

# CSE 461 26wi Section 1 – Socket API, Ping & Traceroute

*First section? I think you mean first socket*

## Socket API

Berkeley sockets, also called BSD sockets, is an API that provides an abstraction on the application layer for network communication. A socket is represented as a file descriptor by the API under the POSIX standards (CSE333) to support a common interface for I/O.

Primitive	Description
SOCKET	Create a new communication endpoint
BIND	Associate a local address (port) with a socket
LISTEN	Announce willingness to accept connections; (give queue size)
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Like all major languages, the `socket` module in Python provides access to the Socket API. For example, the function to create a socket that looks like this:

```
socket.socket(family=AF_INET, type=SOCK_STREAM, ...)
```

→ `family` is the address family. Common ones are listed below:

- ◆ `socket.INET` – Specifies an IPv4 address.
- ◆ `socket.INET6` – Specifies an IPv6 address.

→ `type` is the socket type. Only the two below are generally useful:

- ◆ `socket.SOCK_STREAM` – Creates a socket for TCP connection.
- ◆ `socket.DGRAM` – Creates a socket for UDP connection.

## Exercise 1

Consider the sample server/client code below. Assume we are concerned with serving just a single client.

<pre># Server listener = socket.socket(socket.AF_INET,                socket.SOCK_STREAM) listener.bind(server_address) while True:     connection, client_addr = listener.accept()     connection.recv(n_bytes)  listener.close()</pre>	<pre># Client socket = socket.socket(socket.AF_INET,                socket.SOCK_STREAM) socket.connect(server_address) socket.sendto(message, server_address) socket.close()</pre>
--	--

a) Circle the type of socket being set up

TCP / UDP

b) Briefly explain what this program does. Do you spot any errors?

c) What happens if no client talks to the server?

d) Suppose we launched a server from this code and have a single client sends some bytes to it. How many sockets will be created between the server and client?

e) Would this program work (well) in multi-client scenarios? In other words, if the server is waiting on data from client A, can it accept a connection request from client B?