Through The Clouds:
An Introduction to Cloud Computing

Day 2: Projects

Instructors: Anurag Khandelwal, Ramla Ijaz, Garrett Sager
TA: Joaquin Soto

Yale

Today’s Agenda

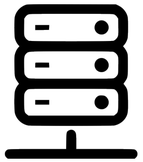
Today’s Agenda

- Networking Basics
- Getting our hands dirty!
  - Simple Ping-Pong Applications
  - [If time permits] Client Registry

Any Lingering Questions from Last Time? :)

Networking Basics

Building your own cloud service

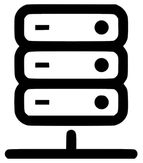


Server

Client

Building your own cloud service

Simple chat service



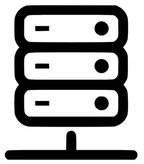
Server

Client

Building your own cloud service

Simple chat service

Chat Server



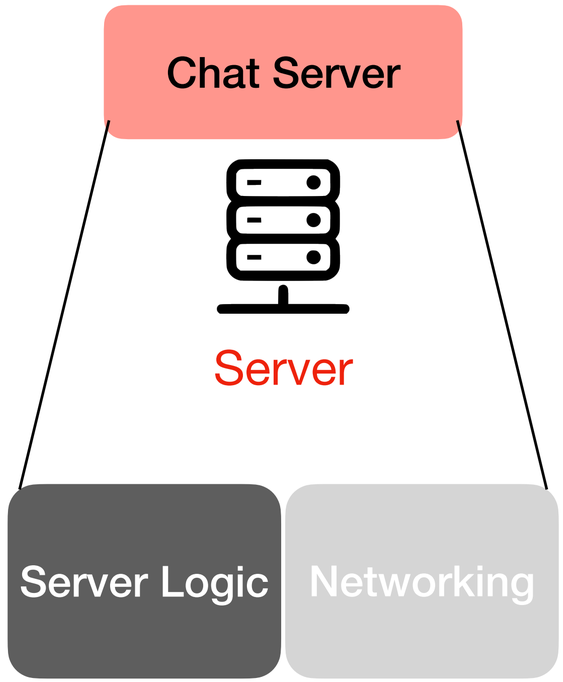
Server

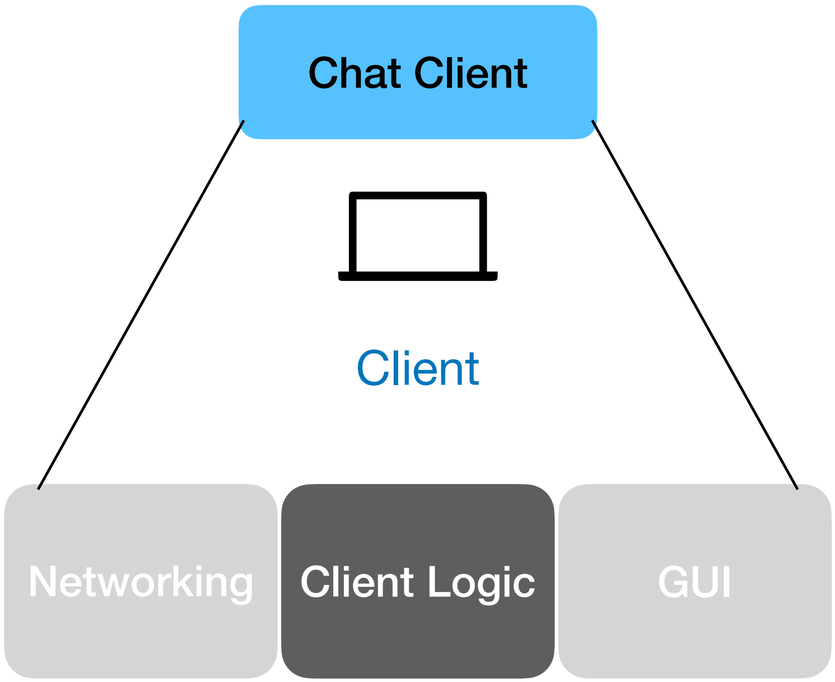
Chat Client

Client

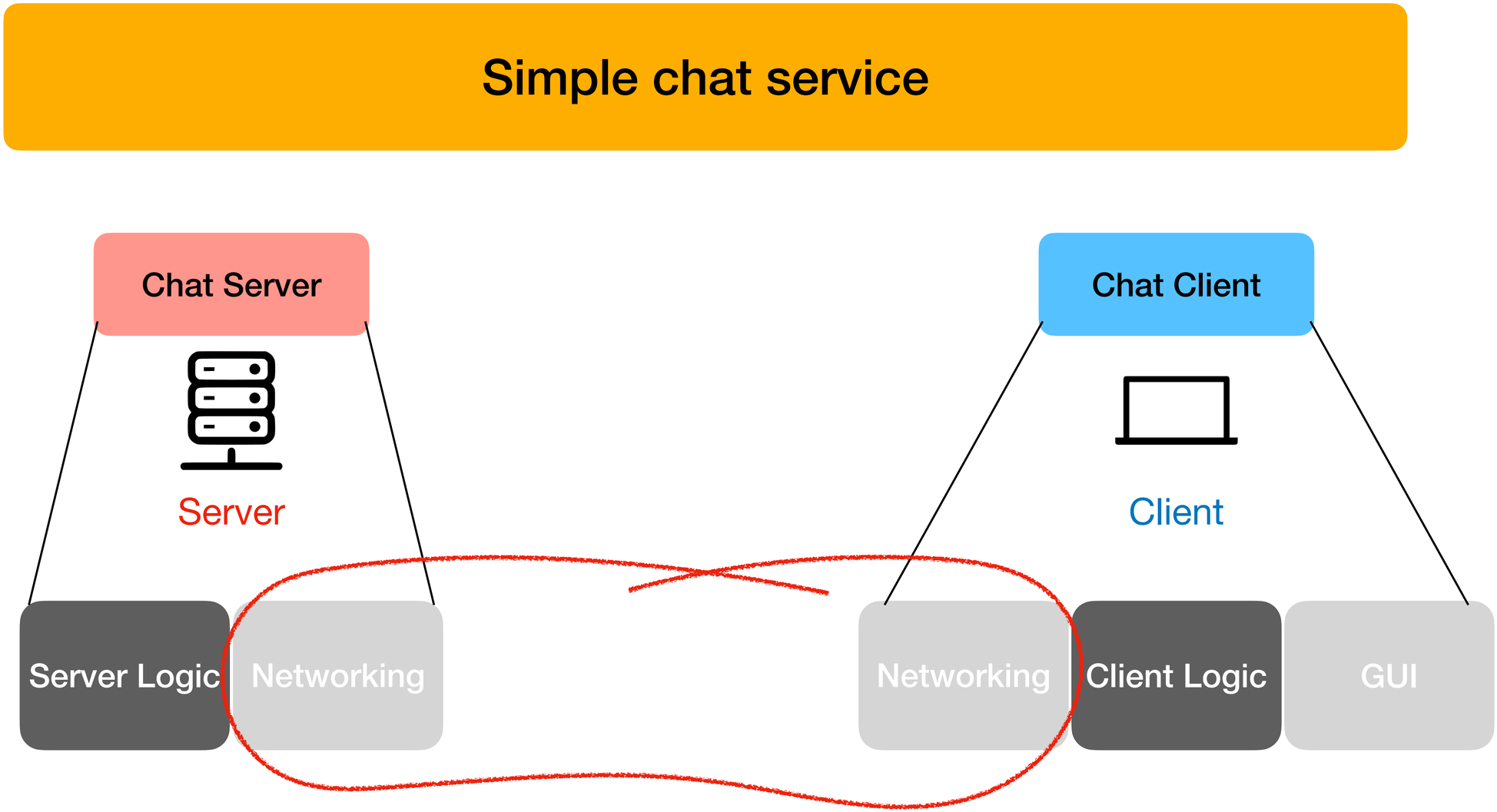
Building your own cloud service

Simple chat service

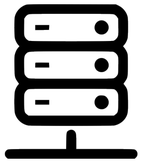




Building your own cloud service



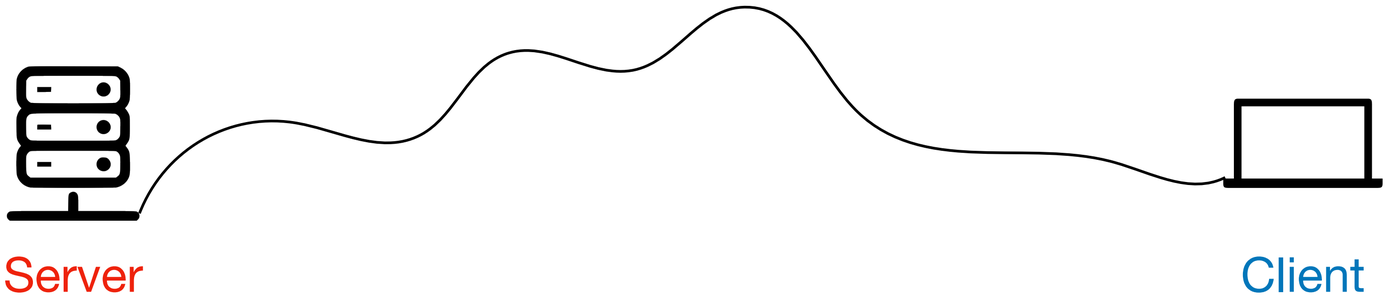
Networking



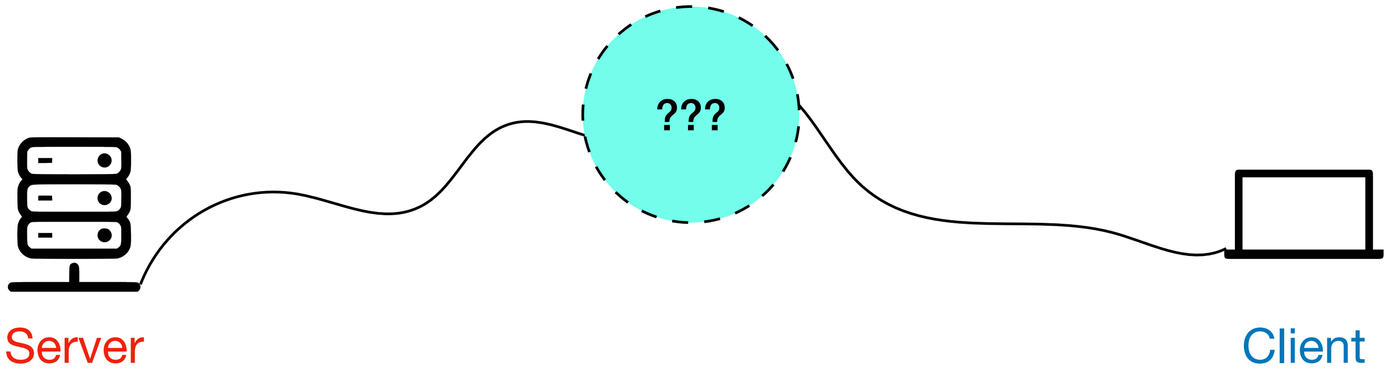
Server

Client

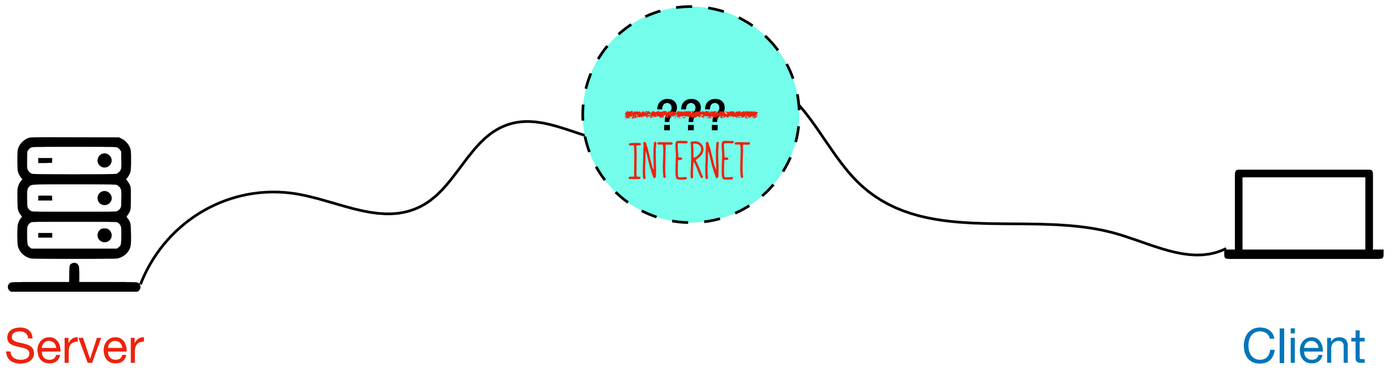
Networking



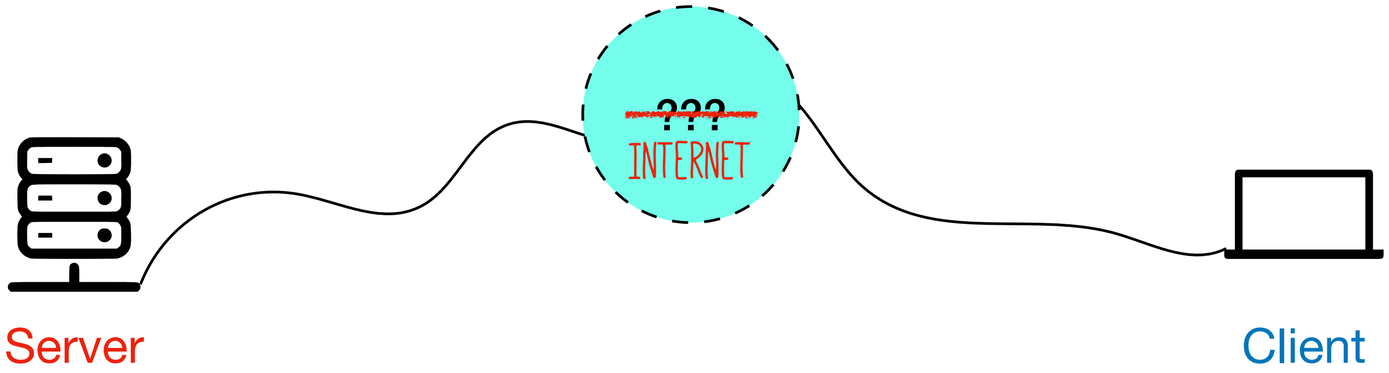
Networking



Networking

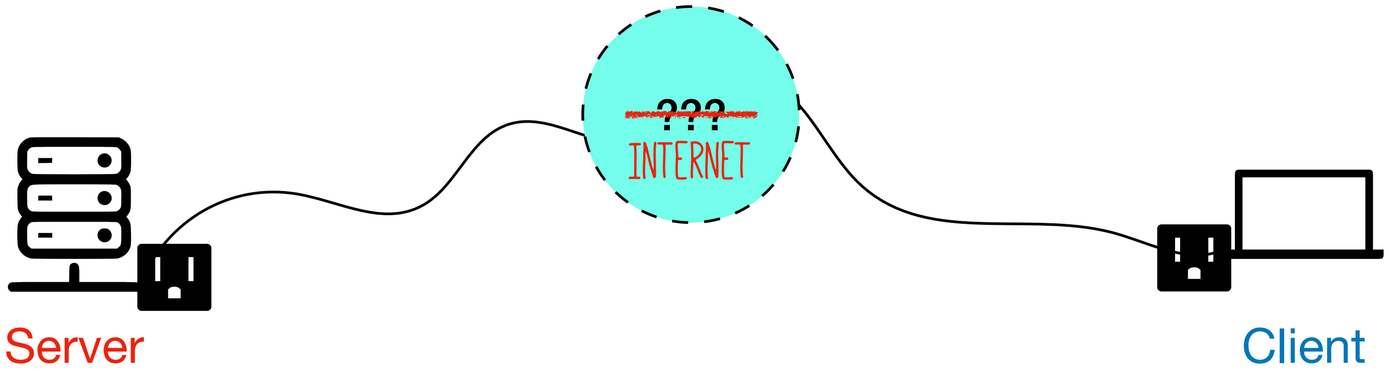


Networking



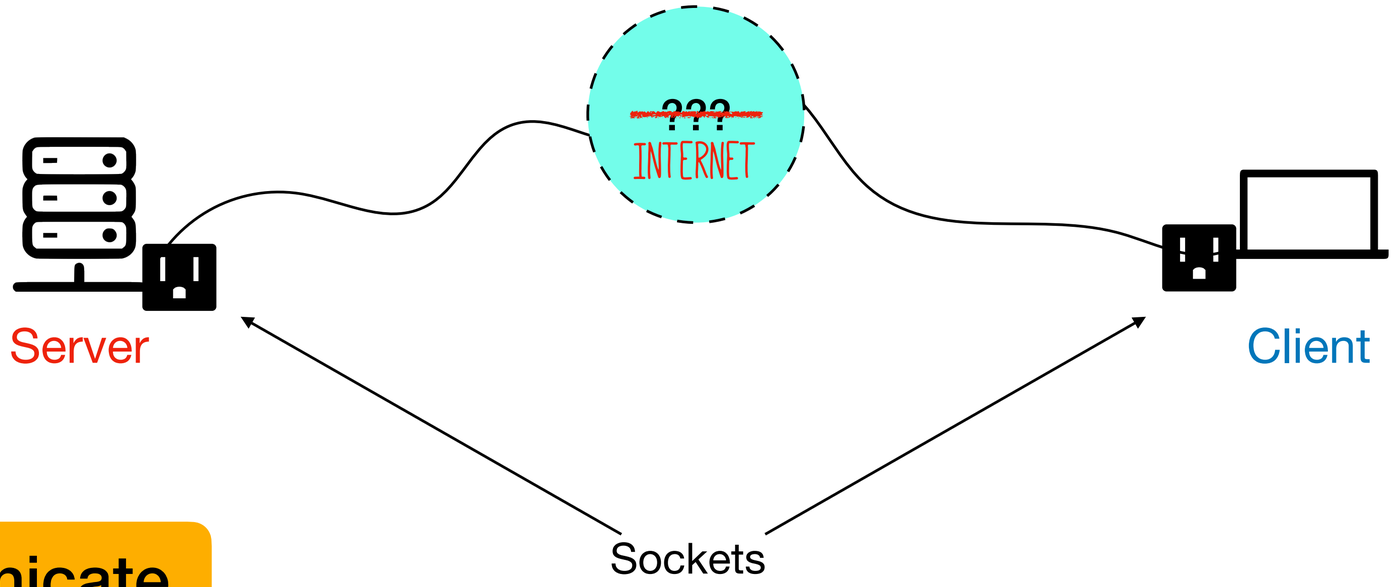
How to communicate over the internet?

Networking



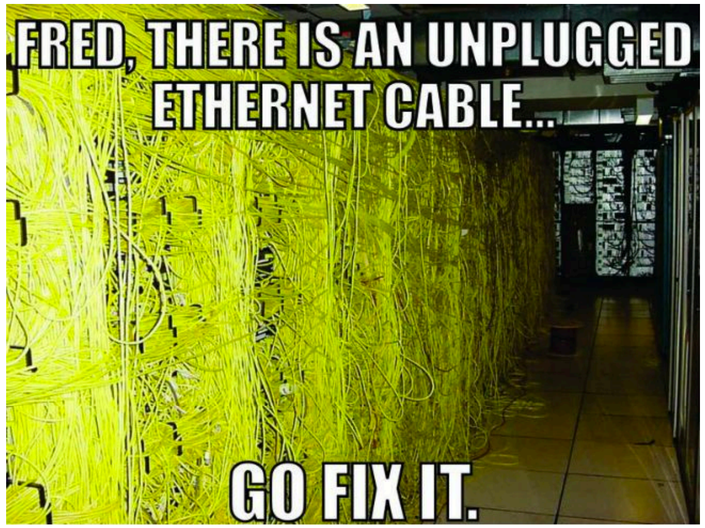
How to communicate over the internet?

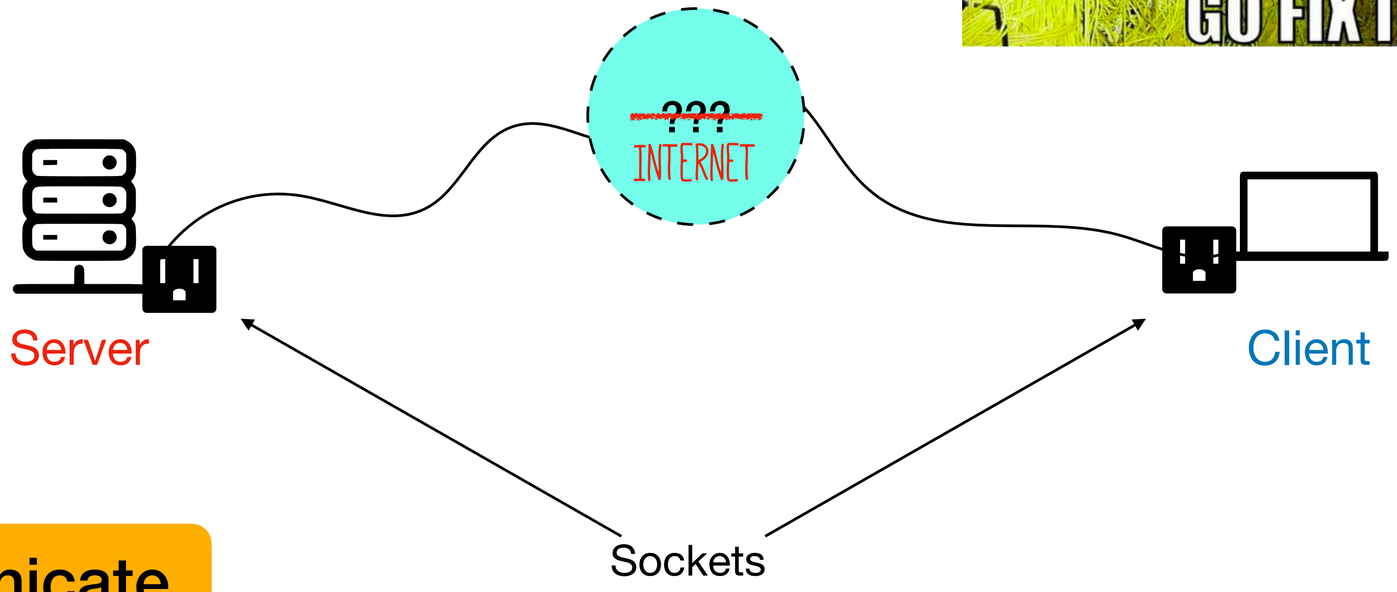
Networking



How to communicate over the internet?

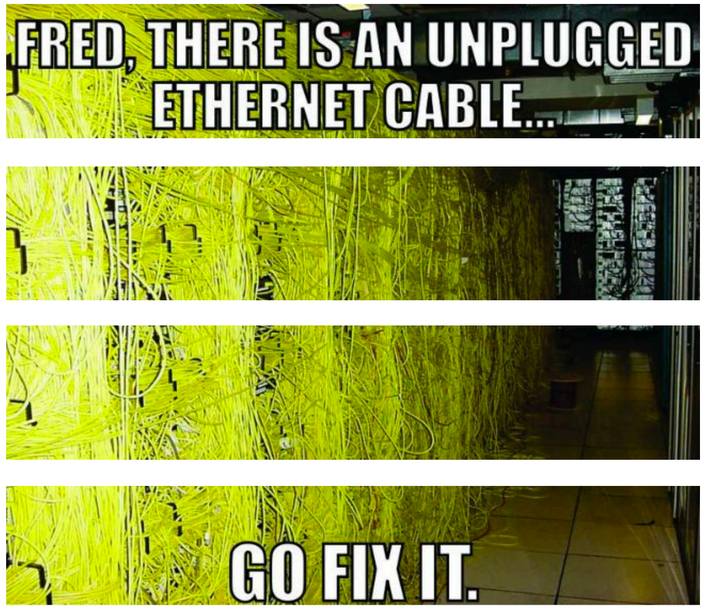
Networking

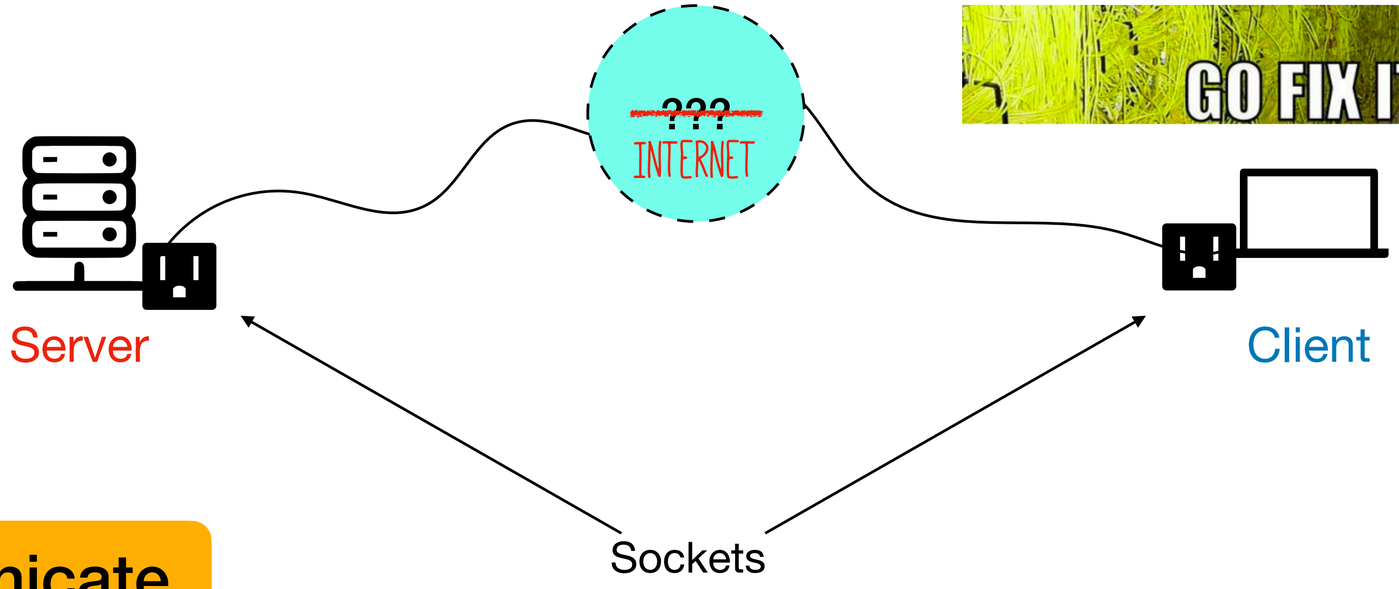




How to communicate over the internet?

Networking

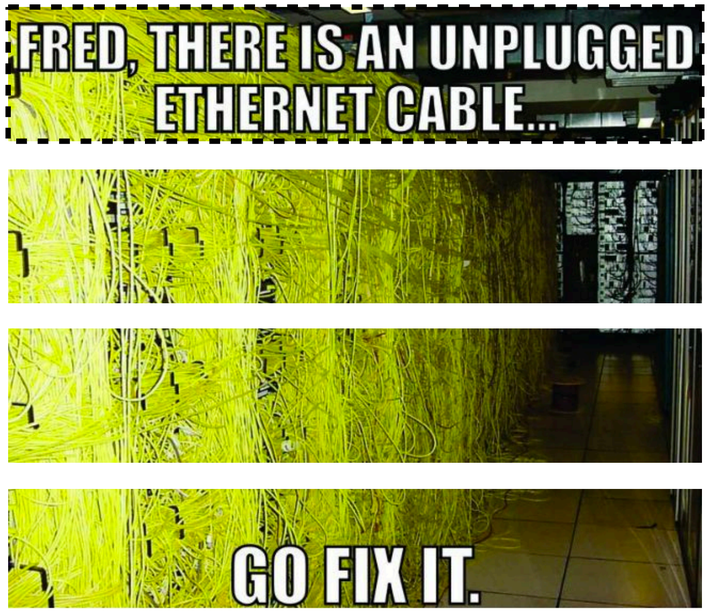


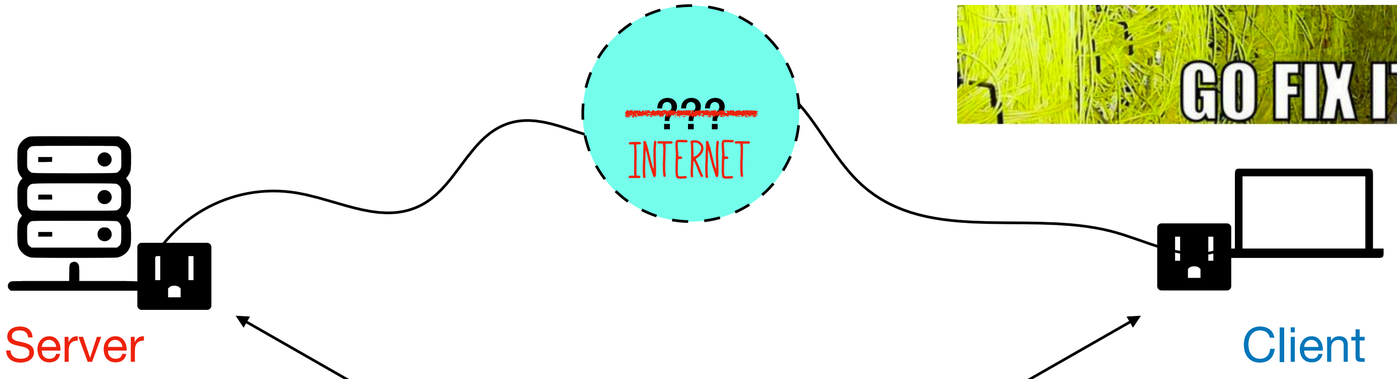


How to communicate over the internet?

Networking

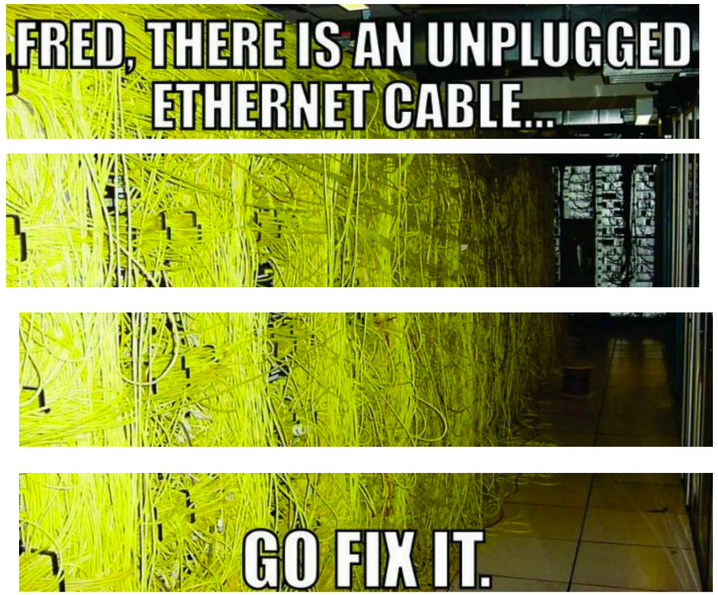
"Packet"

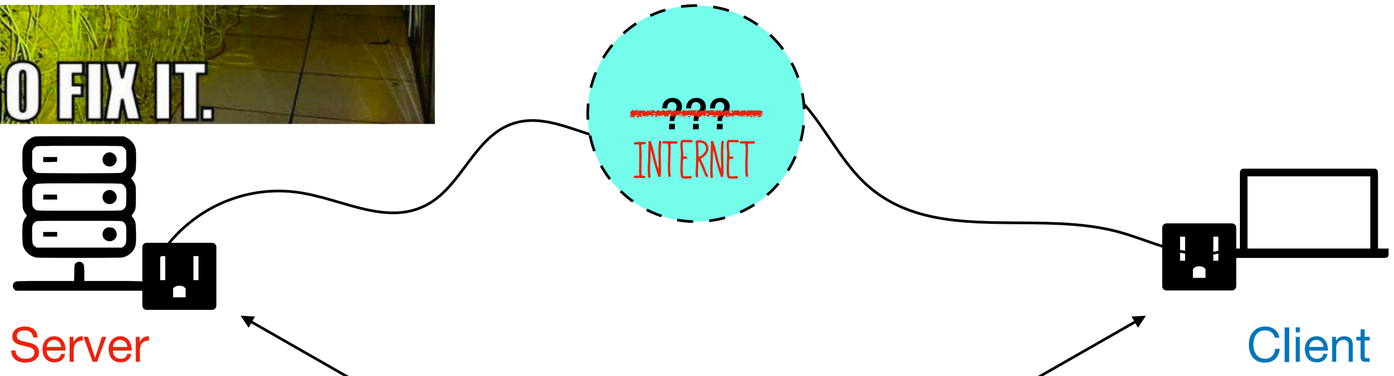




How to communicate over the internet?

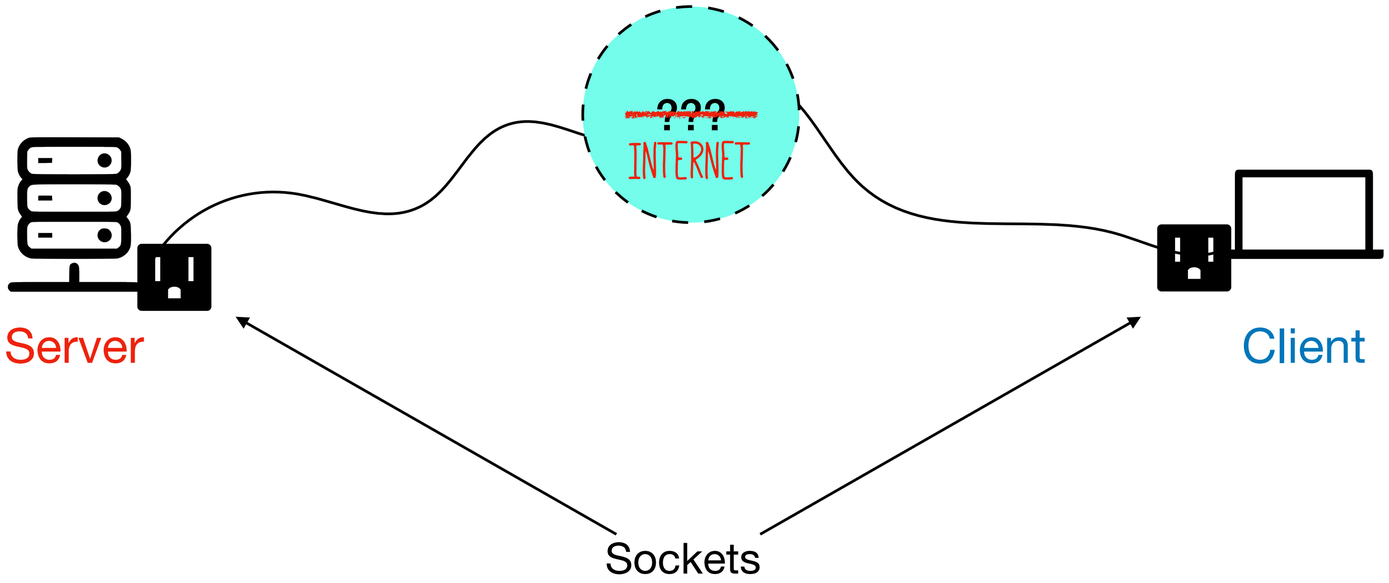
Networking



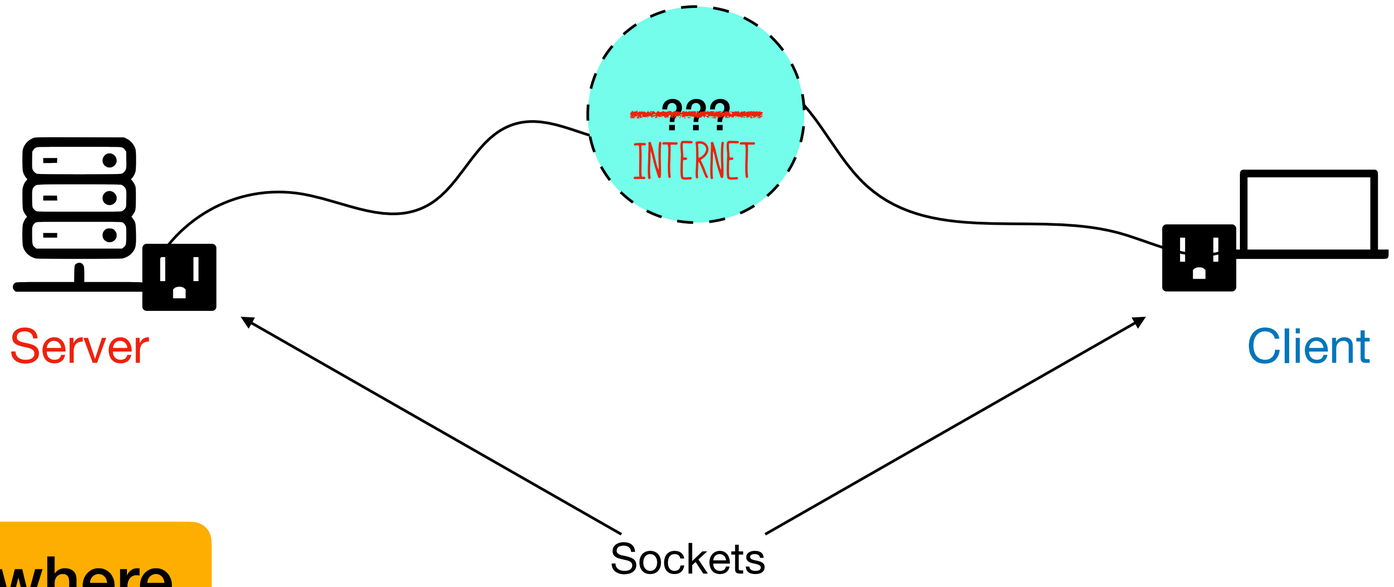


How to communicate over the internet?

Networking

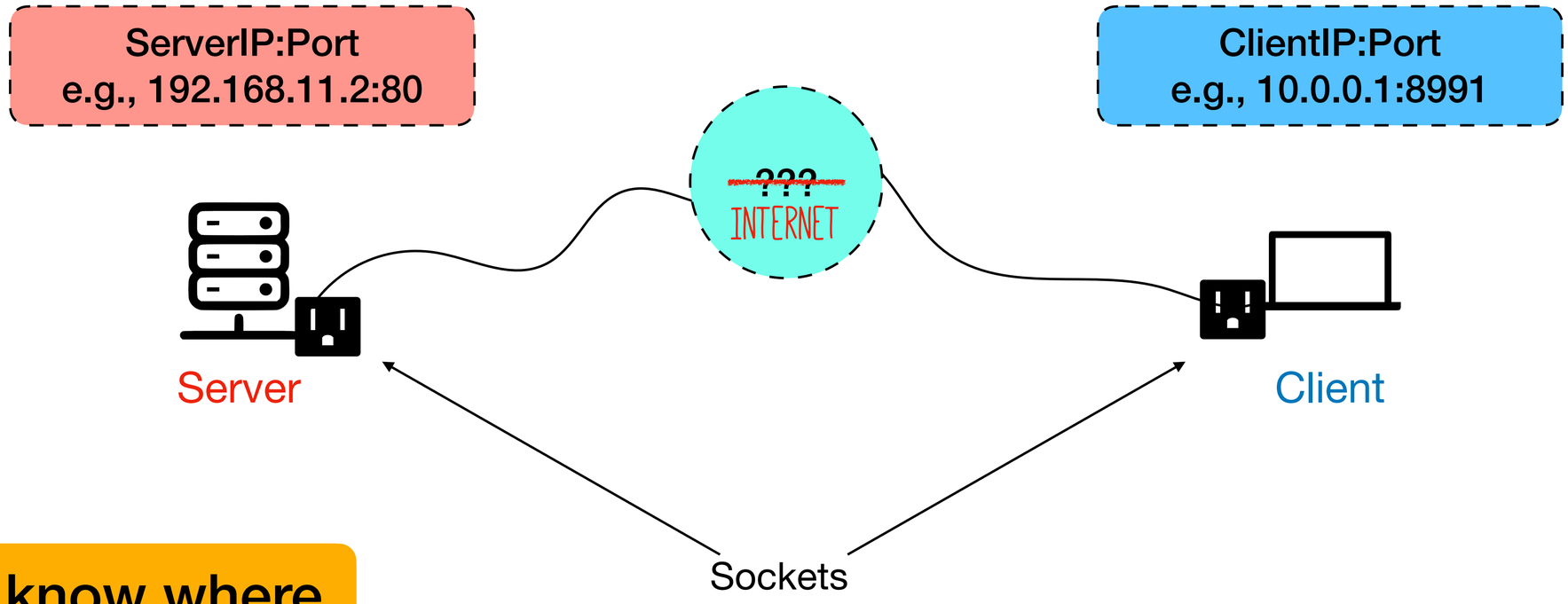


Networking



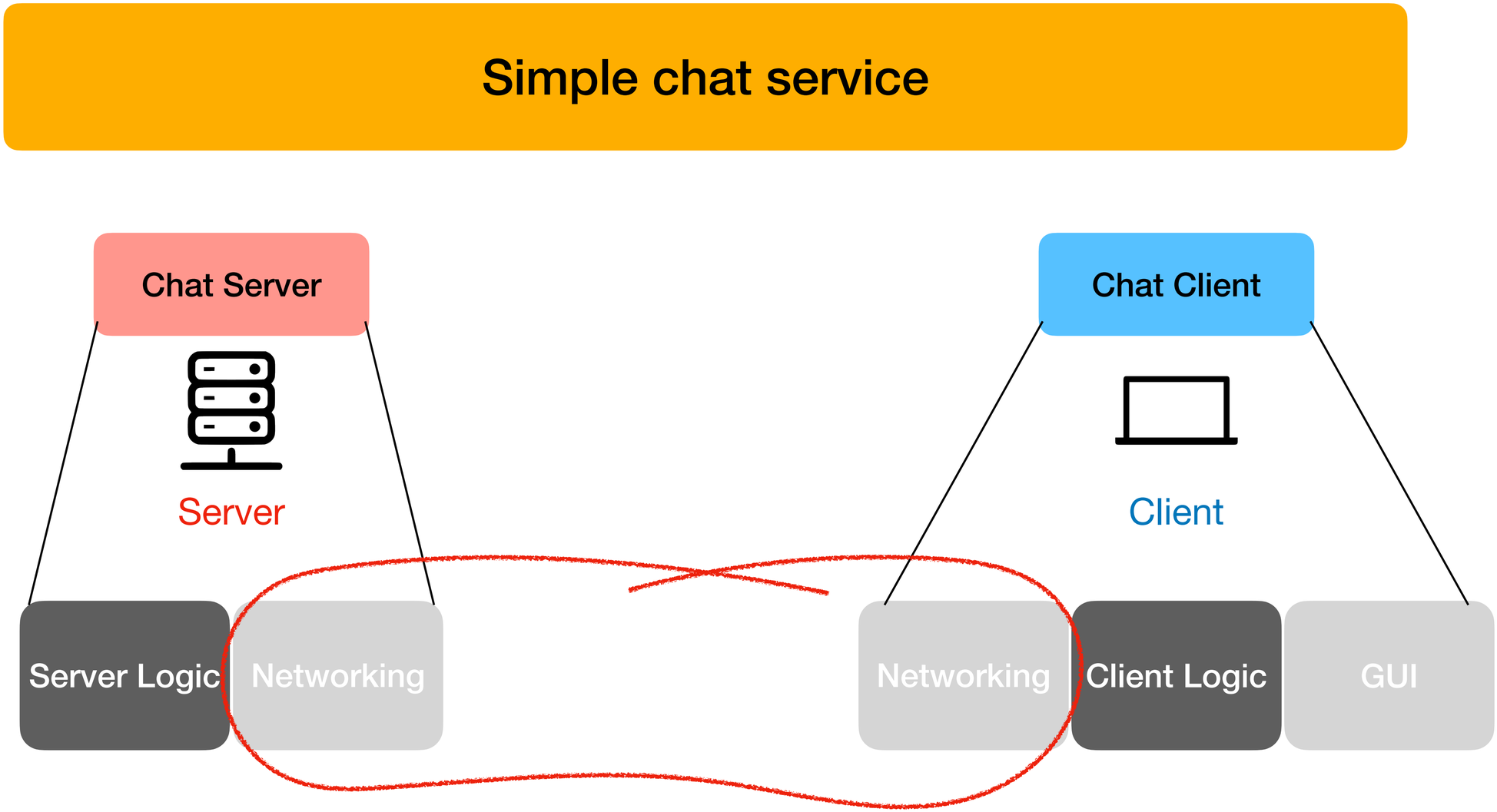
How do I know where the client & server sockets are?

Networking

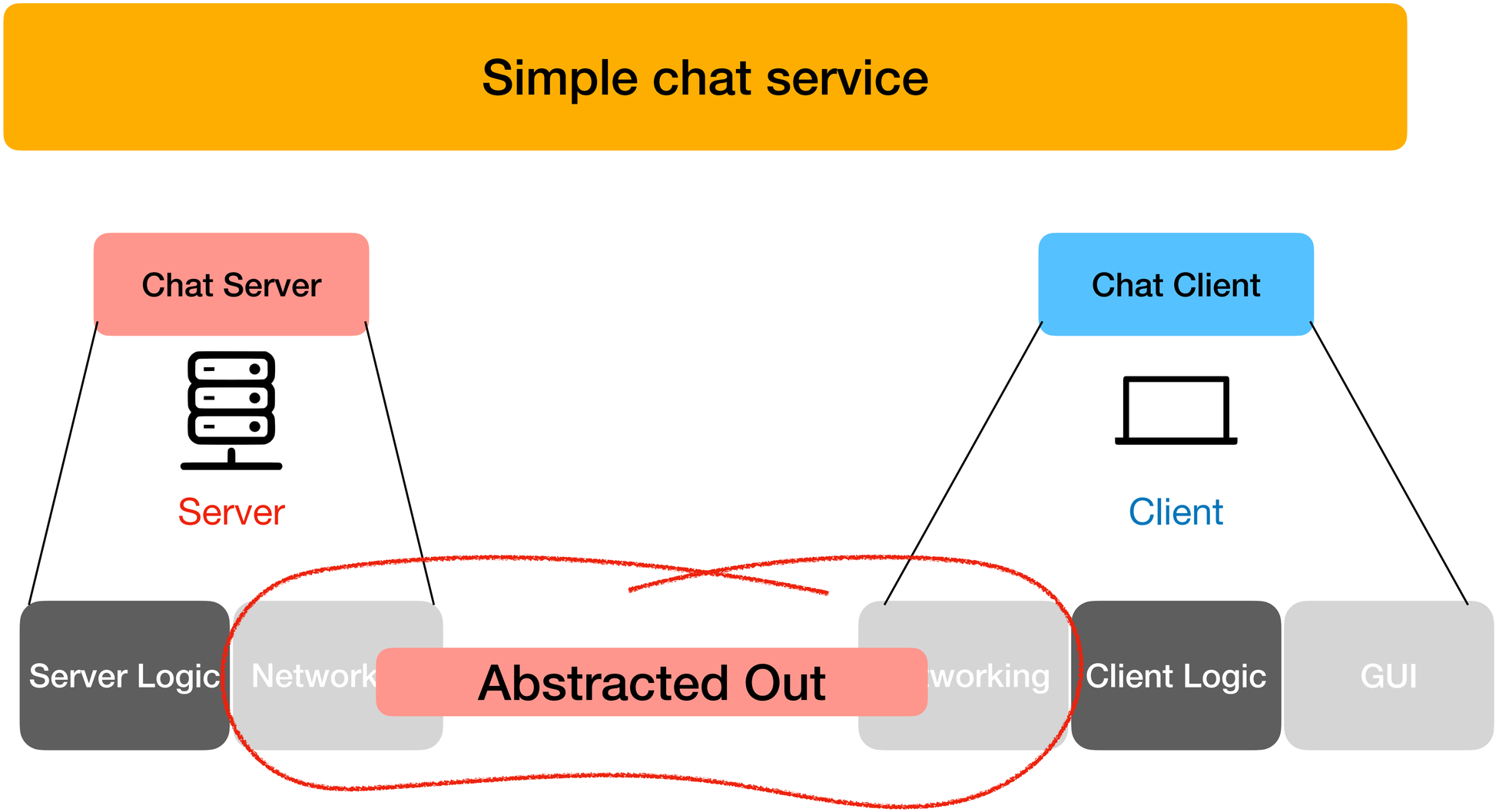


How do I know where the client & server sockets are?

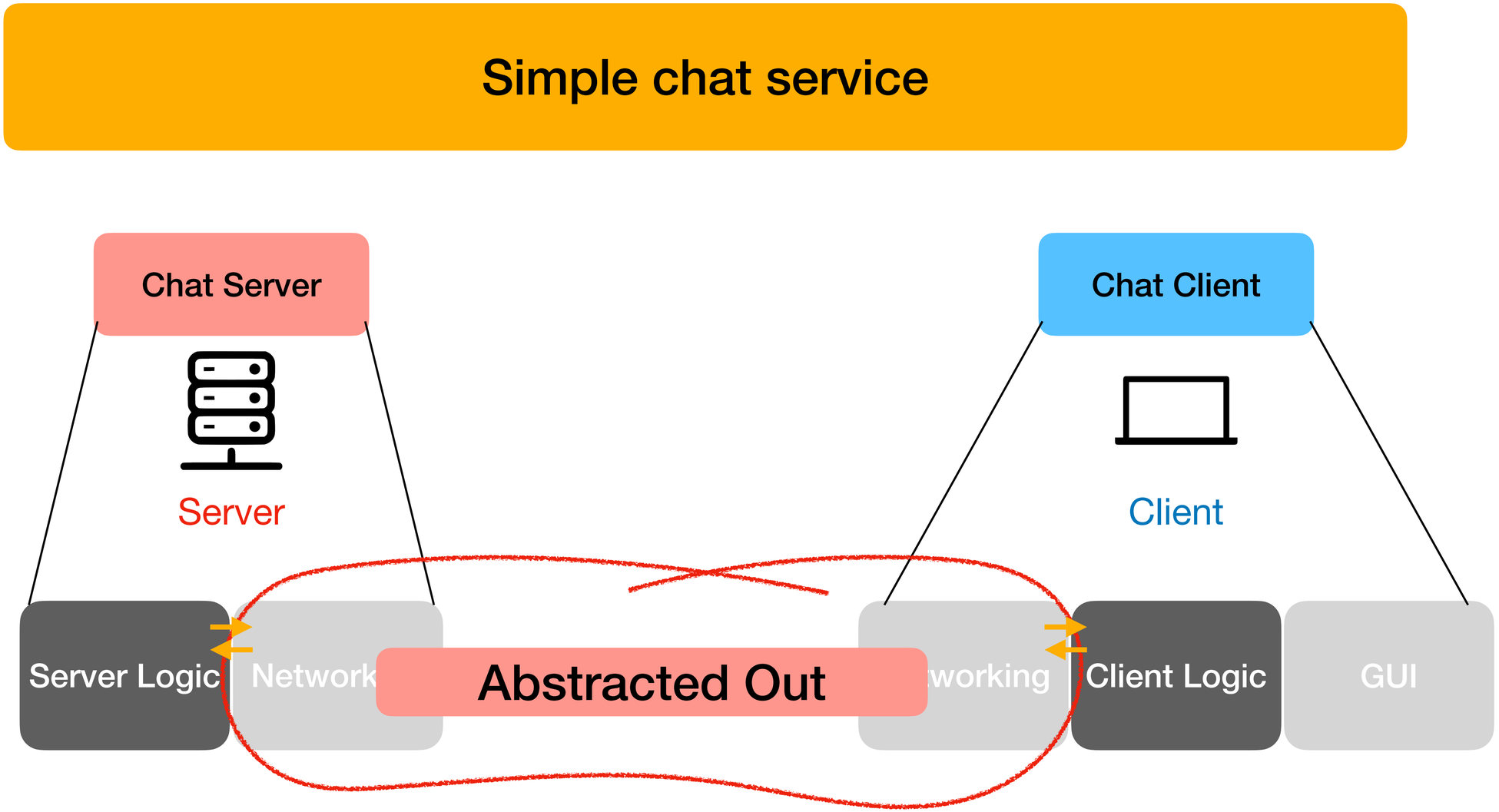
Building your own cloud service



Building your own cloud service



Building your own cloud service



Questions?

Setting up

Getting our hands dirty!

Building Your Chat Application in 4 Simple Steps

Building Your Chat Application in 4 Simple Steps

- Ping pong 1: Single Server, Single Client
- Ping pong 2: Single Server, Two Clients
- Client Registry & Login
- Instant Messaging!

Building Your Chat Application in 4 Simple Steps

- **Ping pong 1: Single Server, Single Client**
- Ping pong 2: Single Server, Two Clients
- Client Registry & Login
- Instant Messaging!

A simple ping-pong application

A simple ping-pong application

A simple ping-pong application

A simple ping-pong application

A simple ping-pong application

How do you do it?

How do you do it?

- The functions you have to edit:
- In client_logic.py:

```
async def client_logic(client):
  # your code here
```

- In server_logic.py:

```
async def server_logic(server, msg):
  # your code here
```

How do you do it?

- The functions you have to edit:
- In client_logic.py:

Abstracts network operations on the client side

```
async def client_logic(client):
  # your code here
```

- In server_logic.py:

```
async def server_logic(server, msg):
  # your code here
```

How do you do it?

- The functions you have to edit:
- In client_logic.py:

```
async def client_logic(client):
  # your code here
```

Abstracts network operations on the client side

- In server_logic.py:

```
async def server_logic(server, msg):
  # your code here
```

Abstracts network operations on the server side

How do you do it?

- The functions you have to edit:
- In client_logic.py:

```
async def client_logic(client):
    # your code here
```

- In server_logic.py:

```
async def server_logic(server, msg):
    # your code here
```

Abstracts network operations on the client side

Abstracts network operations on the server side

Message received by the server (hopefully a “Ping”!)

How do you do it?

- The functions you have to edit:
- In client_logic.py:

```
async def client_logic(client):
    # your code here
```

Abstracts network operations on the client side

```
await client.send_message(msg)
```

- In server_logic.py:

```
async def server_logic(server, msg):
    # your code here
```

Abstracts network operations on the server side

Message received by the server (hopefully a “Ping”!)

How do you do it?

- The functions you have to edit:
- In client_logic.py:

```
async def client_logic(client):
    # your code here
```

Abstracts network operations on the client side

```
await client.send_message(msg)
```

- In server_logic.py:

```
async def server_logic(server, msg):
    # your code here
```

Abstracts network operations on the server side

Message received by the server (hopefully a “Ping”!)

```
await server.send_message(msg)
```

How do you test it?

How do you test it?

- Open **two** terminals
- Go to the directory where the code resides (`pingpong1`) on both terminals

  `cd /path/to/pingpong1` (on Linux & MacOS)
  `cd \path\to\pingpong1` (on Windows)

- On one terminal, run the server:

  ```
  python3 server.py
  ```

- On the other terminal, run the client:

  ```
  python3 client.py
  ```

Demo & Coding