COMPUTER NETWORKING

9/E

A TOP-DOWN APPROACH

KUROSE • ROSS

Preface

Welcome to the ninth edition of *Computer Networking: A Top-Down Approach*. Since the publication of the first edition years ago, our book has been adopted for use at thousands of colleges and universities, translated into languages, and used by many hundreds of thousands of students and practitioners worldwide. We've heard from many of these readers and have been overwhelmed by the positive response.

What's New in the Ninth Edition?

We think one important reason for this success has been that our book continues to offer a fresh and timely approach to computer networking instruction. We've made changes in this ninth edition, but we've also kept unchanged what we believe (and the instructors and students who have used our book have confirmed) to be the most important aspects of this book: its top-down approach, its focus on the Internet, its modern treatment of computer networking, its attention to both principles and practice, and its accessible style and approach. Nevertheless, the ninth edition has been revised and updated substantially.

Readers of earlier editions of our book may recall that in moving from the sixth to the seventh edition, we deepened our coverage of the network layer, expanding material which had been previously covered in a single chapter into a new chapter focused on the so-called "data plane" component of the network layer (Chapter 4) and a new chapter focused on the network layer's "control plane" (Chapter 5). That change turned out to be prescient, as software-defined networking (SDN), one of the most important and exciting advances in networking in decades, has been rapidly adopted in practice—so much so that it's hard to imagine an introduction to

modern computer networking that doesn't cover SDN. In moving from the seventh to the eighth edition, we significantly updated the application layer (Chapter 2), introduced a modern treatment of the transport layer in Chapter 3 by covering the (then new) QUIC protocol, and significantly expanded our coverage of cellular networks in Chapter 7. These changes have also proved prescient as the past five years since the eighth edition have seen important advances in both application-layer infrastructure and transport-layer protocols. Additionally, with the introduction of , cellular networks, and the software-i-zation of cellular networks, the pace of change in wireless and mobile networking has only accelerated since the eighth edition.

The most important changes in this ninth edition are the following:

- **Chapter 1** has been updated to reflect the ever-expanding reach and use of the Internet and its applications. A recent trend now reflected in both Chapters 1 and 2 is the continuing evolution of the Internet's structure in which user applications communicate with numerous small-scale servers over a multi-tiered Internet topology to a flatter structure in which applications increasingly communicate with servers located close to end users in datacenters operated by "hyperscalars" or in large content distribution networks.

- **Chapter 2** has been updated to cover the new protocol and QUIC, while leaving the principles of end-end reliability and congestion control in to Chapter 3. Chapter 2 also contains new material on content distribution networks and updated coverage of "over-the-top" streaming services, both reflecting the ever-increasing prevalence of application services provided by scalable, distributed application-level infrastructure.

- **In Chapter 3,** our foundational treatment of reliable data transfer remains unchanged but our treatment of congestion control principles has been updated to balance coverage of traditional loss-based congestion control techniques with coverage of newer congestion control protocols (BBR in particular) that perform congestion control using measured connection delay and throughput. We also complete our treatment of QUIC (begun in Chapter 2), having studied the principles of reliable data transfer, connection management, and congestion control in this chapter.

- **Chapter 4** covers the network-layer data plane. In addition to updates throughout, Section 4.5 has been restructured to highlight the architectural principles of the Internet, while still covering middleboxes.

- **Chapter 5,** which covers the network-layer control plane, is updated throughout and provides new material on SDN, including Google's Orion network control plane as a case study and adding new material on the "software-ization" of network control. Our separate treatment of network management, and SNMP in particular, has been streamlined.

- **Chapter 6** has minor updates throughout but remains relatively unchanged. The chapter contains a new section on virtual extended LANs and an updated treatment of datacenter networks, reflecting changes over the past five years.

In addition to these changes, we've also updated many sections throughout the book to reflect changes across the breadth of networking. In some cases, we have also retired material from the previous edition. As always, material that has been retired from the printed text can always be found on our book's Companion Website and our authors' Web site.

Audience

This textbook is for a first course on computer networking. It can be used in both computer science and electrical engineering departments. In terms of programming languages, the book assumes only that the student has experience with C, , Java, or Python (and even then only in a few places). Although this book is more precise and analytical than many other introductory computer networking texts, it rarely uses any mathematical concepts that are not taught in high school. We have made a deliberate effort to avoid using any advanced calculus, probability, or stochastic process concepts (although we've included some homework problems for students with this advanced background). The book is therefore appropriate for undergraduate courses and for first-year graduate courses. It should also be useful to practitioners in the networking industry.

What Is Unique About This Textbook?

The subject of computer networking is enormously complex, involving many concepts, protocols, and technologies that are woven together in an intricate manner. To cope with this scope and complexity, many computer networking texts are often organized around the "layers" of a network architecture. With a layered organization, students can see through the complexity of computer networking—they learn about the distinct concepts and protocols in one part of the architecture while seeing the big picture of how all parts fit together. From a pedagogical perspective, our personal experience has been that such a layered approach indeed

works well. Nevertheless, we have found that the traditional approach of teaching —bottom up; that is, from the physical layer toward the application layer—is not the best approach for a modern course on computer networking.

A Top-Down Approach

Our book broke new ground years ago by treating networking in a top-down manner—that is, by beginning at the application layer and working its way down toward the physical layer. The feedback we received from teachers and students alike have confirmed that this top-down approach has many advantages and does indeed work well pedagogically. First, it places emphasis on the application layer (a "high growth area" in networking). Indeed, many of the recent revolutions in computer networking—including the Web, and media streaming—have taken place at the application layer. An early emphasis on application-layer issues differs from the approaches taken in most other texts, which have only a small amount of material on network applications, their requirements, application-layer paradigms (e.g., client-server and peer-to-peer), and application programming interfaces.

Second, our experience as instructors (and that of many instructors who have used this text) has been that teaching networking applications near the beginning of the course is a powerful motivational tool. Students are thrilled to learn about how networking applications work—applications such as e-mail, streaming video, and the Web, which most students use on a daily basis. Once a student understands the applications, the student can then understand the network services needed to support these applications. The student can then, in turn, examine the various ways in which such services might be provided and implemented in the lower layers. Covering applications early thus provides motivation for the remainder of the text.

Third, a top-down approach enables instructors to introduce network application development at an early stage. Students not only see how popular applications and protocols work, but also learn how easy it is to create their own network applications and application-layer protocols. With the top-down approach, students get early exposure to the notions of socket programming, service models, and protocols—important concepts that resurface in all subsequent layers. By providing

socket programming examples in Python, we highlight the central ideas without confusing students with complex code. Undergraduates in electrical engineering and computer science will have no difficulty following the Python code.

An Internet Focus

Although we dropped the phrase "Featuring the Internet" from the title of this book with the fourth edition, this doesn't mean that we dropped our focus on the Internet. Indeed, nothing could be further from the case! Instead, since the Internet has become so pervasive, we felt that any networking textbook must have a significant focus on the Internet, and thus this phrase was somewhat unnecessary. We continue to use the Internet's architecture and protocols as primary vehicles for studying fundamental computer networking concepts. Of course, we also include concepts and protocols from other network architectures. But the spotlight is clearly on the Internet, a fact reflected in our organizing the book around the Internet's five-layer architecture: the application, transport, network, link, and physical layers.

Another benefit of spotlighting the Internet is that most computer science and electrical engineering students are eager to learn about the Internet and its protocols. They know that the Internet has been a revolutionary and disruptive technology and can see that it is profoundly changing our world. Given the enormous relevance of the Internet, students are naturally curious about what is "under the hood." Thus, it is easy for an instructor to get students excited about basic principles when using the Internet as the guiding focus.

Teaching Networking Principles

Two of the unique features of the book—its top-down approach and its focus on the Internet—have appeared in the titles of our book. If we could have squeezed a *third* phrase into the subtitle, it would have contained the word *principles*. The field of networking is now mature enough that a number of fundamentally important issues can be identified. For example, in the transport layer, the fundamental issues include reliable communication over an unreliable network layer, connection

establishment/ teardown and handshaking, congestion and flow control, and multiplexing. Three fundamentally important network-layer issues are determining "good" paths between two routers, interconnecting a large number of heterogeneous networks, and managing the complexity of a modern network. In the link layer, a fundamental problem is sharing a multiple access channel. In network security, techniques for providing confidentiality, authentication, and message integrity are all based on cryptographic fundamentals. This text identifies fundamental networking issues and studies approaches toward addressing these issues. The student learning these principles will gain knowledge with a long "shelf life"—long after many of today's network standards and protocols have become obsolete, the principles they embody will remain important and relevant. We believe that the combination of using the Internet to get the student's foot in the door and then emphasizing fundamental issues and solution approaches will allow the student to quickly understand just about any networking technology.

Student Resources

Student resources are available on the Companion Website (CW) at http://pearson.com/cs-resources, as well as on our authors' website, http://gaia.cs.umass.edu/kurose_ross. Resources include:

- *Additional technical material*. As we have added new material in each edition of our book, we've had to remove coverage of some existing topics to keep the book at manageable length. Material that appeared in earlier editions of the text is still of interest, and thus can be found on the book's Website.
- *Programming assignments*. The Website also provides a number of detailed programming assignments, which include building a multithreaded Web server, building an e-mail client with a GUI interface, programming the sender and receiver sides of a reliable data transport protocol, programming a distributed routing algorithm, and more.

- *Wireshark labs*. One's understanding of network protocols can be greatly deepened by seeing them in action. The Website provides numerous Wireshark assignments that enable students to actually observe the sequence of messages exchanged between two protocol entities. The Website includes separate Wire- shark labs on HTTP, DNS, TCP, UDP, IP, ICMP, Ethernet, ARP, WiFi, TLS and on tracing all protocols involved in satisfying a request to fetch a Web page. We'll continue to add new labs over time.

- *Publisher's Web site*. The Companion Website contains VideoNotes— video presentations of important topics throughout the book done by the authors, as well as walkthroughs of solutions to problems similar to those at the end of the chapter. We've seeded the Website with VideoNotes and online problems for Chapters 1 through 5. As in earlier editions, the Website contains the interactive animations that illustrate many key networking concepts. Professors can integrate these interactive features into their lectures or use them as mini labs.

- *Authors' Website*. In addition to the Companion Website, the authors maintain a public Website, http://gaia.cs.umass.edu/kurose_ross, which contains open resources including video lectures corresponding to sections in this book, additional interactive material for students, and mirrors of publicly available material from the Publisher's Website, such as PowerPoint slides and Wireshark lab materials. Students have told us that the short videos covering the majority of book's sections, knowledge checks and interactive exercises that create (and present solutions for) problems similar to selected end-of-chapter problems have been particularly valuable.

Pedagogical Features

We have each been teaching computer networking for more than years. Together, we bring more than years of teaching experience to this text, during

which time we have taught many thousands of students. We have also been active researchers in computer networking during this time. (In fact, Jim and Keith first met each other as master's students in a computer networking course taught by Mischa Schwartz in 1979 at Columbia University.) We think all this gives us a good perspective on where networking has been and where it is likely to go in the future. Nevertheless, we have resisted temptations to bias the material in this book toward our own pet research projects. We figure you can visit our personal Web sites if you are interested in our research. Thus, this book is about modern computer networking—it is about contemporary protocols and technologies as well as the underlying principles behind these protocols and technologies. We also believe that learning (and teaching!) about networking can be fun. A sense of humor, use of analogies, and real-world examples in this book will hopefully make this material more fun.

Supplements for Instructors

We provide a complete supplements package to aid instructors in teaching this course. This material can be accessed from Pearson's Instructor Resource Center (http://www.pearsonhighered.com/irc). There is also additional instructional material, including Wireshark labs, programming assignments, LMS packages (compatible with Canvas, Moodle, Blackboard and D2L), video lectures, and a mirror of Powerpoint slides, on the authors' web site.

- ***PowerPoint® slides***. We provide PowerPoint slides for all eight chapters. The slides have been updated with this ninth edition. The slides cover each chapter in detail. They use graphics and animations (rather than relying only on monotonous text bullets) to make the slides interesting and visually appealing. We provide the original PowerPoint slides so you can customize them to best suit your own teaching needs. Some of these slides have been contributed by other instructors who have taught from our book.

- ***Solutions***. We provide a solutions manual for the homework problems in the text, programming assignments, and Wireshark labs. We have also developed LMS modules with questions that are similar to (but different than) the review questions and end-of chapter problems, as well as for the Wireshark Labs. These modules have been designed to work with most major LMSs, including Canvas, Moodle, Blackboard and D2L. See the authors' Web site at http://gaia.cs.umass.edu/kurose_ross/interactive for more information.

Chapter Dependencies

The first chapter of this text presents a self-contained overview of computer networking. Introducing many key concepts and terminology, this chapter sets the stage for the rest of the book. All of the other chapters directly depend on this first chapter. After completing Chapter 1, we recommend instructors cover Chapters 2 through 6 in sequence, following our top-down philosophy. Each of these five chapters leverages material from the preceding chapters. After completing the first six chapters, the instructor has quite a bit of flexibility. There are no interdependencies among the last two chapters, so they can be taught in any order. However, the last two chapters depends on the material in the first six chapters. Many instructors first teach the first six chapters and then teach one of the last two chapters for "dessert."

One Final Note: We'd Love to Hear from You

We encourage students and instructors to e-mail us with any comments they might have about our book. It's been absolutely *wonderful* for us to hear from so many

instructors and students from around the world about our first seven editions. We've incorporated many of these suggestions into later editions of the book. We also encourage instructors to send us new homework problems (and solutions) that would complement the current homework problems. We'll post these on the instructor-only portion of the Web site. We also encourage instructors and students to create new interactive animations that illustrate the concepts and protocols in this book. If you have an animation that you think would be appropriate for this text, please submit it to us. If the animation (including notation and terminology) is appropriate, we'll be happy to include it on the text's Web site, with an appropriate reference to the animation's authors.

So, as the saying goes, "Keep those cards and letters coming!" Seriously, please *do* continue to send us interesting URLs, point out typos, disagree with any of our claims, and tell us what works and what doesn't work. Tell us what you think should or shouldn't be included in the next edition. Send your e-mail to kurose@cs.umass.edu and keithwross@nyu.edu.

Acknowledgments

Since we began writing this book in 1996, many people have given us invaluable help and have been influential in shaping our thoughts on how to best organize and teach a networking course. We want to say A BIG THANKS to everyone who has helped us from the earliest first drafts of this book, up to this ninth edition. We are also very thankful to the thousands of readers from around the world—students, faculty, practitioners—who have sent us thoughts and comments on earlier editions of the book and suggestions for future editions of the book. Special thanks go out to:

Al Aho (Columbia University)

Hisham Al-Mubaid (University of Houston-Clear Lake)

Pratima Akkunoor (Arizona State University)

Paul Amer (University of Delaware)

Shamiul Azom (Arizona State University)

Lichun Bao (University of California at Irvine)

Paul Barford (University of Wisconsin)

Bobby Bhattacharjee (University of Maryland)

Steven Bellovin (Columbia University)

Pravin Bhagwat (Wibhu)

Supratik Bhattacharyya (Amazon)

Ernst Biersack (Eurécom Institute)

Shahid Bokhari (University of Engineering & Technology, Lahore)

Jean Bolot (Orange)

Phuthipong Bovornkeeratiroj (University of Massachusetts, Amherst)

Daniel Brushteyn (former University of Pennsylvania student)

Ken Calvert (University of Kentucky)

Evandro Cantu (Federal University of Santa Catarina)

Jeff Case (SNMP Research International)

Jeff Chaltas (Sprint)

Kaushik Chowdhury (University of Texas)

Vinton Cerf (Google)

Byung Kyu Choi (Michigan Technological University)

Bram Cohen (BitTorrent, Inc.)

Constantine Coutras (Pace University)

Lorrie Cranor, Carnegie-Mellon University

John Daigle (University of Mississippi)

Edmundo A. de Souza e Silva (Federal University of Rio de Janeiro)

Philippe Decuetos (former Eurecom Institute student)

Antonio de la Oliva Delgado (U. Carlos III Madrid)

Christophe Diot (Google)

Prithula Dhunghel (Akamai)

Deborah Estrin (Cornell University)

Michalis Faloutsos (University of California at Riverside)

Serge Fdida (Sorbonne University)

Wu-chi Feng (Oregon Graduate Institute)

Sally Floyd (ICIR, University of California at Berkeley)

Paul Francis (Max Planck Institute)

David Fullager (Netflix)

Lixin Gao (University of Massachusetts)

JJ Garcia-Luna-Aceves (University of California at Santa Cruz)

Mario Gerla (University of California at Los Angeles)

David Goodman (NYU-Poly)

Yang Guo (Alcatel/Lucent Bell Labs)

Tim Griffin (Cambridge University)

Max Hailperin (Gustavus Adolphus College)

Bruce Harvey (Florida A&M University, Florida State University)

Michael Hatfield (University of Alaska)

Carl Hauser (Washington State University)

Rachelle Heller (George Washington University)

Phillipp Hoschka (INRIA/W3C)

Wen Hsin (Park University)

Albert Huang (former University of Pennsylvania student)

Cheng Huang (Microsoft Research)

Esther A. Hughes (Virginia Commonwealth University)

Soad Ibrahim (Old Dominion University)

Van Jacobson (Google)

Pinak Jain (former NYU-Poly student)

Jobin James (University of California at Riverside)

Sugih Jamin (University of Michigan)

Atharva Kale (University of Massachusetts)

Shivkumar Kalyanaraman (IBM Research, India)

Jussi Kangasharju (University of Helsinki)

Sneha Kasera (University of Utah)

Wolfgang Kellerer (Technical University of Munich)

Parviz Kermani (U. Massachusetts)

Hyojin Kim (former University of Pennsylvania student)

Leonard Kleinrock (University of California at Los Angeles)

Ed Knightly (Rice University)

David Kotz (Dartmouth College)

Beshan Kulapala (Arizona State University)

Rakesh Kumar (Bloomberg)

Miguel A. Labrador (University of South Florida)

Simon Lam (University of Texas)

Steve Lai (Ohio State University)

Tom LaPorta (Penn State University)

Tim-Berners Lee (World Wide Web Consortium)

Arnaud Legout (INRIA)

Lee Leitner (Drexel University)

Brian Levine (University of Massachusetts)

Chunchun Li (former NYU-Poly student) Yong Liu (NYU-Poly)

William Liang (former University of Pennsylvania student)

Nahid Majd (California State University San Marcos)

Willis Marti (Texas A&M University)

Nick McKeown (Stanford University)

Josh McKinzie (Park University)

Deep Medhi (University of Missouri, Kansas City)

Bob Metcalfe (International Data Group)

Vishal Misra (Columbia University)

Sue Moon (KAIST)

Paul Mockapetris (Nominum)

Jenni Moyer (Comcast)

Erich Nahum (IBM Research)

Katherine Nolin (Bridgewater State University)

Deng Pan (Florida International University)

Christos Papadopoulos (Colorado Sate University)

Guru Parulkar (Open Networking Foundation)

Craig Partridge (Colorado State University)

Radia Perlman (Dell EMC)

Jitendra Padhye (Microsoft Research)

Vern Paxson (University of California at Berkeley)

Kevin Phillips (Sprint)

George Polyzos (Athens University of Economics and Business)

Sriram Rajagopalan (Arizona State University)

Krishnamurthy Raghunandan (Rutgers University)

Ramachandran Ramjee (Microsoft Research)

Ken Reek (Rochester Institute of Technology)

Martin Reisslein (Arizona State University)

Jennifer Rexford (Princeton University)

Leon Reznik (Rochester Institute of Technology)

Pablo Rodrigez (Telefonica)

Sumit Roy (University of Washington)

Catherine Rosenberg (University of Waterloo)

Dan Rubenstein (Columbia University)

Avi Rubin (Johns Hopkins University)

Douglas Salane (John Jay College)

Despina Saparilla (Cisco Systems)

John Schanz (Comcast)

Henning Schulzrinne (Columbia University)

Mischa Schwartz (Columbia University)

Pable Serrano Yanez-Mingot (U. Carlos III Madrid)

Ardash Sethi (University of Delaware)

Harish Sethu (Drexel University)

K. Sam Shanmugan (University of Kansas)

Prashant Shenoy (University of Massachusetts)

Clay Shields (Georgetown University)

Subin Shrestra (University of Pennsylvania)

Bojie Shu (former NYU-Poly student)

Mihail L. Sichitiu (NC State University)

Peter Steenkiste (Carnegie Mellon University)

Tatsuya Suda (University of California at Irvine)

Kin Sun Tam (State University of New York at Albany)

Violet Syrotiuk (Arizona State University)

Don Towsley (University of Massachusetts)

David Turner (California State University, San Bernardino)

Amin Vahdat (Google)

Nitin Vaidya (Georgetown University)

Michele Weigle (Clemson University)

David Wetherall (Google)

Ira Winston (University of Pennsylvania)

Di Wu (Sun Yat-sen University)

Shirley Wynn (former NYU-Poly student)

Raj Yavatkar (Google)

Yechiam Yemini (Columbia University)

Dian Yu (former NYU-Shanghai student)

Ming Yu (State University of New York at Binghamton)

Ellen Zegura (Georgia Institute of Technology)

Honggang Zhang (Suffolk University)

Hui Zhang (Carnegie Mellon University)

Lixia Zhang (University of California at Los Angeles)

Meng Zhang (former NYU-Poly student)

Shuchun Zhang (former University of Pennsylvania student)

Xiaodong Zhang (Ohio State University)

ZhiLi Zhang (University of Minnesota)

Phil Zimmermann (independent consultant)

Mike Zink (University of Massachusetts)

Cliff C. Zou (University of Central Florida)

We also want to thank the entire Pearson team—in particular, Tracy Johnson and Erin Sullivan—who have made this ninth edition possible (and who have put up with two very finicky authors who seem congenitally unable to meet deadlines!). Thanks also to artists, Janet Theurer and Patrice Rossi Calkin, for their work on the beautiful figures in earlier editions of our book, and to Abhijeet Gope for his team's wonderful production work on this edition. Finally, a most special thanks go to our previous editors at Addison-Wesley and Pearson—Matt Goldstein, Michael Hirsch, and Susan Hartman. This book would not be what it is (and may well not have been at all) without their graceful management, constant encouragement, nearly infinite patience, good humor, and perseverance.

Table of Contents

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

Chapter 7

Chapter 8

CHAPTER 1
Computer Networks and the Internet

Today's Internet is arguably the largest engineered system ever created by mankind, with hundreds of millions of connected computers, communication links, and switches; with billions of users who connect via laptops, tablets, and smartphones; and with an array of new Internet-connected "things" including game consoles, surveillance systems, watches, eye glasses, thermostats, robots, cars, and even diapers. Given that the Internet is so large and has so many diverse components and uses, is there any hope of understanding how it works? Are there guiding principles and structure that can provide a foundation for understanding such an amazingly large and complex system? And if so, is it possible that it actually could be both interesting *and* fun to learn about computer networks? Fortunately, the answer to all of these questions is a resounding YES! Indeed, it's our aim in this book to provide you with a modern introduction to the dynamic field of computer networking, giving you the principles and practical insights you'll need to understand not only today's networks, but tomorrow's as well.

This first chapter presents a broad overview of computer networking and the Internet. Our goal here is to paint a broad picture and set the context for the rest of this book, to see the forest through the trees. We'll cover a lot of ground in this introductory chapter and discuss a lot of the pieces of a computer network, without losing sight of the big picture.

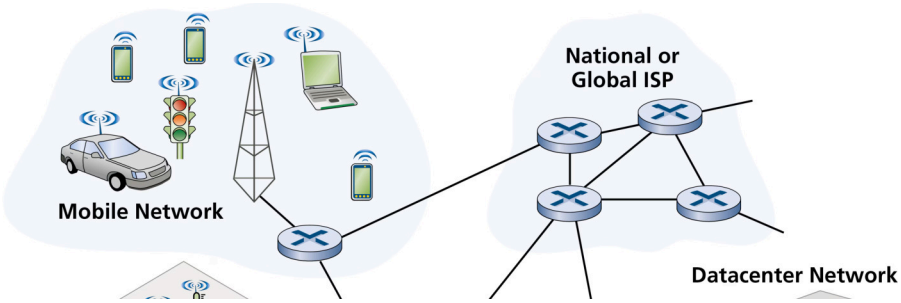
We'll structure our overview of computer networks in this chapter as follows. After introducing some basic terminology and concepts, we'll first examine the basic hardware and software components that make up a network. We'll begin at the network's edge and look at the end systems and network applications running in the network. We'll then explore the core of a computer network, examining the links and the switches that transport data, as well as the access networks and physical media that connect end systems to the network core. We'll learn that the Internet is a network of networks, and we'll learn how these networks connect with each other.

After having completed this overview of the edge and core of a computer network, we'll take the broader and more abstract view in the second half of this chapter. We'll examine delay, loss, and throughput of data in a computer network and provide simple quantitative models for end-to-end throughput and delay: models that take into account transmission, propagation, and queuing delays. We'll then introduce some of the key architectural principles in computer networking, namely, protocol layering and service models. We'll also learn that computer networks are vulnerable to many different types of attacks; we'll survey some of these attacks and consider how computer networks can be made more secure. Finally, we'll close this chapter with a brief history of computer networking.

1.1: What Is the Internet?

In this book, we'll use the public Internet, a specific computer network, as our principal vehicle for discussing computer networks and their protocols. But what *is* the Internet? There are a couple of ways to answer this question. One way is to describe the nuts and bolts of the Internet, that is, the basic hardware and software components that make up the Internet. The other way is to describe the Internet in terms of a networking infrastructure that provides services to distributed applications. Let's begin with the nuts-and-bolts description, using Figure 1.1 to illustrate our discussion.

Figure 1.1: Some pieces of the Internet



1.1.1: A Nuts-and-Bolts Description

The Internet is a computer network that interconnects billions of computing devices throughout the world. Not too long ago, these computing devices were primarily traditional desktop computers, Linux workstations, and so-called servers that store and transmit information such as Web pages and e-mail messages. Increasingly, however, users connect to the Internet with smartphones and tablets—today, approximately two-thirds of world's population are active mobile Internet users [Statista Users 2024]. Furthermore, nontraditional Internet "things" such as TVs, gaming consoles, thermostats, home security systems, home appliances, watches, eye glasses, cars, traffic control systems, and more are being connected to the Internet. Indeed, the term *computer network* is beginning to sound a bit dated, given the many nontraditional devices that are being hooked up to the Internet. In Internet jargon, all of these devices are called **hosts** or **end systems**. By some estimates, there were about devices connected to the Internet in 2025, and the number will reach by 2030 [Statista Connections 2025].

End systems are connected together by a network of **communication links** and **packet switches**. We'll see in Section 1.2 that there are many types of communication links, which are made up of different types of physical media, including coaxial cable, copper wire, optical fiber, and radio spectrum. Different links can transmit data at different rates, with the **transmission rate** of a link measured in bits/second. When one end system has data to send to another end system, the sending end system segments the data and adds header bytes to each segment. The resulting packages of information, known as **packets** in the jargon of

computer networks, are then sent through the network to the destination end system, where they are reassembled into the original data.

A packet switch takes a packet arriving on one of its incoming communication links and forwards that packet on one of its outgoing communication links. Packet switches come in many shapes and flavors, but the two most prominent types in today's Internet are **routers** and **link-layer switches**. Both types of switches forward packets toward their ultimate destinations. Link-layer switches are typically used in access networks, while routers are typically used in the network core. The sequence of communication links and packet switches traversed by a packet from the sending end system to the receiving end system is known as a **route** or **path** through the network.

Packet- switched networks (which transport packets) are in many ways similar to transportation networks transporting vehicles over highways, roads and, intersections. Consider, for example, a factory that needs to move a large amount of cargo to some destination warehouse located thousands of kilometers away. At the factory, the cargo is segmented and loaded into a fleet of trucks. Each of the trucks then independently travels through the network of highways, roads, and intersections to the destination warehouse. At the destination warehouse, the cargo is unloaded and grouped with the rest of the cargo arriving from the same shipment. Thus, in many ways, packets are analogous to trucks, communication links are analogous to highways and roads, packet switches are analogous to intersections, and end systems are analogous to buildings. Just as a truck takes a path through the transportation network, a packet takes a path through a computer network.

End systems access the Internet through **Internet Service Providers (ISPs)**, including residential ISPs such as local cable or telephone companies; corporate ISPs; university ISPs; ISPs that provide WiFi access in airports, hotels, coffee shops, and other public places; and cellular data ISPs, providing mobile access to our smartphones and other devices. Each ISP is in itself a network of packet switches and communication links. ISPs provide a variety of types of network

access to the end systems, including residential broadband access such as cable modem or DSL, high-speed local area network access, and mobile wireless access. ISPs also provide Internet access to content providers, connecting servers directly to the Internet. The Internet is all about connecting end systems to each other, so the ISPs that provide access to end systems must also be interconnected. These lower-tier ISPs are thus interconnected through national and international upper-tier ISPs and these upper-tier ISPs are connected directly to each other. An upper-tier ISP consists of high-speed routers interconnected with high-speed fiber-optic links. Each ISP network, whether upper-tier or lower-tier, is managed independently, runs the IP protocol (see below), and conforms to certain naming and address conventions. We'll examine ISPs and their interconnection more closely in Section 1.3.

End systems, packet switches, and other pieces of the Internet run **protocols** that control the sending and receiving of information within the Internet. The **Transmission Control Protocol (TCP)** and the **Internet Protocol (IP)** are two of the most important protocols in the Internet. The IP protocol specifies the format of the packets that are sent and received among routers and end systems. The Internet's principal protocols are collectively known as **TCP/IP**. We'll begin looking into protocols in this introductory chapter. But that's just a start—much of this book is concerned with networking protocols!

Given the importance of protocols to the Internet, it's important that everyone agree on what each and every protocol does, so that people can create systems and products that interoperate. This is where standards come into play. **Internet standards** are developed by the Internet Engineering Task Force (IETF) [IETF 2025]. The IETF standards documents are called **requests for comments (RFCs)**. RFCs started out as general requests for comments (hence the name) to resolve network and protocol design problems that faced the precursor to the Internet [Allman 2011]. RFCs tend to be quite technical and detailed. They define protocols such as TCP, IP, HTTP (for the Web), and SMTP (for e-mail). There are currently nearly RFCs. Other bodies also specify standards for network components,

most notably for network links. The IEEE 802 LAN Standards Committee [IEEE

1.1.2: A Services Description

Our discussion above has identified many of the pieces that make up the Internet. But we can also describe the Internet from an entirely different angle—namely, as *an infrastructure that provides services to applications*. In addition to traditional applications such as e-mail and Web surfing, Internet applications include mobile smartphone and tablet applications, including Internet messaging, mapping with real-time road-traffic information, music streaming, movie and television streaming, online social media, video conferencing, multi-person games, and location-based recommendation systems. The applications are said to be **distributed applications**, since they involve multiple end systems that exchange data with each other. Importantly, Internet applications run on end systems—they do not run in the packet switches in the network core. Although packet switches facilitate the exchange of data among end systems, they are not concerned with the application that is the source or sink of data.

Let's explore a little more what we mean by an infrastructure that provides services to applications. To this end, suppose you have an exciting new idea for a distributed Internet application, one that may greatly benefit humanity or one that may simply make you rich and famous. How might you go about transforming this idea into an actual Internet application? Because applications run on end systems, you are going to need to write programs that run on the end systems. You might, for example, write your programs in Java, C, or Python. Now, because you are developing a distributed Internet application, the programs running on the different end systems will need to send data to each other. And here we get to a central issue—one that leads to the alternative way of describing the Internet as a platform for applications. How does one program running on one end system instruct the Internet to deliver data to another program running on another end system?

End systems attached to the Internet provide a **socket interface** that specifies how a program running on one end system asks the Internet infrastructure to deliver data to a specific destination program running on another end system. This Internet socket interface is a set of rules that the sending program must follow so that the Internet can deliver the data to the destination program. We'll discuss the Internet socket interface in detail in Chapter 2. For now, let's draw upon a simple analogy, one that we will frequently use in this book. Suppose Alice wants to send a letter to Bob using the postal service. Alice, of course, can't just write the letter (the data) and drop the letter out her window. Instead, the postal service requires that Alice put the letter in an envelope; write Bob's full name, address, and zip code in the center of the envelope; seal the envelope; put a stamp in the upper-right-hand corner of the envelope; and finally, drop the envelope into an official postal service mailbox. Thus, the postal service has its own "postal service interface," or set of rules, that Alice must follow to have the postal service deliver her letter to Bob. In a similar manner, the Internet has a socket interface that the program sending data must follow to have the Internet deliver the data to the program that will receive the data.

The postal service, of course, provides more than one service to its customers. It provides express delivery, reception confirmation, ordinary use, and many more services. In a similar manner, the Internet provides multiple services to its applications. When you develop an Internet application, you too must choose one of the Internet's services for your application. We'll describe the Internet's services in Chapter 2.

We have just given two descriptions of the Internet; one in terms of its hardware and software components, the other in terms of an infrastructure for providing services to distributed applications. But perhaps you are still confused as to what the Internet is. What are packet switching and TCP/IP? What are routers? What kinds of communication links are present in the Internet? What is a distributed application? How can a thermostat or weighing scale be attached to the Internet? If you feel a bit overwhelmed by all of this now, don't worry—the purpose of this book is to introduce you to both the nuts and bolts of the Internet and the principles

that govern how and why it works. We'll explain these important terms and questions in the following sections and chapters.

1.1.3: What Is a Protocol?

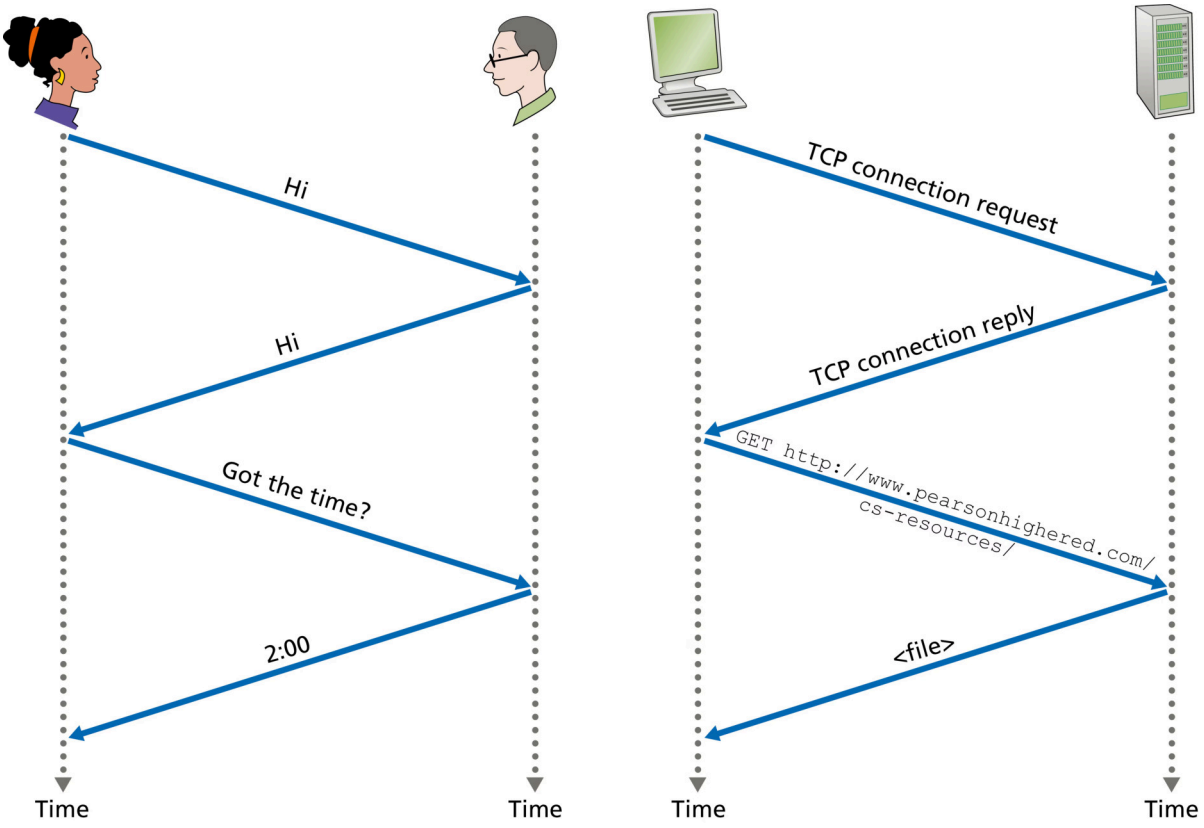
Now that we've got a bit of a feel for what the Internet is, let's consider another important buzzword in computer networking: *protocol*. What is a protocol? What does a protocol do?

A Human Analogy

It is probably easiest to understand the notion of a computer network protocol by first considering some human analogies, since we humans execute protocols all of the time. Consider what you do when you want to ask someone for the time of day. A typical exchange is shown in Figure 1.2. Human protocol (or good manners, at least) dictates that one first offer a greeting (the first "Hi" in Figure 1.2) to initiate communication with someone else. The typical response to a "Hi" is a returned "Hi" message. Implicitly, one then takes a cordial "Hi" response as an indication that one can proceed and ask for the time of day. A different response to the initial "Hi" (such as "Don't bother me!" or "I don't speak English," or some unprintable reply) might indicate an unwillingness or inability to communicate. In this case, the human protocol would be not to ask for the time of day. Sometimes one gets no response at all to a question, in which case one typically gives up asking that person for the time. Note that in our human protocol, *there are specific messages we send, and specific actions we take in response to the received reply messages or other events* (such as no reply within some given amount of time). Clearly, transmitted and received messages, and actions taken when these messages are sent or received or other events occur, play a central role in a human protocol. If people run different protocols (for example, if one person has manners but the other does not, or if one understands the concept of time and the other does not) the protocols do not interoperate and no useful work can be accomplished. The same is true in

networking—it takes two (or more) communicating entities running the same protocol in order to accomplish a task.

Figure 1.2: A human protocol and a computer network protocol



Let's consider a second human analogy. Suppose you're in a college class (a computer networking class, for example!). The teacher is droning on about protocols and you're confused. The teacher stops to ask, "Are there any questions?" (a message that is transmitted to, and received by, all students who are not sleeping). You raise your hand (transmitting an implicit message to the teacher). Your teacher acknowledges you with a smile, saying "Yes . . ." (a transmitted message encouraging you to ask your question—teachers *love* to be asked questions), and you then ask your question (that is, transmit your message to your teacher). Your teacher hears your question (receives your question message) and answers (transmits a reply to you). Once again, we see that the transmission and receipt of messages, and a set of conventional actions taken when these

messages are sent and received, are at the heart of this question-and-answer protocol.

Network Protocols

A network protocol is similar to a human protocol, except that the entities exchanging messages and taking actions are hardware or software components of some device (for example, computer, smartphone, tablet, router, or other network-capable device). All activity in the Internet that involves two or more communicating remote entities is governed by a protocol. For example, hardware-implemented protocols in two physically connected computers control the flow of bits on the "wire" between the two network interface cards; congestion-control protocols in end systems control the rate at which packets are transmitted between sender and receiver; protocols in routers determine a packet's path from source to destination. Protocols are running everywhere in the Internet, and consequently much of this book is about computer network protocols.

As an example of a computer network protocol with which you are probably familiar, consider what happens when you make a request to a Web server, that is, when you type the URL of a Web page into your Web browser. The scenario is illustrated in the right half of Figure 1.2. First, your computer will send a connection request message to the Web server and wait for a reply. The Web server will eventually receive your connection request message and return a connection reply message. Knowing that it is now OK to request the Web document, your computer then sends the name of the Web page it wants to fetch from that Web server in a GET message. Finally, the Web server returns the Web page (file) to your computer.

Given the human and networking examples above, the exchange of messages and the actions taken when these messages are sent and received are the key defining elements of a protocol:

> *A protocol defines the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.*

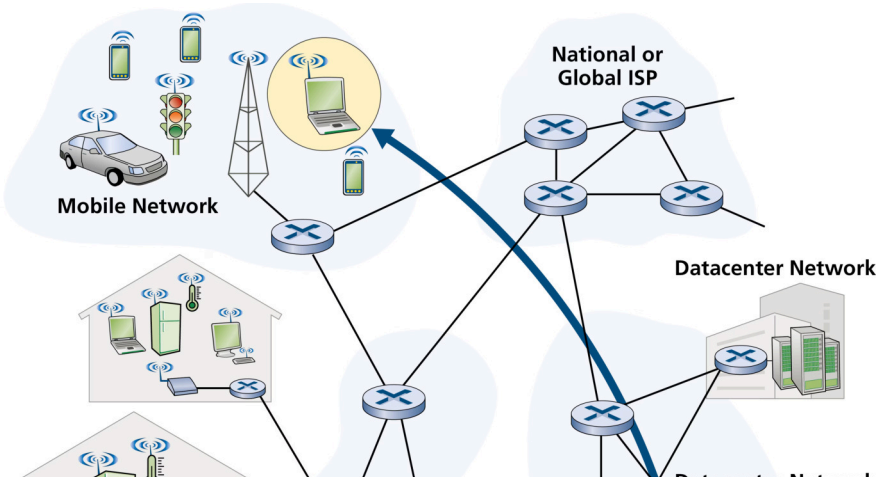
The Internet, and computer networks in general, make extensive use of protocols. Different protocols are used to accomplish different communication tasks. As you read through this book, you will learn that some protocols are simple and straightforward, while others are complex and intellectually deep. Mastering the field of computer networking is equivalent to understanding the what, why, and how of networking protocols.

1.2: The Network Edge

In the previous section, we presented a high-level overview of the Internet and networking protocols. We are now going to delve a bit more deeply into the components of the Internet. We begin in this section at the edge of the network and look at the components with which we are most familiar—namely, the computers, smartphones and other devices that we use on a daily basis. In the next section, we'll move from the network edge to the network core and examine switching and routing in computer networks.

Recall from the previous section that in computer networking jargon, the computers and other devices connected to the Internet are often referred to as end systems. They are referred to as end systems because they sit at the edge of the Internet, as shown in Figure 1.3. The Internet's end systems include desktop computers (e.g., desktop PCs, Macs, and Linux boxes), servers (e.g., Web and e-mail servers), and mobile devices (e.g., laptops, smartphones, and tablets). Furthermore, an increasing number of non-traditional "things" are being attached to the Internet as end systems (see the Case History feature).

Figure 1.3: End-system interaction



End systems are also referred to as *hosts* because they host (that is, run) application programs such as a Web browser program, a Web server program, an e-mail client program, or an e-mail server program. Throughout this book we will use the terms hosts and end systems interchangeably; that is, . Hosts are sometimes further divided into two categories: **clients** and **servers**. Informally, clients tend to be desktops, laptops, smartphones, and so on, whereas servers tend to be more powerful machines that store and distribute Web pages, stream video, relay e-mail, and so on. Today, most of the servers from which we receive search results, e-mail, Web pages, videos and mobile app content reside in large **data centers**. For example, as of late 2024, Google has data centers on four continents, [Google data centers 2024] collectively containing several million servers. Figure 1.3 includes two such data centers, and the Case History sidebar describes data centers in more detail.

Case History

Data Centers and Cloud Computing

Internet companies such as Google, Microsoft, Amazon, and Alibaba have built massive data centers, each housing tens to hundreds of thousands of hosts. These data centers are not only connected to the Internet, as shown in Figure 1.1, but also internally include complex computer networks that interconnect the datacenter's

hosts. The data centers are the engines behind the Internet applications that we use on a daily basis.

Broadly speaking, data centers serve three purposes, which we describe here in the context of Amazon for concreteness. First, they serve Amazon e-commerce pages to users, for example, pages describing products and purchase information. Second, they serve as massively parallel computing infrastructures for Amazon-specific data processing tasks. Third, they provide **cloud computing** to other companies. Indeed, today a major trend in computing is for companies to use a cloud provider such as Amazon to handle essentially *all* of their IT needs. For example, Airbnb and many other Internet-based companies do not own and manage their own data centers but instead run their entire Web-based services in the Amazon cloud, called Amazon Web Services (AWS).

The worker bees in a data center are the hosts. They serve content (e.g., Web pages and videos), store e-mails and documents, and collectively perform massively distributed computations. The hosts in data centers, called blades and resembling pizza boxes, are generally commodity hosts that include CPU, memory, and disk storage. The hosts are stacked in racks, with each rack typically having to blades. The racks are then interconnected using sophisticated and evolving data center network designs. Data center networks are discussed in greater detail in Chapter 6.

1.2.1: Access Networks

Having considered the applications and end systems at the "edge of the network," let's next consider the access network—the network that physically connects an end system to the first router (also known as the "edge router") on a path from the end system to any other distant end system. Figure 1.4 shows several types of