Networking

Sockets

- How do things communicate over the internet? (the simple version)
- This is not a networking class 😁

Some computing resource must bind to a specific port on its host, and then listen for incoming connections.

- Listens on a specific port
- For a HTTP, this software is our web server
- Since a bind must always precede a listen, we will typically omit the bind in our descriptions
- Most socket libraries will take care of this for you
Networking Ports

What's a Port?

- It's basically a door
  - Italian: Porta
  - French: Porte
  - Spanish: Puerta
- I like to think of a port as a door to a building.

Networking Ports

What's a Port?

- If we have some device on the internet with an IP address assigned to it, we can think of that as a building.
- A port then can be thought of as a door to the building.
- Doors can let stuff in or out.

Networking Ports

What's a Port?

- Each port has a number
  - 16 bit unsigned integers
  - 0 - 65535
- Internet Assigned Numbers Authority (IANA) has designated different port ranges for different thing, but there’s nothing stopping you from using them for whatever.
### Networking Ports

**Common Ports**

<table>
<thead>
<tr>
<th>Port Number</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>ssh - Secure Shell</td>
</tr>
<tr>
<td>23</td>
<td>Telnet (unsecure)</td>
</tr>
<tr>
<td>25</td>
<td>SMTP - Simple Mail Transport Protocol (unsecure)</td>
</tr>
<tr>
<td>80</td>
<td>HTTP - HyperText Transport Protocol (unsecure)</td>
</tr>
<tr>
<td>123</td>
<td>NTP - Network Time Protocol</td>
</tr>
<tr>
<td>443</td>
<td>HTTPS - HTTP Secure</td>
</tr>
<tr>
<td>587</td>
<td>SMTP Secure</td>
</tr>
<tr>
<td>3306</td>
<td>MySQL</td>
</tr>
<tr>
<td>25565</td>
<td>Minecraft</td>
</tr>
</tbody>
</table>

### Networking Sockets

- A client then opens a socket to the server
- A socket data stream that sits on top of the network layer provided by the operating system.
- A socket is described by an **IP address**, a **port**, and a **transport protocol**
- For our class, we'll use TCP for our protocol
  - Transmission Control Protocol

### Networking Sockets

- Both sides must **bind** to a port
- The server binds to the well-known port 80, since the clients need to know this
- The client typically uses a random high number available port
- As part of the socket connection, the client tells the server what port it is using
Networking
Sockets

• A web server can listen for and accept connections from many clients.

```
Web Server
listen(80)
```

Networking
Sockets

• Once a socket is connected, the client and server can exchange data according to whatever protocol the server supports.

• For web servers, this is HTTP.

```
GET /index.html HTTP/1.1
Host: example.com
```

Echo Server
The world's worst web server
Create a socket object

Create a server_addr tuple

0.0.0.0 indicates we want to listen on all network interfaces on the host

80 is our port
Bind the socket we created to the local server_addr we defined.

listen on this socket. 5 is the number of backlog connections to accept before the server starts refusing connections.

Wait for a connection, and then accept it. Returns a new connection socket and a client address tuple.
When there is data available on the socket, `recv` the data in 1024 byte chunks, and log it to the console. The `if not data` block will break out of this while loop when the connection is closed.

The Big Picture

- HTML
- CSS
- JavaScript
- Python
- MySQL
Web Servers
The Datacenter Model

Servers We're Responsible For: 7
Web Servers
The Cloud Model

Managed Container Service
Web Server Container
Web Server Container
Web Server Container
Managed Cloud Database

Servers We're Responsible For: 0

Web Servers
Many Different Types

- Apache 2 - httpd
- nginx (pronounced “Engine X”)
- IIS
- Tomcat
- Jetty
- Gunicorn

General Purpose HTTP Servers
Web Servers
Many Different Types

- Apache 2 - httpd
- nginx (pronounced “Engine X”)
- IIS
- Tomcat
- Jetty
- Gunicorn

Revisiting Containers

- We’ve already used containers to run a web server in Homework 2

```
docker run -it --rm -p 8080:80 hw02:latest
```

- Let’s look closer at what those port mappings mean

- Maps port 8080 on your host to the container’s port 80.
We need to access the outside port if we're outside the container.

And the inside port if we're inside the container.
We can run multiple containers, all with the same internal port.
• We can’t map the same port on the host to multiple containers!

• We need separate ports on the host for each container we want to forward traffic to

• Not all containers need their ports mapped to the host.
• Containers can also talk to each other directly, without having to leave the internal docker network