Bradfield
SCHOOL *of* COMPUTER SCIENCE

Computer Networking

January 2021 with Oz Nova

You can't gaze in the crystal ball and see the future. What the Internet is going to be in the future is what society makes it.

Bob Khan, co-creator of TCP/IP

The internet—together with all of the applications it supports, like the web—has become one of the most important forces for technological progress of our era. The highest impact technology companies were once semiconductor companies like Intel, later software companies like Microsoft,[1] and are now internet companies: Facebook, Google and Amazon among the largest in the world. But for all of our reliance on computer networks, few people can boast a strong understanding of them.

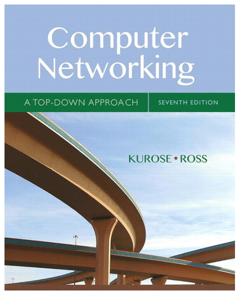
[1] Arguably, Microsoft's continued success is due to their ability to transition into providing their software as services over the internet. Similarly, while Apple makes most of its money from selling iPhones, some argue that their future relevance is predicated on their ability to make money delivering services.

In this course we'll cover the key protocols that enable the internet, the web, and other applications of computer networking. This should empower you not only to make the most of present networking technologies but also to help build the future.

Recommended Resources

Given the amount of content we cover in the course, it's important that you complete the prework for each class, and additionally valuable if you can explore some of the "further study" references.

Our recommended text for the course is Kurose and Ross Computer Networking: A Top-Down Approach (referred to as K&R below). The content changes very little from one edition to another, so the sixth edition (or even fifth, at a pinch) is sufficient and likely much cheaper to buy second hand. The chapter references below are for the seventh edition.



For those who prefer video lectures, we suggest Stanford's Introduction to Computer Networking available via Lagunita, their MOOC platform. Corresponding sections of the course are referred to as "ICN" below.

There is also a worthwhile series of very short videos on miscellaneous networking topics by a former University of Buffalo lecturer. These may be a good place to look for a definition or explanation of a specific topic or protocol.

Finally, some of our suggested further study resources are protocol specifications (particularly RFCs)[3] and papers. These can be incredibly valuable, as networking is a dynamic field where practice often precedes theory and shorter non-academic material better conveys the motivation for a given set of decisions.

1. The Big Picture: Layers of Protocols

Tuesday, 5 January 2021

Our first class explores the history of computer networks and presents the layered model of network protocols. [4] Due to the layered model, application layer protocols like DNS, HTTP, and SMTP can be built in isolation from concerns like routing and reliable delivery. Similarly, TCP can focus on transmission and IP can focus on routing without having to consider the application data.

Pre-class Work

Please read chapter 1 of K&R, particularly 1.1 ("What Is The Internet") 1.5 ("Protocol Layers") and 1.7 ("History of Computer Networking"); or alternatively watch ICN sections 1.0-1.8. Our objective in this section is to build a more complete mental model of the various pieces of computer networks, and how they all interact to facilitate something like "The Internet".

Before you start reading, write an answer to the age old interview question: "what happens when I type www.google.com into my browser's URL bar and hit enter?". Be as detailed as possible, and make note of the topics where you feel you may be lacking detail. We will use this question as a baseline for your current understanding of the topic, and return to it later.

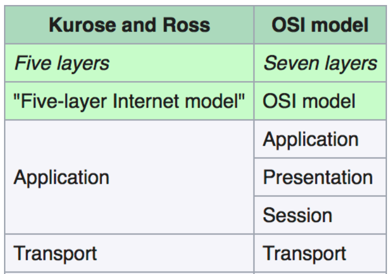
As you work through the prework material, also ask yourself the

[3] Internet protocol standards typically arise as RFCs (Requests For Comment), which are a kind of semi-formal memo that may or may not become adopted by browser and kernel developers, router designers and other stakeholders. This relatively decentralized, consensus-driven "standards" process is arguably a key to the success of the Internet. Interestingly, the humble name "request for comments" reflects the fact that many of the early contributors to what became the Internet were graduate students who wished to avoid alienating their more senior peers.

[4] We use a simplified 5-layer version of the OSI 7-layer networking model, as does the K&R book.

The OSI model was published in 1984, approximately 10 year after the release of TCP/IP. It was intended not only to serve as a rationalized separation of concerns for network protocols, but to replace them all with alternatives designed to be cleaner and better designed than the relatively makeshift TCP/IP and friends. As is often the case with design-by-committee protocols, they never caught on.

What did stick though, due to historical use in textbooks, is the OSI 7 layer model, even if not all 7 are quite reflected in the protocols in use today. As a consequence, many people will use variants of the 7 layer model, as we do. See this Wikipedia section for a comparison of various models, and the article OSI: The Internet That Wasn't for a fascinating account of the battle between OSI and TCP/IP.



Kurose and Ross	OSI model
Five layers	*Seven layers*
"Five-layer Internet model"	OSI model
Application	Application
	Presentation
	Session
Transport	Transport

following questions:

What hardware and software enables what we call the Internet? Which protocol layers do each of these pieces of hardware and software interact with?

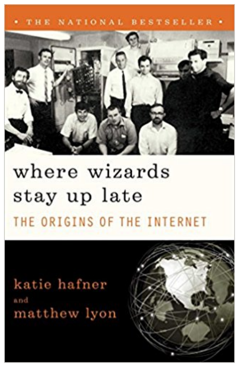
How much does a protocol at one layer need to know about those at others? To what degree do they interact? How does the data corresponding to each protocol combine when ultimately transmitted?

Further Resources

An interactive self-paced version of this class is available at csprimer.com/networks/overview (as a pre-release preview for current networking students).

OSI: The Internet That Wasn't is a fascinating account of the standardization effort that left us with little more than the OSI reference model. The reference model itself was first written up in 1980 by Hubert Zimmermann as the paper OSI Model—The ISO Model of Architecture for Open Systems Interconnection.

For more on the history of the Internet, see the website Brief History of the Internet and the excellent book *Where Wizards Stay Up Late: The Origins Of The Internet*.



2. Lab: Parsing Captured Network Packets

Friday, 8 January 2021

Now that we have a grasp of the layered model of network protocols, we can apply this to our first lab-style class, which involves parsing a file containing captured network packets, and reconstituting a downloaded image.

Pre-class Work

In preparation for class, please review any new concepts from last class, and read the in-class exercise instructions.

In-class Exercise

In class, we will focus on making the layering approach more concrete by writing a program to parse a series of HTTP packets, which are themselves embedded in a series of TCP segments (embedded in IP datagrams [embedded in Ethernet frames]).

Further Resources

An interactive self-paced version of this class is available at csprimer.com/networks/pcap (as a pre-release preview for current networking students).

If you're new to working with binary data, and using JavaScript, you might enjoy the Bytewiser workshopper.

3. The Domain Name System

Tuesday, 12 January 2021

There are perhaps 100 or more application layer protocols; they are common enough that you may be involved in developing one at some point.[6] While we cannot cover them all, our examination of DNS in this lesson—as well as our later exploration of HTTP/1.1 and HTTP/2—should give us a good overall idea of the key concerns and design decisions behind any other application layer protocols.

Many are surprised to learn that DNS is an application layer protocol, and that to resolve a URL into an IP address we must make a network request to a DNS server (identified by its IP address!).[7] This lesson will elucidate how this is possible, and give us a stronger understanding of one of the major services that makes the Internet user friendly.

[6] Consider that in April 2001, Bram Cohen quit his job and started designing what would become BitTorrent. He released the first implementation 2 months later. As a more recent example, Juan Benet has designed a number of peer-to-peer protocols, many of which are maintained now by the well-funded Protocol Labs. While it's unusual to link to a jobs page in a course outline, we do so to make the point that you could apply now for a job that would involve designing and implementing application-layer protocols.

[7] DNS was first designed in 1983. Astonishingly, until its deployment, the mapping between names and addresses was maintained in a simple text file HOSTS.TXT on a server at SRI International (previously Stanford Research Institute) from which every host on the network downloaded a copy! Here is the HOSTS.TXT file from early 1983.

Pre-class Work

For some background on DNS, please read the "DNS—The Internet's Directory Service" chapter of K&R, or watch sections 5.8-5.10 of ICN.

While you're reading, ask yourself these questions:

What are the differences between local, root, TLD and authoritative servers in DNS? What are their roles?

If you query a root DNS server for an A record for "www.google.com", what will it send you in return?[8]

[8] `nslookup` and `dig` are very useful tools, and can be used to check your answer here. See their respective man pages for more.

When might a client prefer a "recursive" over an "iterative" DNS query?

What does DNS have in common with other application layer protocols you know, and how does it differ?

Do you feel that the distributed but hierarchical structure of DNS was a good choice?

Further Resources

An interactive self-paced version of this class is available at csprimer.com/networks/dns (as a pre-release preview for current networking students).

The most illustrative RFCs related to DNS are RFC 1034 and RFC 1035.

For an approachable and fascinating history of DNS, see the paper Development of the Domain Name System.

4. Lab: Write a DNS client

Friday, 15 January 2021

In order to send a DNS message—just like any network application message—we must know the protocol well enough to construct the message correctly. But that's not quite enough! We also require a means of transmitting the message, by having our operating system wrap it in the necessary transport, network and link layer meta data, and to actually transmit the resulting frames over the network. The standard mechanism for achieving this is the socket.[9]

[9] To be specific, it is the "BSD socket" or "Berkeley socket", a specific abstraction around the operating system's networking functionality that was first developed in 1983 and has since become not just a de facto standard but an official (POSIX) one.

In this class, we'll build a simple DNS client, both as our first introduction to socket programming, and to further improve our understanding of this important application layer protocol.

Pre-class Work

For some background on socket programming, please read the sections of Beej's Guide to Network Programming Using Internet Sockets titled "what is a socket" and "system calls or bust". Test your understanding by writing a short program that listens on port 80 and prints what it receives. Use your language of choice for this, but try to use the lowest level socket API your language's standard library provides, rather than some wrapper—the functions you use should resemble the set of system calls described in the article.

Please also read the in-class exercise instructions.

In-class Exercise

In class, you will implement a simple DNS client.

Further Resources

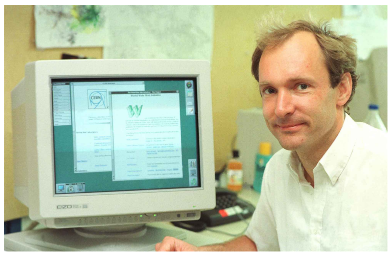
An interactive self-paced version of this class is available at csprimer.com/networks/dns-client (as a pre-release preview for current networking students).

5. Web Protocols

Tuesday, 19 January 2021

HTTP has become a ubiquitous protocol employed well beyond its original purpose of delivering HTML pages to browsers.[10] Given the commonality and usefulness of the protocol, we will spend this class digging into some of the details of the HTTP specification as well as discussing the most important differences between HTTP/1.1 and HTTP/2. We will cover the details of HTTP/2 and other modern and future web protocols in our final class.

[10] Despite the ubiquity of the World Wide Web, few non-engineers recognize that it is quite distinct from the Internet, and developed over 2 decades later. The Internet grew directly out of ARPANET, which commenced development in the late 1960s, whereas Sir Tim Berners-Lee conceived of and first prototyped the Web—including its key protocols HTTP and HTML—in 1989. Pictured below is Berners-Lee at CERN in 1994, demonstrating web pages.



Pre-class Work

Please read the "Web and HTTP" section of chapter 2 of K&R, or sections 5.5-5.6 of ICN. Focus on defining the role of the application layer, and especially the responsibilities of HTTP.

While you're reading, ask yourself these questions:

HTTP/1.0 and HTTP/1.1 are “text based protocols” whereas HTTP/2 is a “binary protocol”. What’s the difference?

HTTP specifies using TCP as the underlying transport layer protocol. Why might the designers of HTTP have specified a particular transport layer protocol, and why pick TCP? Does this mean that it’s not possible to send HTTP data over UDP? What is an example of an HTTP header that effectively requires TCP?

Which HTTP headers might have significant performance implications?

How are cookies transmitted? What is their format? In loading a single web page, how many cookies might be sent?

Further Resources

An interactive self-paced version of this class is available at csprimer.com/networks/http (as a pre-release preview for current networking students).

HTTP/1.0, HTTP/1.1 and HTTP/2 are specified in RFCs 1945, 2616 and 7540 respectively. If you want to really understand these protocols, you should go to the source of truth.

For a general overview of the features in HTTP/2, see the website HTTP/2 FAQs. For a great video on its performance implications, see Yesterday’s perf best-practices are today’s HTTP/2 anti-patterns.

6. Lab: Writing a Caching HTTP Proxy

Friday, 22 January 2021

While many popular web frameworks make it possible to write web applications without a thorough understanding of HTTP, it always helps to be able to go deeper, particularly when developing infrastructural components like load balancers and caches. This lab-style class is designed to develop your understanding of web protocols, and also provide another opportunity to work directly with sockets.

Pre-class Work

In preparation, please review any unfamiliar concepts from our last class, and read the in-class exercise instructions.

In-class Exercise

The exercise for this class will be to write a simple HTTP proxy server that can cache queries and/or load balance.

7. Reliable Data Delivery

Tuesday, 26 January 2021

This class cover the critically important transport layer, and its two major protocols TCP and UDP. We will consider the challenges involved in guaranteeing delivery over an unreliable network, and the implications of how TCP achieves this and its other features such as segmentation. Once we have a good understanding of the trade offs made by TCP, we will have gained an appreciation for when we may wish to ignore TCP entirely and just use UDP instead.[11]

[11] UDP is in essence "not using TCP". David P. Reed is often considered the "designer" of UDP, but in his own words:

"Actually, UDP was 'un-designed' by me and others. By this I mean that UDP was the final expression of a process that today we would call 'factoring' an overcomplex design… UDP was actually 'designed' in 30 minutes on a blackboard when we decided pull the original TCP protocol apart into TCP and IP, and created UDP on top of IP as an alternative for multiplexing and demultiplexing IP datagrams."

Pre-class Work

Please read chapter 3 of K&R ("Transport Layer")—particularly the sections covering UDP and TCP specifically—or alternatively work through unit 2 of ICN ("Transport").

While you're reading, ask yourself these questions:

How are UDP and TCP different? How do their respective headers reflect their differences?

What kinds of applications might prefer UDP to TCP? Why?

What information is exchanged during TCP's "Three way hand-shake"? How are these pieces of information used?

What is a checksum and how can it be used to detect corrupt data?

What is the most minimal version of reliable transport you can think of?

How do participants in a TCP connection recover from a dropped packet?

Finally, because we've added a layer of understanding, try answering our interview question again in more detail: Assuming the use of TCP, what happens when I type www.google.com into my browser's URL bar? Be as detailed as possible, especially regarding TCP's responsibilities.

Further Resources

For historical resources, see Vint Cerf and Bob Kahn's early paper A Protocol for Packet Network Intercommunication. The first RFCs for TCP and UDP are RFC 793 and RFC 768 respectively.

The Computer History Museum has produced a series of video-based oral histories of early pioneers of computing. Bob Kahn's interview by Vint Cerf (part 1 and part 2) is 5.5hrs in total but thoroughly fascinating.

For a quick (2min) animation of the Go-Back-N sliding window protocol, see this video.

8. Routing, and the Structure of the Internet

Friday, 29 January 2021

This class covers the Internet protocol itself, the overall problems of addressing and routing, and how routing tables are maintained without manual intervention.

While static routing is still occasionally used,[12] it's generally much more preferable to use dynamic routing algorithms, so as to account for network change such as unreachable or misbehaving routers. We will explore how some of these protocols effectively perform graph search algorithms to update each host's view of part of the network graph.

[12] For a lighthearted example of static routing, first type `bad.horse` into your browsers URL bar, then open a terminal and type `traceroute bad.horse`.

We'll also explore how the individually managed subnetworks of the Internet known as "autonomous systems" route traffic between one another through the Border Gateway Protocol (BGP).

Pre-class Work

Please read chapters 4 of K&R (“The Network Layer: Data Plane”) with a particular focus on “The Internet Protocol”. If time permits, please also spend some time working through chapter 5 (“The Network Layer: Control Plane”). If using ICN, please work through unit 6 (“Routing”).

While you’re reading, ask yourself these questions:

What is a routing table, and how do routers use them?

Why was IPv6 needed? How do its responsibilities relate to those of Network Address Translation?

What are ”subnets” and ”autonomous systems”, and how are IP addresses used in this context?

What information do individual routers need from other nodes in the system?

How do routers handle packets destined for hosts on their own network? How is this different from how routers handle packets destined for other networks?

How are routing tables updated?

What is ICMP and what is it used for?

Further Resources

The original RFCs for IPv4 and IPv6 are RFC 791 and RFC 2460 respectively. ICMPv6 is defined in RFC 4443.

9. Lab: Implementing Traceroute

Tuesday, 2 February 2021

Traceroute is a useful little program for exploring the structure of the internet. In this class, we’ll further our understanding of the network layer by building a traceroute clone. This will also be an excuse to work directly with raw sockets, which will allow us to send ICMP messages.

Pre-class Work

In preparation for class, please review the Wikipedia ICMP entry as well as the traceroute man page. Please also read the in-class exercise instructions.

In-class Exercise

In class you will implement a simple clone of the traceroute program, using raw sockets and ICMP ping requests.

10. The Link Layer and Local Area Networks

Friday, 5 February 2021

Having proceeded from applications down through the transport and network layers, we still need to account for how data is reliably transmitted via physical media like wires and air. This class covers Ethernet and 802.11 (Wi-Fi) as well as MAC, ARP and other important protocols at the link layer.

Pre-class Work

Please work through as much as you can of chapter 6 of K&R ("The Link Layer and LANs"), particularly "Introduction to the Link Layer", "Multiple Access Links and Protocols", and "Switched Local Area Networks". If using ICN, please work through unit 7 ("Lower Layers").

While you're reading, ask yourself these questions:

What is the difference between a router and a link layer switch?

Do link layer devices know about other devices that are not on their local network?

What services might or might not be offered at the link layer?

What are the unique challenges faced at the link layer?

What are some problems that arise from using a "broadcast medium" and how do link layer devices manage those problems?

Further Resources

The IEEE 802.11 standard is not light reading, but may be interesting to skim through and as a resource. For ARP, the corresponding RFC is RFC 826.

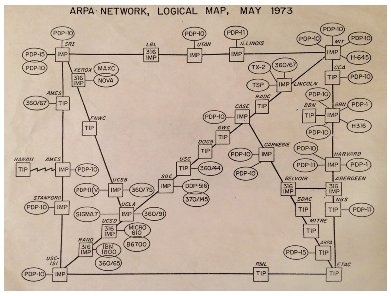
For an account of the early development of Ethernet, see this oral history of Bob Metcalfe, its inventor.

11. Network Security

Tuesday, 9 February 2021

Security was not a first class concern of the pioneers of computer networking,[13] and so is not at the core of its most important protocols so much as retrofitted around them. With time and the increasing popularity of networked applications, it's become clear that security is now a critically important consideration.

[13] Consider that many of the core ideas of TCP were decided upon in 1973, when there were only a few dozen nodes on ARPANET, most of them universities. Pictured below is a logical map of the network at the time.



This class covers the means by which we can secure Internet communication, with a particular focus on public key encryption, SSL/TLS, and the role of certificate authorities. A tremendous amount of work has been invested in these schemes, with the remarkable result that we can now send data securely over a network that we know to be untrustworthy.

Pre-class Work

Please read K&R chapters 8.2 ("Principles of Cryptography") 8.4 ("End-Point Authentication") and 8.6 ("Securing TCP Connections"), or complete Unit 8 of ICN ("Security").

While you're reading, ask yourself these questions:

What is public key encryption, how is it used in TLS, and why is it so important in networking contexts?

What information is exchanged in the TLS handshake? What is the purpose of the cipher-suite exchange? What is the purpose of the hash function?

Why does TLS use public key encryption *and* symmetric key encryption? What are the trade-offs at play? Why not use public/private key encryption for the whole transaction?

What role do certificate authorities play in TLS?

How is public key encryption implemented?

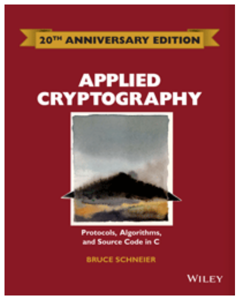
In-class Exercise

If time permits in class, you will implement the RSA algorithm from scratch, to share an encrypted message with a classmate.

Further Resources

For a gentle introduction to some of the concepts covered in class, see this Bradfield article about TLS and signed data.

For an excellent, approachable introduction to cryptography, see the Khan Academy cryptography course. For a more rigorous treatment, we suggest the book *Applied Cryptography* by Bruce Schneier.



There are a number of excellent games and challenges that aim to teach network security concepts. We particularly recommend OverTheWire and the Cryptopals challenges.

Also illustrative as usual are RFCs. TLS is expressed in RFC 5246.

12. Future Network Protocols

Friday, 12 February 2021

Our final class is an opportunity to dive deeper into network protocols that are currently being implemented or designed. We will pay particular attention to protocols intended to improve the web, such as HTTP/2 and HTTP/3 (HTTP over QUIC).

Pre-class Work

In preparation for class, please skim through two short online books: http2 explained and http3 explained, focusing on the new features of each protocol. Test your understanding by asking yourself what problems these protocols were each designed to solve, and how they were addressed at a high level.

Further Resources

For more information on QUIC, see the video QUIC: next generation multiplexed transport over UDP and draft protocol: QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2.