Robustness and the Internet: Design and evolution

Walter Willinger and John Doyle [*]

March 1, 2002

Abstract

The objective of this paper is to provide a historical account of the design and evolution of the Internet and use it as a concrete starting point for a scientific exploration of the broader issues of robustness in complex systems. To this end, we argue that anyone interested in complex systems should care about the Internet and its workings, and why anyone interested in the Internet should be concerned about complexity, robustness, fragility, and their trade-offs.

1 Introduction

Despite the widespread use of the Internet and its impact on practically every segment of society, its workings remain poorly understood by most users. Nevertheless, more and more users take it for granted to be able to boot up their laptops pretty much anywhere (e.g., cafes, airports, hotels, conference rooms) and connect to the Internet to use services as mundane as e-mail or Web browsing or as esoteric as music- or movie-distribution and virtual reality games. The few times the users get a glimpse of the complexity of the infrastructure that supports such ubiquitous communication are when they experience various "networking" problems (e.g., the familiar "cannot connect" message, or unacceptably poor performance), because diagnosing such problems typically exposes certain aspects of the underlying network architecture (how the components of the network infrastructure interrelate) and network protocols (standards governing the exchange of data).

Consider, for example, a user sitting in a cafe and browsing the Web on her laptop. In terms of infrastructure, a typical scenario supporting such an application will include a wireless access network in the cafe; an Internet service provider that connects the cafe to the global Internet; intermediate service providers that agree to carry the user's bytes across the country or around the globe, through a myriad of separately administered, autonomous domains; and another service provider at the destination that hosts the server with the Web page requested by the user. As for protocols, successful Web browsing in this setting will require, at a minimum, standards for exchanging data in a wireless environment (cafe) and standards for the (possibly) different networking technologies encountered when transmitting the user's bit stream over the different domains' wired links within the Internet; a standard for assigning a (temporary) ID or address to the user's laptop so that it can be identified and located by any other device connected to the Internet; a set of rules for providing a single, authoritative mapping between names of hosts that provide Web pages such as `www.santafe.org` and their less mnemonic numerical equivalent (e.g., the address 208.56.37.219 maps to the more informative host name `www.santafe.org`); standards for routing the data across the Internet, through the different autonomous domains; a service that ensures reliable transport of the data between the source and destination and uses the available resources efficiently; and a message exchange protocol that allows the user, via an intuitive graphical interface, to browse through a collection of Web pages by simply clicking on links, irrespective of the format or location of the pages' content. The Internet's success and popularity is to a large degree due to its ability to hide most of this

[*]W. Willinger is with AT&T Labs-Research, Florham Park, NJ 07932-0971, e-mail: `walter@research.att.com`. J. Doyle is Professor of Control and Dynamical Systems, Electrical Engineering, and Bio-Engineering, Caltech, Pasadena, CA 91125, e-mail: `doyle@cds.caltech.edu`.

complexity and give users the illusion of a single, seamlessly connected network where the fragmented nature of the underlying infrastructure and the many layers of protocols remain largely transparent to the user. However, the fact that the Internet is, in general, very successfully in hiding from the user most of the underlying details and intricacies does not make them go away! In fact, even Internet experts admit having more and more troubles getting (and keeping) their arms around the essential components of this large-scale, highly-engineered network that has all the features typically associated with complex systems—too complicated and hence ill-understood at the system-level, but with often deep knowledge about many of its individual components; resilient to designed-for uncertainties in the environment or individual components ("robustness"), yet full of surprising behavior ("emergence"), including a natural tendency for infrequently occurring, yet catastrophic events ("fragility"). During the past 20–30 years, it has evolved from a small-scale research network into today's Internet—an economic reality with mission-critical importance for the national and international economies and for society as a whole.

It is the design and evolution of this large-scale, highly-engineered Internet that we use in this article as a concrete example for discussing and exploring a range of issues related to the study of complexity and robustness in technology in general, and of large-scale, communication networks in particular. We argue that the "robust, yet fragile" characteristic is a hallmark of complexity in technology (as well as in biology; e.g., see [14, 16]), and that is not an accident, but the inevitable result of fundamental tradeoffs in the underlying system design. As complex engineered systems such as the Internet evolve over time, their development typically follows a spiral of increasing complexity to suppress unwanted sensitivities/vulnerabilities or to take advantage of new opportunities for increased productivity, performance, or throughput. Unfortunately, in the absence of a practically useful and relevant "science of complexity," each step in this development towards increasing complexity is inevitably accompanied by new and unforeseen sensitivities, causing the complexity/robustness spiral to continue or even accelerate. The Internet's coming of age is a typical example where societal and technological changes have recently led to an acceleration of this complexity/robustness spiral. This acceleration has led to an increasing collection of short-term or point solutions to individual problems, creating a technical arteriosclerosis [4] and causing the original Internet architecture design to become increasingly complex, tangled, inflexible, and vulnerable to perturbations the system was not designed to handle in the first place.

Given the lack of a coherent and unified theory of complex networks, these point solutions have been the result of a tremendous amount of engineering intuition and heuristics, common sense, and trial and error, and have sought robustness to new uncertainties with added complexity, leading in turn to new fragilities. While the Internet design "principles" discussed in Section 2 below constitute a modest theory itself that has benefited from fragmented mathematical techniques from such areas as information theory and control theory, key design issues for the Internet protocol architecture have remained unaddressed and unsolved. However, it is becoming more and more recognized, that without a coherent theoretical foundation, man-made systems such as the Internet will simply collapse under the weight of their patchwork architectures and unwieldy protocols. With this danger lurking in the background, we provide here a historical account of the design and evolution of the Internet, with the ambitious goal of using the Internet as a concrete example that reveals fundamental properties of complex systems and allows for a systematic study and basic understanding of the tradeoffs and spirals associated with complexity, robustness, and fragility. The proposed approach requires a sufficient understanding of the Internet's history; concrete and detailed examples illustrating the Internet's complexity, robustness, and fragility; and a theory that is powerful and flexible enough to help us sort out generally applicable design principles from historical accidents. The present article discusses the history of the design and evolution of the Internet from a complexity/robustness/fragility perspective and illustrates with detailed examples and well-documented anecdotes some generally applicable principles. In particular, when illustrating in Section 3 the Internet's complexity/robustness spiral, we observe an implicit design principle at work that seeks the "right" balance for a horizontal and vertical separation/integration of functionalities through decentralized control ("horizontal") and careful protocol stack decomposition ("vertical"), and tries to achieve, in a well-defined sense, desirable global network behavior, robustness, and (sub)optimality. Motivated by this observation, the companion article [16] outlines an emerging theoretical foundation for the Internet that illustrates the broader role of theory in analysis and design of complex systems in technology and biology and allows for a systematic treatment of the horizontal and vertical separation/integration issue in protocol design.

1.1 Why care about the Internet?

As discussed in more detail in [16], many of the existing "new theories of complexity" have the tendency of viewing the Internet as a generic, large-scale, complex system that shows organic-like growth dynamics, exhibits a range of interesting "emergent" phenomena, and whose "typical" behavior appears to be quite simple. These theories have been very successful in suggesting appealingly straightforward approaches to dealing with the apparently generic complex nature of the Internet and have led to simple models and explanations of many of the observed emergent phenomena generally associated with complexity, for example, *power-law statistics*, *fractal scaling*, and *chaotic dynamics*. While many of these proposed models are capable of reproducing certain trademarks of complex systems, by ignoring practically all Internet-specific details, they tend to generate great attention in the non-specialist literature. At the same time, they are generally viewed as toy models with little, if any, practical relevance by the domain experts who have great difficulties in recognizing any Internet-specific features in these generic models.

We argue here that only extreme circumstances that are neither easily replicable in laboratory experiments or simulations nor fully comprehensible by even the most experienced group of domain experts are able to reveal the role of the enormous complexity underlying most engineered systems. In particular, we claim in this article that by explicitly ignoring essential ingredients of the architectural design of the Internet and its protocols, these new theories of complexity have missed out on the most attractive and unique features that the Internet offers for a genuinely scientific study of complex systems. For one, even though the Internet is widely used, and it is in general known how all of its parts (e.g., network components and protocols) work, in its entirety, the Internet remains poorly understood. Secondly, the network has become sufficiently large and complicated to exhibit "emergent phenomena"—empirical discoveries that come as a complete surprise, defy conventional wisdom and baffle the experts, cannot be explained nor predicted within the framework of the traditionally considered mathematical models, and rely crucially on the large-scale nature of the Internet, with little or no hope of encountering them when considering small-scale or toy versions of the actual Internet. While the Internet shares this characteristic with other large-scale systems in biology, ecology, politics, psychology, etc., what distinguishes it from these and other complex systems and constitutes its single-most attractive feature is a unique capability for a strictly scientific treatment of these emergent phenomena. In fact, the reasons for why an unambiguous, thorough, and detailed understanding of any "surprising" networking behavior is in general possible and fairly readily available are twofold and will be illustrated in Section 4 with a concrete example. On the one hand, we know in great detail how the individual components work or should work and how the components interconnect to create system-level behavior. On the other hand, the Internet provides unique capabilities for measuring and collecting massive and detailed data sets that can often be used to reconstruct the behavior of interest. In this sense, the Internet stands out as an object for the study of complex systems because it can in general be completely "reverse engineered."

Finally, because of the availability of a wide range of measurements, the relevance of any proposed approach to understanding complex systems that has also been claimed to apply to the Internet can be thoroughly investigated, and the results of any theory of complexity can be readily verified or invalidated. This feature of the Internet gives new insight into the efficacy of the underlying ideas and methods themselves, and can shed light on their relevance to similar problems in other domains. To this end, mistakes that are made when applying the various ideas and methods to the Internet might show up equally as errors in applications to other domains, such as biological networks. In short, the Internet is particularly appropriate as a starting point for studying complex systems precisely because of its capabilities for measurements, reverse engineering, and validation, all of which (individually, or in combination) have either explicitly or implicitly been missing from the majority of the existing "new sciences of complexity." Put differently, all the advantages associated with viewing the Internet as a complex system are lost when relying on only a superficial understanding of the protocols and when oversimplifying the architectural design or separating it from the protocol stack.

1.2 Why care about complexity/robustness?

By and large, the Internet has evolved without the benefit of a comprehensive theory. While traditional but fragmented mathematical tools of robust control, communication, computation, dynamical systems, and statistical physics have been applied to a number of network design problems, the Internet has largely been the result of a mixture of good design principles and “frozen accidents,” similar to biology. However, in contrast to biology, the Internet provides access to the designer’s original intentions and to an essentially complete record of the entire evolutionary process. By studying this historical account, we argue here that the Internet teaches us much about the central role that robustness plays in complex systems, and we support our arguments with a number of detailed examples. In fact, we claim that many of the features associated with the Internet’s complexity, robustness, and fragility reveal intrinsic and general properties of all complex systems. We even hypothesize that there are universal properties of complex systems that can be revealed through a scientifically sound and rigorous study of the Internet. A detailed understanding of these properties could turn out to be crucial for addressing many of the challenges facing today’s network and the future Internet, including some early warning signs that indicate when technical solutions that, while addressing particular needs, may severely restrict the future use of the network and may force it down an evolutionary dead-end street.

More importantly, in view of the companion paper [16] where the emergence of a coherent theoretical foundation of the Internet is discussed, it will be possible to use that theory to evaluate the Internet’s evolutionary process itself and to help understand and compare it with similar processes in other areas such as evolutionary biology, where the processes in question can also be viewed as mixtures of “design” (i.e., natural selection is a powerful constraining force) and “frozen accidents” (i.e., mutation and selection both involve accidental elements). Thus, a “look over the fence” at, for example, biology, can be expected to lead to new ideas and novel approaches for dealing with some of the challenges that lie ahead when evolving today’s Internet into a future “embedded, everywhere” world of total automation and network interconnectivity. Clearly, succeeding in this endeavor will require a theory that is powerful enough to unambiguously distinguish between generally applicable principles and historical accidents, between theoretically sound findings and simple analogies, and between empirically solid observations and superficial data analysis. In the absence of such a theory, viewing the Internet as a complex system has so far been less than impressive, resulting in little more than largely irrelevant analogies and careless analysis of available measurements.

1.3 Background and outlook

While none of the authors of this article had (nor have) anything to do with any design aspects of the original (nor present) Internet, much of the present discussion resulted from our interactions with various members of the DARPA-funded *NewArch Project* [4], a collaborative research project aimed at revisiting the current Internet architecture in light of present realities and future requirements and developing a next-generation architecture towards which the Internet can evolve during the next 10–20 years. Some of the members of the *NewArch*-project, especially Dave Clark, who were part of and had architectural responsibilities for the early design of the Internet, have written extensively about the thought process behind the early DARPA Internet architecture and about the design philosophy that shaped the design of the Internet protocols (see for example, [9, 29, 34, 2]). Another valuable source that provides a historical perspective about the evolution of the Internet architecture over time is the archive of *Requests for Comments (RFCs)* of the *Internet Engineering Task Force (IETF)*[1]. This archive offers periodic glimpses into the thought process of some of the leading Internet architects and engineers about the health of the original architectural design in view of the Internet’s growing pains (for some relevant RFCs, see [8, 23, 7, 6, 10]). In combination, these different sources of information provide invaluable insights into the process by which the original Internet architecture has been designed and has evolved as a result of internal and external changes that have led to a constant demand for new requirements, and this article uses these resources extensively.

The common goal between us “outsiders” and the *NewArch*-members (“insiders”) has been a genuine desire for understanding the nature of complexity associated with today’s Internet and using this understanding to move forward. While our own efforts towards reaching this goal has been mainly measurement- and theory-driven (but with an appreciation for the need for “details”), the insiders approach has been more bottom-up, bringing an immense body of system knowledge, engineering intuition, and empirical observations to the table (as well as an appreciation for “good” theory). To this end, the present discussion is intended as a motivation for developing a theory of complexity and robustness that bridges the gap

[1]`http://www.ietf.org/rfc.html.`

between the theory/measurement-based and engineering/intuition-based approaches in an unprecedented manner and results in a framework for dealing with the complex nature of the Internet that (i) is technically sound, (ii) is consistent with measurements of all kinds, (iii) agrees fully with engineering intuition and experience, and (iv) is useful in practice and for sketching viable architectural designs for an Internet of the future. The development of such a theoretical foundation for the Internet and other complex systems is the topic of the companion article [16].

2 Complexity and designed-for robustness in the Internet

Taking the initial technologies deployed in the pre-Internet area as given, we discuss in the following the original requirements for the design of a "network of networks." In particular, we explore in this section how robustness, a close second to the top internetworking requirement, has influenced, if not shaped, the architectural model of the Internet and its protocol design. That is, starting with a basic packet-switching network fabric, we address here the question of how the quest for robustness—primarily in the sense of (i) flexibility to changes in technology, use of the network, etc., and (ii) survivability in the face of failure—has impacted how the components of the underlying networks interrelate, and how the specifics of exchanging data in the resulting network have been designed.[2] Then we illustrate how this very design is evolving over time when faced with a network that is itself undergoing constant changes due to technological advances, changing business models, new policy decisions, or modified market conditions.

2.1 The original requirements for an Internet architecture

When viewed in terms of its hardware, the Internet consists of *hosts* or endpoints (also called end systems), *routers* or internal switching stations (also referred to as gateways), and *links* that connect the various hosts and/or routers and can differ widely in speed (from slow modem connection to high-speed backbone links) as well as in technology (e.g., wired, wireless, satellite communication). When viewed from the perspective of *autonomous systems* (ASs), where an AS is a collection of routers and links under a single administrative domain, the network is an internetwork consisting of a number of separate subnetworks or ASs, interlinked to give users the illusion of a single, seamlessly connected network (network of networks, or "internet"). A network architecture is a framework that aims at specifying how the different components of the networks interrelate. More precisely, paraphrasing [4], a *"network architecture is a set of high-level design principles that guides the technical design of a network, especially the engineering of its protocols and algorithms. It sets a sense of direction—providing coherence and consistency to the technical decisions that have to be made and ensuring that certain requirements are met."* Much of what we refer to as today's Internet is the result of an architectural network design that was developed in the 1970s under the auspices of the Defense Advanced Research Project Agency (DARPA) of the US Department of Defense, and the early reasoning behind the design philosophy that shaped the Internet protocol architecture (i.e., the "Internet architecture" as it became known) is vividly captured and elaborated on in detail in [9].

Following the discussion in [9], the main objective for the DARPA Internet architecture some 30 years ago was *internetworking* – the development of an "effective technique for multiplexed utilization of already existing interconnected (but typically separately administered) networks." To this end, the fundamental structure of the original architecture (how the components of the networks interrelate) resulted from a combination of known technologies, conscientious choices, and visionary thinking, and led to "a packet-switched network in which a number of separate networks are connected together using packet communications processors called gateways which implement a store and forward packet forwarding algorithm" [9]. For example, the store and forward packet switching technique for interconnecting different networks had already been used in the ARPANET and was reasonably well understood. The selection of packet switching over circuit switching as the preferred multiplexing technique reflected networking reality (i.e., the networks to be integrated under the DARPA project already deployed the packet switching technology) and engineering intuition (e.g., data communication was expected to differ fundamentally

[2]There are other important aspects to this quest for robustness (e.g., interoperability in the sense ensuring a working Internet despite a wide range of different components, devices, technologies, protocol implementations, etc.), but since they are not central to our present discussion.

from voice communication and would be more naturally and efficiently supported by the packet switching paradigm). Moreover, the intellectual challenge of coming to grips with integrating a number of separately administered and architecturally distinct networks into a common evolving utility required bold scientific approaches and innovative engineering decisions that were expected to advance the state-of-the-art in computer communication in a manner that alternative strategies such as designing a new and unified but intrinsically static “multi-media” network could have largely avoided.

A set of second-level objectives, reconstructed and originally published in [9], essentially elaborates on the meaning of the word “effective” in the all-important internetworking requirement and defines a more detailed list of goals for the original Internet architecture. Quoting from [9], these requirements are (in decreasing order of importance),

- *Robustness*: Internet communication must continue despite loss of networks or gateways/routers.
- *Heterogeneity (Services)*: The Internet must support multiple types of communications services.
- *Heterogeneity (Networks)*: The Internet architecture must accommodate a variety of networks.
- *Distributed Management*: The Internet architecture must permit distributed management of its resources.
- *Cost*: The Internet architecture must be cost effective.
- *Ease of Attachment*: The Internet architecture must permit host attachment with a low level of effort.
- *Accountability*: The resources used in the Internet architecture must be accountable.

While the top-level requirement of internetworking was mainly responsible for defining the basic structure of the common architecture shared by the different networks that composed the “network of networks” (or “Internet” as we now know it), this priority-ordered list of second-level requirements, first and foremost among them the robustness criterion, has to a large degree been responsible for shaping the architectural model and the design of the protocols (standards governing the exchange of data) that define today’s Internet. This includes the Internet’s well-known “hourglass” architecture and the enormously successful “fate-sharing” approach to its protocol design [29], both of which will be discussed in more detail below.

In the context of the discussions in [9], “robustness” usually refers to the property of the Internet to be resilient to uncertainty in its components and usage patterns, and—on longer time scales (i.e., evolution)—to unanticipated changes in networking technologies and network usage. However, it should be noted that “robustness” can be (and has been) interpreted more generally to mean “to provide some underlying capability in the presence of uncertainty.” It can be argued that with such a more general definition, many of the requirements listed above are really a form of robustness as well, making robustness the basic underlying requirement for the Internet. For example, in the case of the top internetworking requirement, access to and transmission of files/information can be viewed as constituting the “underlying capability,” while the “uncertainty” derives from the users not knowing in advance when to access or transmit what files/information. Without this uncertainty, there would be no need for an Internet (e.g., using the postal service for shipping CDs with the requested information would be a practical solution), but when faced with this uncertainty, a robust solution is to connect everybody and provide for a basic service to exchange files/information “on demand” and without duplicating everything everywhere.

Clearly, the military context in which much of the early design discussions surrounding the DARPA Internet architecture took place played a significant role in putting the internetworking/connectivity and robustness/survivability requirements at the top and relegating the accountability and cost considerations to the bottom of the original list of objectives for an Internet architecture that was to be useful in practice. Put differently, had, for example, *cost* and *accountability* been the two over-riding design objectives (with some concern for internetworking and robustness, though)—as would clearly be the case in today’s market- and business-driven environment—it is almost certain that the resulting network would exhibit a very differently designed architecture and protocol structure. Ironically though, the very commercial success and economic reality of today’s Internet that have been the result of an astonishing resilience of the original DARPA Internet architecture design to revolutionary changes—especially during the last 10 or so years and in practically all aspects of networking one can think of—have also been the driving forces that have started to question, compromise, erode, and even damage the existing architectural framework. The myriad of network-internal as well as network-external changes and new requirements that have accompanied "the Internet's coming of age" [29] has led to increasing signs of strains on the fundamental architectural structure. At the same time, it has also led to a patchwork of technical long-term and short-time solutions, where each solution typically addresses a particular change or satisfies a specific new requirement. Not surprisingly, this transformation of the Internet from a small-scale research network with little (if any) concern for cost, accountability, and trustworthiness into an economic power house has resulted in a network that is (i) increasingly driving the entire national and global economy, (ii) experiencing an ever-growing class of users with often conflicting interests, (iii) witnessing an erosion of trust to the point where assuming untrustworthy end-points becomes the rule rather than the exception, and (iv) facing a constant barrage of new and, in general, ill-understood application requirements and technology features. As illustrated with some specific examples in Section 3 below, each of these factors creates new potential vulnerabilities for the network which in turn leads to more complexity as a result of making the network more robust.

2.2 The DARPA Internet architecture

When defining what is meant by "the Internet architecture," the following quote from [6] captures at the same time the general reluctance within the Internet community for "dogmas" or definitions that are "cast in stone" (after all, things change, which by itself may well be the only generally accepted principle[3]) and a tendency for very practical and to-the-point working definitions: *"Many members of the Internet community would argue that there is no [Internet] architecture, but only a tradition, which was not written down for the first 25 years (or at least not by the [Internet Architecture Board]). However, in very general terms, the community believes that [when talking about the Internet architecture] the goal is connectivity, the tool is the Internet Protocol, and the intelligence is end-to-end rather than hidden in the network."* To elaborate on this view, we note that connectivity was already mentioned above as the top-level requirement for the original DARPA Internet architecture, and the ability of today's Internet to give its users the illusion of a seamlessly connected network with fully transparent political, economic, administrative, or other sort of boundaries is testimony for the architecture's success, at least as far as the crucial internetworking/connectivity objective is concerned. In fact, it seems that connectivity is its own reward and may well be the single-most important service provided by today's Internet. Among the main reasons for this ubiquitous connectivity are the "layering principle" and the "end-to-end argument," two guidelines for system design that the early developers of the Internet used and tailored to communication networks. A third reason is the decision to use a single universal logical addressing scheme with a simple (net, host) hierarchy, originally defined in 1981.

2.2.1 Robustness as in flexibility: The "hourglass" model

In the context of a packet-switched network, the motivation for using the *layering principle* is to avoid implementing a complex task such as a file transfer between two end hosts as a single module, but to instead break the task up into subtasks, each of which is relatively simple and can be implemented separately. The different modules can then be thought of being arranged in a vertical stack, where each layer in the stack is responsible for performing a well-defined set of functionalities. Each layer relies on the next lower layer to execute more primitive functions and provides services to the next higher layer. Two hosts with the same layering architecture communicate with one another by having the corresponding layers in the two systems talk to one another. The latter is achieved by means of formatted blocks of data that obey a set of rules or conventions known as a *protocol*.

[3]On October 24, 1995, the Federal Networking Council (FNC) unanimously passed a resolution defining the term "Internet." The definition was developed in consultation with members of the Internet community and intellectual property rights communities and states: *The FNC agrees that the following language reflects our definition of the term "Internet". "Internet" refers to the global information system that—(i) is logically linked together by a globally unique address space based on the Internet Protocol (IP) or its subsequent extensions/follow-ons; (ii) is able to support communications using the Transmission Control Protocol/Internet Protocol (TCP/IP) suite or its subsequent extensions/follow-ons, and/or other IP-compatible protocols; and (iii) provides, uses or makes accessible, either publicly or privately, high level services layered on the communications and related infrastructure described herein.*

The main stack of protocols used in the Internet is the five-layer *TCP/IP protocol suite* and consists (from the bottom up) of the physical, link, internetwork, transport, and application layers. The *physical layer* concerns the physical aspects of data transmission on a particular link, such as characteristics of the transmission medium, the nature of the signal, and data rates. Above the physical layer is the *link layer*. Its mechanisms and protocols (e.g., signaling rules, frame formats, media-access control) control how packets are sent over the raw media of individual links. Above it is the *internetwork layer*, responsible for getting a packet through an *internet*; that is, a series of networks with potentially very different bit-carrying infrastructures and possibly belonging to different administrative domains. The *Internet Protocol (IP)* is the internetworking protocol for TCP/IP, and its main task is to adequately implement all the mechanisms necessary to knit together divergent networking technologies and administrative domains into a single virtual network (an "internet") so as to enable data communication between sending and receiving hosts, irrespective of where in the network they are. The layer above IP is the *transport layer*, where the most commonly used *Transmission Control Protocol (TCP)* deals, among other issues, with end-to-end congestion control and assures that arbitrarily large streams of data are reliably delivered and arrive at their destination in the order sent. Finally, the top layer in the TCP/IP protocol suite is the *application layer*, which contains a range of protocols that directly serve the user; e.g., TELNET (for remote login), FTP (the *File Transfer Protocol* for transferring files, SMTP (*Simple Mail Transfer Protocol* for e-mail), HTTP (the *HyperText Transfer Protocol* for the World Wide Web).

This layered modularity gave rise to the "hourglass" metaphor for the Internet architecture [29]—the creation of a multi-layer suite of *protocols*, with a generic packet (datagram) delivery mechanism as a separate layer at the hourglass' waist (the question of how "thin" or "fat" this waist should be designed will be addresses in Section 2.2.2 below). This abstract bit-level network service at the hourglass' waist is provided by IP and ensures the critical separation between an ever more versatile physical network infrastructure below the waist and an ever-increasing user demand for higher-level services and applications above the waist. Conceptually, IP consists of an agreed-upon set of features that has to be implemented according to an Internet-wide standard, must be supported by all the routers in the network, and is key to enabling communication across the global Internet so that networking boundaries and infrastructures remain transparent to the users. The layers below the waist (i.e., physical, and link) deal with the wide variety of existing transmission and link technologies and provide the protocols for running IP over whatever bit-carrying network infrastructure is in place ("IP over everything"). Aiming for a somewhat narrow waist reduces for, or even removes from the typical user the need to know about the details of and differences between these technologies (e.g., Ethernet local area networks (LAN), asynchronous transfer mode (ATM), frame relay) and administrative domains. Above the waist is where enhancements to IP (e.g., TCP) are provided that simplify the process of writing applications through which users actually interact with the Internet ("everything over IP"). In this case, providing for a thin waist of the hourglass removes from the network providers the need to constantly change their infrastructures in response to a steady flow of innovations happening at the upper layers within the networking hierarchy (e.g., the emergence of "killer apps" such as e-mail, the Web, or Napster). The fact that this hourglass design predated some of the most popular communication technologies, services, and applications in use today—and that within today's Internet, both new and old technologies and services can coexist and evolve—attests to the vision of the early architects of the Internet when deciding in favor of the layering principle. "IP over everything, and everything over IP" results not only in enormous robustness to changes below and above the hourglass' waist but also provides the flexibility needed for constant innovation and entrepreneurial spirit at the physical substrate of the network as well as at the application layer.

2.2.2 Robustness as in survivability: "Fate-sharing"

Layered network architectures are desirable because they enhance modularity; that is, to minimize duplication of functionality across layers, similar functionalities are collected into the same layer. However, the layering principle by itself lacks clear criteria for assigning specific functions to specific layers. To help guide this placement of functions among the different layers in the Internet's hourglass architecture, the original architects of the Internet relied on a class of arguments, called the *end-to-end arguments*, that expressed a clear bias against low-level function implementation. The principle was first described in [34] and later reviewed in [6], from where the following definition is taken:

"The basic argument is that, as a first principle, certain required end-to-end functions can only be performed correctly by the end-systems themselves. A specific case is that any network, however carefully designed, will be subject to failures of transmission at some statistically determined rate. The best way to cope with this is to accept it, and give responsibility for the integrity of communication to the end systems. [...]

To quote from [[34]], 'The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)' "

In view of the crucial robustness/survivability requirement for the original Internet architecture (see Section 2.1), adhering to this end-to-end principle has had a number of far-reaching implications. For one, the principle strongly argues in favor of an hourglass-shaped architecture with a "thin" waist, and the end-to-end mantra has been used as a constant argument for "watching the waist" (in the sense of not putting on more "weight", i.e., adding non-essential functionalities). The result is a lean IP, with a minimalistic set of generally agreed-upon functionalities making up the hourglass' waist: algorithms for storing and forwarding packets, for routing, to name but a few. Second, to help applications to survive under failures of individual network components or of whole subnetworks, the end-to-end argument calls for a protocol design that is in accordance with the principle of "soft state." Here "state" refers to the configuration of elements within the network (e.g., routers), and "soft state" means that *"operation of the network depends as little as possible on persistent parameter settings within the network"* [29]. As discussed in [6], *"end-to-end protocol design should not rely on the maintenance of state (i.e., information about the state of the end-to-end communication) inside the network. Such state should be maintained only in the endpoints, in such a way that the state can only be destroyed when the endpoint itself breaks."* This feature has been coined "fate-sharing" by D. Clark [9] and has two immediate consequences. On the one hand, because routers do not keep persistent state information about on-going connections, the original design decision of choosing packet-switching over circuit-switching is fully consistent with this fate-sharing philosophy. The control of the network (e.g., routing) is fully distributed and decentralized (with the exception of key functions such as addressing), and no single organization, company, or government owns or controls the network in its entirety. On the other hand, fate-sharing places most "intelligence" (i.e., information about end-to-end communication, control) squarely in the end points. The network's (i.e., IP's) main job is to transmit packets as efficiently and flexibly as possible; everything else should be done further up in the protocol stack and hence further out at the fringes of the network. The result is sometimes referred to as a "dumb network," with the intelligence residing at the network's edges. Note that the contrast to the voice network with its centralized control and circuit switching technology, where the intelligence (including control) resides within the network ("smart network") and the endpoints (telephones) are considered "dumb" could not be more drastic!

2.2.3 Robustness through simplicity: "Internet transparency"

Much of the design of the original Internet has to do with two basic and simple engineering judgments. On the one hand, the design's remarkable flexibility to changes below and above the hourglass' waist is arguably due to the fact that network designers admittedly never had—nor will they ever have—a clear idea about all the ways the network can and will be used in the future. On the other hand, the network's surprising resilience to a wide range of failure modes reflects to a large degree the network designers' expectations and experiences that failures are bound to happen and that a careful design aimed at preventing failures from happening is a hopeless endeavor, creating overly complex and unmanageable systems, especially in a scenario (like the Internet) that is expected to undergo constant changes. Instead, the goal was to design a system that can "live with" and "work around" failures, shows graceful degradation under failure while still maintaining and providing basic communication services; and all this should be done in a way that is transparent to the user. The result was an Internet architecture whose design reflects the soundness and aimed-for simplicity of the underlying engineering decisions and includes the following key features:

- a connectionless packet-forwarding layered infrastructure that pursues robustness via "fate-sharing,"

- a least-common-denominator packet delivery service (i.e., IP) that provides flexibility in the face of technological advances at the physical layers and innovations within the higher layers, and
- a universal logical addressing scheme, with addresses that are fixed-sized numerical quantities and are applied to physical network interfaces.

This basic structure was already in place about 20 years ago, when the Internet connected just a handful of separate networks, consisted of about 200 hosts, and when detailed logical maps of the entire network with all its links and individual host computers were available The idea behind this original structure is captured by what has become known as the "end-to-end transparency" of the Internet; that is, *"a single universal logical addressing scheme, and the mechanisms by which packets may flow between source and destination essentially unaltered"* [7]. In effect, by simply knowing each other's Internet address and running suitable software, any two hosts connected to the Internet can communicate with one another, with the network neither tampering with nor discriminating against the various packets in flight. In particular, new applications can be designed, experimented with, and deployed without requiring any changes to the underlying network. Internet transparency provides flexibility, which in turn guarantees innovation.

2.3 Complexity as a result of designed-for robustness

The conceptual simplicity portrayed by the hourglass metaphor for the Internet's architectural design can be quite deceiving, as can the simple description of the network (i.e., IP) as a universal packet delivery service that gives its users the illusion of a single, Indeed, when examining carefully the designs and implementations of the various functionalities that make up the different protocols at the different layers, highly engineered, complex internal structures emerge, with layers of feedback, signaling, and control. Furthermore, it also becomes evident that the main reason for and purpose of these complex structures is an over-riding desire to make the network robust to the uncertainties in its environment and components for which this complexity was deemed necessary and justified in the first place. In the following, we illustrate this robustness-driven root cause for complexity with two particular protocols, the transport protocol TCP and the routing protocol BGP.

2.3.1 Complexity-causing robustness issues in packet transport: TCP

IP's main job is to do its best ("best effort") to deliver packets across the network. Anything beyond this basic yet unreliable service (e.g., a need to recover from lost packets; reliable packet delivery) is the application's responsibility. It was decided early on in the development of the Internet protocol architecture to bundle various functionalities into the transport layer, thereby providing different transport services on top of IP. A distinctive feature of these services is how they deal with transport-related uncertainties (e.g., lost packets, delayed packets, out-of-order packets, fluctuations in the available bandwidth) that impact and constrain, at a minimum, the reliability, delay characteristics, or bandwidth usage of the end-to-end communication. To this end, TCP provides a reliable sequenced data stream, while the User Datagram Protocol UDP—by trading reliability for delay—provides direct access to the basic but unreliable service of IP[4]. Focusing in the following on TCP[5], what are then the internal structures, and how complex do they have to be, so that TCP can guarantee the service it promises its applications?

For one, to assure that arbitrarily large streams of data are reliably delivered and arrive at their destination in the order sent, TCP has to be designed to be robust to (at least) lost and delayed packets as well as to packets that arrive out of order. When delivering a stream of data to a receiver such that the entire stream arrives in the same order, with no duplicates, and reliably even in the presence of packet loss, reordering, duplication, and rerouting, TCP splits the data into *segments*, with one segment transmitted in each packet. The receiver acknowledges the receipt of segments if they are "in order" (it has already received all data earlier in the stream). Each acknowledgment packet (ACK) also implicitly acknowledges all of the earlier-in-the-stream segments, so the loss of a single ACK is rarely a problem; a later ACK will

[4]Originally, TCP and IP had been a single protocol (called TCP), but conflicting requirements for voice and data applications soon argued in favor of different transport services running over "best effort" IP.

[5]Specifically, we describe here a version of TCP called TCP Reno, one of the most widely used versions of TCP today. For more details about the various versions of TCP, see e.g. Peterson and Davie [31]).

cover for it, as far as the sender is concerned. The sender runs a timer so that if it has not received an ACK from the receiver for data previously sent when the timer expires, the sender will conclude that the data (or all of the subsequent ACKs) was lost and retransmit the segment. In addition, whenever a receiver receives a segment that is out of order (does not correspond to the next position in the data stream), it generates a "duplicate ACK," that is, another copy of the same ACK that it sent for the last in-order packet it received. If the sender observes the arrival of three such duplicate ACKs, then it concludes that a segment must have been lost (leading to a number of out-of-order segments arriving at the receiver, hence the duplicate ACKs), and retransmits it *without* waiting first for the timer to expire[6].

Next, guaranteeing reliability without making use of the available network resources (i.e., bandwidth) would not be tolerated by most applications. A key requirement for attaining good performance over a network path, despite the uncertainties arising from often highly intermittent fluctuations of the available bandwidth, is that the sender must in general maintain several segments "in flight" at the same time, rather than just sending one and waiting an entire round trip time (RTT) for the receiver to acknowledge it. However, if the sender has too many segments in flight, then it might overwhelm the receiver's ability to store them (if, say, the first is lost but the others arrive, so the receiver cannot immediately process them), or the network's available capacity. The first of these considerations is referred to as *flow control*, and in TCP is managed by the receiver sending an *advertised window* informing the sender how many data segments it can have in flight beyond the latest one acknowledged by the receiver. This mechanism is termed a "sliding window," since each ACK of new data advances a window bracketing the range of data the sender is now allowed to transmit. An important property of a sliding window protocol is that it leads to *self-clocking*. That is, no matter how fast the sender transmits, its data packets will upon arrival at the receiver be spaced out by the network to reflect the network's current carrying capacity; the ACKs returned by the receiver will preserve this spacing; and consequently the window at the sender will advance in a pattern that mirrors the spacing with which the previous flight of data packets arrived at the receiver, which in turn matches the network's current carry capacity.

TCP also maintains a *congestion window*, or CWND, that controls how the sender attempts to consume the path's capacity. At any given time, the sender confines its data in flight to the lesser of the advertised window and CWND. Each received ACK, unless it is a duplicate ACK, is used as an indication that data has been transmitted successfully, and allows TCP to increase CWND. At startup, CWND is set to 1 and the *slow start* mechanism takes place, where CWND is increased by one segment for each arriving ACK. The more segments that are sent, the more ACKs are received, leading to exponential growth. (The *slow start* procedure is "slow" compared to the old mechanism which consisted of immediately sending as many packets as the advertised window allowed.) If TCP detects a packet loss, either via duplicate ACKs or via timeout, it sets a variable called the slow start threshold, or SSTHRESH to half of the present value of CWND. If the loss was detected via duplicate ACKs, then TCP does not need to cut back its rate drastically: CWND is set to SSTHRESH and TCP enters the *congestion avoidance* state, where CWND is increased linearly, by one segment per RTT. If the loss was detected via a timeout, then the self-clocking pattern has been lost, and TCP sets CWND to one, returning to the slow start regime in order to rapidly start the clock going again. When CWND reaches the value of SSTHRESH, *congestion avoidance* starts and the exponential increase of CWND shifts to a linear increase[7].

Clearly, these designed-for features that make TCP robust to the randomly occurring but fully expected failure modes in end-to-end communication (i.e., packet loss, packet delay, out-of-order packets, congestion episodes) are not free but come at a price, namely increased complexity. This complexity reveals itself in the type of adopted engineering solutions which in this case include explicit and implicit signaling (use of ACKs and packet loss), heavy reliance on timers and (hopefully) robust parameter estimation methods, extensive use of control algorithms, feedback loops, etc. The result is a protocol that has performed remarkably well

[6]Imagine, for example, that segments 1 and 2 are sent, that 1 is received and 2 is lost. The receiver sends the acknowledgment ACK(1) for segment 1. As soon as ACK(1) is received, the sender sends segments 3, 4, and 5. If these are successfully received, they are retained by the receiver, even though segment 2 is missing. But because they are out of order, the receiver sends back three ACK(1)'s (rather than ACK(3), ACK(4), ACK(5)). From the arrival of these duplicates, the sender infers that segment 2 was lost (since the ACKs are all for segment 1) and retransmits it.

[7]Congestion control as described here using packet loss as a congestion signal and an *additive-increase-multiplicative-decrease*-type congestion control mechanism at the end points was not part of the original TCP protocol but was proposed in [22] and was subsequently added in the late 1980s in response to observed congestion collapse episodes in the Internet; see also Section 3.2.1 below.

as the Internet has scaled up several orders of magnitude in size, load, speed, and scope during the past 20 or so years (see Section 3.1). Much of this engineering achievement is due to TCP's abilities to allocate network resources in a more or less "fair" or socially responsible manner and nevertheless achieve high network utilization through such cooperative behavior.

2.3.2 Complexity-causing robustness issues in packet routing: BGP

Transport protocols such as TCP "don't do routing" and rely fully on IP to switch any packet anywhere in the Internet to the "correct" next hop. Addressing and routing are crucial aspects that enable IP to achieve this impressive task. As for *addressing*, each network uses a unique set of addresses drawn from a single universal logical addressing space. Each device on the Internet has a unique address that it uses to label its network interface. Each IP packet generated by any of these devices has a source and destination address, where the former references the local interface address and the latter gives the corresponding interface address of the intended recipient of the packet. When handing packets over within the network form router to router, each router is able to identify the intended receiver of each packet. Maintaining sufficient and consistent information within the network for associating the identity of the intended recipient with its location inside the network is achieved by means of *routing protocols*; that is, a set of distributed algorithms that are part of IP and that the routers run among themselves to make appropriate routing decisions. The routing protocols are required to maintain both local and global (but not persistent) state information, for each router must not only be able to identify a set of output interfaces that can be used to move a packet with a given destination address closer to its destination, but must also select an interface from this set which represents the best possible path to that destination. Robustness considerations that play a role in this context include randomly occurring router or link failures and restoration of failed network components or adding new components to the network. The routing protocols in use in today's Internet are robust to these uncertainties in the network's components, and the detection of and routing around failed components remains largely invisible to the end-to-end application—the Internet sees damage and "works" (i.e., routes) around it. The complexity in protocol design that ensures this remarkable resilience to failures in the physical infrastructure of the Internet is somewhat reduced by a division of the problem into two more manageable pieces, where the division is in accordance with separation of the Internet into ASs: each AS runs a local internal routing protocol (or *Interior Gateway Protocol (IGP))*, and between the different ASs, an inter-network routing protocol (or *Exterior Gateway Protocol (EGP))* maintains connectivity and is the glue that ties all the ASs together and ensures seamless communication across AS boundaries. To illustrate the engineers' approaches to tackling this routing problem and discuss the resulting complex internal structures, we focus in the following on the *Border Gateway Protocol (BGP)*, the de-facto standard inter-network routing protocol deployed in today's Internet[8].

In a nutshell, BGP is a "path-vector" routing protocols, and two BGP-speaking routers that exchange routing information dynamically with BGP use TCP as its transport protocol. As a distance-vector protocol, each BGP-speaking router determines its "best" path to a particular destination separately from other BGP-speaking routers. Each BGP-speaking router selects the "next hop" to use in forwarding a packet to a given destination based on paths to those destinations advertised by the routers at the neighboring ASs. Routers exchange paths to destinations in order to facilitate route selection based on *policy*: ASs apply individual, local policies when selecting their preferred routes, usually based on the series of ASs that a given route transits. This feature enables an administratively decentralized Internet—using these policies, ASs can direct traffic to ASs with whom they have business relationships, where traditional network routing protocols would have selected the shortest path. As the network undergoes changes (e.g., link failures, provisioning of new links, router crashes, etc.), BGP uses *advertisement* and *withdrawal* messages to communicate the ensuing route changes among the BGP routers. An advertisement informs neighboring routers that a certain path to a given destination is used and typically includes a number of attributes, such as an AS-path that lists all ASs the advertisement message has traversed, starting with the originating destination AS. A withdrawal is an update message indicating that a previously advertised

[8]It is not our intention here to provide a historical account of the development of routing protocols in the Internet. For the purpose of our discussion, any IGP or EGP that were used in the past or are currently in use would be appropriate; we simply use (version 4 of) BGP because it is the prevalent EGP in today's Internet and is creating a rapidly growing body of literature [33, 36].

destination is no longer available. Advertisements can function as implicit withdrawals if they contain a previously announced destination. A basic feature of BGP is that when a BGP-speaking router advertises to its neighboring router that it has a path for reaching a particular destination, the latter can be certain that the former is actively using that path to reach the destination.

In short, BGP uses distributed computation to tackle the inter-domain routing problem and transmits the necessary information by setting up reliable BGP sessions between any two BGP-speaking routers that have to communicate with one another. Thus, by relying on update messages from their neighbors, BGP-speaking routers find the best paths to the different destinations in an iterative manner, which in turn requires careful attention to potential problems with route instabilities (i.e., oscillations) and slow convergence. For example, to rate-limit advertisements, BGP uses timers associated with the *Minimum Route Advertisement Interval (MRAI)* parameter. The value of MRAI is configurable, although the recommended default value is 30 seconds. When a BGP-speaking router sends a route advertisement for a given destination to a neighbor, it starts an instance of this timer. The router is not allowed to send another advertisement concerning this destination to that neighbor until the corresponding timer has expired. This rate limiting attempts to dampen some of the oscillation inherent in the distance-vector approach to routing. Note that while waiting for the MRAI timer to expire, the router in question may receive many update messages from its neighbors and run its best path selection algorithm internally numerous times. As a result, the router can privately enumerate many alternative choices of its best path without burdening its neighbors with all the (uninformative) intermediate steps. Thus, using the MRAI timer reduces the number of updates needed for convergence but adds some delay to the update messages that are sent. Overall, the BGP specifications [33] explicitly mention five timers (all with recommended default values, but required to be configurable). They either ensure that requests for network resources (e.g., BGP session) will time out if not re-requested periodically, or they attempt to mitigate certain known disadvantages (e.g., route oscillation) associated with relying on distance-vector or similar kinds of routing protocols. In general, determining default values for these timers has been mostly guess work, and little (if anything) is known about their effects (individually, or in terms of "feature interactions") on the dynamics of BGP in today's Internet. In practice, however, these engineering solutions to the routing problem have produced routing protocols that have made network operations extremely resilient to hardware failures, and the ability of the Internet to "route around" failures in a way that is largely transparent to the user is one of the big success stories of the DARPA Internet architecture design.

3 The Internet's complexity/robustness spiral "in action"

Starting out some 30 years ago as a small-scale research network that relied on a novel, but still developing, technology, the Internet has experienced a well-documented and widely-publicized transformation into a crucial force that is increasingly driving the national and international economy and is fueled by enormous IT investments on the order of billions of dollars. As for the original Internet architecture discussed in the previous section, this transformation has been accompanied by constant probing and soul-searching by parts of the Internet research community about the role and relevance of architecture or architectural design principles in such a rapidly changing environment. In the following, we briefly discuss some of the changes that have affected various aspects of the Internet and illustrate with examples their impact on the Internet architecture[9]. In particular, we point out a very typical, but in the long term potentially quite dangerous engineering approach to dealing with network-internal and -external changes, namely responding to demands for improved performance, better throughput, or more robustness to unexpectedly emerging fragilities with increasingly complex designs or untested short-term solutions. While any increase in complexity has a natural tendency to create further and potentially more disastrous sensitivities, this observation is especially relevant in the Internet context, where the likelihood for unforeseen "feature interactions" in the ensuing highly engineered large-scale structure drastically increases as the network continues to evolve. The result is a "complexity/robustness spiral" (i.e., robust design $\rightarrow$ complexity $\rightarrow$ new fragility $\rightarrow$ make design more robust $\rightarrow \ldots$) that—without reliance on a solid and visionary architecture—can easily and quickly get out of control. This section is a reminder that the Internet is only

[9]In this context, it is particularly illuminating to directly compare [6] with [8] and obtain a first-hand assessment of the changing nature of the architectural principles of the Internet.

now beginning to experience an acceleration of this complexity/robustness spiral and, if left unattended, can be fully expected to experience arcane, irreconcilable, and far-reaching robustness problems in the not-too-distant future. Such a perspective may be useful when discussing a new architectural design for the Internet, where the current arguments range from completely abandoning the original architecture and starting from scratch, to gradually evolving it; i.e., sustaining and growing it in response to the new and/or changing conditions associated with the Internet's coming of age.

3.1 The changing Internet

For the last 25 or so years, the Internet has experienced changes in practically all aspects of networking one can think of. These changes are well-known and well-documented (see for example [20]) and are generally associated with the Internet's exponential growth and ever-increasing degree of heterogeneity any which way on looks. For example, in terms of its size as measured by the number of hosts connected the Internet, the number grow from a handful of hosts at the end of 1970 to about 100,000,000 in late 2000—an increase of eight orders of magnitude. In terms of distinct networks that the Internet glues together, the number was 3 in the mid-1970s and has increased to more than 100,000 by mid-2001. As far as ASs are concerned, the global Internet currently consists of close to 10,000 separate ASs, all of which are interlinked in a way that gives users the illusion of a single, seamlessly connected network. A closer look at the communication links within the Internet reveals the sort of heterogeneity the network is dealing with at the physical and technological levels. For one, there are wired, wireless, and satellite links; in the wired world, there are legacy copper wires (e.g., for dial-up modems) and state-of-the-art optical fiber (e.g., for high-speed access and backbone links). The bandwidth of these links can be as low as a few kilobits per second (Kbps) or can reach beyond the Mbps or even Gbps range, varying over an impressive 5–7 orders of magnitude. With satellite links being possibly part of an end-to-end communication, latencies can be 2–3 orders of magnitude larger than for typical land-based wired/wireless communication. A myriad of communication technologies (e.g., LANs, ATM, frame relay, WDM), most of which did not even exist when the original Internet architecture was conceived, can be found at the network access layer where they give rise to a patchwork of widely different types of networks, glued together by IP. IP traffic volume has experienced sustained exponential growth for the last decade or two, with the amount of data traffic in all U. S. networks recently exceeding the total amount of voice traffic. At the transport and application layers, in addition to accommodating an increasing number of new and different protocols, a direct consequence of the Internet's explosive growth has been a wide range of different implementations of one and the same version of a given protocol—some of them fully consistent with and others explicitly violating the relevant standards.

From the perspective of users, services and applications, Internet transparency (see Section 2.2.3) can be viewed as the main enabler and engine of innovation at the higher layers because it has allowed users to develop, experiment with, and deploy new applications without third-party involvement. Using "best efforts," the network simply forwards packets, irrespective of ownership (*who* is sending) and content (*what* is sent), and any enhancements to this basic service must be part of the application and is hence the responsibility of the application designer, not of the network. The end-to-end application neither expects nor relies on support from the intermediate networks that goes beyond basic IP, and in return, the application is ensured that its packets will neither be restricted nor tampered with in flight across the Internet. The fact that at the IP layer, the network is "open" and "free" in the sense of a "commons" has made it possible for killer applications such as e-mail, the Web, or Napster-like services to emerge, flourish, and dominate, despite their often esoteric and ad-hoc origins. They would most likely never have materialized, had the early designers of the Internet decided to compromise this transparency or "openness" by optimizing the network for some known (e.g., telephony) or some to-be-designed application or use. In turn, this commons has created a new competitive market place where an increasing number of increasingly diverse users form fast-growing new communities and interest groups with often vastly different or even opposing objectives, transform existing organizations and businesses, start new companies and develop novel business models, and impact society and world economy as a whole.

With the explosive growth and extreme heterogeneity in its user base and in the ways it has been used, the Internet has experienced a "changing of the guards"—the emergence of powerful new players and the diminishing role of previously dominating constituents and decision makers. For example, by decommissioning the NSFNet in April of 1995[10], NSF created a business opportunity that allowed commercial Internet Service Providers (ISP) to enter and ultimately dominate the Internet business. It also meant a withdrawal of NSF as one of the major policy makers and pioneers for Internet-related activities and research and a transfer of power and responsibility to the private sector. This transfer was accompanied by a re-evaluation of the rank-ordered requirements on the original list of design objectives for an Internet architecture (see Section 2.1). In particular, while accountability was on the original list—way at the bottom, though—it clearly never had a major impact on shaping the architectural design. However, with the power and responsibility for operating, managing, and controlling the global Internet in the hands of private companies that try to compete in the open market place and run a profitable business, accountability in the sense of accurately measuring and efficiently collecting information that is key for running a viable business takes center stage. Similarly, while not even on the list of the original objectives, security and trustworthiness have increasingly become two of the key new requirements, capable of influencing the success (or lack thereof) of new applications and services as they compete in the market place. While in the early days of the Internet, everybody knew one another or relied on the other side to behave as desired, such trustworthy behavior can clearly no longer be assumed; in fact, we have already moved beyond the point where we no longer trust most of the Internet hosts we interact with. In theory, anyone connected to the Internet is vulnerable to malicious attacks and at risk that valuable information will be lost, stolen, corrupted, or misused, or that individual computer, whole intra-networks, or the entire Internet infrastructure will be disrupted. Many of the successful and widely-publicized virus or denial-of-service attacks have exploited flaws and vulnerabilities inadvertently created and distributed by the system vendors and have demonstrated the potential for large-scale economic consequences, especially for enterprises such as eBay and Amazon for which the Internet has reached mission-critical status[11]. Accountability, security, and trustworthiness are examples of requirements that played little or no role in the design of the original Internet architecture but have become more and more crucial during the Internet's coming of age, reaching a similar level of importance that was given to the connectivity and robustness requirements during the early design stages of the Internet architecture. However, incorporating these emerging requirements as "afterthoughts" into the existing architecture has created considerable conceptual and technical challenges and no satisfactory solution exists to date. For other emerging requirements such as various types of quality of service or mobility, where the latter is largely motivated by the increasing penetration of wireless communication and requires the Internet architecture to support ubiquitous mobility, similar observations apply.

3.2 Fragilities that lead to increased complexity: Protocols

The story is well told how the original DARPA Internet architecture gave rise to a network that has been remarkably successful in handling the various "scaling" or growth-related challenges mentioned above, including (i) the sustained exponential growth in the number of users, hosts, and networks that make up the Internet, (ii) the exponential growth in link bandwidth and the increasing heterogeneity in networking technologies, and (iii) the extreme heterogeneity associated with the ways the network is used. This engineering achievement is testimony to the visionary thinking of the original architects of the Internet that gave rise to a network which has succeeded in providing a highly resilient packet delivery service despite experiencing sustained exponential growth over extended periods in time and with respect to a number of key networking components. In fact, a recently published document by a committee of networking experts concluded that despite all the internal and external changes that the Internet has faced during the past 30 or so years, *" [...] the Internet is fundamentally healthy and that most of the problems and issues discussed [...] can be addresses and solved by evolutionary changes within the Internet's current architectural framework and associated processes"* [29].

What are then the main reasons for why the original design of the Internet architecture has enabled

[10]In 1986, the National Science Foundation (NSF) established the National Science Foundation Network (NSFNet) to link six of the nations super-computer centers through 56 kbps lines. In 1990, the ARPANET was officially dissolved, and responsibility for the Internet was passed to the NSFNet. NSF managed the NSFNet from 1987 to 1995, during the first period of explosive growth in the Internet.

[11]Distributed denial-of-service is an example of the fragility created by the end-to-end and trusted endpoint principles. Here the robustness of a simple network creates the fragility to trusted endpoints. In fact, one can argue that the biggest fragility in the Internet is the designed-for easy access that the endpoints are granted.

the network to repeatedly adapt to and continue normal operation despite of the various scaling-related challenges that it has encountered in the process of "growing up"? Furthermore, is it possible to distill the essence of the many engineering solutions that have been proposed within or outside of the framework of the original architectural design and that have been implemented in the network to address problems related to the observed scaling phenomena? To answer these and related questions, we take in the following a closer look at the evolution in time of three of the key protocols in today's Internet: TCP, HTTP, and BGP.

3.2.1 Towards an ever more efficient TCP

Skipping the first important TCP-related change, namely the separation in the late 1970s of TCP into an inter-network layer protocol (IP) and transport-layer protocol (TCP), in version 4 of the respective protocols, RFC-793 [32] describes the functionalities that TCP is required to perform in order to achieve its intended goal—*"to be used as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks."* Moreover, [32] specifies how the required functionalities (i.e., basic data transfer, reliability, flow control, multiplexing, connections, and precedence and security) are implemented and what engineering solutions are used to achieve the aimed-for robustness of TCP. These solutions consist of a range of techniques from control theory, including signaling and feedback (in the form of ACKs), window-based flow control algorithms (so as to not overwhelm the receiver with packets), timers (e.g., for retransmissions), and low-pass filters (for simple RTT estimation used in the context of determining the retransmit timer). The complexity resulting from implementing these engineering solutions makes TCP robust to the uncertainties that are inherent in basic packet transmission (i.e., damaged, lost, corrupted, duplicated, or out-of-order packets, and apparent mismatch between sending and receiving rates) and for which such complexity was selected in the first place.

The "fragility" or vulnerability of this design to rare or unexpected perturbations became apparent in October of 1986, when the network had the first of what became a series of "congestion collapses"—completely unexpected, sudden and drastic (e.g., factor-of-thousand) drops in throughput [22]; while the network continued to transmit packets, the majority of them were retransmission, thereby lowering the effective throughput over the affected links. Subsequently, Van Jacobson [22] identified in the then-prevailing TCP implementations various shortcomings that contributed to these episodes of severe congestion; for example, connections that don't get to equilibrium, i.e., running in a stable mode with a full window of data in transit, either because they are too short or because they restarted after a packet loss. To make TCP robust to this new fragility, he also proposed in [22] a number of engineering solutions that were soon thereafter incorporated into RFC-1122 [3], which lists the requirements for a conformant implementation of TCP as of 1989[12]. The proposed solutions consisted of adding new levels or modifying the existing layer of signaling, feedback, and control; their implementations relied on using new algorithms, especially for RTT variance estimation, for performing exponential retransmit timer backoff, for slow-start, and for dynamic window sizing on congestion (also known as congestion avoidance). The result has been a more complex TCP design that appears to be very resilient to Internet congestion and hence to the fragility for which this increased complexity was deliberately designed for. As we will illustrate in more detail below, this more complex TCP design is inevitably prone to new fragilities, especially in terms of unexpected feature interactions with higher-layer protocols. In this sense, the evolution of TCP illustrates very clearly the intrinsically "robust, yet fragile" character of complex systems design and serves as a concrete example of the complexity/robustness spiral "in action".

Another aspect that not only fuels but typically accelerates this complexity/robustness spiral for TCP is the ever-present push for more efficient and improved throughput, performance, or productivity. In an environment like the Internet, where assumptions underlying the basic TCP design and usage can (and have) suddenly become invalid—even though they appeared at some point in time to be rock-solid and fundamentally sound—as a result of the changing nature of the network, striving for improved efficiency in TCP operations is like trying and hit a constantly moving target. Changes in the key assumptions are generally associated with reduced TCP efficiency and have been a constant source for embellishments to

[12]Since 1989, a number of additional changes to TCP have been made, but they are not central to our discussion and will not be discussed further.

TCP. Without going into the details of these embellishments, there are roughly three separate categories. First, there are the TCP improvements that leave the basic TCP design in tact but attempt to improve TCP's efficiency by using existing "knobs" (i.e., parameter settings) to make TCP more aggressive when it comes to grabbing available bandwidth. The second category consists of a number of different flavors of TCP (e.g., Tahoe, Vegas), where the different flavors are characterized in terms of the type of algorithms that are used for such mechanisms as retransmission, congestion avoidance, and slow start. In contrast to these first two categories, which only alter the actions of the end hosts (by making the proposed changes to the TCP protocol) but not that of the routers, the third type of TCP improvements requires TCP "helpers" (e.g., random early detection, or RED; explicit congestion notification, or ECN). While residing within the network (i.e., in each router) and thus expanding the hourglass' waist, these "helpers" assist TCP in optimizing end-to-end performance. The optimization typically involves some sort of active queueing management (AQM) such as altering the queueing behavior internal to each router by means of, for example, selectively marking or dropping packets.

Another fragility has come with the increasing penetration of wireless communication. TCP was designed for wired networks, and its design makes numerous assumptions that are typical for wired environments. For example, TCP assumes that packet loss is an indication of network congestion, rather than corrupted bits. To provide seamless internetworking between the wired and wireless worlds, TCP has been augmented with auxiliary protocols for (link-layer) local coding and retransmission, but the interactions, for example, between link-level retransmission and transport-level retransmission remain largely ill understood.

Common to all these examples is that the improvements and modifications typically result in a more complex TCP design which, in turn, manifests itself in the implementation of additional mechanisms, more sophisticated algorithms, and more versatile signaling and feedback strategies. At the same time, being specifically designed to enhance TCP performance by exploiting (potentially transient) changes in the assumptions that underlied the early design of TCP, the resulting improvements also tend to make the ensuing TCP design more brittle and vulnerable, primarily to a range of yet unknown or ill-understood feature interactions that are inevitable, given the degree of complexity of these improvements and the scale of the network where they are deployed. After all, aspects that include the size of the network, finite link capacities, and design features drive the dynamics of protocols such as TCP and couple the different simultaneous connections sharing a given link in intricate and complex ways, introducing significant and complicated interactions in time, among active connections, and across the different networking layers. Thus, the creation of new fragilities due to unknown feature interactions at all layers looms as a consequence of a possibly mis-guided quest for improved TCP performance.

3.2.2 Improving Web efficiency: HTTP

The "robust, yet fragile" character of the Internet can also be observed when moving from the transport layer to the application layer. To this end, we briefly consider the evolution of the design of the *Hypertext Transfer Protocol (HTTP)*. HTTP is an application-layer transfer protocol that relies on TCP for transport and is used by the World Wide Web (WWW) to retrieve information from distributed servers. In contrast to TCP where a considerable effort went into getting the design "right" before deploying the protocol in the network, HTTP is often referred to as a "success disaster"—an insatiable demand (fueled by the explosive popularity of the Web) for a protocol that had escaped into the real world, without any systematic evaluation of its design and without a careful assessment of its potential side effects. Fittingly, RFC-1945 [1], a document that describes most of the component specifications and implementations for HTTP (or HTTP/1.0, as it is commonly referred to), only appeared in 1996, when there were already millions of Web users and when Web-related traffic had started to dominate in the Internet.

Conceptually, HTTP/1.0 is a simple request-response protocol—the client establishes a connection to the remote server and then issues a request; in turn, the server processes the request, returns a response, and closes the connection. Much of the complexity in the design specifications of HTTP/1.0 as documented in [1] is due to ensuring ease-of-use, scalability, and flexibility. Robustness in HTTP/1.0 is largely addresses by relying on TCP for data transport, and therein lies the cause for a fragility in its design that became more and more obvious with the explosive growth of the Web. This fragility became known as the "World Wide Wait" phenomenon—a slowdown of Web performance to the point where user-perceived end-to-end performance becomes unacceptable. The root cause of this phenomenon was already known in 1994, when it was pointed out in [35] that certain design aspects of HTTP/1.0 interacted badly with TCP, and that these feature interactions caused problems with user-perceived latency as well as with server scalability. However, the problem was essentially ignored until after 1996, when the Internet's performance problems suddenly became a reality due to the Web's explosive growth.

To illustrate the unanticipated design mismatch between HTTP/1.0 and TCP, note that a typical Web page includes embedded images. Since HTTP/1.0 cannot ask for multiple objects (e.g., embedded images) within the same request (and hence within the same TCP connection), fetching each object corresponds to a separate request, and hence separate TCP connection. To make things worse, while much of TCP's original design was concerned with long-lived connections (e.g., transferring large files), actual TCP connections, and in particular TCP connections commonly encountered in an HTTP/1.0 message exchange, are short-lived (i.e., consisting of just a few packets). As a result, most objects are transferred before their corresponding TCP connections complete their slow start phase; that is, much of HTTP/1.0 traffic is transported when TCP is at its least efficient. Thus, instead of relying on the strengths of TCP, HTTP/1.0 succeeded in the opposite, i.e., designing for its weaknesses. This and numerous other fragilities in the design of HTTP/1.0 have been addressed and resolved in RFC-2616 [18] that specifies the requirements for version 1.1 of HTTP, i.e., HTTP/1.1. [18] describes a considerably more complex design than that given in [1], and while succeeding in aligning HTTP operations more closely with TCP (e.g., through the introduction of persistent connections and pipelining), the new protocol supports features and provides for mechanisms that may well give rise to new fragilities in the form of unexpected feature interactions (i.e., with IP-related functionalities), unanticipated performance bottlenecks, or cascading failure events. For more details about HTTP, its evolution from HTTP/1.0 to HTTP/1.1, protocol descriptions for both versions, etc., we refer to [24], where further feature interactions between HTTP and TCP are discussed as well.

3.2.3 Coping with scale: BGP

BGP was build on experience gained with EGP and EGP usage in the NSFNet backbone. Each AS uses an interior routing system to maintain a coherent view of the connectivity within the AS, and uses BGP to maintain adjacency information with it's neighboring ASs, thereby creating a global view of the connectivity of the system. As far as BGP is concerned, most of the "fragilities" encountered during its evolution from EGP to (version 4) of BGP and its numerous subsequent modifications have been due to scaling issues, e.g., exponential growth of the number of Internet hosts, ASs, routing table entries, or the size of individual ASs. For example, as far as the latter property is concerned, originally, BGP deployments were configured such that that all BGP speakers within the same AS had be fully meshed so that any external routing information had to be redistributed to all other routers within that AS. This feature of BGP as specified in [33] led to well-documented scaling problems and was subsequently rectified by the introduction and specification of BGP "route reflection" (e.g., see [36]).

A more serious scaling-related fragility of BGP that does not seem to have an easy solution concerns the problem caused by the explosive growth of the sizes of BGP routing tables, especially in the backbone. Recall that the routing tables used by BGP contain network-wide connectivity information. Connectivity is expressed as a preference for "shortest paths" to reach any destination address, modulated by the connectivity policies used by the different ASs. Each entry in the routing table refers to a distinct route. The attributes of the route are used to determine the "best" path from a given AS to the AS that is originating the route. Here, determining the "best" path means using an AS as the basic element of computing and finding out which routing advertisement and associated next AS hop address are the most preferred. The elements of the BGP routing domain are routing entries, expressed as a span of addresses; all addresses within each routing entry share a common origin AS and a common connectivity policy. The total size of the BGP routing table is thus a metric of the number of distinct routes within the Internet, where each route describes a contiguous set of addresses which share a common origin AS and a common reachability policy. As the size of the Internet increases and more and more devices need addresses and seek connectivity—resulting in a growing number of distinct provider networks and ASs and, in turn, an ever increasing number of distinct connectivity policies—the task of maintaining coherent reachability information becomes more formidable and results in ever larger routing tables [21]. However, the limited capabilities (e.g., memory) of even the latest routing devices mandate that this growth must be managed, and herein lies a vulnerability that the original BGP design did not anticipate.

This routing table scaling crises became acute in 1994/95 when the explosive demand for Internet address space (largely fueled by an insatiable demand for top-level domain names for Web businesses) threatened to overwhelm the capacities of the then-available routers. This fear led to the deployment of *Classless Inter-Domain Routing (CIDR)*[13] which provides some control over the number of routes a router has to remember. This was achieved by assigning addresses in a hierarchical manner that forces addresses to be assigned in blocks. The idea behind CIDR was to support hierarchical provider address aggregation; that is, a network provider is assigned an address block from an address registry, and the provider announces this entire block into the exterior routing domain as a single routing entry with a single connectivity policy. Customers of this provider use a sub-allocation from this address block , and these smaller routing entries are aggregated by the provider and are not directly passed into the exterior routing domain. By providing a new set of mechanisms and functionalities for supporting CIDR, this engineering solution was incorporated into BGP in version 4 (BGP4) and has somewhat contained the observed growth of the routing tables. However, more recent measurements suggest another onset of rapid routing table growth around year 2000 [21].

While scaling has been (and is likely to continue to be) the root cause for many of the BGP-induced fragilities, there are examples of other types of BGP-related vulnerabilities that add to the picture of a "live" Internet complexity/robustness spiral. For one, the observed slow convergence of BGP [25] remains a topic of current research, and while it is suspected to be influenced by certain BGP design features (e.g., choice of timer values, tie-breaking techniques), as of yet, no solutions have been proposed or implemented. There have been many examples of modifications to and extension of BGP4 (e.g., AS confederations, BGP communities) that increase the complexity of the protocol by providing new functionalities and, at the same time, reduce the management complexity of maintaining the Internet. Another example deals with various security-related considerations, including an increasing awareness for the need of a BGP that is secure against disruptions such as accidental BGP mis-configurations (which, ironically, become more likely as BGP as a protocol becomes more complex and difficult to configure) and malicious attacks that are directly or indirectly targeted at compromising local or global connectivity information stored in the routing tables. In this context, a recent (preliminary) study [11] of the impact of the Internet propagation of Microsoft worms (such as Code Red and Nimda) on the stability of the global routing system is particularly illuminating. The study offers compelling evidence for a strong correlation between periods of worm propagation and BGP message storms, causing widespread degradation in the end-to-end utility of the global Internet. While most of the traffic appeared to traverse the Internet's backbone links relatively unperturbed, most of the links at the Internet edge appeared to suffer serious performance degradation during the worms' probing and propagation phases. The prime suspects listed in [11] for this undesirable network behavior include congestion-induced failures of BGP sessions due to timeouts; flow-diversity induced failures of BGP sessions due to router CPU overloads; pro-active disconnection of certain networks (especially corporate networks and small ISPs); and failures of other equipment at the Internet edge (e.g., DSL routers and other devices). In either case, the total amount of BGP message traffic appears to grows exponentially with the number of edge domains that feel the effects of the worm. Compared to point failures in the core Internet infrastructure, such as power outages in telco facilities or fiber cuts, which are fully expected to occur and represent classic instances of designed-for vulnerabilities, these worm-induced instabilities of the Internet's routing system define a completely novel class of BGP-related fragilities. These new problems will require timely and effective solutions, or next-generation worms causing cascading BGP storms that will seriously degrade not only the access side of the network but also its core loom as real possibilities.

3.3 Fragilities that lead to increased complexity: Address space

Recall that a key feature of the original Internet architecture design was a single universal logical addressing scheme, with addresses that are fixed-sized (i.e., 32 bits) numerical quantities and are applied to physical

[13]Originally, the IP address space included a concept of *classes* of networks (Classes A–E, with a flat space of host addresses within each class), and a standard engineering solution is—as soon as scaling problems arise—to add hierarchy where there was none.

network interfaces (for naming the node and for routing to it). With 32 bits of address space, a maximum of some 4 billion different hosts can be addressed, although in practice, due to some inefficiency in address space allocation, a few hundred million addressable devices is probably a more realistic limit. In any case, while at the time of the design of the original Internet architecture, 4 billion (or 400 million) was considered to be a safe enough upper limit that justified the "hard-wiring" of 32 bits of address space into the Internet's design, the explosive growth of the network soon identified the very design assumption of a single fixed-sized address space as the main culprit behind a very serious but completely unpredicted fragility—the potential exhaustion of available IP address space. The alarming rate of IP address depletion as a result of the tremendous demand for and popularity of top-level domain names for Web businesses increased the overall awareness for this fragility that was simply a non-issue in the original design of the Internet architecture. Starting in the early 1990s, when the problem of IP address space exhaustion was first discussed, either in terms of being imminent or as a real possibility for the not-too-distant future, a number of engineering solutions have been considered for dealing with this problem. In the following, we illustrate some of these solutions and discuss how the various proposed designs contribute to an acceleration of the Internet's complexity/robustness spiral.

3.3.1 Keep architecture, change IP: IPv6

The obvious and straight-forward solution to the (real or perceived) IP address space exhaustion problem is to increase the address space. Version 6 of IP, or IPv6 as it became known, takes this approach, and its design includes 128 bits of address space. IPv6 is specified in RCF-1883 [15] and provides a number of enhancements over the existing IP (i.e., IPv4). The appealing feature of IPv6 is that conceptually, it simply represents a new generation of the IPv4 technology and fully conforms to the original Internet architecture. Unfortunately, the simplicity of this solution is deceiving, and the complications (and costs) arising from a potential transition of the Internet from IPv4 to IPv6 are substantial and have been one of the major road blocks for a widespread deployment of IPv6 to date.

While IPv6 promises to restore the simplicity as well as expand the functionality of the original Internet architecture, the increased complexity and new fragilities associated with the next-generation IP protocol are due to the need for a smooth transition period during which IPv4 and IPv6 have to be able to co-exist. In the absence of viable alternatives, such a transition period will result in doubling the number of service interfaces, will require changes below and above the hourglass' waist, and will inevitably create interoperability problems. Historically, the Internet has experienced only one such comparable transition phase, namely on January 1983, when there was a transition of the ARPANET host protocol from the original Network Control Protocol (NCP) to TCP/IP [26]. As a "flag-day" style transition, all hosts were required to convert simultaneously, and those that didn't convert had to communicate via rather ad-hoc mechanisms. The transition was carefully planned over several years before it actually took place and went very smoothly. However, what worked in 1983 when the network connected some 300 hosts that required changes, simply does not scale to a network with a few 100,000,000 hosts. While the Internet has responded well to incremental deployment of new technologies "out" at the edges of the network and "up" in the protocol stack, scalable solutions to the problem of deploying new technologies that require changes "inside" the network (e.g., at the IP-layer) are largely non-existent.

3.3.2 Change architecture, keep IP: NAT

An alternative approach to dealing with the IP address space exhaustion problem is to work within the framework of the current IP protocol (IPv4) and allow, if necessary, for new features or capabilities, even if they explicitly compromise the original Internet architecture. For example, address reuse is an obvious method for solving the address depletion problem (at least in the short term), and motivated the introduction of Network Address Translators (NAT) [17]. A NAT is a network device that is placed at the border of a stub domain or intranet (e.g., private corporate network) and has a table consisting of pairs of local IP addresses and globally unique addresses. Since the IP addresses inside the stub domain are not globally unique and are reused in other domains, NATs were envisioned to ease the growth rate of IP address use. A significant advantage of NAT boxes is that they can be deployed without any changes to the network (i.e., IP). However, NAT boxes directly violate the architectural principle of "Internet transparency" (see Section 2.2.3), whereby addresses in packets are not compromised during transport and are carried unchanged from source to destination. While the arguments in favor of or against Internet transparency often take on religious tones, the loss of transparency due to NAT is not just academic (see also [2] for a more thorough assessment of this problem in the context of the “end-to-end arguments”). In fact, the deployment of NATs adds up to complexity in applications design, complexity in network configuration, complexity in security mechanisms, and complexity in network management. For example, NATs break all applications that assume and rely on globally unique and stable IP addresses. If the full range of Internet applications is to be used, complex work-around scenarios are needed; e.g., NATs have to be coupled with Application Layer Gateways (ALG), inserted between application peers to simulate a direct connection when some intervening protocol or device prevents direct access. ALGs terminate the transport protocol and may modify the data stream before forwarding; e.g., identify and translate packet addresses. Thus, new applications may need NAT upgrades to achieve widespread deployment, which could have far-reaching implications on the Internet’s ability in the future to support and guarantee flexibility and innovation at the application layer.

The NAT technology was introduced in 1994 with the specific concern that *“NAT has several negative characteristics that make it inappropriate as a long term solution, and may make it inappropriate even as a short term solution”* [17]. Even though the need to experiment and test what applications may be adversely affected by NAT’s header manipulations before there was any substantial operational experience was articulated loud and clear in [17], networking reality is that there has been widespread deployment and use of NATs in the Internet. Reasons for this popularity of NAT as a technology[14] include (i) lower address utilization and hence a delay for IPv6 roll-out, (ii) ease of renumbering when providers change, (iii) masks global address change, and (iv) provides a level of managed isolation and security (typically as part of an already installed firewall system) between a private address realm and the rest of the Internet. In this sense, NAT is another example of a “success disaster,” but in comparison to HTTP, the architectural and practical implications are possibly more severe. For one, NAT boxes move the Internet from the traditional peer-to-peer model, where any host is in theory addressable, towards a client-server model, in which only certain hosts are addressable from the global Internet. The potential that NAT—as a solution to the address depletion problem—severely curtails future usability of the network and directs the evolution of the Internet into a dead-end street looms as a real possibility. Furthermore, due to their immense popularity and widespread deployment, NAT boxes make diagnosing network faults much harder and introduce many more single points of failure, which in turn increases the fragility of entire network.

4 Outlook

Despite the drastic changes that the Internet has experienced during the transition from a small-scale research network to a mainstream infrastructure, its ability to scale (i.e., adapt to explosive growth), to support innovation (i.e., foster technological progress below and above the hourglass’ waist), and to ensure flexibility (i.e., follow a design that can and does evolve over time in response to changing conditions) has been remarkable. The past 20–30 years have been testimony to the ingenuity of the early designers of the Internet architecture, and the fact that by and large, their original design has been maintained and has guided the development of the network through a “sea of change” to what we call today’s Internet is an astounding engineering achievement. However, as argued in Section 3 above, the Internet is only now beginning to experience changes that pose serious threats to the original architectural design. Moreover, these threats have a tendency to create a patchwork of technical solutions that, while addressing particular needs, may severely restrict the future use of the network and may force it down an evolutionary dead-end street. To correct this development, theoretical foundations for large-scale communication networks and an analytical theory of protocol design are needed for formulating and addressing pertinent questions such as “What are fundamentally sound and practically relevant architectural design guidelines for escaping the presently observed complexity/robustness spiral and for developing effective engineering approaches that contain the inevitable fragilities of the anticipated network infrastructures, but don’t automatically increase complexity?” or “What sort of protocol design principles will lead to what sort of tradeoffs

[14]Beyond its intended use, the NAT technology has recently also been used as an alternative to BGP for route-optimization in multi-homing, where businesses use two or more diverse routes to multiple ISPs to increase robustness to peering link failures.

between complexity and robustness, and will these designs ensure provably scalable protocols?"

To this end, note that one way to calibrate a theory of the Internet and the level of comfort that it provides for understanding today's Internet is by the number of instances with which networking researchers observe "emerging phenomena"—measurement-driven discoveries that come as a complete surprise, baffle (at first) the experts, cannot be explained nor predicted within the framework of the traditionally considered mathematical models, and rely crucially on the large-scale nature of the Internet, with little or no hope of encountering them when considering small-scale or toy versions of the actual Internet. At the same time, we are here interested in theoretical foundations of the Internet that allow for a systematic and integrated treatment of both the "horizontal" (spatially distributed) and the "vertical" (protocol stack) aspect of the Internet—a requirement that emerges very clearly from our studies of the original design and evolution of the Internet in the previous sections. To this end, we revisit the arguments made in Section 1 for why the Internet is unique as a starting point for a scientific study of complex systems and illustrate them in the context of a "classic" example of an Internet-related emergent phenomenon—self similar Internet traffic.

Intuitively, the self-similarity discovery refers to the empirical finding that measured traffic rates on links in the Internet (i.e., number of packets or bytes that traverse a given link per time unit) exhibit *self-similar* (or "fractal-like") behavior in the sense that a segment of the traffic rate process measured at some time scale looks or behaves just like an appropriately scaled version of the traffic rate process measured over a different time scale; see [27] and the important follow-up studies [30] and [13]. More precisely, these and numerous subsequent empirical studies described pertinent statistical characteristics of the temporal dynamics of measured traffic rate processes and provided ample evidence that measured Internet traffic is consistent with *asymptotic second-order self-similarity* or, equivalently, *long-range dependence (LRD)*; i.e., with autocorrelations $r(k)$ that decay like a power for sufficiently large lags k:

$$r(k) \sim c_1 k^{-\beta}, \quad \text{as } k \to \infty,$$

where $0 < \beta < 1$ and c_1 is a positive finite constant. These empirical findings were in stark contrast to conventional wisdom and to what traditionally-used models assumed about actual Internet traffic, namely exponentially-fast decaying autocorrelations and, in turn, a classical white noise behavior when measuring traffic rates over large time scales[15]. Moreover, the ease with which it was possible to distinguish between assumed and actual traffic rate processes was surprising as well as striking.

The observed self-similar characteristic of Internet traffic constitutes a classic "emergent" phenomenon because, as an ubiquitous scaling property, it is apparently not put in by hand or design and thus invites a range of potential explanations, from being a consequence of dynamical instabilities or bifurcations (where the details of the system's design, architecture, and components are largely irrelevant), all the way to arising naturally within the confines of the Internet's hourglass protocol architecture (where it can be understood in terms of underlying architectural guidelines and design principles). Consider for example the following mathematical construction that fits in well with the layering architecture of the Internet. At the application layer, *sessions* (i.e., FTP, HTTP, TELNET) arrive at random (i.e., according to some stochastic process) on the link and have a "lifetime" or session length during which they exchange information. This information exchange manifests itself for example at the IP layer, where from the start until the end of a session, IP packets are transmitted in some bursty fashion. Thus, at the IP layer, the aggregate link traffic measured over some time period (e.g., 1 hour) is made up of the contributions of all the sessions that—during the period of interest— actively transmitted packets. Mathematically, this construction is known to give rise to LRD or, equivalently, asymptotic second-order self-similarity, provided the session arrivals follow a Poisson process and, more importantly, the distribution $F(x)$ of the session "lifetimes" T are *heavy-tailed with infinite variance*; that is, as $x \to \infty$,

$$1 - F(x) = P[T > x] \sim c_2 x^{-\alpha},$$

where $1 < \alpha < 2$ and c_2 is a positive finite constant. Originally due to Cox [12], the main ingredient of this construction is the heavy-tailedness of the session durations. Intuitively, this property implies that there

[15]Note that contrary to widely-held believes, LRD does not preclude smooth traffic; in fact, depending on the mean, variance, and β-value of the underlying process, modeled traffic rates can range from highly bursty to very smooth. In particular, traffic can be smooth and can exhibit LRD at the same time, thereby enriching the conventional perspective that tends to equate smooth traffic with Poisson-type traffic.

is no “typical” session length but instead, the session durations are *highly variable* (i.e., exhibit infinite variance) and fluctuate over a wide range of time scales, from milliseconds to seconds, minutes, and hours. It is this basic characteristics at the application layer that, according to Cox's construction, causes the aggregate traffic at the IP layer to exhibit self-similar scaling behavior[16]. Moreover, the LRD parameter β of the aggregate traffic can be shown to be linearly related to the heavy-tail index α of the session lifetimes or sizes; in fact, we have $\beta = \alpha - 1$ (e.g., see [12, 28, 38].

What makes the Internet unique in terms of approaching the study of complex systems from a genuinely scientific perspective is its unmatched ability for validating in an often unambiguous, thorough, and detailed manner any proposed explanation of or new hypothesis about an emergent phenomenon. In fact, it is the ability of the Internet for providing enormous opportunities for measuring and collecting massive and detailed relevant data sets and for completely ”reverse engineering” unexpected discoveries and surprises that gives rise to unique capabilities for developing and validating new theories for the complex Internet. Take, for example, Cox's construction. First note that it clearly identifies the type of additional measurements that should be collected and/or analyzed to test for the presence of the cause (i.e., heavy-tailed property) that is claimed to explain the phenomenon of interest (i.e., self-similar scaling). These measurements range from the application layer, where relevant variables include session durations (in seconds) and sizes (in number of bytes or packets), to the transport layer (where the items of interest are TCP connection durations and sizes), all the way to the network layer (where IP flow durations and sizes constitute the traffic variables of interest; here, an IP flow is made up of successive IP packets that satisfy certain common features). With the exception of some session-related variables (see for example [37], these layer-specific measurements can be readily extracted from the packet-level traces that are typically collected from individual links within the network. Indeed, while heavy-tailed distributions had been essentially unheard of in the networking context prior to the “emergence” of self-similarity, the numbers of empirical studies in support of the ubiquitous nature of heavy tails in measured Internet traffic at the different layers have steadily increased since then and have identified the heavy-tailed characteristic of individual traffic components as an *invariant* of Internet traffic for the past 10 or so years, despite the sometime drastic changes the network has undergone during that period; see for example [30, 38, 13].

The implications for this networking-based explanation of the self-similarity phenomenon are far-reaching and unexpected. On the one hand, the fact that we can explain self-similar scaling in terms of the statistical properties of the individual sessions, connections, of flows that make up the aggregate link traffic shows the power of measurements that are available on the Internet to scientifically validate any given model or explanation and to “reverse engineer” unexpected discoveries and surprises. It also suggests that the LRD nature of network traffic is mainly caused by user/application characteristics (i.e., Poisson arrivals of sessions, heavy-tailed distributions with infinite variance for the session durations/sizes)—heavy-tailed files or Web objects (or other application-specific items) are streamed out onto the network in a TCP-controlled manner to create strong temporal correlations (i.e., LRD) in the aggregate traffic seen on individual links within the network[17]. On the other hand, the fact that the underlying heavy-tailed property of the various traffic-related entities appears to be a robust feature of network traffic suggests a number of new questions: Should the network be redesigned to handle self-similar aggregate traffic/heavy-tailed connections? If so, can it be redesigned? Is this self-similar/heavy-tailed aspect of network traffic a permanent feature of application demand or an historical artifact? If the former holds true, what causes application demands to be intrinsically heavy-tailed; i.e., where do the heavy tails come form? Attempts to answer these and other questions have motivated the development of a theory of the Internet that is firmly based on some of the general principles reported in this paper and derived from a study of the design of the Internet and the evolutionary path that it has taken to date. Developing and illustrating this theory will be the main focus of the companion paper [16].

[16]A closely related earlier construction, originally due to Mandelbrot [28], relies on the notion of an *on/off process* (or, more generally, a *renewal-reward process*), but uses the same basic ingredient of heavy-tailedness to explain the self-similarity property of the aggregate link traffic.

[17]Note that without TCP or some TCP-type congestion control, the highly variable session workloads are capable of creating aggregate link traffic that can be very different from what we observe in today's Internet.

References

[1] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext Transfer Protocol — HTTP/1.0. *Network Working Group RFC-1945*, May 1996.

[2] M. S. Blumenthal and D. D. Clark. Rethinking the design of the Internet: The end-to-end argument vs. the brave new world. *ACM Transactions on Internet Technology*, Vol. 1, No. 1, 2001.

[3] B. Braden, Editor. Requirements for Internet Hosts — Communication Layers. *Network Working Group RFC-1122*, October 1989.

[4] B. Braden, D. Clark, S. Shenker, and J. Wroclawski. Developing a next-generation Internet architecture. `http://www.isi.edu/newarch/DOCUMENTS/WhitePaper.ps`

[5] J. M. Carlson and J. Doyle. Highly optimized tolerance: A mechanism for power laws in designed systems. *Phys. Rev. E***60**, pp. 1412–1428, 1999.

[6] B. Carpenter, Editor. Architectural principles of the Internet. *Network Working Group RFC-1958*, June 1996.

[7] B. Carpenter. Internet transparency. *Network Working Group RFC-2775*, February 2000.

[8] B. Carpenter and P. Austein (editors). Recent changes in the architectural principles of the Internet. *IAB, Internet Draft*, February 2002.

[9] D. D. Clark. The design philosophy of the DARPA Internet protocols. *Proc. of the ACM SIGCOMM'88*, in: *ACM Computer Communication Reviews*, Vol. 18, No. 4, pp. 106–114, 1988.

[10] D. Clark, L. Chapin, V. Cerf, R. Braden, and R. Hobby. Towards the future Internet architecture. *Network Working Group RFC-1287*, December 1991.

[11] J. Cowie, A. Ogielski, B. J. Premore, and Y. Yuan. Global routing instabilities during Code Red II and Nimda worm propagation. `http://www.renesys.com/projects/bgp_instability`, Oct. 2001.

[12] D. R. Cox. Long-range dependence: A review. in *Statistics: An Appraisal*, eds. David, H. A. and David, H. T. (Iowa State University Press), pp. 55–74, 1984.

[13] M. Crovella and A. Bestavros. Self-similarity in World Wide Web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking* **5**, pp. 835–846, 1997.

[14] M. E. Csete and J. Doyle. Reverse engineering of biological complexity. Preprint, 2002.

[15] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. *Network Working Group RFC-1883*, December 1995.

[16] J. Doyle, S. Low, J. Carlson, F. Paganini, G. Vinnicombe, and W. Willinger. Robustness and the Internet: Theoretical Foundations. Preprint, 2002.

[17] K. Egevang and P. Francis. The IP Network Address Translator (NAT). *Network Working Group, RFC-1631*, May 1994.

[18] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol—HTTP/1.1. *Network Working Group RFC-2616*, June 1999.

[19] S. Floyd and V. Jacobson. The synchronization of periodic routing messages. *Proc. ACM SIGCOMM'93*, San Francisco, CA, 33–44, 1993.

[20] S. Floyd and V. Paxson. Difficulties in simulating the Internet. *IEEE/ACM Transactions on Networking*, Vol. 9, No. 4, pp. 392–403, 2001.

[21] G. Huston. Architectural requirements for Inter-Domain Ruting in the Internet. *Internet Architecture Board draft-iab-bgparch-00.txt*, February 2001.

[22] V. Jacobson. Congestion avoidance and control. *Proc. of the ACM SIGCOMM'88*, in: *ACM Computer Communication Reviews*, Vol. 18, No. 4, pp. 314–329, 1988.

[23] M. Kaat. Overview of 1999 IAB Network Layer Workshop. *Network Working Group RFC-2956*, October 2000.

[24] B. Krishnamurthy and J. Rexford. *Web Protocols and Practice.* Addison-Wesley, Boston, MA, 2001.

[25] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. *Proc. ACM SIGCOMM'00*, Stockholm, Sweden, pp. 175–187, 2000.

[26] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff. A brief history of the Internet.
`http://www.isoc.org/internet/history/brief.shtml`.

[27] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the Self-Similar Nature of Ethernet Traffic (Extended Version). *IEEE/ACM Transactions on Networking* **2**, pp. 1–15, 1994.

[28] B. B. Mandelbrot. Long-run linearity, locally Gaussian processes, H-spectra and infinite variances. *International Economics Review* **10**, pp. 82–113, 1969.

[29] Computer Science and Telecommunications Board (CSTB), National Research Council. *The Internet's Coming of Age.* National Academy Press, Washington, D.C., 2001.

[30] V. Paxson and S. Floyd. Wide-area traffic: The failure of Poisson modeling. *IEEE/ACM Transactions on Networking* **3**, pp. 226–244, 1995.

[31] L. L. Peterson and B. L. Davie. *Computer Networks, A Systems Approach.* Morgan Kaufmann, San Francisco, CA, 1996.

[32] J. Postel, Editor. Transmission Control Protocol. *RFC-793*, September 1981.

[33] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). *Network Working Group RFC-1771*, March 1995.

[34] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Transactions on Computer Systems*, Vol. 2, No. 4, pp. 277–288, 1984.

[35] S. Spero. Analysis of HTTP performance.
`http://www.w3.org/Protocols/HTTP/1.0/HTTPPerformance.html`

[36] J. W. Stewart *BGP4, Inter-Domain Routing in the Internet.* Addison-Wesley, Boston, MA, 1999.

[37] W. Willinger and V. Paxson. Where mathematics meets the Internet. *Notices of the AMS* **45**, pp. 961–970, 1998.

[38] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level. *IEEE/ACM Transactions on Networking* **5**, pp. 71–86, 1997.