Announcements
- HW due Friday
- Review for final - two weeks from yesterday
About Networking: definitions

URL: http://www.mathcs.slu.edu

translated by domain name server - DNS

IP address: 165.134.12.1

IPv4
Networking (cont).

The network protocol, such as http, smtp, ssh, dhcp, etc., tells the computer what the connection is trying to do.

To help quickly distinguish types of individual connections, each network connection specifies a port which it tries to connect on. These range from 0 to 65535.
Ports:

Can choose arbitrary numbers, but some conventions exist.

<table>
<thead>
<tr>
<th>Port #</th>
<th>Connection type</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>ssh</td>
</tr>
<tr>
<td>53</td>
<td>DNS</td>
</tr>
<tr>
<td>80, 8080</td>
<td>http</td>
</tr>
</tbody>
</table>
Sockets

To manage network connections, programmers use an abstraction called a socket. When 2 machines connect, each establishes a dedicated socket. The OS uses this socket as an input/output for data going along this connection.

In Python, this is just an object imported from the socket library—a bit like file I/O.
Ex:

```python
from socket import socket
s = socket()  # use library
s.connect(('mathcs.slu.edu', 80))  # will raise error if can't connect
print(s.recv(1024))
s.close()  # characters
# also s.send(data)
```
Models for network communication

Many models out there.

Two common ones:

1. **Client - server**:
   - Server waits for a client to connect & request data, then sends that data.

2. **Peer-to-peer**:
   - Two connected machines w/ roughly the same capabilities.
Protocols.

In any model, need an established protocol.

Each machine needs to know what is expected to happen next, so that their programming can handle the communication.
Ex: daytime protocol

The daytime protocol is a very simple one running on port 13.

When a connection opens, the server immediately sends a string containing day info about the day and the time.

Format: \n 54169 07-03-10 02:43:18, 52 0 0 594.5 UTC(NIST) \n
Managed time.nist.gov
An example script

We’ll code a program that connects to this API and prints the day and time in a nice format.

Note: mathcs.slu.edu only works from turing.time.nist.gov and won’t work on campus.
Servers

A bit more complex than clients which make a request, wait, and then process the request.

Servers initialize and need to wait for clients to contact them.

TCPStream class is provided to help with this setup.

We'll set up a class that inherits from a BaseRequestHandler class.
TCP Server

- Will initialize socket, & save it as self.request.

  So we can use send and recv (as the server!)

- We'll set up our server & put it in a loop (like our Tkinter stuff) so it will always listen.

- Our example will be an echo server