Please join the **eduroam** wifi network.

If it asks you for a username/password, use your SUNET ID (first part of email) and password, and accept any questions about certificates.
A Grain Of Sand
The distinction between “virtual” and “physical” is fuzzy.

It may be small, but every piece of data, every process, involves the physical presence, absence, or movement of at least electrons.

The internet lets us affect real objects, anywhere, anywhere.

Whether it’s just causing a CPU to do some work, or adjusting the position of a mechanical arm constructing a car– our actions over a network set off a veritable rube goldberg machine that actually makes physical changes somewhere else– as far as across the globe!
Learning Goals

- Understand basics of computer networking
  - Circuit-switched vs packet-switched networks
  - Ethernet, IP, TCP; NAT, DHCP, DNS
- Understand the structure and use of Ethernet, and IPv4 addresses
- Understand how a packet is routed around Stanford’s campus
- Understand how a basic home network works
- Obtain a rudimentary understanding of DNS
- Understand what a server and client are
- **Understand how to use ping and traceroute tools**
- **Understand how to use nslookup and dig tools**
- **Run your first server and have a friend/classmate connect to it!**
The Inception of Modern Computer Networking

- **ARPANET** was the progenitor of the modern internet.
- It was created with the goal of facilitating resource sharing between faraway computers, especially for research purposes.
- Key innovation: packet-switched wide area network
Here to There and Back Again

[demo circuit switched network]
Here to There and Back Again

[ demo packet switched network ]
Concepts from the Demo

**Packets (the envelope)**

Chuncked-up data, including any information about how to transport the data and where to.

**Routers (that’s you!)**

Computers whose special job is to ingest incoming packets and very quickly decide where to forward those packets to. (They often have special hardware to make this happen faster.)

**Addresses**

We wrote things like “back left of the room,” but computers that want to talk to each other must have an address, and often have multiple of different kinds for different purposes. More on this in a minute.

**Hops**

Each time the “packet” was “forwarded” (each time the envelope exchanged hands), that was a “hop.”
Routers *(that’s you!)*

By the way, consumer “routers” you might buy at the store or be given by your ISP are actually much more than just a router: they’re usually a router, firewall, access point, and network switch all at once.
Addresses I

IP Addresses

These are used to identify a computer on a network. That means that an IP address is unique to the network it’s in.

IP Addresses are structured in a particular way that makes routing efficient. More on this in a moment.

IP Addresses are used for getting information across a network, even if it needs to hop through many different points to get there.

Ethernet Address

These are used to identify a computer globally. They are unique across all devices*, and cannot be changed*.

Ethernet addresses are used to send information locally, i.e. to other computers you are directly connected to.
IP Addresses

These are used to **identify a computer on a network**. That means that **an IP address is unique to the network it’s in**.
IP Addresses

These are used to identify a computer on a network. That means that an IP address is unique to the network it’s in.

If that network is your home network, your IP addresses will be unique only in that network. (This is commonly referred to as a Local Area Network, or a LAN).
IP Addresses

These are used to **identify a computer on a network**. That means that an *IP address is unique to the network it’s in*.

If that network is **your home network**, your IP addresses will be unique only in that network. (This is commonly referred to as a Local Area Network, or a LAN).

If that network is **the Internet**, your IP address(es) will be unique on the entire internet. (This is commonly referred to as a Wide Area Network, or a WAN).
IP Addresses

These are used to **identify a computer on a network.** That means that an **IP address is unique to the network it’s in.**

If that network is **your home network**, your IP addresses will be unique only in that network. (This is commonly referred to as a Local Area Network, or a LAN).

If that network is **the Internet**, your IP address(es) will be unique on the entire internet. (This is commonly referred to as a Wide Area Network, or a WAN).

Sometimes, you need to go **between different networks** (e.g. between a LAN and the Internet). In order to do this, you use something called **Network Address Translation (or NAT)**.
Here is what your typical IPv4 address looks like:

192.168.1.1
Here is what your typical IPv4 address looks like:

```
192.168.1.1
```

It might look like a set of four numbers, but it’s really just a special format of a single 32-bit number, chunked up in 8-bit sections.
Here is what your typical IPv4 address looks like:

192.168.1.1

IP Addresses are **hierarchical**. Networks are defined in terms of this hierarchy.
Here is what your typical IPv4 address looks like:

192.168.x.x

IP Addresses are **hierarchical**. For example, any IP address starting with 192.168 (that is, starting with 1100000010101000) is defined as being for LANs.
Here is what your typical IPv4 address looks like:

192.168.0.0/16

One common way to define a subnet is using CIDR notation. The above says that “the first 16 bits are locked in, and the rest of them are for the network”
Here is what your typical IPv4 address looks like:

```
192.168.1.0/24
```

This is a common LAN subnet used by home routers.
Here is what your typical IPv4 address looks like:

![IP address diagram](image)

Every network has two special addresses: the first address (e.g. 192.168.1.0, for the above network) is unused (it defines the network). The last address (e.g. 192.168.1.255 for the above network) is called the broadcast address. Messages sent to broadcast are received by all computers on the network.
Here is what your typical IPv4 address looks like:

\[10.0.0.0/24\]

Here’s another common LAN network.
Here is what your typical IPv4 address looks like:

```
10.0.0.0/8
00001010xxxxxxxxxxxxxxxxxxxxxxxxxxx
```

In fact, any IP address in the subnet 10.0.0.0/8 is reserved for LANs!
Here is what your typical IPv4 address looks like:

10.0.10.0/23

00001010.00000000.0000101x.xxxxxxxxx

Of course, subnets don’t have to only be /8s, /16s, or /24s— they can be any number of bits you want*. For example, my home network uses this subnet.
This is one of the blocks of public IPv4 space that Stanford owns:

128.12.0.0/16

10000000.0001100.xxxxxxxxx.xxxxxxxxx

I.e., this is a set of IP addresses for the public Internet (not local area networks!) that Stanford owns exclusively. Nobody else can use these.
Meanwhile, most home internet connections give you only a single, **temporary** public IP address. Some of them don’t even give you that.

\[
172.217.14.78/32
\]

A /32 is a single IP address.
Meanwhile, most home internet connections give you only a single, temporary public IP address. Some of them don’t even give you that.

172.217.14.78/32

A /32 is a single IP address. Unless you’re running an ISP or extremely large business, you’ll likely be given an IP address from a DHCP (Dynamic Host Configuration Protocol) server.
Addresses II: IP Addresses: Subnets

Meanwhile, most home internet connections give you only a single, temporary public IP address. Some of them don’t even give you that.

172.217.14.78/32

A /32 is a single IP address. Unless you’re running an ISP or extremely large business, you’ll likely be given an IP address from a DHCP (Dynamic Host Configuration Protocol) server. DHCP servers oversee some set of IP addresses and give them out to computers that ask, ensuring no duplicates.
However: Public IP addresses are owned by people/companies, and private IP addresses have duplicates and are given dynamically. We need some way to identify specific computers in order to send them information.

Enter: MAC Addresses (a.k.a. Ethernet addresses).
Addresses III: Ethernet Addresses

Here is what your typical MAC address looks like:

2A:78:A3:5F:5D:02
Here is what your typical MAC address looks like:

```
2A:78:A3:5F:5D:02
```

MAC Addresses are also a chunked-up single number, in this case, a 48-bit number.
Here is what your typical MAC address looks like:

2A:78:A3:5F:5D:02

001010100111100010100011010111110101110100000010

They are split into two parts: the **OUI** (an ID given to a particular manufacturer) and the **NIC ID** (the ID for your computer*).
Here is what your typical MAC address looks like:

\[ 2A:78:A3:5F:5D:02 \]

001010100111100010100011010111110101110100000010

Ethernet addresses (MAC addresses) are used to send packets link-locally (i.e. to other computers on the same logical “wire” as you. Usually the same as your LAN.)
Visiting My Dorm Room: I

Let’s send a message to my dorm room!

“Hello World!”
Let’s send a message to my dorm room!
Let’s send a message to my dorm room!
Visiting My Dorm Room: I

Let’s send a message to my dorm room!

“Please make sure my data gets to where I’m sending it, in order, resending any information that gets dropped along the way.”
Let’s send a message to my dorm room!

We use **encapsulation** to layer information that we need to transfer our message across the network.
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49
Visiting My Dorm Room: I

**ROUTING TABLE:**
- 0.0.0.0/0 via 128.12.10.1
- 128.12.10.0/24 via eth0 local

**SOURCE IP:**
- 128.12.10.141

**DEST IP:**
- 128.12.11.49
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

ROUTING TABLE:
0.0.0.0/0 via 128.12.10.1
128.12.10.0/24 via eth0 local
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

ROUTING TABLE:
0.0.0.0/0 via 128.12.10.1
128.12.10.0/24 via eth0 local

TO EVGR C
Visiting My Dorm Room: I

**SOURCE IP:** 128.12.10.141

**DEST IP:** 128.12.11.49

**ROUTING TABLE:**
- 0.0.0.0/0 via 128.12.10.1
- 128.12.10.0/24 via eth0 local
Visiting My Dorm Room: I

**SOURCE IP:** 128.12.10.141

**DEST IP:** 128.12.11.49

**ROUTING TABLE:**
- 0.0.0.0/0 via 128.12.10.1
- 128.12.10.0/24 via eth0 local
Visiting My Dorm Room: I

ROUTE from local to router 1

Intro: Ethernet. Routing tables. CIDR. Encapsulation

TO EVGR C

ROUTING TABLE:
0.0.0.0/0 via 128.12.10.1
128.12.10.0/24 via eth0 local

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49
Visiting My Dorm Room: I

ROUTING TABLE:
0.0.0.0/0 via 128.12.10.1
128.12.10.0/24 via eth0 local

SOURCE IP:
128.12.10.141

DEST IP:
128.12.11.49
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

ARP: Hey! Who has 128.12.10.1?
Visiting My Dorm Room: I

**SOURCE IP:**
128.12.10.141

**DEST IP:**
128.12.11.49

**ARP:**
Hello! I have **128.12.10.1**, and my MAC address is **34:F3:00:AA:BB:CC**
Visiting My Dorm Room: I

**SOURCE MAC:**
8D:FF:12:DD:EE:FF

**DEST MAC:**
34:F3:00:AA:BB:CC
Visiting My Dorm Room: I

**SOURCE MAC:** 8D:FF:12:DD:EE:FF

**DEST MAC:** 34:F3:00:AA:BB:CC
Visiting My Dorm Room: I

**SOURCE IP:**
128.12.10.141

**DEST IP:**
128.12.11.49
Visiting My Dorm Room: I

Routing Table:

- 0.0.0.0/0 via 128.12.0.1
- 128.12.0.0/24 via enp0s0 local
- 128.12.10.0/24 via enp1s7 local
- 128.12.11.0/24 via 128.12.0.2
- 128.12.12.0/24 via 128.12.0.3
- 128.12.13.0/24 via 128.12.0.4

Source IP: 128.12.10.141
Destination IP: 128.12.11.49
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

ROUTING TABLE:
0.0.0.0/0 via 128.12.0.1
128.12.0.0/24 via enp0s0 local
128.12.10.0/24 via enp1s7 local
128.12.11.0/24 via 128.12.0.2
128.12.12.0/24 via 128.12.0.3
128.12.13.0/24 via 128.12.0.4
Visiting My Dorm Room: I

SOURCE IP: 128.12.10.141

DEST IP: 128.12.11.49

ROUTING TABLE:
- 0.0.0.0/0 via 128.12.0.1
- 128.12.0.0/24 via enp0s0 local
- 128.12.10.0/24 via enp1s7 local
- 128.12.11.0/24 via 128.12.0.2
- 128.12.12.0/24 via 128.12.0.3
- 128.12.13.0/24 via 128.12.0.4
Visiting My Dorm Room: II

SOURCE IP:
128.12.10.141

DEST IP:
128.12.11.49
Visiting My Dorm Room: II

ROUTING TABLE:
0.0.0.0/0 via 128.12.0.1
128.12.0.0/24 via enp0s0 local
128.12.10.0/24 via enp1s7 local
128.12.11.0/24 via 128.12.0.2
128.12.12.0/24 via 128.12.0.3
128.12.13.0/24 via 128.12.0.4
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Visiting My Dorm Room: II
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Visiting My Dorm Room: III

Wi-Fi AP

PoE Switch

Router + Firewall + Switch

Server

Server

Server
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Extends network to wireless clients

...also NAT, DNS, DHCP...

Extends network to more wired clients (and can power them)

For messing around on

Hosts applications I've developed

Hosts media for myself, friends, and family

Hosts media for myself, friends, and family

I've developed applications for messing around on

…the media for myself, friends, and family...
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Visiting My Dorm Room: III
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)
Visiting My Dorm Room: III

SOURCE IP: 128.12.10.141

DEST IP: 128.12.11.49
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Visiting My Dorm Room: III

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

"Hey! That's me!"
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Visiting My Dorm Room: III

WEB SERVER

MINECRAFT SERVER

SSH SERVER
Routing to a server in my dorm room

WEB SERVER
LISTEN: TCP PORT 80
LISTEN: TCP PORT 443

MINECRAFT SERVER
LISTEN: TCP PORT 25565

SSH SERVER
LISTEN: TCP PORT 22

Visiting My Dorm Room: III
Visiting My Dorm Room: III

**WEB SERVER**
LISTEN: TCP PORT 80
LISTEN: TCP PORT 443

**MINECRAFT SERVER**
LISTEN: TCP PORT 25565

**SSH SERVER**
LISTEN: TCP PORT 22

**SOURCE PORT:**
43224

**DEST PORT:**
80
Visiting My Dorm Room: III

**WEB SERVER**
LISTEN: TCP PORT 80
LISTEN: TCP PORT 443

**MINECRAFT SERVER**
LISTEN: TCP PORT 25565

**SSH SERVER**
LISTEN: TCP PORT 22

SOURCE PORT:
43224

DEST PORT:
80
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Visiting My Dorm Room: III

WEB SERVER
LISTEN: TCP PORT 80
LISTEN: TCP PORT 443

MINECRAFT SERVER
LISTEN: TCP PORT 25565

SSH SERVER
LISTEN: TCP PORT 22
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

WEB SERVER
LISTEN: TCP PORT 80
LISTEN: TCP PORT 443

MINECRAFT SERVER
LISTEN: TCP PORT 25565

SSH SERVER
LISTEN: TCP PORT 22
Routing to a server in my dorm room

**Intro: TCP for multiplexing (ports)**

- **WEB SERVER**
  - LISTEN: TCP PORT 80
  - LISTEN: TCP PORT 443

- **MINECRAFT SERVER**
  - LISTEN: TCP PORT 25565

- **SSH SERVER**
  - LISTEN: TCP PORT 22

"Hello World!"
Ping

The ping tool is useful for checking if a host is alive and reachable.

$ ping <destination>
$ ping 128.12.11.49

Traceroute

The traceroute (tracert on Windows) tool is useful for checking the path/route to a given host.

$ traceroute <destination>
$ traceroute 128.12.11.49
Delving Deeper

[demo wireshark]
Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Going Deeper On My Network

- Wi-Fi AP
- PoE Switch
- Router + Firewall + Switch
- Server
- Server
- Server
Going Deeper On My Network

Routing to a server in my dorm room

Intro: TCP for multiplexing (ports)

Wi-Fi AP

Routing to a server in my dorm room

Wi-Fi AP

PoE Switch

Router + Firewall + Switch

Server
My Network

LAN Network: 10.0.10.0/24

Router

WAN IP: 128.12.11.49
LAN IP: 10.0.10.1

Switch

No IP Address: ethernet only

IP: 10.0.10.2
WiFi Access Point

IP: 10.0.10.10
Server

IP: 10.0.10.10
Note that the inside of this box is a different network than outside (it is off the public internet). This means that the router must use NAT to translate between them.
My Network

Note that the inside of this box is a different network than outside (it is off the public internet). This means that the router must use NAT to translate between them.

Also note that the entire inner network has only one Internet IP address (given to the router).
Also note that the entire inner network has only one Internet IP address (given to the router).

Running HTTP on port 80
Port Forwarding:

Forward incoming data on TCP port 80 to 10.0.10.10

Also note that the entire inner network has only one Internet IP address (given to the router).
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

LAN Network: 10.0.10.0/24
LAN IP: 10.0.10.1

WAN IP: 128.12.11.48

Running HTTP on port 80

IP: 10.0.10.2
WiFi Access Point

Switch

Server
IP: 10.0.10.10

IP given via DHCP: 19.0.11.72
Port Forwarding

Port Forwarding: Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

SOURCE IP: 128.12.10.141
DEST IP: 128.12.11.49

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

SOURCE PORT: 43224
DEST PORT: 80

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

SOURCE IP: 10.0.10.1
DEST IP: 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

SOURCE IP: 10.0.10.1
DEST IP: 10.0.10.10

NAT:
Note how the source and destination have been rewritten for the local network!

Running HTTP on port 80
Port Forwarding

Port Forwarding: Forward incoming data on TCP port 80 to 10.0.10.10

NAT: The router must remember this translation in order for both parties to communicate

SOURCE IP: 10.0.10.1
DEST IP: 10.0.10.10

LAN Network: 10.0.10.0/24
LAN IP: 10.0.10.1
IP: 10.0.10.2
WiFi Access Point

Switch
No IP Address: ethernet only

Router
IP: 128.121.148.1

Server IP: 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

LAN Network: 10.0.10.0/24
Router
Switch
WiFi Access Point
Server
IP: 10.0.10.2
IP: 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

- **LAN Network:** 10.0.10.0/24
- **IP:** 10.0.10.1
- **WAN IP:** 128.12.11.48
- **Server IP:** 10.0.10.10
- **Running HTTP on port 80**
Port Forwarding

Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Port Forwarding

Port Forwarding:
Forward incoming data on TCP port 80 to 10.0.10.10

Running HTTP on port 80
Most interactions over the network have **two sides:**

1. The **Server**, which has resources or services that you want to access and use.
2. The **Client**, which accesses/uses those resources or services.
Servers are just computers.

Anything can be a server! The only thing it needs to do to qualify as a server is:

(1) **Run** an application that **listens** for connections on a particular port

That’s it!
Demo time!

[demo hosting and port forwarding]
Now you try!

[demo hosting and connecting locally]
Okay, but you never type in \texttt{128.12.11.49} when accessing websites, for example. You’d prefer to type in, say, \texttt{jonak.link}!

The system that allows this to happen is called the \textbf{Domain Name System}, or DNS.
Basically Just A Table

Originally, all hostnames were in a file *manually* maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have *records* that map *names* to *data*, within a *zone* (i.e. a domain). For example:

```
jonak.link. 300IN A  76.76.21.21
```
The Internet’s Yellowpages

Basically Just A Table

Originally, all hostnames were in a file *manually* maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have **records** that map **names** to **data**, within a **zone** (i.e. a domain). For example:

```
jonak.link. 300IN A  76.76.21.21
```
^Name
Basically Just A Table

Originally, all hostnames were in a file *manually* maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have *records* that map *names* to *data*, within a *zone* (i.e. a domain). For example:

```
jonak.link. 300IN A 76.76.21.21
```

^TTL (how long you can cache this record)
Basically Just A Table

Originally, all hostnames were in a file manually maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have records that map names to data, within a zone (i.e. a domain). For example:

jonak.link. 300IN A 76.76.21.21

^for the internet
The Internet’s Yellowpages

Basically Just A Table

Originally, all hostnames were in a file manually maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have records that map names to data, within a zone (i.e. a domain). For example:

```
jonak.link. 300IN A 76.76.21.21
```

^A-type record

(An A record contains IPv4 addresses)
Basically Just A Table

Originally, all hostnames were in a file *manually* maintained by Stanford Research Institute (SRI). This, obviously, didn’t scale.

Now it’s a fancy table! You have records that map names to data, within a zone (i.e. a domain). For example:

```
jonak.link. 300IN A 76.76.21.21
```

^value
The Internet’s Yellowpages

[demo nslookup and dig]
The Internet’s Yellowpages
Computers aren’t flexible. They need specific rules to be able to talk with each other. These are called protocols.
Computers aren’t flexible. They need **specific rules** to be able to talk with each other. These are called **protocols**.

One common protocol is **HTTP**, which we use for loading webpages. By convention, HTTP servers listen on port **80**.
You can use the `ssh` ("secure shell") command to securely create a remote terminal. That means you’re connected and running all your commands on the remote computer!

- This is the #1 way to connect to remote servers/etc
- It is secure—you can create secure proxies and port-forwards between your computer and a server.
- It’s pretty lightweight
- You can also transfer files using the protocol (`scp`)
Administrivia

- Assignment 3 has been released and is due next Wednesday, May 3rd!
- Reminder that our office hours are listed below our faces on our website, https://cs45.stanford.edu