Goals for Today’s Class

- Course overview
  - Goals of the course
  - Structure of the course
  - Learning the material
  - Programming assignments
  - Course grading
  - Academic policies

- Key concepts in data networking
  - Protocols
  - Layering
  - Resource Allocation
  - Naming
What You Learn in This Course

- **Skill:** networking and network programming
  - Socket programming
  - Designing and implementing protocols
  - Network configuration
- **Knowledge:** how the Internet works
  - IP protocol suite
  - Internet architecture
  - Applications (Web, e-mail, P2P, VoIP, ...)
- **Insight:** key concepts in networking
  - Protocols
  - Layering
  - Resource Allocation
  - Naming

Structure of the Course (1\textsuperscript{st} Half)

- **Start at the bottom**
  - Link technologies (Ethernet, wireless, ...)
- **Then study the “narrow waist” of IP**
  - IP best-effort packet-delivery service
  - IP addressing and packet forwarding
- **And how to build on top of the narrow waist**
  - Transport protocols (TCP, UDP)
  - Domain Name System (DNS)
  - Glue (ARP, DHCP, ICMP)
  - End-system security and privacy (NAT, firewalls)
- **Take a look at sockets**
  - How applications view the Internet
Structure of the Course (2nd Half)

- Then how to get the traffic from here to there
  - Internet routing architecture (the “inter” in Internet)
  - Intradomain and interdomain routing protocols
- Building applications
  - Web and content-distribution networks
  - E-mail
  - Peer-to-peer file sharing
  - Multimedia streaming and voice-over-IP
- Other approaches to building networks
  - Circuit switching (e.g., ATM, MPLS, …)
  - More on wireless networks, multicast, …

Learning the Material

- Lecture (Mirela Damian)
  - When: Th 6:15-9:00 in G87
  - Office hours: M 1:30 – 2:30, Th 4:30 – 5:30
  - Slides available online at course Web site
- Textbooks
- Online resources
  - E.g. on socket programming
Assignments

• OpNet Labs
  – Network simulation experiments
• Hands-on Labs (G291)
  – Get real experience and using real equipment
• Programming
  – Sockets assignment
  – Transport protocol assignment
• Research Project
  – Group project
  – Ends up in a peer-review presentation (graduates)
  – Topic due before April 2\textsuperscript{nd}

Grading and Schedule

• One exam (35%)
• Assignments (35% G, 40% UG)
• Labs (OpNet, hands-on) (15% G, 25% UG)
• Research project (15% G)
Policies: Write Your Own Code

Programming in an individual creative process much like composition. You must reach your own understanding of the problem and discover a path to its solution. During this time, discussions with friends are encouraged. However, when the time comes to write code that solves the problem, such discussions are no longer appropriate - the program must be your own work.

Policies: Write Your Own Code

If you have a question about how to use some feature of C, UNIX, etc., you can certainly ask your friends or the TA, but do not, under any circumstances, copy another person’s program. Letting someone copy your program or using someone else’s code in any form is a violation of academic regulations. "Using someone else’s code" includes using solutions or partial solutions to assignments provided by commercial web sites, instructors, preceptors, teaching assistants, friends, or students from any previous offering of this course or any other course.
Okay, so let’s get started… with a crash course in computer networking

Key Networking Concepts

- Protocol Architecture
- Protocol Layers
- Encapsulation
- Network Abstractions
Key Networking Concepts

- **Protocols**
  - Speaking the same language
  - Syntax and semantics

- **Layering**
  - Standing on the shoulders of giants
  - A key to managing complexity

- **Resource allocation**
  - Dividing scarce resources among competing parties
  - Memory, link bandwidth, wireless spectrum, paths, …
  - Distributed vs. centralized algorithms

- **Naming**
  - What to call computers, services, protocols, …

Sending a packet from Argon to Neon
Sending a packet from *Argon* to *Neon*

DNS: What is the IP address of "neon.tcpip-lab.edu"?
Sending a packet from Argon to Neon

DNS: The IP address of "neon.tcpip-lab.edu" is 128.143.71.21

128.143.71.21 is not on my local network. Therefore, I need to send the packet to my default gateway with address 128.143.137.1.
Sending a packet from Argon to Neon

ARP: What is the MAC address of 128.143.137.1?

ARP: The MAC address of 128.143.137.1 is 00:e0:f9:23:a8:20
Sending a packet from *Argon* to *Neon*

128.143.71.21 is on my local network. Therefore, I can send the packet directly.
Sending a packet from Argon to Neon

ARP: What is the MAC address of 128.143.71.21?

ARP: The MAC address of 128.143.137.1 is 00:20:af:03:98:28
Sending a packet from *Argon* to *Neon*

Communications Architecture

- The complexity of the communication task is reduced by using multiple protocol layers:
  - Each protocol is implemented independently
  - Each protocol is responsible for a specific subtask
  - Protocols are grouped in a hierarchy

- A structured set of protocols is called a communications architecture or protocol suite
TCP/IP Protocol Suite

- The TCP/IP protocol suite is the protocol architecture of the Internet
- The TCP/IP suite has four layers: Application, Transport, Network, and Data Link Layer
- End systems (hosts) implement all four layers. Gateways (Routers) only have the bottom two layers.

Functions of the Layers

- **Data Link Layer:**
  - **Service:** Reliable transfer of frames over a link
  - **Functions:** Framing, media access control, error checking

- **Network Layer:**
  - **Service:** Move packets from source host to destination host
  - **Functions:** Routing, addressing

- **Transport Layer:**
  - **Service:** Delivery of data between hosts
  - **Functions:** Connection establishment/termination, error control, flow control

- **Application Layer:**
  - **Service:** Application specific (delivery of email, retrieval of HTML documents, reliable transfer of file)
  - **Functions:** Application specific
TCP/IP Suite and OSI Reference Model

The TCP/IP protocol stack does not define the lower layers of a complete protocol stack.

Assignment of Protocols to Layers

- ping
- HTTP
- Telnet
- FTP
- DNS
- SNMP
- TCP
- UDP
- ICMP
- IGMP
- DHCP
- ARP
- Ethernet
- RIP
- PIM
- OSPF
- Network Layer
- Data Link Layer
- Transport Layer
- Application Layer
The Internet Protocol Graph

The waist facilitates interoperability

Alternative View of the Internet Architecture

- Applications are free to bypass TCP/UDP
- May use IP or the Network protocols directly
Layered Communications

• A layer entity can only communicate with:
  1. a peer layer entity using a Peer Protocol
  2. adjacent layers to provide services and to receive services

Service Primitives

Communication services are invoked via function calls. The functions are called **service primitives**
Communication services are invoked via function calls. The functions are called service primitives.
**Recall:** A layer N+1 entity sees the lower layers only as a service provider.
Service Primitives

**Recall:** A layer N+1 entity sees the lower layers only as a service provider

![Diagram showing N+1 Layer Entity, N+1 Layer Peer Protocol, and Service Provider]

Layers in the Example

![Diagram showing HTTP, TCP, IP, Ethernet, and their protocols with example IP addresses]
Layers in the Example
Layers in the Example

Establish a connection to 128.143.71.21 at port 80

Open TCP connection to 128.143.71.21 port 80
Layers in the Example

HTTP
TCP
IP
Ethernet

Send a datagram (which contains a connection request) to 128.143.71.21

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Layers in the Example

HTTP
TCP
IP
Ethernet

Send IP datagram to 128.143.71.21

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21
Layers in the Example

HTTP
TCP
IP
Ethernet

Send the datagram to 128.143.137.1
argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Layers in the Example

HTTP
TCP
IP
Ethernet

Send Ethernet frame to 00:e0:f9:23:a8:20
argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21
Layers in the Example

HTTP
TCP
IP
Ethernet
argon.tcpip-lab.edu
128.143.137.144

HTTP
TCP
IP
Ethernet
router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

HTTP
TCP
IP
Ethernet
router137.tcpip-lab.edu
128.143.71.1

HTTP
TCP
IP
Ethernet
neon.tcpip-lab.edu
128.143.71.21

Frame is an IP datagram

Send IP data-gram to
128.143.71.21
Layers in the Example

HTTP
TCP
IP
Ethernet

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Send the datagram to 128.143.7.21

Layers in the Example

HTTP
TCP
IP
Ethernet

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Send Ethernet frame to 00:20:af:03:98:28
Layers in the Example

HTTP
TCP
IP
Ethernet

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Frame is an IP datagram

Layers in the Example

HTTP
TCP
IP
Ethernet

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:f9:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

IP datagram is a TCP segment for port 80
Layers in the Example

HTTP
TCP
IP
Ethernet

argon.tcpip-lab.edu
128.143.137.144

router71.tcpip-lab.edu
128.143.137.1
00:e0:ff:23:a8:20

router137.tcpip-lab.edu
128.143.71.1

neon.tcpip-lab.edu
128.143.71.21

Layers and Services

• **Service provided by TCP to HTTP:**
  – reliable transmission of data over a logical connection

• **Service provided by IP to TCP:**
  – unreliable transmission of IP datagrams across an IP network

• **Service provided by Ethernet to IP:**
  – transmission of a frame across an Ethernet segment

• **Other services:**
  – DNS: translation between domain names and IP addresses
  – ARP: Translation between IP addresses and MAC addresses
Encapsulation and Demultiplexing

- As data is moving down the protocol stack, each protocol is adding layer-specific control information.

![Diagram showing encapsulation and demultiplexing](image)

Encapsulation and Demultiplexing in our Example

- Let us look in detail at the Ethernet frame between Argon and router, which contains the TCP connection request to Neon.

- This is the frame in hexadecimal notation.

```
00e0 f923 a820 00a0 2471 e444 0800 4500
002c 9d08 4000 8006 8bff 808f 8990 808f
4715 065b 0050 0009 465b 0000 0000 6002
2000 598e 0000 0204 05b4
```
Sending a packet from *Argon* to *Neon*
Encapsulation and Demultiplexing

destination address
source address
type

6 bytes

Ethernet Header IP Header TCP Header Application data Ethernet Trailer

4 bytes

CRC

Encapsulation and Demultiplexing: Ethernet Header

00:e0:79:23:a8:20
0:a0:24:71:e4:44
0x0800

6 bytes

Ethernet Header IP Header TCP Header Application data Ethernet Trailer

4 bytes

CRC
Encapsulation and Demultiplexing:
IP Header

32 bits

version (4 bits) | headlen | DSCP | ECN | Total Length (in bytes) (16 bits)

Identification (15 bits) | Flags (3 bits) | Fragment Offset (13 bits)

TTL Time-to-Live (8 bits) | Protocol (8 bits) | Header Checksum (16 bits)

Source IP address (32 bits)

Destination IP address (32 bits)

Encapsulation and Demultiplexing:
IP Header

32 bits

<table>
<thead>
<tr>
<th>0x4</th>
<th>0x0</th>
<th>0x0</th>
<th>0x0</th>
<th>44h</th>
</tr>
</thead>
<tbody>
<tr>
<td>9d08</td>
<td>010</td>
<td>0000000000000000000000000000000000000000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>128h</th>
<th>0x06</th>
<th>8bff</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.143.137.144</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128.143.74.21</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ethernet Header | IP Header | TCP Header | Application data | Ethernet Trailer
Encapsulation and Demultiplexing: TCP Header

- 32 bits

<table>
<thead>
<tr>
<th>Source Port Number</th>
<th>Destination Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence number (32 bits)</td>
<td></td>
</tr>
<tr>
<td>Acknowledgement number (32 bits)</td>
<td></td>
</tr>
<tr>
<td>Flags</td>
<td>window size</td>
</tr>
<tr>
<td>TCP checksum</td>
<td>urgent pointer</td>
</tr>
<tr>
<td>Max. segment size</td>
<td></td>
</tr>
</tbody>
</table>

Option: maximum segment size

Ethernet Header | IP Header | TCP Header | Application data | Ethernet Trailer

Encapsulation and Demultiplexing: TCP Header

- 32 bits

<table>
<thead>
<tr>
<th>1627_{16}</th>
<th>88_{10}</th>
</tr>
</thead>
<tbody>
<tr>
<td>607835_{10}</td>
<td></td>
</tr>
<tr>
<td>0_{10}</td>
<td></td>
</tr>
<tr>
<td>6_{10}</td>
<td>000000_{2}</td>
</tr>
<tr>
<td>0x598e</td>
<td>0000_{2}</td>
</tr>
<tr>
<td>2_{10}</td>
<td>1460_{10}</td>
</tr>
</tbody>
</table>

Ethernet Header | IP Header | TCP Header | Application data | Ethernet Trailer

Ethernet frame
Encapsulation and Demultiplexing: Application data

```
00e0 f923 a820
00a0 2471 e444
0800 4500 002c
9d08 4000 8006
8bff 808f 9990
808f 4715 065b
0050 0009 465b
0000 0000 6002
2000 598e 0000
0204 05b4
```

```
Ethernet Header  IP Header  TCP Header  Application data  Ethernet Trailer
```

---

Encapsulation and Demultiplexing: Application data

```
00e0 f923 a820
00a0 2471 e444
0800 4500 002c
9d08 4000 8006
8bff 808f 9990
808f 4715 065b
0050 0009 465b
0000 0000 6002
2000 598e 0000
0204 05b4
```

No Application Data in this frame

```
Ethernet Header  IP Header  TCP Header  Application data  Ethernet Trailer
```

---

38
Different Views of Networking

- Different Layers of the protocol stack have a different view of the network. This is HTTP’s and TCP’s view of the network.

Network View of IP Protocol
Network View of Ethernet

- Ethernet’s view of the network

![Diagram of Ethernet network](image)

Protocol Standardization

- Communicating hosts speaking the same protocol
  - Standardization to enable multiple implementations
  - Or, the same folks have to write all the software
- Standardization: Internet Engineering Task Force
  - Based on working groups that focus on specific issues
  - Produces “Request For Comments” (RFCs)
    - Promoted to standards via rough consensus and running code
    - E.g., RFC 1945 on “HyperText Transfer Protocol – HTTP/1.0”
  - IETF Web site is http://www.ietf.org
What if the Data Doesn’t Fit?

Problem: Packet size
- On Ethernet, max IP packet is 1500 bytes
- Typical Web page is 10 kbytes

Solution: Split the data across multiple packets

GET index.html
What if the Data gets Dropped?

Problem: Lost Data

Solution: Timeout and Retransmit
What if the Data is Out of Order?

Problem: Out of Order

ml → inde → x.ht → GET

GET x.htindeml

Solution: Add Sequence Numbers

ml 4 → inde 2 → x.ht 3 → GET 1

GET index.html
Resource Allocation: Queues

- Sharing access to limited resources
  - E.g., a link with fixed service rate
- Simplest case: first-in-first out queue
  - Serve packets in the order they arrive
  - When busy, store arriving packets in a buffer
  - Drop packets when the queue is full

Resource Allocation: Congestion Control

- What if too many folks are sending data?
  - Senders agree to slow down their sending rates
  - ... in response to their packets getting dropped
- The essence of TCP congestion control
  - Key to preventing congestion collapse of the Internet
Transmission Control Protocol

- **Flow control: window-based**
  - Sender limits number of outstanding bytes (window size)
  - Receiver window ensures data does not overflow receiver

- **Congestion control: adapting to packet losses**
  - *Congestion window* tries to avoid overloading the network (increase with successful delivery, decrease with loss)
  - TCP connection starts with small initial congestion window

Conclusions

- **Course objectives**
  - Network programming, how the Internet works, key concepts in networking

- **Key concepts in networking**
  - Protocols, layers, naming, resource allocation