Computer Networks

The Socket API (Project 1) & Traceroute (HW 1)

(§1.3.4, 6.1.2-6.1.4)

Originally By David Wetherall (djw@), Modified By Rithik Duggempudi(rrd3@)

UNIVERSITY of WASHINGTON

About Me

I’m Rithik!

Senior in CSE

I love distributed systems, network programming and computer security

This is my 3rd quarter as a TA

About Me

I’m Daniel!

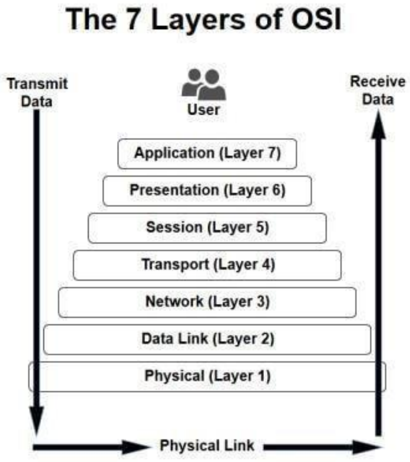
Final year in BS/MS

I too love security, systems, and network programming

This is my 3rd quarter as a TA

Network-Application Interface

- Defines how apps use the network
  - Application Layer APIs
  - Lets apps talk to each other
  - hides the other layers of the network



The 7 Layers of OSI

Transmit Data

User

Receive Data

Application (Layer 7)

Presentation (Layer 6)

Session (Layer 5)

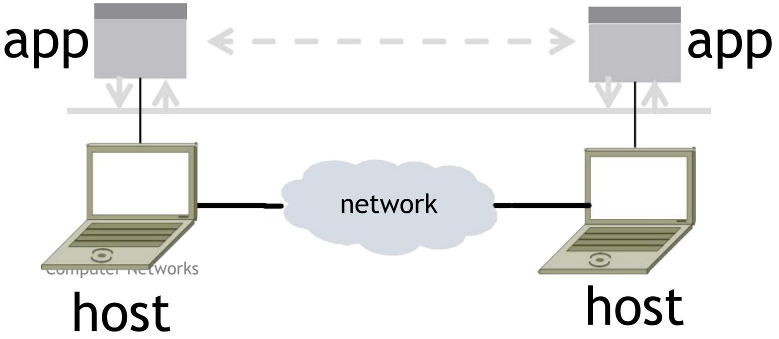
Transport (Layer 4)

Network (Layer 3)

Data Link (Layer 2)

Physical (Layer 1)

Physical Link



Project 1

- Simple Client
  - Send requests to attu server
  - Wait for a reply
  - Extract the information from the reply
  - Continue…
- Simple Server
  - Server handles the Client requests
  - Multi-threaded

Project 1

- This is the basis for many apps!
  - File transfer: send name, get file (§6.1.4)
  - Web browsing: send URL, get page
  - Echo: send message, get it back

- Let’s see how to write this app …

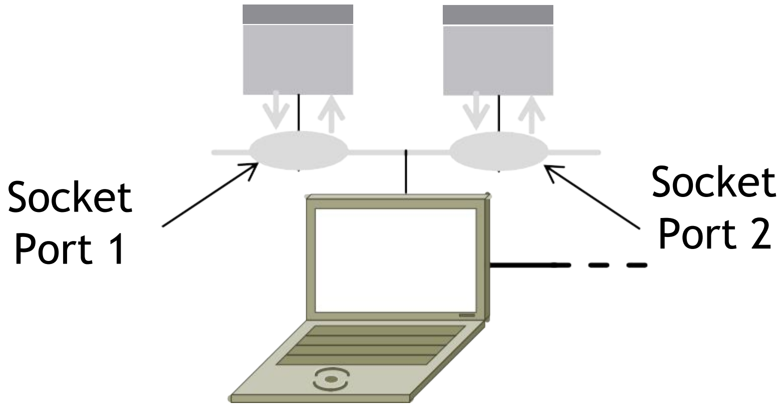
Socket API (Generalized)

- Simple application-layer abstractions (APIs) to use the network
  - The network service API used to write all Internet applications
  - Part of all major OSes and languages; originally Berkeley (Unix) ~1983

- Two kinds of sockets
  - Streams (TCP): reliably send a stream of bytes
  - Datagrams (UDP): unreliably send separate messages

Socket API (2)

- Sockets let apps attach to the local network at different ports
  - Ports are used by OS to distinguish services/apps using internet

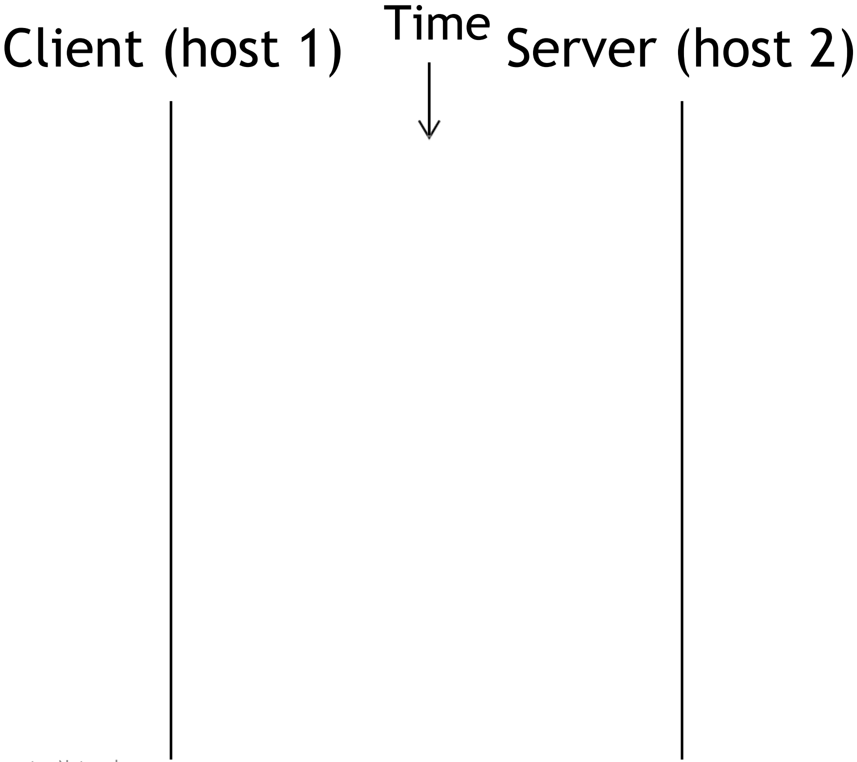


Socket API (3)

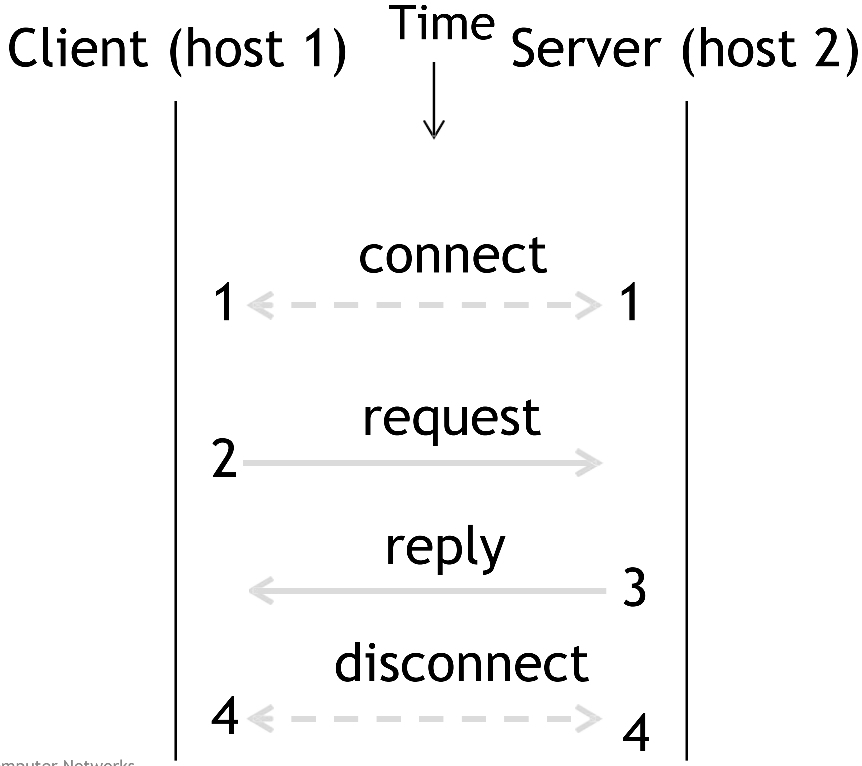
Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address (port) with a socket
LISTEN	Announce willingness to accept connections; (give queue size)
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data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

https://docs.oracle.com/javase/8/docs/api/java/net/Socket.html
https://docs.oracle.com/javase/8/docs/api/java/net/ServerSocket.html

Using Sockets



Using Sockets (2)



Client (host 1)

Time

Server (host 2)

connect

1

1

request

2

reply

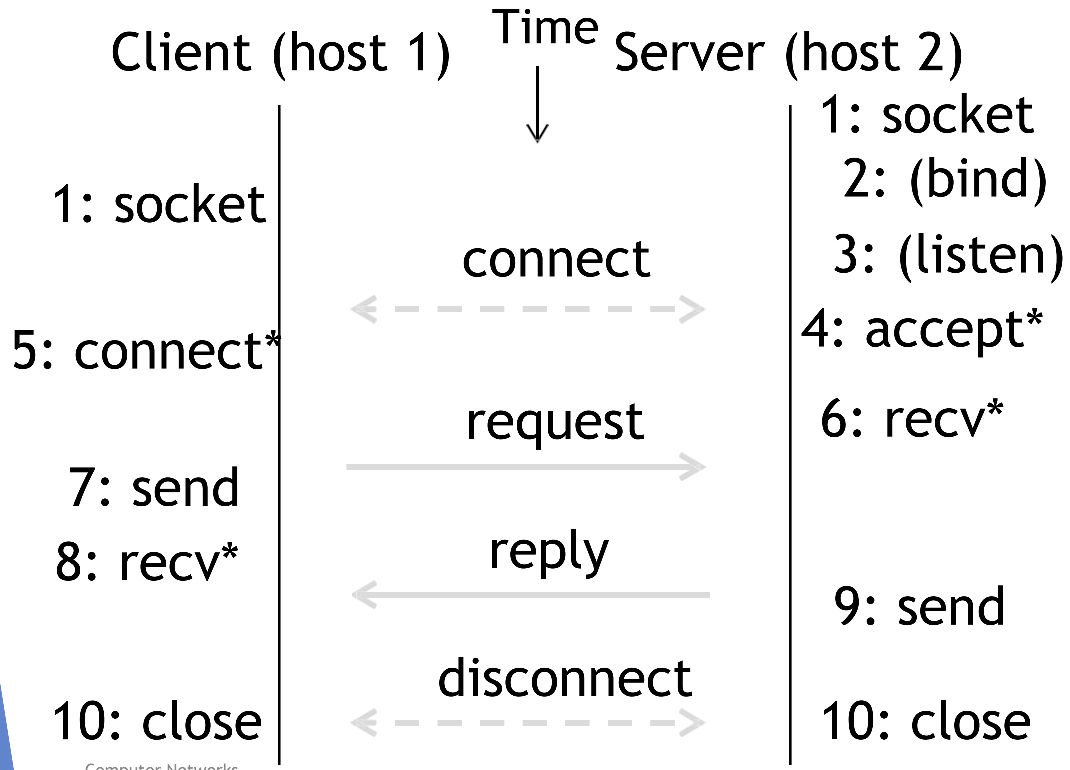
3

disconnect

4

4

Using Sockets (3)



Client (host 1)

Time

Server (host 2)

1: socket

2: (bind)

1: socket

3: (listen)

connect

4: accept*

5: connect*

request

6: recv*

7: send

8: recv*

reply

9: send

disconnect

10: close

10: close

*= call blocks

Client Program (outline)

```
socket()  // make socket
getaddrinfo() // server and port name
          // www.example.com:80
connect() // connect to server [block]
…
send()       // send request
recv()       // await reply [block]
…          // do something with data!
close()      // done, disconnect
```

Server Program (outline)

```
socket()        // make socket
getaddrinfo()   // for port on this host
bind()          // associate port with socket
listen()        // prepare to accept
                connections
accept()        // wait for a connection [block]
…
recv()          // wait for request
…
send()          // send the reply
close()         // eventually
                disconnect
```

Java Examples with Socket & ServerSocket

- Server
- Client

```
ServerSocket listener = new
    ServerSocket(9090);  try {
      while (true) {
        Socket socket =
        listener.accept();  try {
          socket.getInputStream();
        } finally {
          socket.close();
        }
      }
    }
    finally {
      listener.close();
    }
```

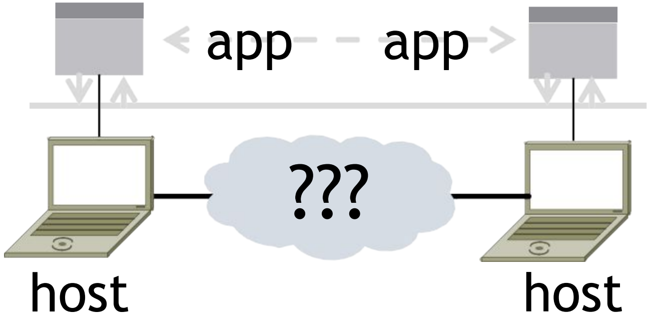
```
Socket socket = new Socket(server,
9090);  out =
    new PrintWriter(socket.getOutputStream(), true);
socket.close();
```

- http://cs.lmu.edu/~ray/notes/javanetexamples/
- https://docs.oracle.com/javase/tutorial/networking/datagrams/clientServer.html
- https://docs.oracle.com/javase/tutorial/networking/sockets/index.html

Questions?

Traceroute

- Apps talk to other apps with no real idea of what is inside the network
  - This is good! But you may be curious …
- Peeking inside the Network with Traceroute



Traceroute

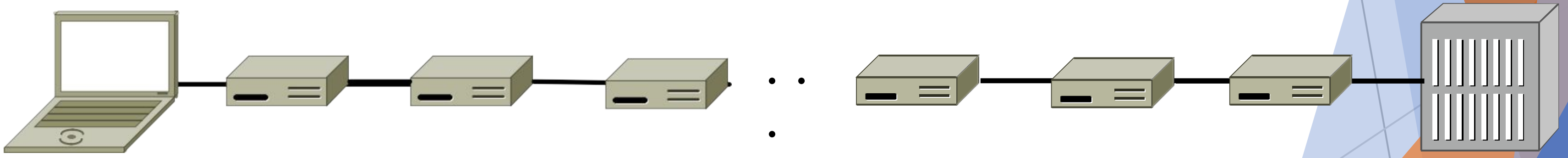
- Widely used command-line tool to let hosts peek inside the network
  - On all OSes (tracert on Windows)
  - Developed by Van Jacobson ~1987
  - Uses a network-network interface (IP) in ways we will explain later

Van Jacobson

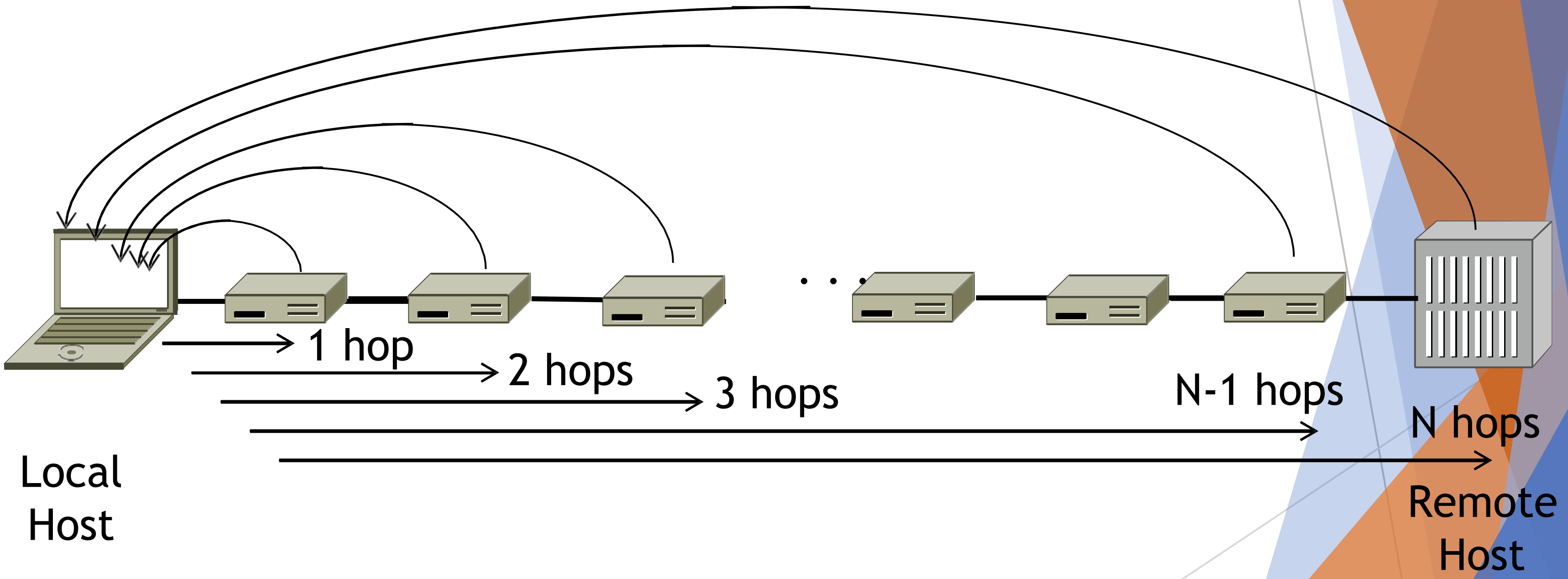
: Credit: Wikipedia (public domain)

Traceroute

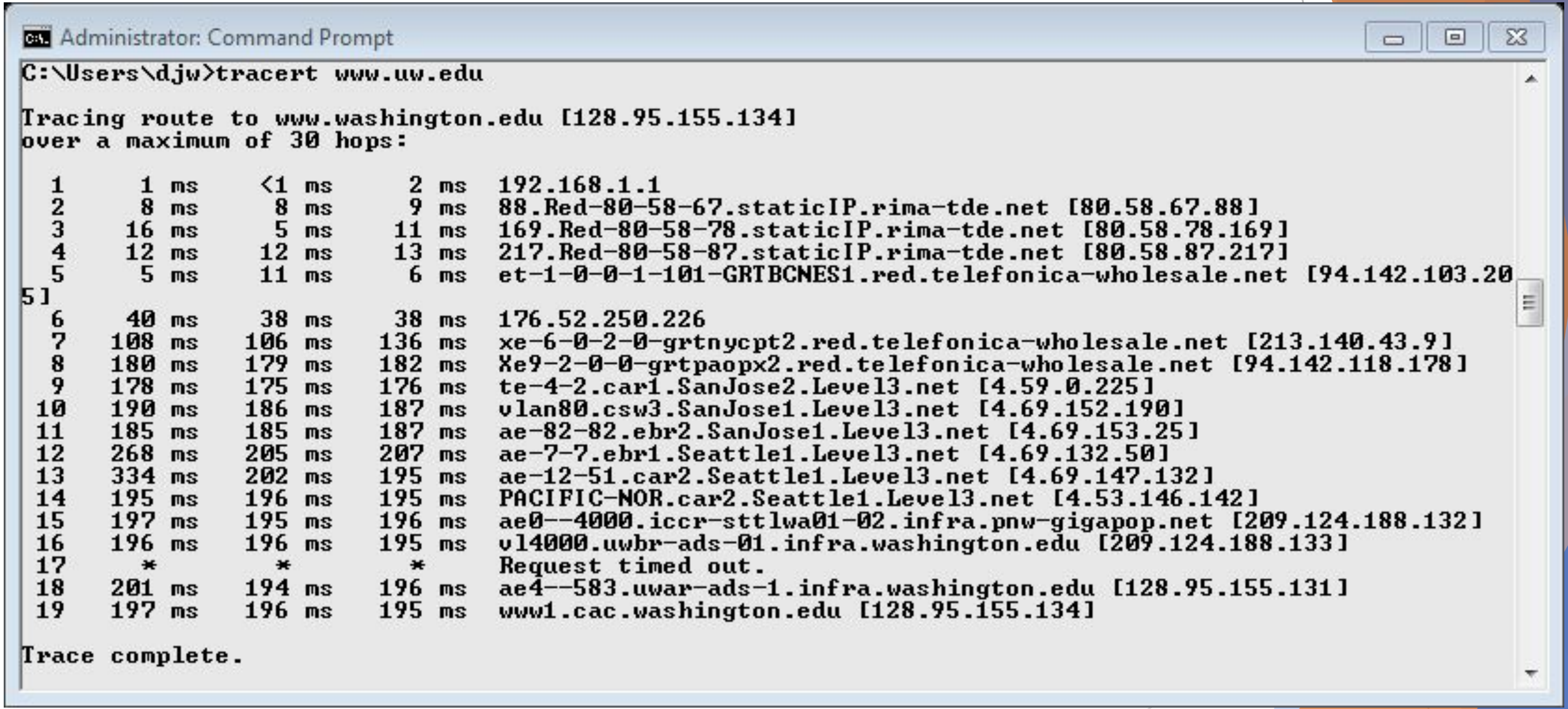
- Probes successive hops to find network path
- TTL: time-to-live



Traceroute



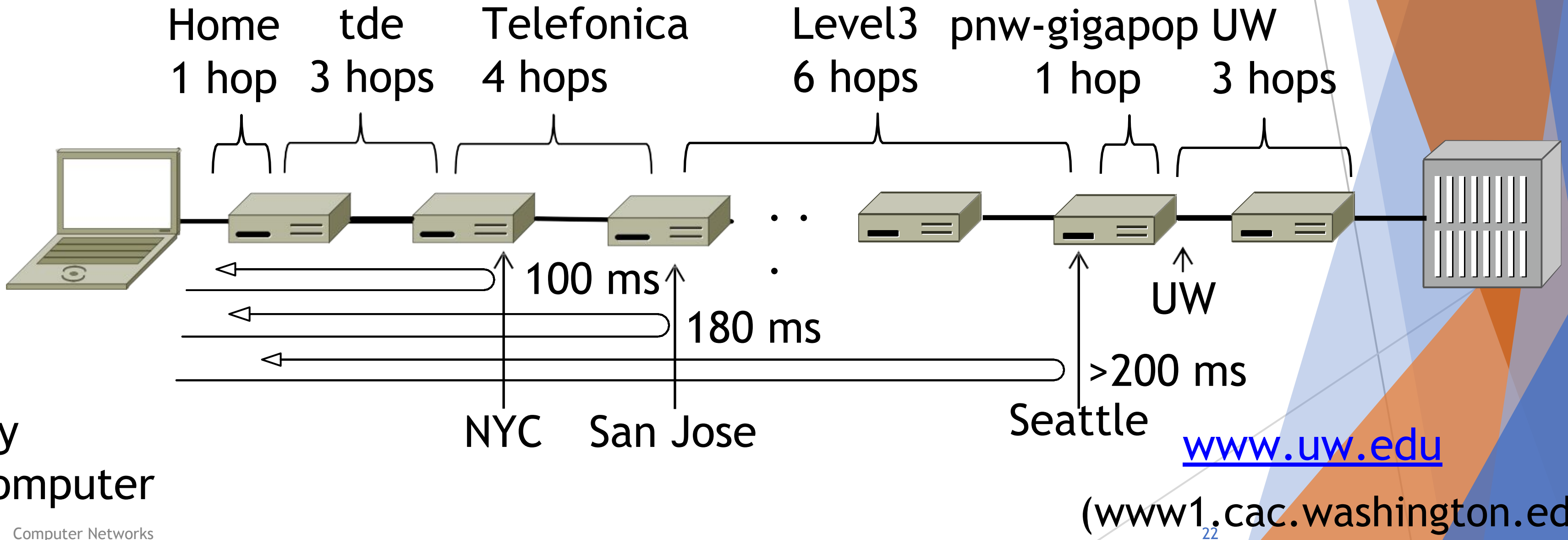
Using Traceroute



```
C:\Users\djw>tracert www.uw.edu

Tracing route to www.washington.edu [128.95.155.134]
over a maximum of 30 hops:

  1     1 ms    <1 ms     2 ms  192.168.1.1
  2     8 ms     8 ms     9 ms  88.Red-80-58-67.staticIP.rima-tde.net [80.58.67.88]
  3    16 ms     5 ms    11 ms  169.Red-80-58-78.staticIP.rima-tde.net [80.58.78.169]
  4    12 ms    12 ms    13 ms  217.Red-80-58-87.staticIP.rima-tde.net [80.58.87.217]
  5     5 ms    11 ms     6 ms  et-1-0-0-1-101-GRTBCNES1.red.telefonica-wholesale.net [94.142.103.20
5]
  6    40 ms    38 ms    38 ms  176.52.250.226
  7   108 ms   106 ms   136 ms  xe-6-0-2-0-grtnycpt2.red.telefonica-wholesale.net [213.140.43.9]
  8   180 ms   179 ms   182 ms  Xe9-2-0-0-grtpaopx2.red.telefonica-wholesale.net [94.142.118.178]
  9   178 ms   175 ms   176 ms  te-4-2.car1.SanJose2.Level3.net [4.59.0.225]
 10   190 ms   186 ms   187 ms  vlan80.csw3.SanJose1.Level3.net [4.69.152.190]
 11   185 ms   185 ms   187 ms  ae-82-82.ebr2.SanJose1.Level3.net [4.69.153.25]
 12   268 ms   205 ms   207 ms  ae-7-7.ebr1.Seattle1.Level3.net [4.69.132.50]
 13   334 ms   202 ms   195 ms  ae-12-51.car2.Seattle1.Level3.net [4.69.147.132]
 14   195 ms   196 ms   195 ms  PACIFIC-NOR.car2.Seattle1.Level3.net [4.53.146.142]
 15   197 ms   195 ms   196 ms  ae0--4000.iccr-sttlwa01-02.infra.pnw-gigapop.net [209.124.188.132]
 16   196 ms   196 ms   195 ms  v14000.uwbr-ads-01.infra.washington.edu [209.124.188.133]
 17     *        *        *     Request timed out.
 18   201 ms   194 ms   196 ms  ae4--583.uwar-ads-1.infra.washington.edu [128.95.155.131]
 19   197 ms   196 ms   195 ms  www1.cac.washington.edu [128.95.155.134]

Trace complete.
```

Using Traceroute (2)

- ISP names and places are educated guesses



END