CONTENTS

1

Introduction

1.1. What and for whom

This is an exciting time in computer networking. The Internet is one of the most influential inventions of all time—a research experiment that, within our own lifetimes, escaped from the lab to become a global communications infrastructure. The Internet has ever-wider reach, with billions of users today and a future that promises to connect the world's entire population. We see seemingly non-stop innovation in networked devices (the Internet of Things), link technologies (fiber optics, cellular radio, microwaves, short-wavelength radio), and applications (social networks, telemedicine, finance, virtual worlds, smart factories, connected cars, environmental monitoring, power grids, blockchains). To make our lives more interesting, there is also non-stop innovation in threats to the security and privacy of Internet users. After a long series of scientific achievements [29], many network elements are now programmable, so the potential of networks to provide new services has been greatly expanded. Students who choose to study networking will have an ample supply of interesting problems, and a solution to any one of them might change the world.

Ironically, this is also a time when many people believe that the architecture of the Internet has become rigid and unchanging—"ossified" is a favorite word—and that its resistance to evolution is holding back progress. §1.2 summarizes this opinion and the Internet history that supports it.

The "architecture" of something is always a description of it. A style of description that people elevate to the prime position of an "architecture" is a representation of the most important, organizing characteristics, and the characteristics that best relate the structure of the artifact to its purpose and functions.

We believe that the Internet architecture has evolved dramatically since its early days, and that it continues to evolve. Its evolution is poorly understood because too many people rely on a description of it, called here the "classic" Internet architecture, that was valid around 1993 but is valid no longer.

Our goal in this book is to explain how the Internet has evolved up to this time, how it works now, and where it might be going. This goal demands a better style or

language of description, one that encompasses the Internet's original or "classic" architecture, its current architecture, and all its possible future architectures. Consequently, the book introduces a new style or language of description, called *compositional network architecture*, to play this crucial role.

In the next few sections, §1.2 explains the "classic" Internet architecture and the arguments for ossification. In contrast to §1.2, §1.3 presents the evidence for Internet evolution. Then §1.4 explains our purposes and approach in more detail, and also provides a preview of the remainder of the chapter.

We expect this book to be interesting to anyone involved with today's computer networks, regardless of whether they are engineers (network designers and operators), academics (faculty and graduate students), researchers, or product developers. The book is not an introductory text, however, and readers will get the most out of this book if they have the kind of general knowledge taught in an undergraduate networking course.

1.2. A brief conventional history of the Internet

1.2.1. Fundamental concepts

Before the Internet, electronic communications were carried by broadcast networks for radio and television, and by the Public Switched Telephone Network (PSTN).

The dominant characteristic of the PSTN is "circuit switching." A "circuit" is a pathway of physical resources, sufficient to carry the bandwidth of a voice call. In circuit switching, at the beginning of a call the network allocates a circuit end-to-end which is dedicated to that call. The circuit resources are not released until the call ends. A large physical link, termed a "trunk," can support many circuits, but its upper limit on circuits is inflexible. A telephone circuit is only fully utilized when both people are talking continuously at the same time. Thankfully most people listen *some* of the time, so most circuits in the PSTN are under-utilized.

Two other characteristics of the PSTN are also relevant to Internet history. First, in most countries the PSTN was a legal monopoly, and therefore under centralized control. Second, the PSTN is a person-to-person network, and the person-to-network interface is a telephone handset, with very limited capabilities. Considering that the PSTN pre-dated modern computers by about 70 years, it is not surprising that the original user interface was primitive by today's standards. But the user interface remained simple throughout the lifetime of the PSTN, primarily because the duty of a public utility (a legal monopoly) is to serve all customers reliably, whether they have the latest equipment or not. As technology improved and customers wanted more sophisticated services, the PSTN became much more complex to provide them—despite the limitations of the user interface.

In the 1970s and 1980s computers were becoming more powerful and widely available, and the Internet was conceived as a means to connect them [50]. Compared to the PSTN, two of its concepts were revolutionary:

- It replaced circuit switching with "packet switching," in which packets from many sessions share physical links, and statistical multiplexing can ensure that physical resources are well-utilized.
- It would connect independent, autonomous networks with individual designs, tailored to their own physical media and customer requirements. There would be no centralized control, only voluntary cooperation for the sake of interoperation.

This thinking led to the defining "end-to-end principle" [23, 78], which creates a sharp divide between the Internet and the user machines that it serves (Figure 1.1). The principle states that the functions of the network should be minimized, so that basic service is efficient and no one pays for services that they don't use. Many functions can be implemented in the endpoint user machines, because they are so easily programmable. Furthermore, key examples show that user endpoints are often the best place to provide functions such as reliability [78]. As an oft-mentioned test of minimality, network components should be gateways and routers without per-flow state, where a flow is a set of packets from one source to one destination.

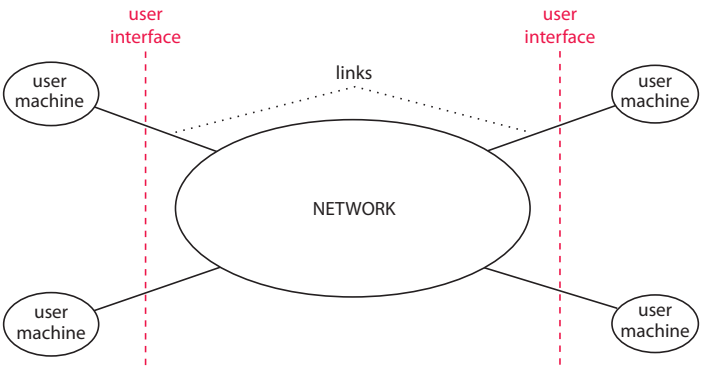

FIGURE 1.1. The user interface to a network, in the classic Internet architecture. The placement of the user interface reflects the view that user machines assume very little about the network, and the network assumes very little about the user machines. **Throughout this book, red is used to point out the most important parts of figures.**

The end-to-end principle was summarized by the slogan "smart edge, dumb core." This slogan was widely emphasized in technical discussions because of its direct contrast with the PSTN, which had developed an elaborate and expensive core, yet still served the same old handsets.

1.2.2. *The classic Internet architecture*

The fundamental concepts of the Internet soon coalesced into the ubiquitous description we call the "classic" Internet architecture. The Internet is described as organized into layers, where "layer" has its usual meaning in engineering. A layer is a set of functions within a hierarchy of other function sets (layers), such that upper layers depend upon lower layers and know enough about them to use them, but lower layers do not know about or depend upon upper layers. In short, "layers" are modules in a dependency hierarchy.

The classic architecture is standardized in the IPv4 protocol suite. It has four layers above the physical layer, each providing a distinct set of functions (see Figure 1.2): a "link" layer providing best-effort local packet delivery over heterogeneous physical networks, a "network" layer providing best-effort global packet delivery across autonomous networks, a "transport" layer providing communication services such as reliable byte streams (TCP) and datagram service (UDP), and an "application" layer. According to this description a typical packet has four headers: Ethernet (or some other physical network), IP, TCP, and HTTP (or some other application protocol). The contemporaneous Open Systems Interconnection reference model [42] is similar: it has seven layers including the physical layer, with "session" and "presentation" layers between the transport and application layers.

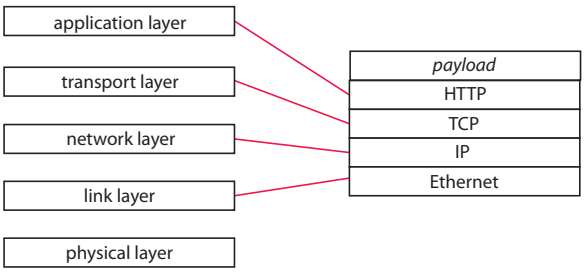

FIGURE 1.2. Layers in the classic Internet architecture on the left, with examples of corresponding packet headers on the right. Headers lower in the diagram come earlier in the actual packet.

So focused were the Internet founders on their five layers that most Internet terminology is based on them. A link-layer forwarder is a "switch," while a network-layer forwarder is a "router"—and there are no other forwarders. Data is transmitted in units called "frames" (link layer), "packets" (network layer), "segments" or "datagrams" (transport layer), and "messages" (application layer).

The last major change in IPv4 was made in 1993 [34], which is just about the time when use of the World-Wide Web began its explosive growth. Simplicity [54] made the Internet a runaway success. Its open nature allowed competition between network

providers in local markets, which further accelerated its development as a high-capacity, low-cost, global network.

1.2.3. *Success and ossification*

There is no need to dwell on the continued success of the Internet since 1993, as it is now part of everyone's lives. By the 2000s the classic Internet architecture had matured, and ever since then experts have been pointing out its shortcomings, calling for evolutionary change, and decrying its "ossification" [40]. The prevailing belief about Internet evolution is that: (i) no network provider will change its IP network unless there is immediate economic benefit, (ii) IP network changes will have no economic benefit unless they are universally deployed, and therefore (iii) the Internet architecture is extremely difficult to change.

The most obvious form of potential evolution is the adoption of IPv6, which is the next major version of the Internet Protocol. IPv6 was conceived in the late 1990s, and its standardization was completed during the 2000s. The main difference between the two versions is that IPv6 has a much larger address space (128 bits versus 32 bits). This conferred relatively little economic benefit to network providers during the 2010s, which accounts for its slow deployment during that decade.[1] Now that the available IPv4 address space is exhausted, new IP addresses are valuable, and the adoption of IPv6 is accelerating. Nevertheless, the adoption of IPv6 does not change the classic concepts as illustrated by Figures 1.1 and 1.2.

Networking researchers have long been aware of ossification and struggled against it. Most prominently, the "clean slate" movement [30] and the Future Internet Architectures program [31] advocated complete redesign of the Internet. However, the products of these efforts (e.g., [6, 86, 87, 99]) tended to be specialized in some way, and not compatible enough to merge into one unified design. One doesn't hear much about "clean slates" anymore.

Nevertheless, there is still a regular succession of papers about Internet evolution, including [3, 10, 46, 68, 72]. These papers are in general agreement that the architecture of the Internet is the classic architecture presented above (with IP referring to IPv4 and/or IPv6), and that the architecture is not evolving. A key piece of evidence is the IP protocol suite (Figure 1.3). The hierarchy in the figure is the hierarchy of dependency, with exactly the same layers as in Figure 1.2, and the figure shows some of the protocols available in each layer. Several of these papers [3, 68] emphasize that the most important part of the protocol suite is its "narrow waist" at the transport and network layers. The narrow waist is in fact resistant to change, because so many other protocols and so much network infrastructure depend on it. To summarize the ossification argument, the architecture of

1. In effect, IPv4 and IPv6 are separate and co-existing Internets. Providers of IPv6 networks are usually also providers of IPv4 networks, sometimes even sharing the same machines between the networks.

the Internet is primarily IPv4 and/or IPv6, and in particular their network and transport protocols, which have changed only slightly over three decades.

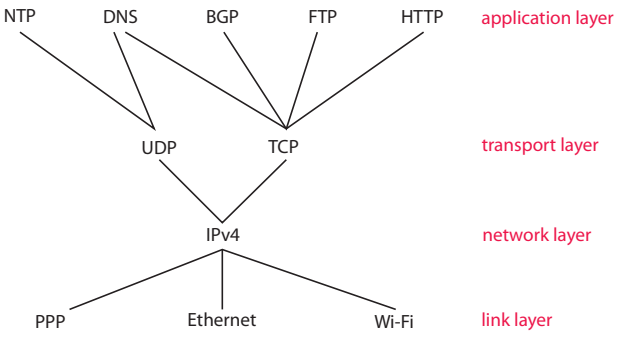

FIGURE 1.3. A sample from the Internet protocol suite.

1.2.4. *Teaching about networking*

Meanwhile, as the Internet has gained steadily in importance, academics have been teaching about networks. What goes into an undergraduate networking course? More or less, it is the classic Internet architecture, in a bottom-up or top-down order. As an electrical-engineering colleague of ours at Princeton once said, "I took a networking class in college. I fell asleep at the start of the semester with the IP header on the screen, and woke up at the end of the semester with the TCP header on the screen."

The classic Internet architecture is good for teaching because it is both relevant and concrete. Nevertheless, its use gives rise to some common complaints. Teachers want to keep their courses up-to-date, but most of the new material is an exception to the classic Internet architecture, and does not fit the conceptual framework. Students might feel there is a seemingless endless flood of details (and acronyms!), all of apparently equal significance.[2] And it is increasingly difficult to find enough room in the curriculum for everything that should be covered.

Graduate students usually learn about networking from reading research papers. We know from our own experience that, when approaching research in an unfamiliar area, it helps tremendously to have some context for it. The context helps us understand what fundamental problem the research is trying to solve, and why the problem exists in the first place. It also helps us ask the right questions while we read the paper. Sometimes context is missing because the reader is assumed to know it, but often it is missing because the classic Internet architecture provides no language in which to express it.

2. Also known as a plethora of protocols, a heap of header formats, a big bunch of boxes, and a ton of tools [74].

1.2.5. *Networking research*

Networking research has a very special asset: the largest computer system on the planet. The Internet interacts with almost every aspect of life, so everyone is a stakeholder [24] and its economic importance cannot be overestimated. It is completely decentralized, being assembled dynamically from parts without the benefit of mutual trust.

Yet networking research has its chronic problems, one of which is the lack of precise and consistent terminology. We have assigned to students a research paper in which there are four (among others) fields in packet headers, each corresponding to an important concept. In the paper, these four fields are referred to by two, three, four, and seven different terms respectively. Even worse, the set of terms used for one concept overlaps with the set of terms used for another concept![3] We point out this case not to complain about technical writing, but because it seems symptomatic of a more serious disease. In the current state of the field, almost any choice of terminology is vague, ambigous, or misleading in some way, and there is no consensus for writers and readers to rely on. Sometimes it seems as if, with no way to improve the situation, writers have simply given up.

Another chronic problem is the prevailing belief that the Internet is not evolving, and its stifling effect on research. (This explains why there has been so much interest in the research community on "clean slate" architectures.) Research that cannot be deployed easily within the classic Internet architecture gets poor reviews, discouraging investigation in all but a few approved directions.

In particular, we believe that there is far too little research interest in problems faced by software developers trying to build new Internet applications, if the problems are caused by entrenched Internet features. One of the founding principles of the Internet was enabling user innovation, and we fear it is being sacrificed because the belief in Internet ossification is too strong. For example, this book will point out simple changes that could have made IPv6 more flexible for application programmers, but may have been considered too radical, or not considered at all.

Among Internet applications, one of the most interesting is distributed systems with large replicated databases. Today the divide between distributed systems and networking has narrowed [75], but there is still little understanding of the common ground between them, and how it might be exploited to the benefit of both disciplines.

1.3. An alternative view of the Internet

In our alternative view, the Internet has a history of purposeful change rather than ossification, although the IP protocol suite has been relatively static.

3. And this is a *good* paper, referenced in this book.

1.3.1. *Past evolution*

Since the finalization of IPv4 in 1993, the Internet has met the following challenges not accommodated by the original architecture:

- Today, most networked devices are mobile.
- There has been an explosion of threats to security and privacy.
- While originally offering services that were different from the PSTN and broadcast networks, the Internet then won the contest for best overall global network. As a result, it has had to grow to support most of the world's telecommunication infrastructure and entertainment distribution.
- Enterprises now need massive computing infrastructures, often supported by cloud computing.
- In a deregulated, competitive world, network providers must control costs by allocating resources dynamically, rather than provisioning networks with static resources for peak loads.

The magnitude of these challenges raises an obvious question: How has all this been possible without evolution of the Internet architecture? The answer, which we first reported in [96], is that the Internet architecture *has* evolved—it has been evolving continually since 1993, and it is evolving now.

Two features of the current Internet show evolution most clearly: middleboxes and tunnels. A middlebox is a network component, other than a forwarder, inserted in the paths of some packets. A tunnel is a network structure with entrance and exit points, both of which are network components. At the entrance point, some packets are encapsulated in extra network headers with the address of the exit as the destination; at the exit point, the packets are decapsulated to expose their original headers, and processed as usual. Middleboxes and tunnels are not independent, as one purpose of a tunnel is to ensure that packets pass through a middlebox at the exit point of the tunnel, before they travel to their final destinations.

Neither middleboxes nor tunnels are part of the classic Internet architecture, and in fact middleboxes are deprecated by the end-to-end principle, yet today they are *everywhere*. Many networks have approximately as many middleboxes as forwarders. The networks of Internet service providers have a tangle of tunnels at multiple levels. Middleboxes and tunnels can no longer be dismissed as rare or unimportant exceptions—to the unbiased eye, this is how the Internet works.

To make this point concretely, Figure 1.4 shows the headers of a typical packet in the AT&T backbone in 2013 [81]. The presence of extra headers means that there is tunneling. Instead of the expected four headers (Figure 1.2), the packet has eleven headers, giving clear evidence that the network architecture is not what it used to be. Another odd thing, inexplicable according to the classic Internet architecture, is that there are three network (IP) headers and two transport (TCP and UDP) headers.

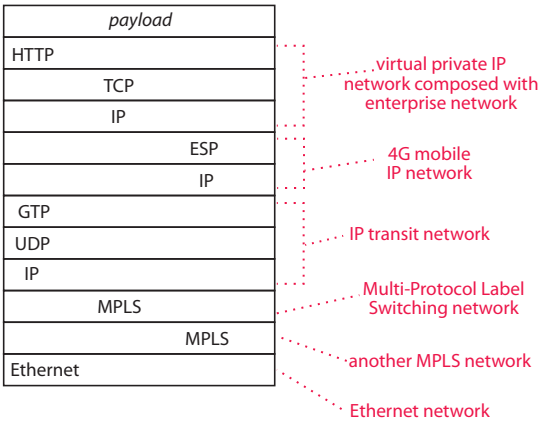

FIGURE 1.4. Headers of a typical packet in the AT&T backbone network of 2013, illustrating evolution of the Internet architecture.

The middleboxes and tunnels of today's Internet are providing services such as mobility, security, cloud computing, and content distribution. These require new functions and network components that maintain per-flow state. Much of this is built on the IPv4 and/or IPv6 standard, because it is a good general-purpose network design.

The problem today is not the middleboxes and tunnels, but the fact that the architectural descriptions of the past give us no help in understanding or improving them. For this reason, our book introduces a new style or language for architectural description, henceforth called a "model," to play this indispensable role. It will assign identity, structure, and meaning to all the middleboxes and tunnels, so that their purposes and coordination in this magnificent engineering artifact become apparent.

1.3.2. *The current Internet*

We now give a brief explanation of how the Internet produced the packet in Figure 1.4. Although the example is complex, it can be understood piece by piece. Implicitly, we are sneaking in concepts of the new architectural model to make the explanation modular and coherent. Please be aware that the body of this book will explain each piece of the new model slowly, carefully, and with many other examples. For now, even a sketchy understanding of this example is good enough!

1.3.2.1. END-TO-END COMMUNICATION

To begin with a description everyone agrees on, Figure 1.5 shows two user machines connected by the Internet. In each machine there is a stack of protocol implementations selected from the IP protocol suite as shown in Figure 1.3. In the machine on the left, an application program produces an HTTP message to be sent. Then the message passes

FIGURE 1.5. There is a protocol stack in each of two user machines connected by the Internet. Not shown: (i) protocols below IP, e.g., the Ethernet protocol; (ii) links and forwarders in the path between the user machines.

downward through the stack. Each protocol implementation encapsulates the message in a protocol-specific header (and possibly footer), and may perform many other functions related to the purpose of the protocol.

From the IP implementation module, the message leaves the user machine as a packet (or more than one, if the message is large). The packet travels along the dotted line, which represents a path of forwarders and links, until it is delivered to the user machine on the right. From there the packet travels upward through a matching protocol stack, each implementation module stripping off its own protocol-specific headers, and possibly performing other functions such as packet assembly, until it becomes a message that is delivered to the application. The dashed lines indicate that the TCP modules have direct (but virtual) communication with each other through TCP messages, and the HTTP modules have direct communication through HTTP messages. Messages travel between the two machines in both directions, of course, but only one direction is illustrated.

In the classic Internet architecture, additional communication services would be provided by adding extra protocols in the stack. These might include the services of the "session" and "presentation" layers of the Open Systems Interconnection reference model, which would be inserted between HTTP and TCP in Figure 1.5. As the example unfolds, we will show that this mechanism is insufficient to provide security, mobility, and many other services that have been added as the Internet has evolved.

1.3.2.2. ASSEMBLING NETWORKS WITH BRIDGING

In this example, the user machine on the right is a Web server in an enterprise network. For security, the enterprise network has an "intrusion-detection" middlebox that reconstructs the TCP byte stream (just as the destination endpoint does) and scans it for the signatures of known viruses and other security threats. The middlebox is shown in Figure 1.6. The figure is drawn as if the middlebox views HTTP messages as undifferentiated data in a TCP byte stream, but it could have HTTP-specific filtering as well.

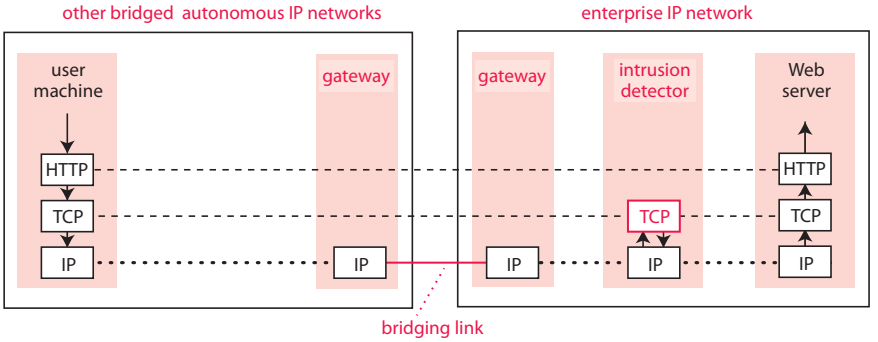

FIGURE 1.6. Bridged IP networks in the Internet. In the enterprise network, there is an intrusion-detection middlebox for security.

Figure 1.6 begins to show that the Internet contains many autonomous IP networks—each with its own administrative authority. Although most of these networks are still treated as one blob, the enterprise network has been separated from the others. This network is connected to at least one other IP network by bridging, which means that there are one or more links crossing the boundary between them. Bridging is a way to assemble autonomous networks into a larger network architecture.

Although the bridged IP networks all use the same general-purpose design, their different administrative authorities impose different goals and policies. For example, some of the many "other IP networks" included in the same box in Figure 1.6 are wide-area networks. The goal of routing in these networks is to find, for each packet destination, the most efficient path (e.g., short or uncongested) along which packets to that destination should be forwarded. In contrast, routing in the enterprise network need not be driven exclusively by path efficiency, because paths in the enterprise network are shorter. Security matters more, so one goal of enterprise routing is to forward packets through the necessary security middleboxes on the way to their destinations.

1.3.2.3. MORE SECURITY

To carry the theme of security further, in this example the user machine on the left belongs to an employee of the enterprise. As we can see from Figure 1.6, the employee machine is now connected to an IP network outside the enterprise. This is insecure because enterprise packets can be read or tampered with as they traverse public networks. To protect against this threat, packets traveling outside the enterprise network are encrypted.

Encryption is performed by the Encapsulating Security Payload (ESP) protocol (which belongs to the IPsec suite of IP security protocols), as shown in Figure 1.7. Here ESP is used in what is called "transport" mode, with ESP in the protocol stack below TCP. In the enterprise network, the employee's packets are routed to a "virtual private network"

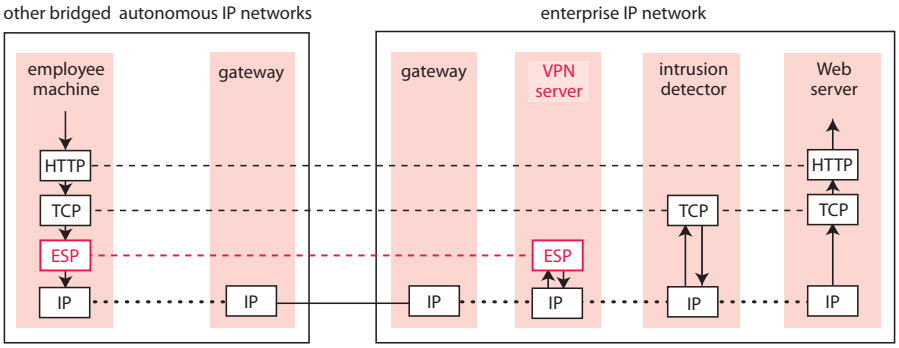

FIGURE 1.7. Encryption provides further security.

(VPN) server, where the ESP module strips off the ESP header, decrypts the packet, and passes it back down to the IP module.

This scheme is simple, but it introduces as many problems as it solves. First of all, the employee may be using the network in a coffee shop, airport, or other public place. The IP name in the source field of its packets will belong to the local network and be unknown to the enterprise network. In most enterprise networks, the gateway is a firewall that would drop the employee machine's initial packet (a TCP SYN) as a security risk.

In addition to the firewall, there may be other problems:

- How does the enterprise network know that these particular packets should be routed through the VPN server?
- If the Web server is meant for use only by employees, it may not have a public IP name. (In this case it will have a name in the private IP namespace only, which is not meaningful or reachable from the public Internet.) How does the employee's machine in a coffee shop or airport direct the packets to this particular enterprise machine?
- The VPN server will require the employee machine to present authenticating credentials, probably including a user name and password typed by the employee. But only ordinary TCP/IP packets leave the VPN server, so how does the intrusion detector or Web server find out the user name of the packet source, if it is needed for additional screening or customization based on the employee's role?

As examples of why mere stacking of transport and application protocols is insufficient for today's Internet, these problems are just the tip of the iceberg. And with this many problems in such a simple example, there should not be a separate, ad hoc solution for each one!

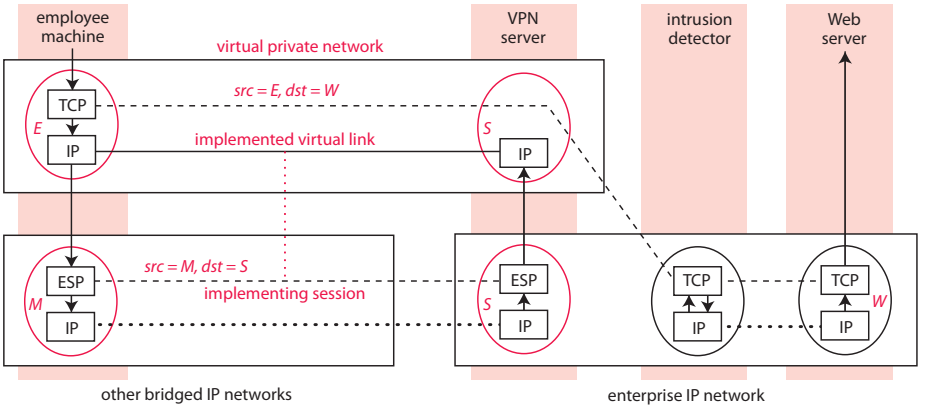

FIGURE 1.8. A virtual private network is layered on bridged IP networks. HTTP modules and gateways between the machines are still present but no longer shown.

1.3.2.4. ASSEMBLING NETWORKS WITH LAYERING

Fortunately there is a straightforward way to solve all these problems, which is shown in Figure 1.8. In the figure we see that a "virtual private network" is really a network, enclosed in its own box.

Now two of the machines in the figure are participating in two networks each. Each machine has, for each network it participates in, a network member with a name in the namespace of the network. Members are pictured as ellipses, and their names are shown in Italics. Each member contains (at least conceptually[4]) the protocol stack it uses to participate in the network. Because the virtual private network is also an IP network, members at both levels have normal IP protocol stacks. Don't worry about the gap between two members in the enterprise network, as it will be filled in later.

In Figure 1.8 the virtual private network is brought together with the bridged networks at the lower level by layering. Layering is another way, in addition to bridging, to assemble networks into a larger network architecture. When a network is layered on one or more bridged networks, it always means exactly the same thing—that a link in the upper network (overlay network) is virtual, and is implemented by a session in the lower networks (underlay). A session is simply a set of messages/packets that go together from the viewpoint of its endpoints. In this instance of layering, the ESP session between the employee machine and the VPN server is implementing the virtual link in the virtual private network between the same machines.

How does layering actually work? In the virtual private network, when a TCP message is being sent from *E* on the virtual link, it is of course encapsulated in an IP header, then sent. The sending is guided by layering data in the IP implementation module. This data tells the module to pass the message to *M*. In *M* the message is treated simply as ESP

4. There is no guarantee that the software and hardware in the machine will be organized exactly like this.

payload, encrypted, encapsulated in ESP and IP headers, and sent out on some link in *M*'s network. This way of using ESP is called "tunnel mode." When the message/packet is delivered to *S* in the enterprise network, it is stripped of its IP and ESP headers, decrypted, and passed upward (again guided by layering data in the IP implementation) to the member of the virtual private network. In that member, it is handled exactly as if it were received on the virtual link.

Now that the left side of Figure 1.8 has two layered IP networks, packets crossing the public Internet have two IP headers with two source/destination pairs. Furthermore, a machine participating in two IP networks can have a different name in each, as the employee machine does. It can also have the same name in each, as the VPN server does. This increased flexibility solves the problems noted in §1.3.2.3, as follows:

- The packets' outer IP header has the public IP name *S* as destination. The firewall accepts all packets with this destination and forwards them only to the VPN server. So the firewall knows what to do with these packets, and its actions are safe because the VPN server performs its own authentication of the packet source.
- Providing that the packets are authentic, the VPN server strips off their outer IP and ESP headers, revealing another IP header with destination *W*—the private name of the Web server. This name is meaningful now that the packets are within the scope of the private enterprise network, even though it was not meaningful in the public Internet. The destination name tells the enterprise network exactly where the packet should go.
- The source name *E* is a private name assigned to this particular employee within the enterprise network. Because the VPN server has made sure this employee machine has the right to use *E*, the intrusion detector and Web server can use it as a trustworthy indication of who is sending the packets.

Thinking architecturally, note that another name for a virtual link is "tunnel." Note also that both the VPN server and intrusion detector are middleboxes.

1.3.2.5. MORE LAYERING

Next we reveal that the employee machine is a cellphone. So the cellphone's lower-level connection to the Internet is actually through a 4G (4th Generation) mobile network, as shown in Figure 1.9. Although the label "4G" is correct for 2013, when the packet was observed in the AT&T backbone, everything in this example is true for 5G as well.

Adding this detail will give us more instances of layering. One of them is that the 4G network is layered on one or more radio networks, which are not IP networks and are not shown in the figure. The virtual link *R* connects *M* to a cell tower, and is implemented by a radio network. When the cellphone roams to another cell tower, it will be connected through a different virtual link to the new tower.

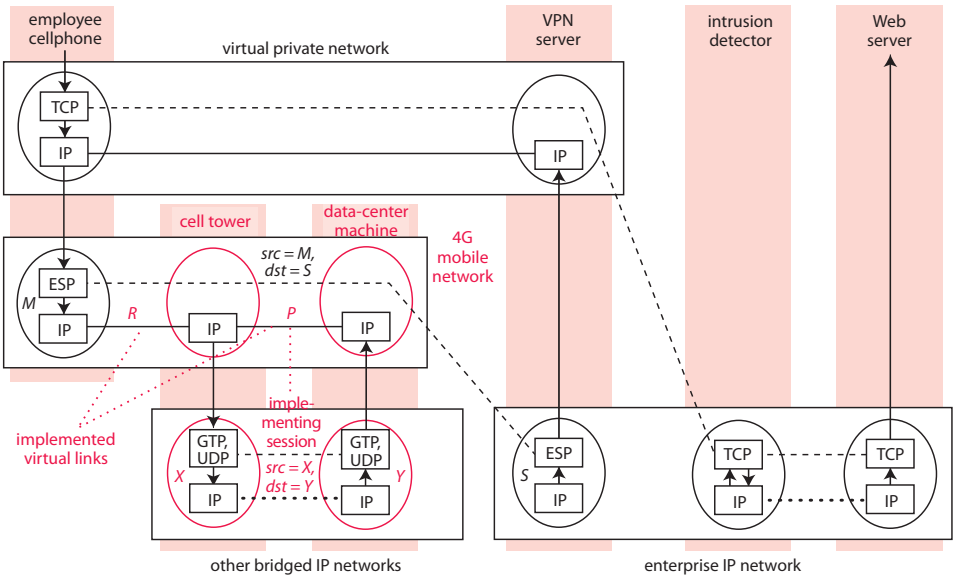

FIGURE 1.9. The virtual private network is layered on a 4G mobile network. The 4G network is layered on IP networks and radio networks. Note that the virtual link *R* is implemented by the radio network, which is not shown.

Later in the book, mobility service in networks will be explained in detail. Meanwhile, Figure 1.9 is an oversimplification of one approach to mobility, but it is faithful enough for present purposes. For this kind of mobility, in this example, most of the necessary computation takes place in a data center, probably far from many of the radio networks. In the figure, a data-center machine is connected to the cell tower via virtual link *P*. The 4G network uses special routing to keep track of the current path to *M* and to forward its packets along that path, which begins with link *P*. Later, when the cellphone is connected by radio to a different cell tower, it will be reached from the data-center machine over a different virtual link rather than *P*. This kind of routing must be separate from ordinary Internet routing, because the frequency of updates is much greater than ordinary wide-area routing can handle.

As shown in Figure 1.9, the virtual link *P* is implemented on ordinary IP networks, statically connecting remote cell towers to centralized data centers. In the lowest-level IP networks, the protocol GTP[5] performs the required encapsulation. The IP standard requires that GTP be encapsulated in a more common session protocol, which is usually UDP.

In Figure 1.8, along some part of their path, packets from the employee cellphone had two IP headers and two source/destination pairs, both of them necessary. In Figure 1.9, along the path between the cell tower and the data-center machine, packets have three

5. See Chapter 4 for the complete expansion of this acronym.

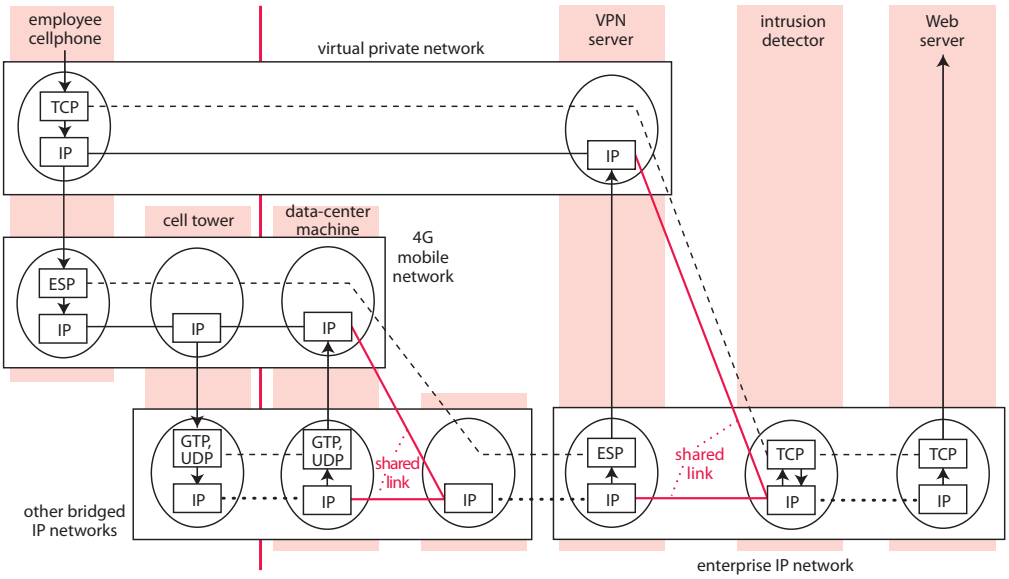

FIGURE 1.10. Subduction assembles networks by means of shared links.

IP headers and three source/destination pairs, all of them necessary. Now *M* is a mobile destination, which cannot be found in ordinary wide-area forwarding tables. The 4G network's path to it is implemented by radio networks and largely static IP networks, where the new IP sources and destinations *X* and *Y* can get the packets across long distances between the data center and a cell tower.

1.3.2.6. ASSEMBLING NETWORKS WITH SUBDUCTION

Finally, all we have to do is fill in the two connectivity gaps at the lowest level in Figure 1.9. The gap between the VPN server and intrusion detector seems to involve some kind of interoperation between the virtual private network and the enterprise network, but what is it? Bridging assembles peer networks, but these two networks are not peers, because the virtual private network is layered (partly) on the enterprise network. Layering is for hierarchical assembly, not peer-to-peer assembly.

The answer is that we need a third way to assemble networks, called subduction, as shown in Figure 1.10. Subduction[6] relies on shared (in a special sense) links, two of which fill the former gaps. Shared links can be virtual, but for purposes of explanation let's assume these are physical. Thus there is only one physical link between the VPN server and intrusion detector, and there is only one physical link between the data-center machine and the unnamed machine to its right.

6. The origin of this word will be explained in Chapter 4.

A shared link is always visualized as shown in Figure 1.10: it looks like two separate links sharing a common endpoint. One of these parts is a peer-to-peer link, which may be a bridging link between networks or a link internal to a network. The other part acts as a bridging link between an overlay network and an underlay network, although it is not actually a bridging link because it crosses levels of the hierarchy. At the common endpoint, the network member sees a single link that is normal in every way; if the link is two-way, it can both send and receive packets on the link.

On the other end of the shared link, there are endpoints in two different network members on the same machine. These two members must have a small amount of data to coordinate use of the link. If a packet is being received, there is a mechanism to decide whether it goes to the overlay or the underlay (never both). If a packet is being sent, there is a mechanism to decide whether the forwarding rules of the overlay or the underlay (never both) apply to it. The advantage of this abstraction is that, in almost every way, the behavior of subduction is simply the behavior of bridging plus the behavior of layering.

Subduction is extremely important in today's Internet, because (as Figure 1.10 shows) different user machines participate in different networks, and there are discontinuities in the layering. Subduction is a major enabler of Internet innovation, because it smooths over these discontinuities, allowing new businesses and user associations to add network services as they please.

This brief explanation has now covered the three IP headers in Figure 1.4. If a packet monitor were placed along the path of a packet, where the vertical red line in Figure 1.10 crosses the path, it would see exactly these IP headers (the non-IP headers will be explained later in the book). Now the network labels in Figure 1.4 should make more sense, as we know that the three IP headers belong to three different IP networks, assembled with bridging, layering, and subduction. In each network's header, there is a header for a forwarding protocol (here always IP) and one or more session protocols. The session protocol determines which packets go together from the viewpoint of the session endpoints. If the session is implementing a link in a higher-level network, the packets in the session are exactly those sent and received on the implemented link.

1.3.2.7. SUMMARY OF THE EXAMPLE

As we have pointed out before, virtual links are the same as tunnels. In this realistic example of today's Internet, tunnels and middleboxes are plentiful. And the reality has strayed very far from anything encompassed by the classic Internet architecture.

Nevertheless, our presentation of the example has imposed structure on it, so we could give each tunnel and middlebox an identity, and explain the role it is playing. We can, for example, predict exactly which sources and destinations would be found in each IP header, wherever in the physical topology packets are sampled.

1.3.3. *Future evolution*

Despite the challenges that the Internet has evolved to meet since 1993, the world has not run out of challenges yet. For example:

- There is a well-known "digital divide" between developed and developing countries. Citizens of developing countries need new forms of Internet access that are lower in cost, more tolerant of delays and low speeds, and based on a wider range of resources.
- Providers of popular Internet services are large, wealthy companies. They want to gain more competitive advantage by investing in better Internet connectivity all the way from their servers to their users.
- Because of privacy concerns, most Internet traffic is now encrypted. Many security mechanisms developed over the last two decades rely on inspecting the payloads of packets for malware and the signatures of other threats; these mechanisms do not work on encrypted data.
- There is rapidly increasing automation of critical infrastructure, such as power grids. These are becoming more complex and also more integrated through the Internet. This enhanced connectivity must not be allowed to compromise their safety and reliability.
- In many areas of computing, there are increasing expectations that services essential to society should meet specification standards higher than the Internet Engineering Task Force's "rough consensus and running code," and that there should be some reason to believe their implementations satisfy their specifications. Verification of correctness properties, including some security properties, is now practical [13]. The new challenge is to apply these reasoning techniques at a higher level of abstraction, so that they relate directly to the experience of network users.

Our general impression is that this list exemplifies "tussle" in the sense of [24]: the need for different outcomes in different parts of the Internet ecosystem, because of the ongoing contention among parties with conflicting interests.

At the same time, people with intimate knowledge of network software agree that it is too complex. They believe that its complexity threatens reliability, and must be addressed as urgently as other well-known problems [19].

In summary, the challenge of future Internet evolution is to increase the Internet's flexibility in function and performance while at the same time reducing its complexity and increasing its trustworthiness.

1.4. Purposes and a new approach

The primary purpose of this book is to describe the real Internet architecture, explaining how it evolved to its current state and how it continues to evolve. This is interesting in

its own right, but—much more important—it may help guide evolution in the future. We will show that diversity of function and performance is well on its way, and probably needs no additional push. In contrast, we offer some new ideas for reducing complexity and increasing trustworthiness, without slowing the pace of innovation.

If nothing else, better descriptions of the Internet will improve teaching and research by providing a common framework for discussion. With a common framework of concepts and terminology, it will become much easier to analyze architectural alternatives and design trade-offs. It will also become easier to recognize alternatives and trade-offs that cross the conceptual boundary between networks and distributed systems.

An essential ingredient of this work is a new model of network architecture, to replace the still-ubiquitous classic model. The need is easy to comprehend. Based on our study of examples such as in § 1.3.2, evidence for the evolution of Internet architecture is everywhere, and it is overwhelming. If this is so, why do so many people still think that the Internet's architecture has not evolved? The answer is that the prevailing model of Internet architecture is misleading, and has prevented people from seeing the ongoing changes.

The new model has many characteristics, but two characteristics have been the basis and inspiration for everything else:

- The model is completely *general*. Without generality, it is hard to imagine how one could study evolution. Because the new model is general, the past and present Internet architectures are instances of it, as are all the prescriptive "future Internet architectures" that have been proposed, along with many other alternatives. The future Internet architecture—whatever it is—will also be an instance of this model.
- It is *precise* and *formalizable*. Without precision, we cannot be sure what we are talking about, but we can (almost) be sure that others will understand it differently. The best way to ensure precision is define the model formally, and use automated tools to see what consequences emerge from the definitions.

In the remainder of this chapter, § 1.5 introduces the new model, compares it to the classic Internet architecture, and discusses its characteristics further. Finally, § 1.6 gives a chapter-by-chapter overview of the book.

1.5. A new model of network architecture

1.5.1. Fundamental concepts

The new model of network architecture is based on these ideas:

- A *network* is a module of network architecture. For example, Figure 1.10 has four networks (although one of them is actually an abbreviation for many networks bridged together).

- A *networking environment* or *network ecosystem* contains many such *network* modules, each with its own purpose, geographical span, membership scope, and level of abstraction. Figure 1.10 shows a piece of the Internet ecosystem.
- The *architecture* of a network ecosystem is, in its primary description, a rich, flexible *composition* of networks. Networks are composed or assembled by means of three operators, *bridging*, *layering*, and *subduction*.

Because of these key ideas, the new model is called *compositional network architecture*. In this phrase and other uses, "network architecture" is being used in a common, general sense, as a substitute for the more precise "architecture of network ecosystems."

In compositional network architecture, all networks have namespaces, members, links, routing, forwarding protocols such as IP, session protocols such as TCP, and directories. The list of basic components and functions is always the same, because each network is a self-contained microcosm of networking. Despite this commonality, the networks can have very different designs, as embodied by their *particular* namespaces, protocols, topologies, resource strategies, and other characteristics.

For example, in §1.3.2, we pointed out that IP routing in wide-area networks and in the enterprise network use different approaches because they have different goals—path efficiency without centralized control versus routing through middleboxes. Routing in the 4G mobile network is different from either, because its major goal is fast updating to track the movement of cellphones. And routing in the virtual private network is different from all three! A VPN is topologically very simple because there is a direct (virtual) link between each pair of members that need to communicate. So routing in a VPN is vestigial, meaning that it is so simple it is unnoticeable.

Network designs go far beyond IP. Other designs are meant to connect different types of machine for different purposes, so they make different performance assumptions and have different membership and authentication procedures. Some are meant to be local, and some can span the globe. Some are practical for only a few members, and some can handle millions. Some are meant to be static, and some can be used where components are highly dynamic. Some have concrete physical links, and some have abstract virtual links. Each network design has its own namespace, defined for its own purposes. Each network design has its own session protocols and services. And each network design has its own policies and control algorithms in accordance with its own assumptions and goals about such concerns as trust, security, performance, and economics.

In this book we will put on display a zoo of real network designs, including exotic beasts as well as domesticated animals. These will include networks in which members have no names, networks in which session protocols are executed by the infrastructure components rather than the user machines, and networks in which important routing and forwarding functions are performed by—literally—wild animals. Nevertheless, all these networks can be described with the same basic template, and all of them can be composed with IP networks within the Internet ecosystem.

Of the three composition operators on networks, the operator that shapes compositional network architecture most forcefully is layering. You have already seen multiple instances of layering in §1.3.2. Every instance of layering is the same: a link in the upper network (overlay) is being implemented by a session in the lower network or networks (underlays). Both links and sessions are communication channels, and all networks have them. Although most instances of layering in §1.3.2 are composing IP networks with other IP networks, because links and sessions are universal concepts, any kind of network can be layered on any other kind of network. *This is the feature of compositional network architecture that makes networks composable like Lego blocks.* We have found this definition of layering to be broadly applicable.[7]

For another perspective, let's look at the user interface to a network, as shown in Figure 1.11. The user of a network is a distributed system needing communication services. The machines that host the distributed system must also host members of the network. A module of the distributed system sends a message by giving it to the network member through the operating system or hardware of the machine. Later, network members deliver the message to modules of the distributed system on one or more other machines. Note the contrast with Figure 1.1, in which user machines are not considered part of the network. Compositional network architecture acknowledges that sometimes user machines participate very actively in their networks.

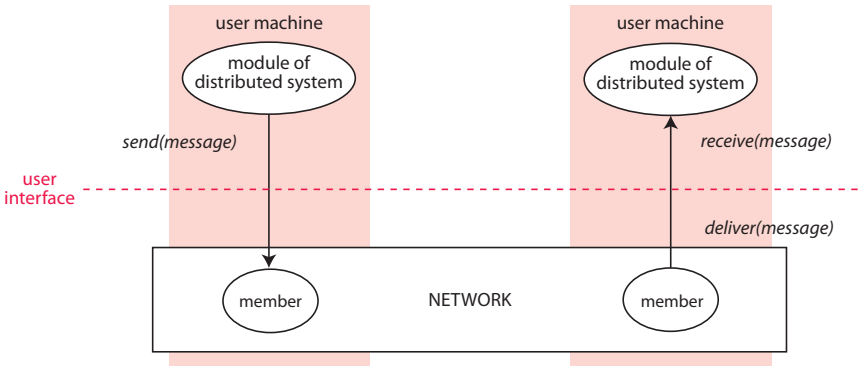

FIGURE 1.11. The user interface to a network in compositional network architecture.

Figure 1.11 also reveals the secret of how the Internet has evolved so much without major changes in the IP protocol suite. Look at the layering interface between networks, shown abstractly in Figure 1.13. The user interface to a network is *exactly* the same as the layering interface between networks! From the perspective of a network, it can be used by a distributed application system or by an overlay network, and the difference will be

7. In fact, we know of *no* exceptions to it. But there's no way to prove the absence of unknown exceptions.

irrelevant. So new communication services and modes of usage can be added freely to the existing Internet, simply by layering new networks on top of it.

1.5.2. *Brief comparison to the classic Internet architecture*

The most easily misunderstood difference between compositional network architecture and the classic Internet architecture is the meaning of layering. As we have seen, the classic Internet architecture has a fixed number of layers. Each layer performs a specific, fixed set of functions. The relationship among layers, seen in both Figures 1.2 and 1.3, is simply one-way dependency, from top to bottom.

In the new model, networks composed by layering are also modules in a dependency hierarchy, but the resemblance ends there. Beyond the mere existence of a dependency hierarchy, there are differences in the nature of layers, the interfaces between layers, and the number of layers:

- In the classic architecture, a layer is simply a set of related functions. In the new model, a layer has a much more specific structure. It is a *self-contained network,* with all the basic mechanisms including a namespace, routing, forwarding, session protocols, and directories. As a consequence, layers/modules in the new model are different from layers in the classic model (see Figure 1.12). In particular, an IP network contains forwarders, links, routing, and forwarding, *and* the IP session protocols such as TCP and UDP.
- Because a layer in the classic architecture can implement any set of functions, each interface between layers must be specifically tailored to the functions above and below. But layers/modules in the new model all have a fundamental similarity as self-contained networks, so the layering interface between any two networks is the same. Again, this is the feature of compositional network architecture that makes networks composable like Lego blocks.
- In the classic architecture, there is a fixed set of layers. In the new model, the depth of layering in a network architecture is unconstrained.

Figure 1.12 shows how layers in the two architectures correspond. Similar functions are placed at the same level in different layer stacks. Application networks will be discussed in Chapter 3.

Another major difference between the two models concerns the possible roles that middleboxes can play. In Figure 1.13, for example, there is a middlebox in the overlay network. In this network, the middlebox is a trusted component of the infrastructure, while the two session endpoints are untrusted user members.

Now consider the role of the middlebox machine in the underlay networks. In these networks, its member is an ordinary untrusted user, while the forwarders are trusted infrastructure components. This is exactly the right model if the middlebox machine is owned

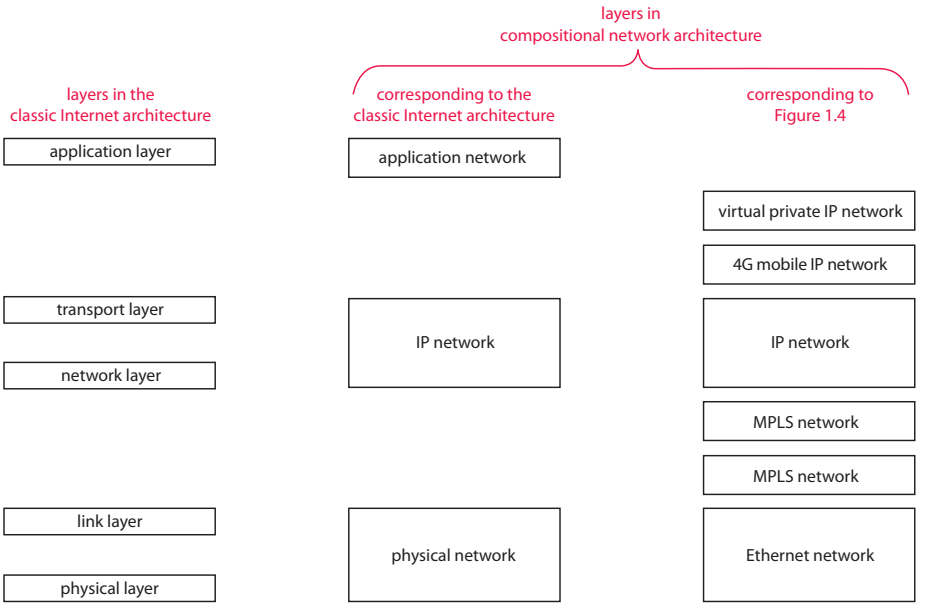

FIGURE 1.12. Correspondence of layers in the two architectures. Similar functions have similar vertical placements.

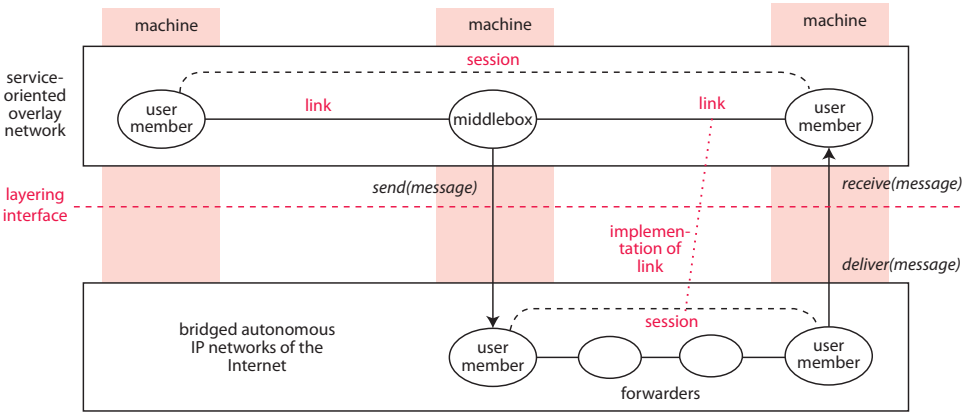

FIGURE 1.13. In compositional network architecture, machines hosting middleboxes can play different roles in different networks.

and controlled by the administrative authority of the service network, but not owned or controlled by the administrative authority of its Internet service provider.

This example shows how middleboxes can be deployed wherever they are needed, even if they are not part of the infrastructure of basic Internet service. It illustrates the

idea behind an updated version of the end-to-end principle, suitable for the Internet's current and future architecture, which will be presented in Chapter 4. In the classic Internet architecture, this nuanced view of middleboxes cannot be represented.

1.5.3. Characteristics of the model

1.5.3.1. MORE ON ACCURACY AND PRECISION

According to Wikipedia,

> In the fields of science and engineering, the accuracy of a measurement system is the degree of closeness of measurements of a quantity to that quantity's true value. The precision of a measurement system, related to reproducibility and repeatability, is the degree to which repeated measurements under unchanged conditions show the same results.

Since a measurement of a network is a very simple model of it, our interpretation of "accuracy" and "precision" applies Wikipedia's definition in an expansive sense, to "measuring" a network ecosystem by making a structural and functional model of it.

With this interpretation, a degree of precision predicts what will happen if you ask two people to describe some aspect of networking, large or small, in terms of the new model. A high degree of precision means that there is one right answer, which is not a matter of judgment or opinion, so that both people will give the same description. And if they do not, there is a definite reason why one answer is wrong. Precision is important because, without it, people cannot communicate reliably.

Compared to the prevailing standards in networking, compositional network architecture is very precise. For example, in [48] there is a discussion of whether the Address Resolution Protocol (ARP) belongs to the network layer or link layer of the classic Internet architecture. The authors are not able to come to a conclusion. In compositional network architecture, on the other hand, there is no ambiguity whatsoever: session protocols are part of each network, and ARP is an Ethernet session protocol. This fact is not dependent on the purpose of ARP, which is to create a partial, distributed directory mapping the IP namespace to the Ethernet namespace. It is also not dependent on the reason why directory-building is accomplished by an Ethernet protocol rather than an IP protocol, which is that the protocol relies on Ethernet broadcast service.

As for improved accuracy, there is no shortage of examples, such as the packet in Figure 1.4, which cannot be explained by the classic Internet architecture. This entire book is full of such examples, but they are most heavily concentrated in Chapter 4.

Accuracy and precision are also important because they make it possible to explain things in a way that a person can learn by heart and then compare constantly to his or her own observations. For example, in a section of their book [67] called "Perspective: Virtual networks all the way down," Peterson and Davie write:

> For almost as long as there have been packet-switched networks, there have been ideas about how to virtualize them, starting with virtual circuits. But what exactly does it mean to virtualize a network? . . . The hard part is grappling with the idea of virtual networks being nested (encapsulated) inside virtual networks, which is networking's version of recursion.

Our model shows that virtualization of networks is network composition by layering, and it is completely straightforward, although extremely rich in its purposes and properties. Why has this straightforward operator not been defined earlier? Peterson and Davie have some sense of the right answer—they mention encapsulation, which is a big part of layering because messages from an overlay network are encapsulated in messages of the underlay network before they are transmitted. But the layers in the classic Internet architecture are not the right layers, because they don't put network function (e.g., forwarding) and network usage (i.e., sessions) together, so they don't have Lego-like interfaces. This is the piece of the puzzle that has been missing until now.

For another perspective on compositional network architecture, consider its contrast to Plutarch [25] and a recent update of it [18]. Plutarch is interesting because it has a concept of a "context" that is similar to our concept of a network (although there are important differences). For both concepts, internal diversity and external interoperation or composition are the norm. One major difference is that in Plutarch, composition of contexts is accomplished by means of interfaces called "interstitial functions," which are not further defined—because the set of interstitial functions needed for each pair of networks is expected to be different. In the new model, on the other hand, interfaces between networks are formally defined and always the same except for data stored in tables.

Another major difference between the new model and Plutarch is that the focus in Plutarch is on communication between contexts to discover names, capabilities, and other control information. In other words, the focus is on the control plane. Compositional network architecture, on the other hand, formalizes the data plane: all the operational state of a network, including packet headers, forwarding tables, composition tables, session state, and packet processing that will be needed at runtime. This is sufficient to provide a view of network architecture that is both broad and deep.

In fact, we know from long experience that without precise description of the data plane, it can be almost impossible to know for sure what all the architectural options are, which options are actually feasible in a given situation, which pairs of options are really different and which are fundamentally the same, and what extended consequences an option might have. Our debates on these questions went on and on until we found the right data-plane model for each architectural option—and then the answers to the questions became clear.

1.5.3.2. ON TERMINOLOGY

The first step in achieving precision is to use clear, well-defined terminology. Terminology is often a contentious subject, because it is the foundation of abstraction. To name a

concept, type of object, data structure, algorithm, or other abstraction is to say that it is worth talking about. We all instinctively know this is important and want to get it right. The terminology in this book is carefully chosen and in some ways unusual, so we are taking the time to explain exactly why we are defying convention in this way.

What to name?

The first decision in naming is to identify some abstraction as needing a name. Much of the trouble with the confusing paper in §1.2.4 would have been avoided if the authors had recognized how central the four header fields were to their presentation, and agreed early on what they would be called. The next decision, what to name it, will be covered below.

We have pointed out that every layer of the classic Internet architecture has its own terminology. Referring back to Figure 1.4, we do not want separate names for concepts in the three different types of network found in this figure (Ethernets, MPLS networks, and IP networks), let alone the six different layers, let alone all the other network designs and layer hierarchies that will be found in this book. Furthermore, compositional network architecture emphasizes the similarities among all networks. Ultimately, the only choice that makes sense is to introduce terms for all the parts, functions, mechanisms, etc., of a network, and to use *the same terms for all networks.*

What to name it?

Now that we have identified a precise concept worth naming, what name do we choose? This is difficult because every decent term has been used often, and may have many specific connotations and associations in people's minds, often different for different people.

One possibility is to make up a completely new word, so that no one has any preconceptions about it. The trouble with this choice is that everyone has to learn a lot before they can understand anything we say. Also, language is a social signifier, and people who use a lot of artificial words no one else knows are signifying that they are a cult.

Instead of starting a cult, we prefer to use the common term that fits the concept best. This will mislead some people some of the time, which seems to be the lesser of the two evils. Readers of this book should be aware that terms introduced in *Italics* are given precise definitions and used consistently as defined throughout the book. These terms and their definitions can all be found in the Glossary.

1.5.3.3. MODULARITY, REPETITION, AND PATTERNS

The best way to expedite evolution of a complex system is *modularity.* In a modular architecture there are well-defined module interfaces, so that modules can easily be changed without changing their environments, and can easily be combined in new ways.

We have already mentioned that network software is too complex, and that its complexity threatens reliability. The only feasible way to make software simpler without losing

functionality is to find *repetition of function*, and to replace many idiosyncratic implementations of the same function with a few well-engineered implementations. With modularity, repetition of functions is easier to find. The obvious differences among module instances make it possible to understand the requirements and trade-offs governing a particular instance of a function, and to select the right implementation for each module accordingly.

In compositional network architecture, modules are networks, and all networks have the same basic parts and functions, identified by the same names. This means that there is repetition, from network to network, of parts to manage and functions to carry out. Some functions, such as routing functions, have varied goals and diverse techniques. On the other end of the spectrum, however, layering and subduction are logically the same in all networks. Here there is real potential for taking advantage of a few well-engineered implementations, differing primarily on the number of parameters and the speed of the implementation technology (e.g., hardware versus software).

Another benefit of recognizing modularity and repetition is the discovery of *patterns*, where each pattern is a problem common to many different networks, along with a range of related solutions.[8] A pattern may be suitable for some networks and not for others. We have ample evidence from writing surveys on mobility [94] and network security [98], as well as this book, that compositional network architecture facilitates recognition of patterns.

We believe that patterns have great potential for improving the teaching of networking. Patterns are far more meaningful than the details of isolated networks. They also supply the much-needed context of why and where, in a network ecosystem, a problem arises. Knowing that there are different solutions to the same problem, we want to know why there are different solutions, and this is where all the good stuff is. Clarifying the choices requires understanding the important considerations that drive engineers in their decision-making. For these same reasons, the knowledge embodied in patterns is memorable and reusable, when the student encounters something new.

Patterns can also help with the ever-expanding curriculum, because they are abstractions. Thus a pattern can be presented in expansive detail, or it can be presented concisely and more abstractly, where space in the curriculum is tight.

Just as patterns provide a framework for knowledge and teaching of networking, they can also help to consolidate and generalize research results. The range of solutions captured in a pattern can often be arranged in a "design space" with well-defined, possibly quantitative, dimensions. This would help in relating results from different experiments, and could ultimately systematize engineering trade-offs.

8. With thanks to Day's seminal and inspiring book [26].

1.5.3.4. MORE ON GENERALITY AND FORMALITY

In many ways generality and formality would seem to be at odds with each other: formality means rigid definitions, so how can a formal model cover every possibility?

Compositional network architecture emphasizes network boundaries and interfaces, rather than network internals. So compositional network architecture does aim to be complete with respect to network-composition operators, including the data and packet processing necessary to implement composition. The model is much less complete for intra-network concerns, covering primarily topology, basic forwarding, and some session state. Examples in the book will show that these are modeled abstractly enough to describe a wide range of network designs.

The Recursive Network Architecture (RNA) [84] is based on a view of network architecture similar to ours, but, in contrast, it is embodied in a would-be-universal "metaprotocol." So it is far less general than compositional network architecture. It is also much harder to use as a key to understanding the Internet, because functions performed by many different mechanisms in the Internet ecosystem must all be squeezed (in imagination) into the particular mechanisms of the RNA metaprotocol.

For both inter-network and intra-network aspects of behavior, the new model should usually be interpreted as necessary rather than sufficient. Many of the network mechanisms described as "control plane," such as routing state, do not appear in the model at all. Consequently, a particular network design may contain many things not described by the formal model, and not constrained by it.

The exception implicit in "*usually* interpreted as necessary" above is that in a particular network, a particular object can be *vestigial*. Formally, this often means that if it were present it would be making or representing a choice from a set with one or zero members. For example, even though the model states that a network header has a protocol-selection field, if the network has only one session protocol then the field is vestigial and not necessary.

For those who appreciate formality or are interested in verification, the formal model is available at `compositionalnetarch.org`. This is completely optional reading, as the body of this book defines each aspect of compositional network architecture rigorously, if not formally.

Formality enables automated reasoning, including code analysis, verification, and code generation. Automated reasoning can improve quality in a cost-effective manner, especially because it can be applied continually throughout the life cycle of hardware and software. Automated reasoning is the only way to provide guaranteed properties and a high level of quality assurance.

1.6. Organization of the book

This section provides a brief synopsis of the contents of each chapter. Note that Chapters 2 and 3 give a complete introductory overview of Internet basics. Their primary purpose is

to describe these well-known facts using compositional network architecture, so that the new model is explained thoroughly. Secondarily, however, they give a general impression of what it might be like to teach networking using compositional network architecture.[9]

Chapter 2: Describing networks and services

Chapter 2 covers the internal structures, functions, and mechanisms of networks. These include network components, routing and forwarding, sessions and session protocols, and authority and management. The chapter covers each part in a general way that encompasses all network designs, and introduces terminology to establish common ground among all networks.

Chapter 2 also introduces the first of five forms of composition in compositional network architecture. It is called *protocol embedding*, and is used for composing network services inside networks.

As examples, Ethernet networks and IP networks are described in detail. A highlight of this chapter is a viewing of four other animals in the zoo—unusual network designs showing the malleability and descriptive power of the new model.

In addition, this chapter introduces terminology for description of network services. Although unlikely to be seen as a highlight by most readers, this is important because relating services to architecture is a recurring theme of the book.

Chapter 3: Composing networks and services

Chapter 3 introduces the first two composition operators on networks, *bridging* and *layering*. The discussion of bridging explains how the hierarchical bridging relation among IP networks is shaped equally by physical factors and business relationships. It also introduces another mechanism for service composition inside networks: *compound sessions*.

The primary examples of layering are the ubiquitous ones: layering IP networks on Ethernets, and layering the World-Wide Web (as a service-oriented network) on the IP networks of the Internet. In addition, we present five other examples of special-purpose networks and how they fit into the Internet ecosystem with layering. Two of these examples are well known in academia (Tor, cloud computing), while others are less studied but very widely used (MPLS networks, Virtual Local Area Networks, and performance-enhancing overlay networks as exemplified by the pioneering ideas of Resilient Overlay Networks). A highlight of this section is a detailed account of cloud computing and how it exploits the power of layering.

By the end of this chapter, you will have learned about two mechanisms for composing network services inside networks: *protocol embedding* and *compound sessions*. You will have also learned about two operators for composing networks, *bridging* and

9. Because of the introductory material, this book might even be used, experimentally, as an introductory textbook for intellectually mature students.

layering. There is one more composition operator to come in Chapter 4, called *subduction*. Together, these five concepts put the "compositional" in compositional network architecture.

Chapter 4: The real Internet architecture

Chapter 4 is the heart of the book, as it pulls the previous chapters together to describe today's Internet in terms of compositional network architecture. To do this, it introduces the third network-composition operator, *subduction,* which combines bridging and layering as needed at the "bleeding edge" of Internet evolution. It is well worth understanding, because it is the enabler of innovation in the Internet.

We distinguish the lowest-level IP networks of the Internet, bridged together to achieve global reachability, as *the base Internet*. Roughly speaking, the base Internet is the same as the classic Internet of 1993. As the Internet ecosystem has evolved, many additional networks have been added, both above and below the base Internet, by means of layering and subduction. These additions are characterized by three patterns in this chapter: layering for routing scalability and flexibility, layering for sharing or "slicing" resources, and layering/subduction for enhancing network services. Here the example in §1.3.2 is presented in fuller detail, as an illustration of all the patterns that describe the real Internet architecture.

The chapter also discusses past and future Internet evolution, focusing on two major topics. First, past evolution has completely undermined the letter of the original end-to-end principle, but not its spirit. We show why this is true, and define a new version of the end-to-end principle, suitable for today's Internet. Second, we discuss evolutionary trends affecting the base Internet, now and in the forseeable future.

Chapter 5: Patterns for enhanced network services

Chapter 5 is full of patterns, all of which elucidate the ways that network services can be added to basic network communication. Enhanced network services overcome obstacles to communication inherent in basic networking, such as network failures, security threats, endpoint limitations, bad side-effects of good features, and the challenge of finding the right network member with which to communicate.

The patterns in this chapter connect services to architecture, showing step-by-step why certain services require specific features of compositional network architecture. Major examples include mobility, inter-network multicast, aspects of security and privacy, and firewall traversal.

The highlight of this chapter is the repeated use of a very small set of mechanisms to extend the Internet ecosystem, incrementally, with a very large set of services. This is possible because each mechanism is defined with the greatest possible generality, and because each mechanism is a fundamental part of compositional network architecture. In

other words, we are *not* answering the question, "What are all the ways to add each particular service?" Rather, we are answering the question, "How can all services be added, with good performance and efficiency, with the fewest different mechanisms?" The potential payoff is simpler, better-engineered implementations without loss of functionality.

Chapter 6: Ideas for a better Internet

Chapter 6 explores what it might mean to use the new model, not just descriptively, but also prescriptively—as a means of designing and building networks. We cover such diverse topics as Internet standards, implementation and optimization of compositional network architecture, and our thoughts about teaching courses on networking.

One highlight of this chapter is a presentation of three case studies, showing that the modularity of compositional network architecture can be very effective in providing detailed specifications of network properties, including user-level service properties, and in making their automatic verification scalable. The section makes a direct link between verification and security.

Another highlight of this chapter is a section bringing together ideas and examples from across the book, for the purpose of inspiring research on principles of layered architectures. These principles both support and take advantage of the new end-to-end principle. They also relate the basic characteristics of layering of networks to dynamic properties of networks and to network performance.

1.7. Bon voyage

In 2001, a report from a committee of the National Research Council [59] said:

> The traditional Internet model pushes the intelligence to the edge, and calls for a simple data forwarding function in the core of the network. Does this continue to be the correct model? A number of ad hoc functions are appearing in the network, such as NAT boxes, firewalls, and content caches. There are devices that transform packets, and places where the network seems to operate as an overlay on itself (e.g., virtual private networks). Do these trends signal the need to rethink how function is located within the network? What aspects of modularity need to be emphasized in the design of functions: protocol layering, topological regions, or administrative regions? Is there a need for a more complex model for how applications should be assembled from components located in different parts of the network?

In this book, you will find answers to all of these questions.

We are eager to hear what you think of compositional network architecture—your questions, your comments, your complaints, and what it made you think about. We thank you very much for your attention, hope you have a good time reading, and don't forget to write!

INDEX