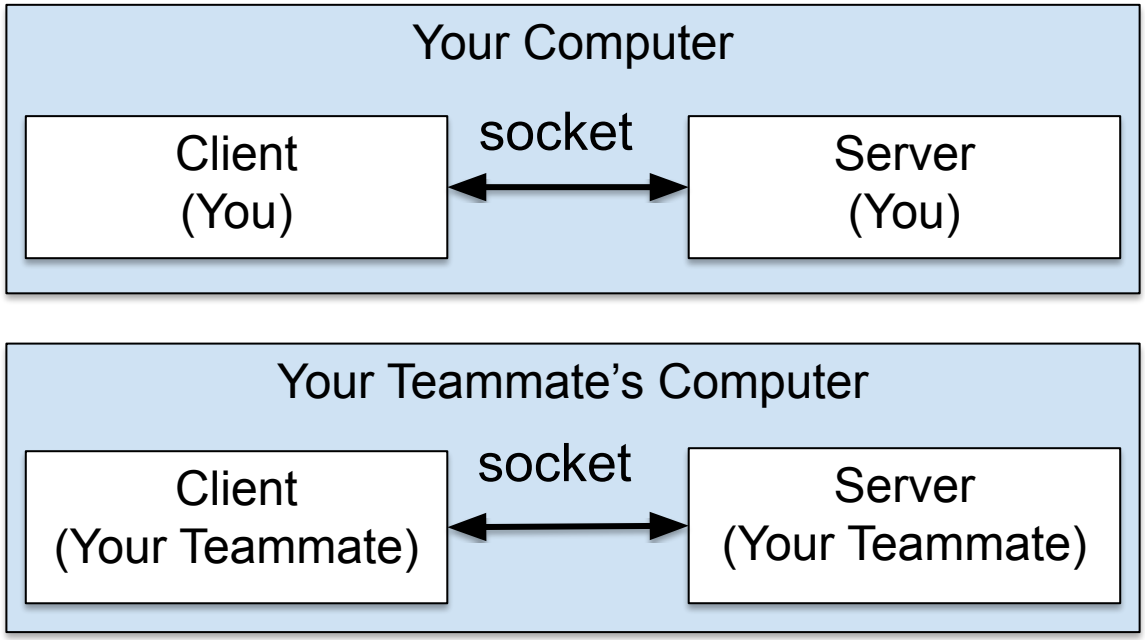


Agenda

- Key concepts
- Client/server computing
- **Client/server computing in COS 333**
- Network programming: daytime example
- Network programming: echo example

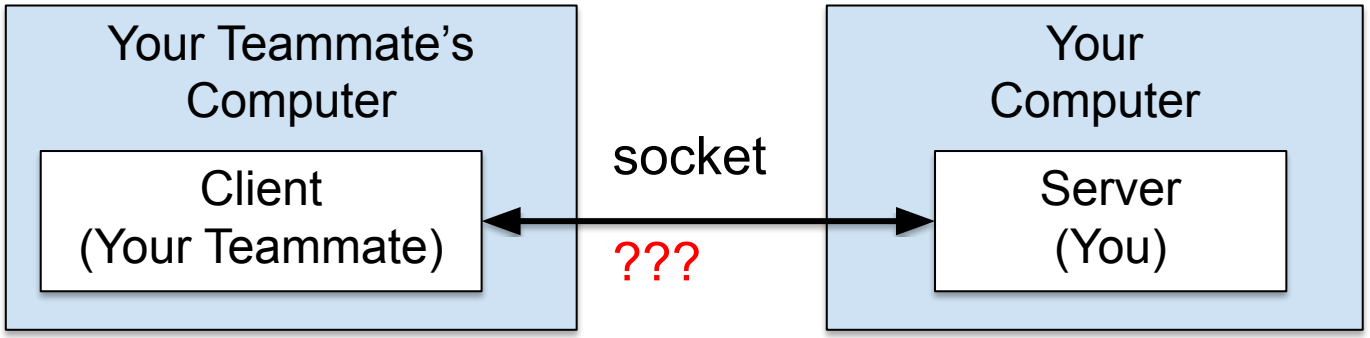
Client/Server in COS 333

Option 1: Run server on local computer
Run client on same local computer



Client/Server in COS 333

Option 2: Run server on local computer
Run client on different local computer



To determine IP address of your computer:

Mac/Linux: `ifconfig`

MS Windows: `ipconfig`

Won't work if either computer is not on Eduroam

Client/Server in COS 333

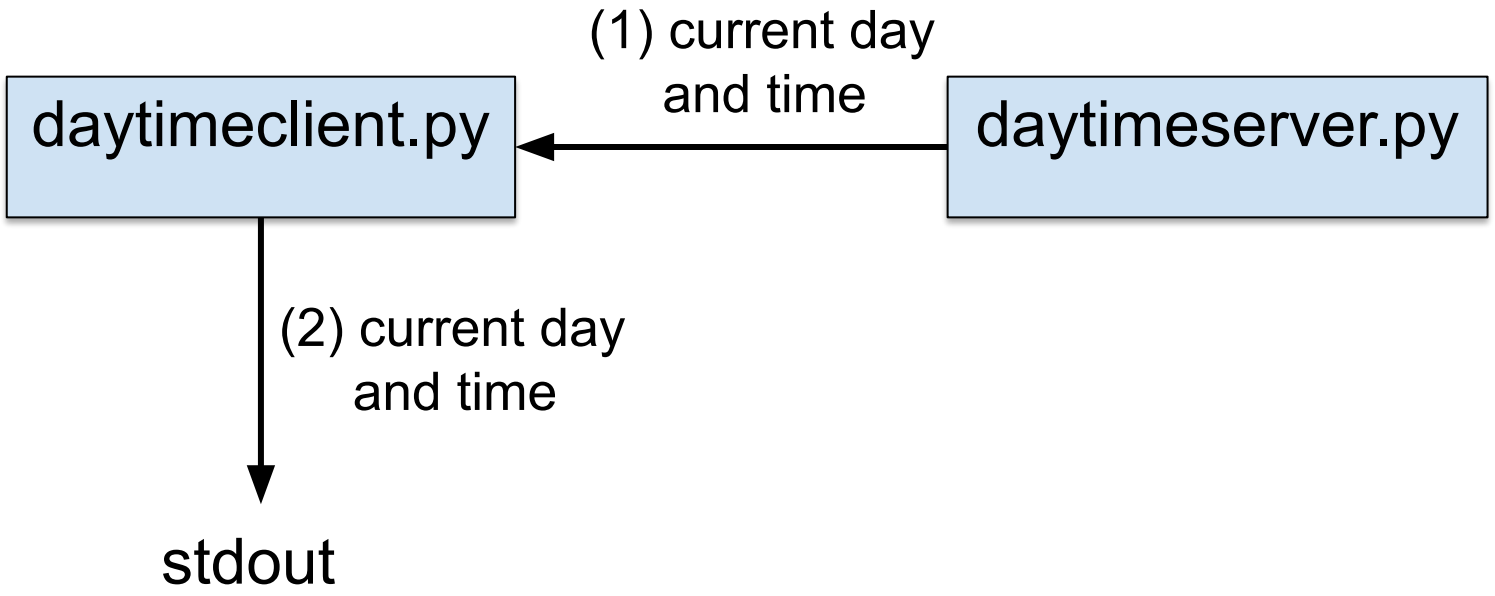
- Suggestions:
  - Use **option 1** during development
  - Use **option 2** to test network comm
    - Working alone =>
      - Use your computer and a COS 333 instructor's computer during office hours?

Agenda

- Key concepts
- Client/server computing
- Client/server computing in COS 333
- **Network programming: daytime example**
- Network programming: echo example

Network Programming: daytime

See **daytime** app



Network Programming: daytime

- See **daytime** app (cont.)

Server: On host 192.168.1.8

```
$ python daytimeserver.py 55555                          (1)
Opened server socket                                     (1)
Bound server socket to port                              (1)
Listening                                                (1)
Accepted connection                                      (3)
Opened socket                                            (3)
Server IP addr and port: ('192.168.1.8', 55555)          (3)
Client IP addr and port: ('192.168.1.8', 50252)          (3)
```

Client

```
$ python daytimeclient.py 192.168.1.8 55555 (2)
Sun Feb 13 14:47:15 2022                    (4)
$
```

Can you get the app to work on your computer? On your computer and a neighbor's computer?

Network Programming: daytime

- See **daytime** app (cont.)

Server: On host time-a.nist.gov at port 13

Client

```
$ python daytimeclient.py time-a.nist.gov 13      (1)
                                                  (2)
59622 22-02-12 19:34:35 00 0 0 635.1 UTC(NIST) *  (2)
$
```

Network Programming: daytime

- See **daytime** app (cont.)
  - Code structure

Baseline:

```
sock = socket(…)
…
…
sock.close()
```

What's wrong with the baseline version?

Better:

```
sock = socket(…)
try:
    …
    …
finally:
    sock.close()
```

Better still:

```
with socket(…) as sock:
    …
    …
```

Network Programming: daytime

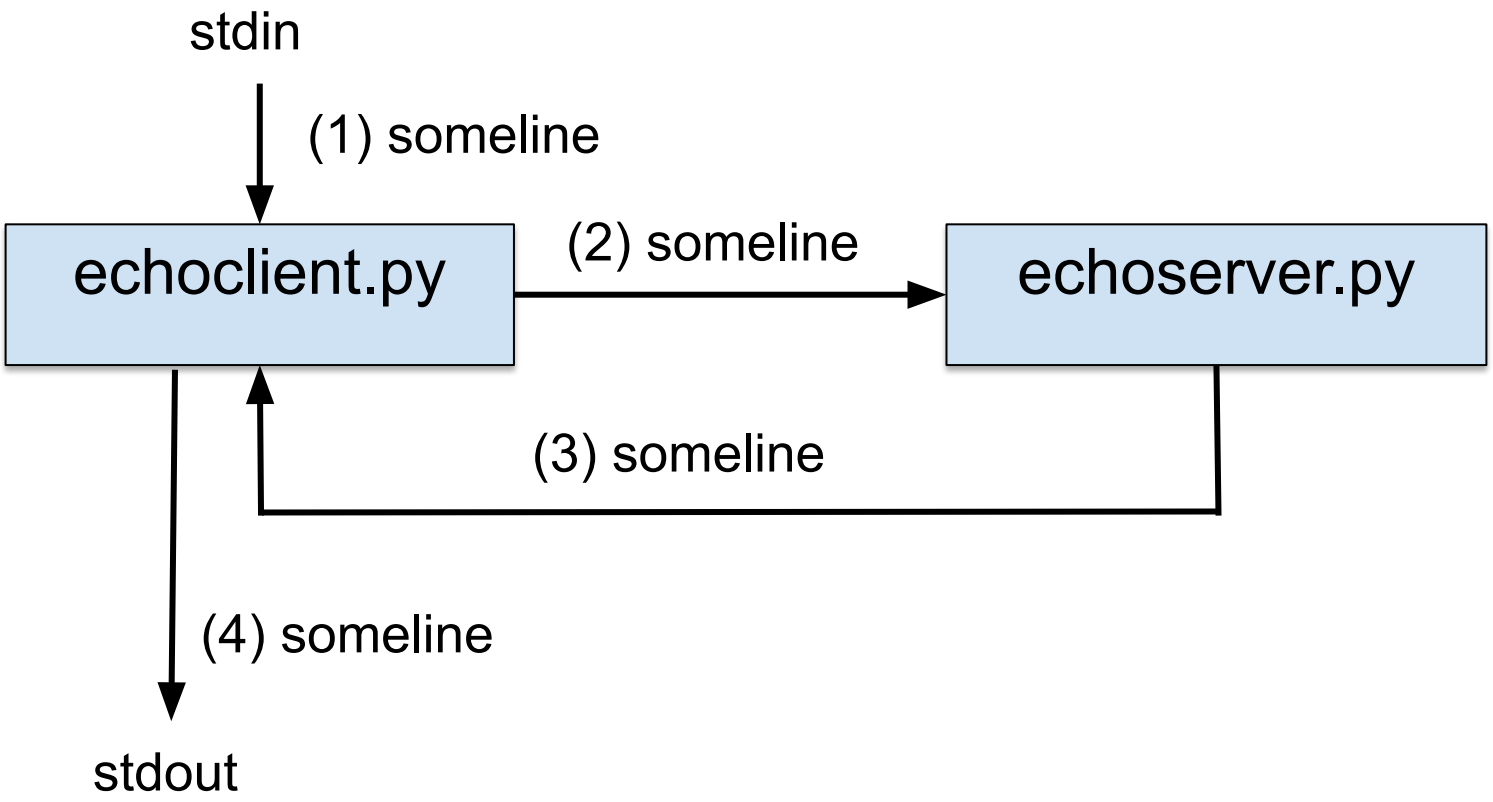
- See **daytime** app (cont.)
  - **daytimeclient.py**
  - **daytimeserver.py**

Agenda

- Key concepts
- Client/server computing
- Client/server computing in COS 333
- Network programming: daytime example
- **Network programming: echo example**

Network Programming: echo

See **echo** app



Network Programming: echo

- See **echo** app (cont.)

Server: On host 192.168.1.8

```
$ python echoserver.py 55555                                  (1)
Opened server socket                                          (1)
Bound server socket to port                                   (1)
Listening                                                     (1)
Accepted connection                                           (3)
Opened socket                                                 (3)
Server IP addr and port: ('192.168.1.8', 55555)               (3)
Client IP addr and port: ('192.168.1.8', 50851)               (3)
Read from client: Hello, COS 333.                             (3)
Wrote to client: Hello, COS 333.                              (3)
```

Client

```
$ python echoclient.py 192.168.1.8 55555 (2)
Hello, COS 333.                          (2)
Hello, COS 333.                          (4)
$
```

Network Programming: echo

- See **echo** app (cont.)
  - **echoserver.py**
  - **echoclient.py**

Network Programming: echo

- See **echo** app (cont.)
  - **echoserver.py** works with:
    - echoclient.py
    - An echo client written in Java, C, …
  - **echoclient.py** works with:
    - echoserver.py
    - An echo server written in Java, C, …

Lecture Summary

- This lecture covered:
  - Network programming key concepts
  - Client/server programming
  - Client/server programming in COS 333
  - Network programming in Python
    - How to compose a client
    - How to compose a server