Federal Communications Law Journal

Volume 63 | Issue 2 Article 3

3-2011

The End-to-End Argument and Application Design: The Role of Trust

David D. Clark
MIT Computer Science and Artificial Intelligence Laboratory

Marjory S. Blumenthal
Georgetown University

The End-to-End Argument and Application Design: The Role of Trust

David D. Clark*

Marjory S. Blumenthal**

---

* David Clark is a senior research scientist at the MIT Computer Science and Artificial Intelligence Laboratory, where he has worked since receiving his Ph.D. there in 1973. Since the mid 70s, Dr. Clark has been leading the development of the Internet; from 1981–1989 he acted as Chief Protocol Architect in this development and chaired the Internet Activities Board. His current research looks at redefinition of the architectural underpinnings of the Internet and the relation of technology and architecture to economic, societal, and policy considerations. Dr. Clark is past chairman of the Computer Science and Telecommunications Board of the National Academies, and he has contributed to a number of studies on the societal and policy impact of computer communications. Support for Dr. Clark's effort on this research was provided by the U.S. Office of Naval Research grant number N00014-08-1-0898.

** Marjory S. Blumenthal is associate provost, Academic at Georgetown University. Between July 1987 and August 2003, she served as founding Executive Director of the National Academies Computer Science and Telecommunications Board (CSTB; http://cstb.org). She is a member of the Advisory Board of the Pew Internet & American Life Project and the Center for Strategic and International Studies Commission on Cybersecurity; she is a fellow of the National Academy of Public Administration; she chairs the External Advisory Board of the Center for Embedded Networked Sensing at UCLA; and she is a RAND adjunct and an Office of Naval Research grantee. This work was supported by a grant from the U.S. Office of Naval Research grant number N00014-09-1-0037.

I. INTRODUCTION

Applications are the *raison d'être* of the Internet. Without e-mail, the Web, social media, VoIP and so on, the Internet would be (literally) useless. This fact suggests that the structure of applications, as well as the structure of the Internet itself, should be a subject of study, both to technologists and those who are concerned with the embedding of the Internet in its larger context. However, the Internet, as the platform, may have received more attention and analysis than the applications that run on it.

The original end-to-end argument[1] was put forward in the early 1980s as a central design principle of the Internet, and it has remained relevant and powerful as a design principle, even as the Internet has evolved.[2] However, as we will argue, it does not directly speak to the design of applications. The original end-to-end paper poses its argument in the context of a system with two parts, the communications subsystem and "the rest."[3] That paper says: "In a system that includes communications, one usually draws a modular boundary around the communication subsystem and defines a firm interface between it and the rest of the system."[4] Speaking generally, what the end-to-end argument asserts is that application-specific functions should be moved up out of the communications subsystem and into "the rest" of the system. But the argument, as stated, does not offer advice about how "the rest" should be structured. That paper equates the "rest of the system" with the application, and the application with the end points. It says: "The function in question

1. J.H. Saltzer, D.P. Reed & D.D. Clark, *End-to-End Arguments in System Design*, 2 ACM TRANSACTIONS ON COMPUTER SYS. 277 (1984).

2. This may reflect path dependence—the Internet remains young enough that it should not be surprising to see a common set of underlying uses persist.

3. Saltzer et al., *supra* note 1, at 278.

4. *Id.*

can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible."[5]

Applications and services on the Internet today do not just reside at the "end points"; they have become more complex, with intermediate servers and services provided by third parties interposed between the communicating end points. Some applications such as e-mail have exploited intermediate servers from their first design. E-mail is not delivered in one transfer from original sender to ultimate receiver. It is sent first to a server associated with the sender, then to a server associated with the receiver, and then finally to the receiver. By one interpretation, all of these intermediate agents seem totally at odds with the idea that function should be moved out of the network and off to the end points. In fact, the end-to-end argument, as described in the original paper, admits there are interpretations that are diametrically opposed. When we consider applications that are constructed using intermediate servers, we can view these servers in two ways. An Internet purist might say that the "communications subsystem" of the Internet is the set of connected routers; servers are not routers, but are connected to routers; as such, servers are outside the "communications subsystem." This reasoning is compatible with the end-to-end argument of placing servers anywhere in "the rest" of the system. On the other hand, these servers do not seem like "ends," and thus they seem to violate the idea of moving functions to the ends. These issues are prominent today, thanks to the emergence of cloud computing—which involves specific sorts of servers—and the tendency of some popular discourse to treat "the cloud" as a new incarnation of the Internet itself.[6]

The original end-to-end paper, because it uses a simple two-part model of the communications subsystem and "the rest," does not directly speak to the situation where "the rest" has structure. The purpose of this Article is to offer an interpretation of the end-to-end argument, drawing on the original motivation and reasoning, that is applicable to today's application design and today's more complex world of services and service providers.

A. What Is an End Point?

Part of the definitional problem, of course, is to define the end point.

---

5. *Id.* (emphasis omitted).

6. *See, e.g.*, Phil Dotree, *Cloud Computing: The Most Important Technology of 2010*, ASSOCIATED CONTENT FROM YAHOO! (Jan. 13, 2010), http://www.associatedcontent.com/article/2585171/cloud_computing_the_most_important.html.

There is an intuitive model that is often adequate: if computer A is sending a file to computer B (to use the example of "careful file transfer" from the original paper[7]), then A and B are end points. However, they are end points in two ways that are subtly different. In the original example, the end points are the literal source and destination of the data being sent across the communications subsystem. They are also the end points in that they are the prime movers in the activity—they are directly associated with the principals that actually wanted to accomplish the action. Intermediate nodes, whether at the packet level or application service level, seem to play a supporting role, but they are not the instigators of the action, or the nodes that wanted to see it accomplished.

The original paper provides a hint as to the importance of this distinction. Using a telephone call as an example, it points out that the ultimate end points are not the computers, but the humans they serve.[8] As an illustration of human-level end-to-end error recovery, one person might say to another: "[E]xcuse me, someone dropped a glass. Would you please say that again?"[9] The humans are the prime movers in the activity, the ultimate end points. The computers are just their agents in carrying out this objective.

In the case of a phone call, the humans and the computers are colocated. It makes no sense to talk about making a phone call unless the person is next to the phone. So one can gloss over the question of where the human principal is. But in the case of careful file transfer, the location of the person or persons instigating the action and the location of the computer end points may have nothing to do with each other. As an example, there might be one person, in (say) St. Louis, trying to do a careful file transfer from a computer in San Francisco to a computer in Boston. Now, what and where are the end points?

The person in St. Louis might undertake a careful file transfer in three stages. First, she might instruct the computer in San Francisco to compute a strong checksum of the file (i.e., a measure of the bits it contains) and send it to her in St. Louis. Then she might instruct the two computers to carry out the transfer. Third, the person might instruct the computer in Boston to compute the same strong checksum and send it to St. Louis, where she can compare the two values to confirm that they are the same. In this case, the computers in San Francisco and Boston are the end points of the *transfer*, but they seem just to be agents (intermediaries) with respect to the person in St. Louis. With respect to the instigation of the transfer, there seems to be one principal (one end point) located in St. Louis.

---

7. Saltzer et al., *supra* note 1, at 278.
8. *See id.* at 284–85.
9. *Id.* at 285.

It might seem that this example serves to further confuse the story, rather than clarify it. But if we explore one step deeper, we can begin to find some clarity. The example above, building on the example in the original paper, referred to the overall activity as "careful file transfer." It is important to ask, why is that sequence of steps being careful? It is careful only in the context of an assumed failure mode—that is, loss or corruption of information during transfer. But why does the end user assume that the computation of the checksum will not fail? Why does the end user assume that the checksum returned by the computer is actually the checksum of the file, as opposed to some other value? Why does the end user assume that the file transferred today is the same as the file stored earlier? Why does the end user assume that the file will still be there at all? A prudent end user would be careful about these concerns as well. Perhaps the file was copied to Boston because the computer in San Francisco is crash prone or vulnerable to malicious attack. Perhaps this move was part of a larger pattern of "being careful." Perhaps, in a different part of the story, the end user in St. Louis has the computer in San Francisco compute the strong checksum on multiple days and compares them to see if they have changed. All of these actions would represent "being careful" in the context of some set of assumed failures.

But if there is *no* part of the system that is reliable, being careful is either extremely complex and costly, or essentially impossible. For example, the end user cannot protect against all forms of failure or malice using the comparison of strong checksums, because it may not be possible to detect if one of the computers deliberately corrupts the file but returns the checksum of the correct version. Ultimately, being careful has to involve building up a process out of component actions, some of which have to be trustworthy and trusted.

II. RELIABILITY AND FUNCTION PLACEMENT

The example of careful file transfer in the original paper can help us to explore the relevance of the end-to-end argument to today's world. It points to the need to define what it means to be careful in a more general sense. Being careful implies making a considered and defensible judgment about which parts of the system are reliable and which parts are failure prone or open to malicious attack—being careful today implies a degree of risk management. Using careful design implies constructing a set of checks and recovery modes that can compensate for the unreliable parts. The end user in St. Louis, moving a file from San Francisco to Boston, presumably has decided to place some level of trust in those two computers. She has also designed the pattern of information movement and storage to make the overall outcome reliable, based on the assumed level of reliability and trust

of the component parts, including the computers and the communications subsystem that connect them. The trust assumptions are made by the end user (who is, at one level, the end point), and the computers are trusted agents that act on behalf of the end user.

Why does the above view of "being careful" motivate us, in the context of the original end-to-end argument, to move functions out of the communications subsystem and into the end nodes? The original paper lists several reasons:

- In some respects, it is technically very hard to make a communications subsystem fully reliable. In a system with statistical sharing, for example, there is a probability of packet loss. Such imperfections are technical consequences of rational technical design.
- Adding mechanisms to the communications subsystem adds to its complexity, and complexity seems to make systems less reliable, as well as more costly.[10]
- The communications system may not be fully trustworthy. The original paper recognizes this issue—it talks about the peril of having the communications subsystem do encryption on behalf of the end node: "[I]f the data transmission system performs encryption and decryption, it must be *trusted* to securely manage the required encryption keys."[11]
- The providers of the communications subsystem may not be motivated to provide service with the level of reliability the end user desires and can depend on.[12]

There is an explicit assumption in the original paper that the communications subsystem is unreliable.[13] This assumption is justified (both then and now) for the reasons listed above. But there is an *implicit* assumption that the end node *is* reliable and trustworthy. The example of "careful file transfer" in the original paper[14] assumes that the end node can compute a checksum reliably and perform other actions designed to compensate for the unreliability of the communications. It also assumes, implicitly, that the two ends trust each other. One end wants to send the file to the other, and the other wants to receive it. Presumably, the interests of

10. Technical advances and a more mature understanding of the system, as well as a desire to add new features, have led to increasing complexity of the communications substrate of the Internet. It is an interesting question as to whether that has reduced the overall reliability of the Internet, but this Article does not focus on issues of this sort of complexity.

11. Saltzer et al., *supra* note 1, at 282 (emphasis added).

12. *Id.* at 287.

13. *See generally id.*

14. *See id.* at 278–82.

the two ends are aligned in this respect. But let us challenge these assumptions and see what happens.

What if the two ends do not trust each other? This situation is common today. People receive e-mail but worry that it is spam or contains a virus. They are willing to receive it (because it is worth the risk), but they do not trust the sender. Now what does it mean to be careful? This is a real-world situation, so we can see what the real-world answer is. People deploy spam filters, virus checkers, and so on. And where is that done? Sometimes it is done at the receiving end point of the mail transfer, and sometimes it is done "in the middle," at one of the mail relay points. Is this a violation of the end-to-end argument?

- As a practical matter, performing these functions at an intermediate point makes sense, because, assuming that the end user trusts the intermediary, it may be more reliable and more convenient.
- The operator of the end node (the end user) may not want to go to the effort of providing the service with the desired level of reliability.
- By performing the function at an intermediate point, the service may have access to more information; for example, a mail filter may be better able to detect spam if it can compare mail going to many recipients.
- By performing the function at an intermediate point, the end user can avoid the cost and overhead of at least temporarily storing and then transferring unwelcome traffic across the communications subsystem to the ultimate end point.
- The end node might have a vulnerability that would allow a virus to attack it before a virus checker on that machine could detect it. Doing the check at an intermediate point can protect the end node from a vulnerability the end user cannot rectify.
- Pre-positioning information at an intermediate point can make the subsequent delivery more responsive as well as more reliable. Replicated intermediate points can specifically improve reliability.

What we see is that function is migrating to the point where it can be done most reliably and efficiently. In some cases, this migration is "naturally" toward the ultimate end points (because of "natural" limits to the reliability of the communications subsystem), but in other cases function may migrate away from the end point to a service point somewhere else in the network.

When we look at the design of applications, we can see different approaches to structure based on different views of those functions that are

reliable and trustworthy and those that are not. Here are two examples.

"Careless" mail transfer. E-mail, an early application for the Internet, has no end-to-end assurance of delivery or data integrity.[15] The mail is sent via a series of servers, any of which might lose the mail. Yet there is no end-to-end confirmation. E-mail seems almost an "anti-careful" file transfer, in contrast to the first example of the original paper. What was the reasoning that made the original design for Internet e-mail come out that way? The original motivation for designing e-mail systems to use forwarding servers was that the sender and the receiver might not be connected to the Internet at the same time, and if the transfer had to be done in one step, it might never succeed. Using an intermediate server is an obvious solution. But for this approach to work with reasonable overall reliability, the servers that relay mail have to be built to a very high standard of availability, reliability, and trustworthy operation. And indeed, each stage of the mail transfer is expected to be "very careful." Given this level of attention to reliability of the intermediate nodes, no end-to-end confirmation was considered necessary. So the overall reliability is built out of a cascade of these steps, rather than an end-to-end confirmation. E-mail is not "careless"; it is just based on a different set of assumptions about which parts of the system are reliable.[16]

What happens if this assumption of reliable delivery is violated? Here is a story passed on by someone who spent two years as a volunteer in Africa, where she was forced to use an e-mail server that often crashed or otherwise lost mail.[17] The end users created a manual reliability mechanism, which was to put sequence numbers in the subject line of each piece of e-mail, and send human-to-human acknowledgements of the sequence numbers by return e-mail. In other words, they added an end-to-end confirmation to deal with the unreliable servers.[18]

Content distribution. Today, much Web content is not delivered to the ultimate recipient directly from the Web server belonging to the original creator, but via a content delivery network (CDN)—a collection of

15. Later enhancements to Internet e-mail have provided the option of end-to-end integrity and authenticity checks, often using digital signatures. *See, e.g.*, *Understanding Digital Signatures*, U.S. COMPUTER EMERGENCY READINESS TEAM, http://www.us-cert.gov/cas/tips/ST04-018.html (last visited Feb. 20, 2011). These checks are seldom used today, perhaps because they do not address delivery assurance, something for which tools are lacking. Return-receipt features are used sometimes, but can be ignored by recipients, thereby undermining their value.

16. The same logic can be seen in the recent development of delay- or disruption-tolerant networking; different circumstances give rise to different assumptions about which parts of a system are reliable. *See, e.g.*, *Home*, DELAY TOLERANT NETWORKING RES. GROUP, http://www.dtnrg.org/wiki (last visited Feb. 20, 2011).

17. Interview with Libby Levison in Cambridge, Mass. (2001).

18. *Id.*

servers that cache the content and deliver it on demand. This, like e-mail, has no end-to-end confirmation of correct delivery. Is this design being careful? Is it trustworthy? Commercial CDNs such as Akamai[19] depend on their reputation as a reliable and trustworthy provider. There are no features built into the web standards that assure that they are reliable; there is only the discipline of the competitive marketplace. If they were not reliable and trustworthy, they would go out of business. So they build highly reliable systems, the content creators trust them, and the result is a more efficient overall system.

A. Application-Specific Semantics

There is another aspect to the end-to-end argument, which is that different applications have different semantics—different definitions of what it means to be "reliable" or "correct." In the context of network data transfers, for example, some applications may define "correct" operation as perfect delivery of every byte as sent, while another application may define "correct" as delivery within some time limit, with as few errors and omissions as possible. Putting some mechanism to enhance reliability into the communications subsystem runs the risk of adding a mechanism that does not meet the needs of the application. However, when we look at the placement of application-level function inside "the rest," this argument has less relevance. Wherever application-level components are placed, they can be designed so that they are aware of the application-level semantics. This line of reasoning has been used to argue explicitly for the placement of application-aware components throughout the network, because these components can then be aware of *both* local conditions in the network and application-level requirements.[20]

III. THE CENTRALITY OF TRUST

The previous discussion has used the words "reliable" and "trustworthy" in loose equivalence. However, the distinction is very important. Reliability is a technical concept, and relates to the correct operation of a component or system under specific circumstances. The concept of trust is a broader concept. A component may not be trustworthy

19. *See* AKAMAI, http://www.akamai.com (last visited Feb. 20, 2011).

20. *See, e.g.*, Samrat Bhattacharjee et al., Commentary, *Commentaries on "Active Networking and End-to-End Arguments,"* IEEE NETWORK, May/June 1998, at 66–67. Similar reasoning has also informed planning for the so-called Next Generation Networks by the International Telecommunications Union, where desires by some to support priority access and such applications as telephony have focused attention on in-network mechanisms. *See, e.g., ITU-T Study Group 13 – Future Networks Including Mobile and NGN*, INT'L TELECOMM. UNION, http://www.itu.int/ITU-T/studygroups/com13/questions.html (last visited Feb. 20, 2011).

even though it is technically reliable, because it is operated by an agent with interests and motivations that are not aligned with the end user—the principal who wants to undertake the action. Early experience with public cloud services, including social media, illustrate this concern.[21] Trust or trustworthiness thus includes some of the issues associated with security, and security is recognized as something that can and often should be addressed at multiple points in a system.[22]

A. Multiple Stakeholders

Why would one agent or server be more trustworthy than another? In many applications today, different parts of the application belong to different actors. An ISP may provide a mail server, a third party may provide a web cache or a component of what is displayed on a web page, or a peer system may provide a music-sharing server. The difference in the degree of trustworthiness relates to the motivation and roles of the different actors, and their external influences, which range from economic incentives[23] to legal requirements or constraints.

In many cases, the interests of the different actors are nominally aligned, notwithstanding differences in status or role. End users want to send and receive mail, and ISPs attract customers by providing this service, so both the end user and the ISP want the same thing to happen. The ISP may not want to perform the function exactly as the end user would prefer, and this misalignment is either tolerated or corrected via economic means (competition to provide the service) or through the technical design of the protocol, which allows the trusted elements at each end to compensate for and recover from the failures of the other agents. Recent controversy over privacy on Facebook, a provider of social media services, reflects conflicting incentives facing service providers, who seek to attract and retain both users and advertisers (which want access to users).[24]

---

21. One of the Authors has been examining the potential for the cloud to be a platform for malice from either providers or other users. *See, e.g.*, Marjory S. Blumenthal, *Is Security Lost in the Clouds?*, CONFERENCE ON COMMUNICATION, INFORMATION AND INTERNET POLICY (2011), http://www.tprcweb.com/images/stories/2010%20papers/Blumenthal_TPRC2010.pdf.

22. For example, two mutually trusting end nodes can use encryption to preserve integrity and prevent unwanted disclosure, but preventing attacks that flood the network or disrupt availability by harming network control mechanisms can only be accomplished inside the network.

23. *See* Jonathan Anderson & Frank Stajano, *Not That Kind of Friend: Misleading Divergences Between Online Social Networks and Real-World Social Protocols (Extended Abstract)* (forthcoming in Springer LNCS), http://www.cl.cam.ac.uk/~jra40/publications/2009-SPW-misleading-divergences.pdf (discussing economic incentive weakness) (last visited Feb. 20, 2011).

24. *See, e.g.*, Emily Steel & Geoffrey A. Fowler, *Facebook in Privacy Breach*, WALL ST. J., Oct. 18, 2010, at A1, *available at*

But sometimes, there are actors in the system with motivations that are adverse, rather than aligned. Music lovers of a certain disposition choose to share copyrighted material; the rights-holders try to prevent this. Some end users may prefer to have private conversations; law enforcement (and, in some countries, other governmental elements) wants the ability to intercept conversations.

To understand this situation, one must do an analysis from the perspective of all the actors. Each actor, from its own perspective, has the same ambition about reliable and trustworthy execution of its requirements—but they have different requirements. Performing this analysis will reveal that sometimes one actor's end is another actor's middle, and sometimes the actors fight over the ends. From the perspective of trust, different actors will have different views about which servers and services they can trust, and in this respect, these different servers and services represent different "ends" of the application.

Lawful intercept. Lawful intercept, or government-ordered "wiretapping," is usually conceived as being implemented in the "middle" of the network. One approach is to carry out lawful intercept within the communications subsystem (e.g., the routers of the Internet). This would imply finding a router (perhaps one very close to the end node) that the traffic of interest is likely to pass through. Another idea is to identify some service at a higher layer (an "application layer" service) that is involved in the communication, and implement the intercept there. In the e-mail system, the mail servers are a natural point of intercept. For instant messaging, the IM server would be the target.

In order for an interceptor (lawful or otherwise) to locate a node or server through which the content is flowing, it may be necessary (or at least helpful) if this actor can constrain the set of choices, both technical and commercial, that the end user can exploit. If, because of technical design or economic or policy reasons, the end node is forced to use a particular server that can be easily identified, this makes the intercept much easier to carry out. If the end user can be prevented from using encryption (an obvious "end-to-end" reliability enhancement from the perspective of the communicating end users), the effectiveness of the intercept improves. Accordingly, the legal authorities might try to limit the use of encryption, either by influencing the development of standards, legal restrictions, making encryption hard to use and understand, and so on.[25]

---

http://online.wsj.com/article/SB10001424052702304772804575558484075236968.html.

25. The Internet Engineering Task Force has addressed these concerns for over a decade, declining to accept the task of designing corresponding protocols. *See* Brian E. Carpenter & Fred Baker, *IAB and IESG Statement on Cryptographic Technology and the Internet*, IETF RFC 1984 (rel. Aug. 1996), http://www.ietf.org/rfc/rfc1984.txt; Brian E. Carpenter & Fred Baker, *IETF Policy on Wiretapping*, IETF RFC 2804 (rel. May 2000),

In several countries, as government sophistication about the Internet has grown, so, too, have efforts to monitor and control use, both of which can involve forms of interception. Attempts to visit certain websites, to search the web for certain words, to blog using certain words, to send e-mail to certain recipients, or to send e-mail using certain words have been affected by such government efforts. Even use of anonymizing services can be affected if it constitutes a pattern that can be recognized and constrained.[26] The year 2010 saw a number of countries attempt to prevent use of BlackBerry communication because of its strong encryption, forcing adaptation by BlackBerry as it sought to balance demands from governments and from end users.[27] In some of these countries, regulation of speech and other conduct serves to control Internet access and use, making it, from the perspective of many end users, less trustworthy regardless of ISP or other service provider. An international perspective makes clear that reliability is only one element of trustworthiness and that a well-functioning market is only one kind of force influencing a provider's behavior. Moreover, growth in intergovernmental discussion and cooperation in dealing with cybercrime, spam, and malware—notwithstanding different national stances about such individual rights as privacy and freedom of expression—suggests that pressures for systems to inspect and filter will continue to grow.[28]

Music sharing. The copyright holders for music and other content have taken a more direct approach to achieving their rights-protection aims—they are taking the fight to the end points themselves. They do this in a number of ways. For example, they have tried introducing their own (untrustworthy, from the end user's point of view) end nodes into some peer-to-peer systems to disrupt the delivery of illicitly shared content, and they attempt to identify sources of that content and take nontechnical (e.g.,

---

http://www.ietf.org/rfc/rfc2804.txt.

26. *See* Julien Pain, *Bloggers, the New Heralds of Free Expression, in* HANDBOOK FOR BLOGGERS AND CYBER-DISSIDENTS 5, 6 (Reporters Without Borders Sept. 2005), http://www.rsf.org/IMG/pdf/handbook_bloggers_cyberdissidents-GB.pdf [hereinafter HANDBOOK].

27. *See, e.g.*, Margaret Coker et al., *U.A.E. Puts the Squeeze on BlackBerry*, WALL ST. J., Aug. 2, 2010, at B1, *available at* http://online.wsj.com/article/SB10001424052748704702304575402493300698912.html?KEYWORDS=uae+puts+the+squeeze+on+blackberry; Bibhudatta Pradhan & Mark Lee, *India Seeks Permanent BlackBerry Solution from RIM, Pillai Says*, BLOOMBERG BUSINESSWEEK (Sept. 16, 2010), http://www.businessweek.com/news/2010-09-16/india-seeks-permanent-blackberry-solution-from-rim-pillai-says.html.

28. The U.S. is not immune: A defense contractor announced a product aimed at monitoring social media use by client enterprise personnel in late 2010. *See Raytheon Unveils Cybersecurity Product*, UNITED PRESS INT'L (Sept. 17, 2010), http://www.upi.com/Business_News/Security-Industry/2010/09/17/Raytheon-unveils-cybersecurity-product/UPI-15531284735793/.

legal) action against them.[29] This is a classic example of end nodes that communicate even though they have no mutual trust and adverse interests. The long-term strategy of the rights-holders is to influence the hardware manufacturers to build what they call "trusted systems," which prevent the end users from performing certain actions on data that the rights-holders deem unacceptable. The term for this may be "trusted system," but it begs the question of "trusted by whom?"

B. "Good Guys" and "Bad Guys"

As we have noted in several places in this Article, while the original end-to-end paper used examples in which the two end points had a common interest in communicating, today more and more users who choose to communicate do not trust each other. Whether it is e-mail designed to defraud as in the case of phishing, a node in a peer-to-peer content distribution system that is designed to nab copyright violations, or a website that attempts to download malware or third-party tracking software onto an unsuspecting client, the Internet is full of examples where there is good reason for the ends of a communication not to trust each other.

In this context, the end-to-end argument is a two-edged sword. Since the end-to-end argument leads to a general-purpose network in which end users can run the application of their choice, without constraint from the network, it empowers both the "good guys" and the "bad guys." As the Internet seems to be increasingly overrun with bad guys, some security advocates deem the end-to-end argument itself as too dangerous to tolerate, since it is an enabler for bad guys. Further, the proliferation of malware transmitted by e-mail and the web provides some with an argument against end-to-end encryption, on the grounds that it makes filtering such material by service providers harder and therefore facilitates its circulation. On the other hand, the Internet Engineering Task Force has emphasized the value of end-to-end security, taking what some might call a "good guy"-centric position that, because in part of rampant exploitation of compromised end systems, development and use of secure protocols by end systems is critical for the Internet to serve the purpose of an international infrastructure.[30]

---

29. Peer Media Technologies offers "noninvasive" techniques (such as posting of false files and propagation of false signals) aimed at limiting illicit transfers of copyrighted materials on peer-to-peer networks. *See* PEER MEDIA TECH., http://www.peermediatech.com/services.html (last visited Feb. 20, 2011). A discussion of what have been called pollution and poisoning can be found in Nicolas Christin et al., *Content Availability, Pollution and Poisoning in File Sharing Peer-to-Peer Networks*, *in* EC '05 PROCEEDINGS OF THE 6TH ACM CONFERENCE ON ELECTRONIC COMMERCE, 68, 68, 75–77 (2005).

30. *See* Jeffrey Schiller, *Strong Security Requirements for Internet Engineering Task Force Standard Protocols*, IETF RFC 3365 (rel. Aug. 2002), http://www.ietf.org/rfc/rfc3365.txt. Schiller's 2002 RFC reiterates and amplifies the

We will revisit this point at several points in this Article. However, our overall approach is to reframe the end-to-end argument in terms of trust (where trust exists, and between which parties), rather than in terms of physical location (e.g., an "end point"). In this approach, adding protection to keep the bad guys from harming the good guys is consistent with (and integral to) the end-to-end argument, rather than being at odds with it.

IV. THE NEW END-TO-END

The discussion of what it means to be careful provides a framework for proposing a reformulation of the end-to-end argument for today's context: we can replace the end-to-end argument with a "trust-to-trust argument." The original paper said: "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."[31] The generalization would be to say: The function in question can completely and correctly be implemented only with the knowledge and help of the application *standing at a point where it can be trusted to do its job in a reliable and trustworthy fashion.* Trust, in this context, should be determined by the ultimate end points—the principals that use the application to fulfill their purposes. Because the locus of trust is naturally at the ends, where the various principals are found, "trust-to-trust" is preferable to "end-to-end" from the point of view of the principals, because it more directly invites the important question of "trusted by whom?" That question, in turn, relates to questions that implicate application design, notably "who gets to choose which service is used?" or "which parts of an application are in which service modules?" Answers to these questions illuminate who controls what aspects of an application.

To reconstruct the end-to-end argument in the context of trust, we proceed in two steps. We first look at the range of options that each participant in the communication can take, based on their individual choices about trust, and then we look at the range of options that arise *jointly*, depending on the degree to which the various communicants trust each other. Trust-to-trust acknowledges that, unlike when the original paper was written, there is more reason for one end to question the trustworthiness of another and therefore more reason to seek something beyond simple end-to-end communication. As we noted in our earlier paper, the population of end users has become more diverse, and this raises questions for the end-to-end argument. [32]

"Danvers Doctrine" agreed to in 1995. *Id.* at 3.

31. Saltzer et al., *supra* note 1, at 278 (emphasis omitted).

32. Marjory S. Blumenthal & David D. Clark, *Rethinking the Design of the Internet: The End-to-End Arguments vs. The Brave New World*, 1 ACM TRANSACTIONS ON INTERNET

A. *Trust Options for the Individual End Node*

Each end user, to the extent the option is available, must make decisions about where services should be positioned so that they can be performed in a trustworthy manner. They can be positioned on a computer that is directly associated with the end user (the classic "end node" of the original paper), or they can be delegated to a service provider elsewhere in the network. A marketplace of providers and subscribers gives the end user control over which provider is selected to perform the service. Given choice, users can be expected to select services and service providers that they deem trustworthy. Only if the principals at the edge of the network, where they connect to it, are constrained from making choices about what agents to use, and are thus constrained to depend on agents that are not trustworthy, is this natural pattern of edge-determined trust broken. The above anecdote about problematic e-mail in Africa illustrates this point.[33] First, of course, the mail relay was unreliable. But second, the end users had no reasonable alternative but to use the mail relay of their ISP—they could not choose to move to another one, for reasons of ISP policy and pricing. There was thus no market incentive to motivate the provider to be reliable or trustworthy. This story also shows how end users may respond to untrustworthy agents by adding a new layer that they believe they can trust, in that case by trusting each other to use sequence numbers properly.

There are many reasons why the end user might be constrained in some way from making a choice to select trustworthy services and forced to use a service, whether or not she trusts it. An ISP can try to force its customers to use its own e-mail servers (most end users today depend on the DNS servers of the ISP, which influence where traffic is directed, without even thinking about whether it is wise to do so); and some ISPs try to force the end user to use an ISP-provided web proxy. Certain applications may be designed so there are few (or no) choices available to the prospective users as to the provider of the service. For example, a dominant social media service provider, such as Facebook, defines both hidden and visible aspects of its service; the user has no view into and no control over the hidden aspects. More generally, there are countries where all local agents may not be trustworthy, for reasons other than their use of inadequate or unreliable technology. For example, government interception may diminish the trustworthiness of all services available locally.[34] And in

---

TECHNOLOGY 70, 74 (2001).

33. *See supra* note 18 and accompanying text.

34. The OpenNet Initiative tracks such government-based interception. *See* OPENNET INITIATIVE, http://opennet.net/ (last visited Feb. 20, 2011). China now requires cell phone users to register for new accounts with their names to facilitate monitoring of the increasingly mobile Internet. *See* Loretta Chao, *China Starts Asking New Cellphone Users for ID*, WALL ST. J., Sept. 1, 2010, *available at*

developed, as well as developing countries, there is a growing number of reasons, including private sector monitoring for commercial purposes and selective blocking and filtering of communication, for end users to question the trustworthiness of available agents in at least some regards.

Constraint also comes from the operating system (and the browser) of the end user's computer. As we will discuss, the end user is more or less forced today to use one of a very small set of operating systems (and browsers). Whether or not the end user trusts those operating systems, convenience drives end users to use them. The power of convenience as a driver is manifest in the rapid growth in use of smartphones and other mobile devices, which support mobile Internet use.

In most of the examples we have listed of this sort, and in most countries today, the provider of the service has some motivation to provide services in a reasonably reliable and trustworthy manner. There are enough checks and balances in the system (through market or legal/regulatory mechanisms) to discipline a provider. But the match of expectations is often not perfect, as illustrated by the surge in concerns about privacy motivated by social media and other public cloud applications, and the end user is often forced into various sorts of compromises.

One of the most problematic situations is where a user is constrained to use an ISP that is not trustworthy. The ISP may indeed forward traffic correctly, but may monitor or log it. In this case, users with sufficient skills and knowledge invoke services (such as encryption) that disguise what is being sent. In other cases, the ISP may block the sending of certain traffic, or to certain destinations. Here, sophisticated users may invoke some sort of "higher-level" forwarding service, so that the ultimate destination of the communication is not visible to the ISP. Some dissidents in censorship-prone regimes resort to third parties in different countries to get their messages out on their behalf, perhaps without attribution.[35] Tools such as onion routing[36] can be used to disguise both the content and the destination of a transmission; it essentially overlays the routing algorithm of the ISP with a separate routing scheme carried out by (presumably) more trustworthy nodes.

---

http://online.wsj.com/article/SB10001424052748704791004575465190777886192.html?KEYWORDS=china+requires+id+cellphone+customers.

35. *See* Nart Villeneuve, *Technical Ways to Get Round Censorship*, *in* HANDBOOK, *supra* note 26, at 63, 75. The U.S. government has funded the development of software for this purpose. *See, e.g.*, *Freegate*, DYNAMIC INTERNET TECH., http://www.dit-inc.us/freegate (last visited Feb. 20, 2011).

36. For a description of onion routing, see TOR PROJECT: ANONYMITY ONLINE, http://www.torproject.org (last visited Feb. 20, 2011).

B. *Delegation of Function*

E-mail and content distribution as described above, as well as the example of checking for viruses and spam, illustrate what we might call *delegation* of function to a trusted agent. The content producers trust the CDN, and they delegate the delivery function to it. In most cases, end users trust their mail agents (in contrast to the story about the African service), and they delegate the transfer of mail to these services. We could draw a circle around each end point and the servers (including supporting services such as the DNS) the user has chosen to trust, and (at the application layer) we could call this an *application* end point.

Figure 1 illustrates how the e-mail system might be drawn. To get between parts of this "higher-level" end point it will be necessary to make use of the lower-layer communications subsystem, and there will be reliability mechanisms designed and used at that level. At this lower level, the end-to-end argument will apply as each part of the service communicates with the other parts. At a higher level, there is a different interpretation of the end-to-end argument, as one application end point talks to the other application end point.

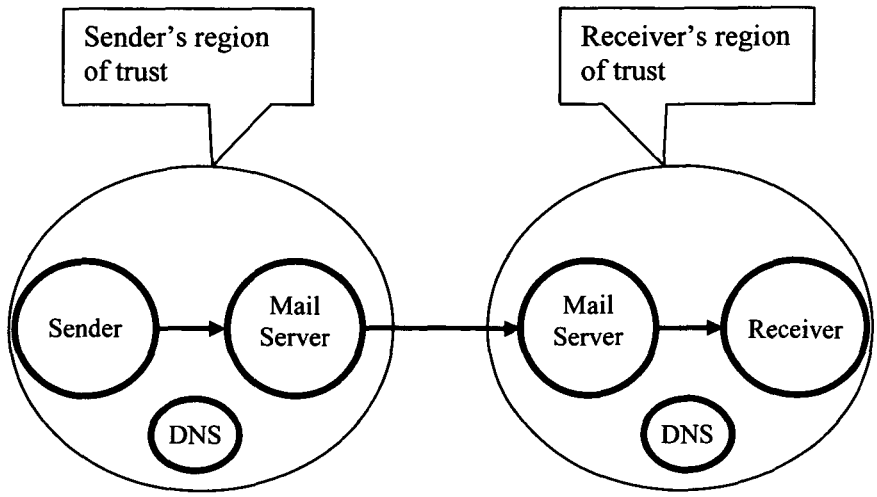

Figure 1: Regions of trust in email forwarding system

C. *Mandatory Delegation*

In the real world, there are many circumstances where an individual user does not have control over which services and servers to use. Perhaps the most obvious example is the context of employment, where individual

employees are constrained by corporate policy to use their computers in certain ways, to use only certain servers (e.g., their corporate e-mail or instant message servers), and so on. The fact that the user is forced to use these services does not automatically make them untrustworthy. Some employees may be entirely comfortable with the way their employers operate their IT infrastructure; others may have fears about surveillance, logging, or other practices. But whatever judgment the employee makes about the trust to place in his or her circumstances, he or she has no realistic control over the situation (although alternative platforms and applications may be chosen for personal use).

There is a useful analog between this situation and a duality that arises in the security community. Certain security controls are cataloged as "discretionary access controls," or DACs, and "mandatory access controls," or MACs.[37] MACs originated from the need to enforce rules for the proper handling of classified information, and access decisions were taken away from the individual at his computer and given to a system security administrator, who would impose access controls based on corporate or institutional security policies.[38] Because the individual user had no control over these mechanisms, they were called *mandatory*, which is a word that signals that somebody other than the end user has the discretion to control them.[39]

One way to capture the range of situations that apply to the individual end user is illustrated in Figure 2.

37. *See, e.g.*, SYS. SEC. STUDY COMM. ET AL., COMPUTERS AT RISK: SAFE COMPUTING IN THE INFORMATION AGE 251 (1991).

38. *See id.*

39. An illustrative example of such a control in the network context is an intrusion detection system. Such systems look at incoming and outgoing traffic to attempt to detect patterns that suggest an ongoing attack. They can benefit from seeing the traffic to and from many nodes, not just one. They are often installed by network managers (so they are mandatory from the perspective of the end user), and they are generally viewed as benign.

	Likely degree of end user trust: Lower	Likely degree of end user trust: Higher
Mandatory selection of mechanism	Monopoly provider Government intervention	Employee
User choice or discretion over mechanism selection	Uncommon: given choice and knowledge, users select trusted option	Competitive market with consumer choice

Figure 2: Range of options for choice and trust

D. When End Users Do Not Trust Each Other

In the analysis above, we defined "trust end points" by drawing circles around end points and the various trusted services to which they have chosen to delegate functions. But once we have drawn these circles, an important question remains—do the different trust end points trust each other?

For most of the examples in the original paper, the answer is yes. In the example of careful file transfer, the two end points are collaborating to make the transfer reliable. But as we hinted above (using the example of viruses), we often communicate with other end points that are not trustworthy and/or that we do not choose to trust. How do we deal with this situation?

One class of response is to try to devise and deploy defenses inside each circle of trust that are robust enough that the other end point cannot inflict any material damage. We deploy virus checkers, spam filters, and so on, and then we cautiously try to exchange e-mail.

But the other class of response is to invoke the services of a mutually trusted third party to remove some of the risk of the interaction. I do not trust you, you do not trust me, but we both trust this other party— perhaps that other party can help us interact. The real world is full of examples of this sort—trusted institutions of all sorts are what make contemporary, economically developed society function, from banks to contract law and courts to credit card companies and various sorts of negotiators. In the real world, when two parties view each other with suspicion, they seldom try to

resolve the problem on their own.

And we see more and more the emergence of online analogs. For example, credit card companies, because they can verify the identity of all parties and because they protect against fraudulent actions, act to add trust so that transactions can be completed between untrusting end users and merchants. Providers of digital certificates assist in the authentication of communicants that may not trust each other. And today, a variety of projects aim to provide identity-management services in ways that suggest new categories of third-party actors facilitating trust.[40] By providing assurance-supporting services such as identity management and insurance against specific risks, such third parties permit untrusting parties to decide to take the risk of interacting. More directly, many applications are designed so that services developed and operated by the designer of the application are interposed between the end users. When two end users communicate using popular instant messaging applications today, they do not directly connect across the communications subsystem. Instead, the IM communications are relayed through an IM server run by the service itself. This service enhances many aspects of the overall function. For example, the centralized management of identities provides level of confidence to the users about the identities.. Second, the service provides isolation between the end users. Since end users do not communicate directly, they need not reveal low-level information such as IP addresses to each other, which prevents them from attacking each other directly across the communications subsystem.

Similarly, eBay, interposed between buyer and seller, provides a neutral meeting ground (more specifically, a neutral place where markets can be made). eBay also illustrates the role of reputation in assessing trustworthiness: eBay is a third party that facilitates communication about reputation and implied trustworthiness. This is one way that identity can be invoked for trust.

For some applications, for example multi-player games, it is fairly obvious that much of the implementation of the game resides on servers rather than on the end nodes of the players. This structure arises both because there is a great deal of shared information (about the game and its state of play) among the players that must be coordinated, and also because the players must be kept from cheating. The players certainly want to communicate, but they just as certainly do not trust each other.

Here is a partial list of functions that a trusted third party might

40. *See, e.g.*, HIGGINS: OPEN SOURCE IDENTITY FRAMEWORK, http://www.eclipse.org/higgins/ (last visited Feb. 20, 2011); SHIBBOLETH, http://shibboleth.internet2.edu/ (last visited Feb. 20, 2011) (explaining the single sign-on approach); OPENID, http://openid.net/ (last visited Feb. 20, 2011).

perform:

- Manage identity, in many ways
- Facilitate appraisal of reputation
- Provide robust authentication (prevent identity theft, prevent fraud)
- Control the release of attributes (limit what one party can see about others, e.g., IP addresses)
- Preserve anonymity (extreme form of controlled release—sender wants to hide all aspects of his identity from receiver)
- Protect end users from each other
- Prevent attacks
- Regulate and filter content
- Prevent cheating (e.g., in games)
- Provide mutual assurance and guarantees (escrow, fraud insurance, nonrepudiation)

Sometimes the third party software is directly interposed in the communication path between the end nodes, as with instant messaging, games, eBay, and the like. In other cases, the third party is not literally in the communication path between the two untrusting users but is invoked by one or both of those parties to augment the trustworthy nature of the overall transaction. It is tempting to try to analyze the implications of trusted third parties for the end-to-end argument by looking to see if the third party is literally in the path of communication. If we allow ourselves to fall back to a lower-level view of end-to-end, looking at the role of the communications subsystem, models where the third party is "off to the side" (invoked by one of the end nodes) might seem more "end-to-end." But we would argue that this lower-level detail is irrelevant in an analysis of trust, which is the basis for our higher-level model. If two parties decide to involve a trusted third party, then that party is in the middle of the "path of trust," regardless of whether that party is in the middle of the packet flow. We should not be concerned with how the packets flow, but instead look at which aspects of the trust depend on our mutual dependence on that third party, and which aspects we can determine for ourselves.

The choice as to whether to invoke a third party to enhance trust in a particular application is usually not up to the individual user. It will usually be embedded into the design of the specific application at hand; in other words, the designer of the application has control over the patterns of communication and thus the "architecture of trust." Whether or not a buyer and a seller on eBay have reason to trust each other, they must interact in the context of the marketplace defined by eBay. This fact begs the obvious question as to whether it is any more reasonable for end users to trust the third-party service provider than to trust each other. One way to try to

answer this question would be by analogy to the original end-to-end argument, where one might argue that it is better for the end nodes to solve what problems they can by themselves, because involving a third party can only add to the complexity, and perhaps to the lack of certainty about trust.[41] An issue for the design and operation of such third parties, as recently publicized identity-theft cases illustrate, is to avoid having them emerge as a bigger, let alone just another, source of vulnerability. To some observers who are concerned about the loss of personal control, the use of certain kinds of remotely provided services (services "in the cloud") is a major source of risk.[42] But the outcome of the analysis, in this case as in the original paper, is not a dogmatic stricture but a preference to be validated by the facts of the situation. And this construction by analogy may be nonsense. While there are specific reasons to assume that the communications system will be unreliable, there is no similar reason to assume that third-party services are intrinsically unreliable. The decision will be based on a trust assessment, as well as considerations of convenience and utility. So perhaps at this level there should not be a preference for end-to-end patterns of communication, but a preference for the use of third-party services and multiway patterns of communication—that is the kind of thinking that has contributed to growth in demand for cloud services.

In the marketplace of the 2000s, a number of developments shift activities away from end nodes. "Service oriented architecture" (SOA) is a buzzphrase for accessing a variety of applications and data over a network. It is linked to a model in which end users, within some enterprises, access what they need from servers as they need it, rather than investing in capabilities at their individual end nodes. It is also a concept fundamental to social media and various public cloud applications. For example, Google's move to provide office-productivity capabilities aims to motivate end users, as individuals and as members of enterprises, to use capabilities hosted on its servers rather than at the end nodes (or servers controlled by the enterprise).[43] This mode of operation, combined with a style of operating the end node in which no new software or functions can be downloaded or installed, tries to accomplish stable operation through delegation and outsourcing.

---

41. This is a big question for cloud computing, at least for the public cloud services. *See* Blumenthal, *supra* note 21.

42. *See, e.g.*, Richard Stallman, *What Does That Server Really Serve?*, BOS. REV. (Mar. 18, 2010), http://bostonreview.net/BR35.2/stallman.php (revised version available at http://www.gnu.org/philosophy/who-does-that-server-really-serve.html).

43. *Stay Connected and Be More Productive*, GOOGLE APPS, http://www.google.com/apps/ (last visited Feb. 20, 2011).

V. THE ULTIMATE INSULT

The erosion of the end-to-end argument is often equated to the emergence of intermediate servers and services not located at the end points. As we have argued, this is not necessarily so. If the end user has a choice and can pick services that he trusts, this can be seen as delegation and the creation of a distributed end point. The more fundamental erosion of the end-to-end argument is that the end user can no longer trust his own end node—his own computer. There are forces, both lawful and unlawful, that try to shift the balance of control and trust away from the end user toward other parties such as rights holders. Malicious software such as spyware and key loggers—sent by malicious end systems—try to attack the reliability and trustworthy nature of typical end user activities by penetrating the end node computer and turning it against the end user or against other end users. Criminal elements make surreptitious use of large numbers of end nodes owned or used by others via botnets that attack, send spam, and otherwise make mischief for yet other end users. Legitimate businesses seeking advertiser support tolerate tracking software that can compromise end user privacy.[44]

Whatever the cause for distrust, what is the future of the end-to-end argument if the end user cannot trust his own computer to behave reliably? This trend could signal the end of end-to-end, and more catastrophically, the end of any ability to make rational trust assumptions at all. If the end user cannot trust her own computer, what can she trust?

A. Can We Take Back the End Node?

One response to end users' diminishing ability to trust their own end nodes might be further delegation, as mentioned above: to move away from using the computer as a platform for trustworthy activities, and to move those activities to servers provided by operators who seem to be able to offer them reliably. This approach would signal the return (yet again) of the thin client and a "services architecture" for applications. Using our analysis, what would be required to make this work? First, this scheme would still require a trustworthy path of communication from the end user to the service. This path has to reach all the way to the human user—this implies that what the end user sees on the screen is what the service wanted

44. *See* Nick Wingfield, *Microsoft Quashed Effort to Boost Online Privacy*, WALL ST. J., Aug. 2, 2010, at A1, *available at* http://online.wsj.com/article/SB10001424052748703467304575383530439838568.html; Steve Stecklow, *On the Web, Children Face Intensive Tracking*, WALL ST. J., Sept. 17, 2010, at A1, *available at* http://online.wsj.com/article/SB10001424052748703904304575497903523187146.html?mod=WSJ_article_RecentColumns_WhatTheyKnow.

to put there.[45] The potential for a key logger on the client, no matter how thin the client, destroys the trustworthy nature of the scheme. The need for a trusted path might lead to a model of end node software where the machine has a fixed set of software and no ability to download any active code or new applications. Second, to make this scheme viable, service providers who declare that they are going to offer a trustworthy service must be able to do so. If their servers are susceptible to being infested with spyware or are readily intercepted for censorship or surveillance purposes, we are no better off.

Another approach is to try to reclaim control of the end node, both by reducing vulnerability (bugs) and by allowing the end user to know what is in the system. Part of the appeal of Linux is that since the code is open, skilled programmers can read it and try to verify that there are not any intentionally installed controls and features that make the machines using it less trustworthy and less suited to the needs of the end user.

VI. DESIGN FOR DELEGATION

If we agree that it is useful in certain cases for end nodes to delegate functions to servers and services within the network, then applications have to be designed to make this both possible and easy. The application has to be broken up into parts connected by well-specified protocols that seem to represent useful functional building blocks. This act of modularization, of course, takes a lot of judgment, and is probably best suited to be the subject of a book, rather than an article. Assuming that the application has been properly modularized, there are then some further points that arise from the discussion of trust and the reality of both trusted and untrusted third parties.

First, one can ask whether the modularization of the application allows the trust assumptions to be violated in unexpected ways. For example, one of the ways that untrusted third parties can insert themselves into an application is by interjecting themselves into the path of a well-specified protocol—the sort that is designed to allow functional decentralization—and playing the part of the other communicant. One of the implications of an open and documented protocol is that since any actor can "speak the language," it may be possible for a third party to insert itself into the middle of a path and pretend that it is the intended destination of

45. This idea is not new, of course. It relates to the idea of a "Trusted Path" in secure computer systems, as articulated in the Trusted Computer System Evaluation Criteria. DEP'T OF DEF. STANDARD, Trusted Computer System Evaluation Criteria No. DoD 5200.28/STD (1985), *available at* http://csrc.nist.gov/publications/history/dod85.pdf. This reference defines a Trusted Path as "[a] mechanism by which a person at a terminal can communicate directly with the Trusted Computing Base," which emphasizes that the trusted path must reach all the way to the human user to be effective. *Id.* at 113.

the conversation.[46] A (mostly) harmless example of this occurs quite often when an Internet user at a hotel or WiFi hot-spot tries to send mail. It is often the case that the connection back to the Simple Mail Transfer Protocol (SMTP) server chosen by the end user is redirected to a different SMTP server operated by the local provider. The hotel intends this to be a helpful feature (it solves the problem that not all SMTP servers will accept connections from distant parts of the network), but at a philosophical level, it represents a complete overriding of the end user's right to choose which service to use. Protocols should be designed so that the end user who makes the choice of which service and servers to use maintains control over that choice. Distributed elements should always be able to tell which other elements they are talking to, and it should not be possible to subvert the protocol so that untrusted parties can exploit them to insert themselves. Tools (often based on encryption) that provide assurance about identity and nondisclosure can ensure that only the services chosen by the end nodes are the ones being used.

Second, trust is with respect to a given role. I may be willing to trust a router to forward my packets—or, putting this differently, there may be enough constraints that I can count on the router to forward my packets even if I do not fully trust it—but I may not trust it to protect my packets from disclosure. If the protocols that are designed to allow functional decentralization and delegation are designed so that the capabilities of the servers and services are limited to the intended functions, then we need not make as strong a trust assumption about these devices, which will provide more flexibility regarding which services we are prepared to choose. For example, if different parts of the application payload are encrypted and/or signed (so an intermediate cannot see or change them) and other parts are revealed, this can allow servers to be employed without having to trust them to preserve all aspects of the information.[47]

An important aspect of application design applies to protocols and mechanisms that can operate both in the context where the end users trust each other and where they do not. If the end users have the choice among

46. In security parlance, when a malicious node manages to insert itself into the middle of a conversation, pretending to each of the communicants to be the other communicant, this is called a "man in the middle" attack. It may give the attacker (or more generally the third party with adverse interests) the ability to see and modify anything that is being transmitted.

47. Of course, if the design process for the application included an explicit discussion about which parts of the payload should be encrypted or revealed, this might trigger vigorous advocacy among the different stakeholders as to how the application should be designed. There is a parallel with the debates that occurred during the design of IPsec—the IP level encryption standard—where there were competing views as to which parts of the original packet header should be hidden and the eventual development of two alternatives (Encapsulating Security Payload and Authentication Header) that offer a different answer to this question.

invoking a third party, using mutual checks and constraints, or communicating openly based on mutual trust; and if the application can easily adapt to all of these modes, then it becomes more practical for the end users to operate in each of these modes and to move among them as they deem appropriate.

Research is driving some new approaches to the architecture of social media applications that restore some control to end users. Recent research projects illustrate the impact of different choices about modularizing applications. The Lockr system, for example, decouples information about an end user's social network from distribution of content to members of that network, allowing end users to limit the number of services with which they share their social network information.[48] It also provides for asymmetric relationships among people in a social network and revocation of relationships.[49] Another approach is taken by the authors of the proposed PrPl "person-centric" infrastructure for storing and sharing information with "fine-grained access-control."[50] These specific examples illustrate that application design and modularity can enhance or reduce options for user choice. Different designers will have different motivations to offer or constrain choice, and thus control the degree to which a user can make personal decisions about trust within specific applications. Our earlier example of e-mail illustrated an application based on a design that gives the user choice.

We have taken the view here that if some principal chooses to trust some agent and, for example, delegates function to it, this should lead to a system that is just as trustworthy as a system in which all the functions are carried out on the end node. The IETF has explored this space, and its analysis illustrates the limits of its willingness to depend on trust, as assessed by the user, as a building block of a trustworthy system. Several years ago, an IETF working group was proposed to design what was called Open Pluggable Edge Services, or OPES.[51] The OPES proposal was essentially an architecture for delegation, and it triggered a controversy in the IETF that led to a policy assessment of the OPES concept by the Internet Architecture Board.[52] This assessment reached several of the same

48. *See* Amin Tootoonchian et al., *Lockr: Better Privacy for Social Networks*, INTERNATIONAL CONFERENCE ON EMERGING NETWORKING EXPERIMENTS AND TECHNOLOGIES (CONEXT) (2009), http://conferences.sigcomm.org/co-next/2009/papers/Tootoonchian.pdf.

49. *Id.*

50. Seok-Won Seong et al., *PrPl: A Decentralized Social Networking Infrastructure*, ACM WORKSHOP ON MOBILE CLOUD COMPUTING & SERVICES: SOCIAL NETWORKS AND BEYOND (MCS) (2010), http://prpl.stanford.edu/papers/mcs10.pdf.

51. *Description of Working Group*, OPEN PLUGGABLE EDGE SERVICES (OPES), http://datatracker.ietf.org/wg/opes/charter/ (last visited Feb. 20, 2011).

52. Memorandum from Sally Floyd & Leslie Daigle, *IAB Architectural and Policy Considerations for Open Pluggable Edge Services*, IETF RFC 3238 (rel. Jan. 2002),

conclusions that we do:

- Delegation is only acceptable if one end or the other has explicitly put it in place (that is, injection of service elements by unrelated actors should not be permitted by the architecture).[53]
- Messages being sent to the service element should be explicitly addressed to the element, and tools such as encryption should be used to ensure that only the expected elements are participating in the delegation.[54]

However, after reaching these conclusions, its analysis suggests that the IAB had an instinctive reaction that services delegated to a server were somehow intrinsically less trustworthy than services running locally on the host. The assessment called for the addition to the architecture of technical means for an end node (or the principal using the end node) to be able to check or review what the service element had done. It says:

> [W]e recommend that the IESG require that the OPES architecture protect end-to-end data integrity by supporting end-host detection and response to inappropriate behavior by OPES intermediaries. We note that in this case by "supporting end-host detection", we are referring to supporting detection by the humans responsible for the end hosts at the content provider and client.[55]

One could see this recommendation as arising from the traditional roots of the Internet, where the users are technically sophisticated and able to fall back on technical intervention to validate what a server is doing. In today's Internet, most users do not have the skills to verify (technically) what a program is doing, whether it is running on their own machine or on a server. Today, most users select and use a program based on some assessment of its suitability and trustworthy nature, no matter where it runs.

VII. REINTERPRETING THE END-TO-END ARGUMENT

If this Article represents a significant (re)interpretation of the original end-to-end argument, it is part of a larger tradition of reinterpretation. Perhaps because the argument is described in the original paper as much by example as by definition, there has been a rich history of assertion and speculation about how to interpret the end-to-end argument, and what it really means. This section surveys some of that history to put our Article into a larger context.

The original paper states the end-to-end argument in terms of how function must be placed to achieve correct operation and to align with

---

http://www.ietf.org/rfc/rfc3238.txt.

53. *See id.* at 13.

54. *See id.*

55. *Id.* at 1.

application-level semantics. There is an implication that a system built according to this approach is more general, in that it is not designed to support a specific, known set of applications. However, the benefit of generality is implicit—it is not directly argued in the paper. This virtue is often associated with the *open* nature of the Internet, although the word "open" hardly appears in the paper.[56]

The importance of openness was spelled out for a broad audience in an interpretive work crafted by a committee involving the authors of this Article and others from networking and other fields. Published and extensively presented in 1994, *Realizing the Information Future: The Internet and Beyond*[57] articulated in plain English the virtues of the Internet and served to educate a wide range of U.S. and foreign policy makers, industry executives, and civil society leaders about the concept of an "Open Data Network," exemplified by the Internet. The Open Data Network is defined as open to users, service providers, network providers, and change,[58] and the book calls for research to further the development of "general and flexible architecture" for networking and the development of security architecture.[59] It also noted that the logic of an Open Data Network implied the unbundling of higher-level applications and services from lower-level networking functions.[60]

The authors of the original paper expanded on the implications of the end-to-end argument for application innovation in a 1998 paper,[61] motivated by a research program called Active Networks.[62] Beginning

---

56. Note that a well-known amplifier of the end-to-end argument, IETF RFC 1958, also does not use the word "open"; it appears that more social and economic experience with the Internet was needed before the concept was broadly appreciated. *See* Brian Carpenter, *Architectural Principles of the Internet*, IETF RFC 1958 (rel. June 1996), http://www.ietf.org/rfc/rfc1958.txt.

57. *See generally* NRENAISSANCE COMMITTEE, COMPUTER SCI. AND TELECOMM. BD., NAT'L RES. COUNCIL, REALIZING THE INFORMATION FUTURE: THE INTERNET AND BEYOND (1994).

58. *Id.* at 44.

59. *Id.* at 93.

60. *Id.* at 51.

61. David P. Reed et al., Commentary, *Commentaries on "Active Networking and End-to-End Arguments,"* IEEE NETWORK, May/June 1998, at 69–70. This states, among other things, that

> [p]art of the context of an end-to-end argument is the idea that a lower layer of a system should support the widest possible variety of services and functions, to permit applications that cannot be anticipated. . . . Higher-level layers, more specific to an application, are free (and thus expected) to organize lower-level network resources to achieve application-specific design goals efficiently (application autonomy).

Id. at 70.

62. *See generally* David L. Tennenhouse & David J. Wetherall, *Towards an Active Network Architecture*, COMPUTER COMM. REV., April 1996. The Active Networks program

shortly thereafter, as Internet virtues became more evident to a wider range of people, other authors championed the open nature of the Internet, focusing on its ability as a platform to support a wide range of unanticipated and unplanned applications. This open nature has economic and social impacts, which, as we noted in our earlier paper cited above, have motivated rhetoric by advocates of various sorts. Most prominently, Larry Lessig has used the end-to-end argument as the basis for a defense of the open nature of the Internet as an enabler of third-party innovation and what has become known as "network neutrality."[63] David Reed, one of the authors of the original paper, has reflected on the roots of the end-to-end argument, the push by telecommunications companies for more centralized control as the broadband market grows, and the chilling effect on innovation associated with in-network chokepoints.[64] Another author of the original paper, Jerry Saltzer, has chronicled "gatekeeping restrictions" arising in cable-company Internet service.[65] He has been quoted as noting that such restrictions are at odds with the end-to-end argument and,

---

was a DARPA-sponsored research project to explore a novel networking approach in which packets carry code that can be executed by routers to modify their operation. While this idea might be seen as the antithesis of the end-to-end approach, as it could move application or service-specific function into every router, the commentary cited below gives a nuanced view. *See infra* note 64.

63. *See, e.g.*, Lawrence Lessig, *It's the Architecture, Mr. Chairman*, BERKMAN CENTER FOR INTERNET AND SOC'Y, HARVARD U. (1996), http://cyber.law.harvard.edu/works/lessig/cable/Cable.html. Lessig observes,

> The Internet has a constitution. Its architecture is this constitution—the way the net is coded, its design, the principles that govern its control. Like any constitution, this architecture embeds certain values. These values have consequences. In the case of the Internet, they have produced the greatest space of innovation that we have seen this century. . . . The value at stake is a design principle called "end-to-end."

Id. at 1. Similar ideas are expressed at greater length in a variety of Lessig's writings around the turn of the century. *See, e.g.*, Lawrence Lessig, *The Internet Under Siege*, FOREIGN POL'Y, Nov. 1, 2001, *available at* http://www.foreignpolicy.com/articles/2001/11/01/the_internet_under_siege.

64. David P. Reed, *The End of the End-to-End Argument*, REED'S LOCUS (Apr. 2000), http://www.cs.sfu.ca/~vaughan/teaching/431/papers/ReedEndOfTheEndToEnd.pdf ("Today's applications (eCommerce storefronts, telephone calls routed over IP networks, streaming video broadcast of Hollywood movies, and banner-ad-sponsored web pages) are being used to justify building in idiosyncratic mechanisms into the network's core routers and switches. Though it is clearly not possible to meet the requirements of today's hot applications solely with functionality in the network's core, we are being asked to believe that this is the only possible architecture. Implicitly, we are being told that the impact of building these structures into the network is worth the cost of erecting major barriers to future innovation. . . . In the Internet's end-to-end design, the default situation is that a new service among willing endpoints does not require permission for deployment. But in many areas of the Internet, new chokepoints are being deployed so that anything new not explicitly permitted in advance is systematically blocked.").

65. Jerome H. Saltzer, "Open Access" Is Just the Tip of the Iceberg (Oct. 22, 1999) (unpublished article), http://mit.edu/Saltzer/www/publications/openaccess.html.

therefore, a threat to innovation.[66] He continues to observe shrewdly that people are not passive in the face of such corporate conduct, suggesting that effective responses can arise from consumer behavior and/or government regulation.[67]

Barbara van Schewick, in her dissertation[68] and book,[69] has undertaken an extensive analysis of the economics of the Internet market, which she prefaces with a thorough and careful review of work that interprets and contextualizes the original end-to-end argument in various ways. Van Schewick asks what it means to adhere to the original argument when its own authors varied the wording over time. In the original paper, the authors wrote: "The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system."[70] In their 1998 commentary on the end-to-end argument and active networks, they wrote a somewhat different sentence: "[A] function or service should be carried out within a network layer only if it is needed by all clients of that layer . . . , and it can be completely implemented in that layer."[71] Van Schewick calls the earlier version "narrow" and the later version "broad," and then considers how the economics vary with the version.[72] The analysis in this Article is consistent with either version of the end-to-end argument.

In addition to openness and flexibility, simplicity (of the communications subsystem) has also been identified as a benefit of the end-to-end argument. The original authors discuss these benefits of the end-to-end argument in their 1998 commentary, where they argue for the architectural benefit of "moving function from lower layers to more application-specific layers . . . ."[73] They explain that "building complex functions into a network implicitly optimizes the network for one set of uses," arguing that "an end-to-end argument . . . strongly suggests that enthusiasm for the benefits of optimizing current application needs by

66. *Id.*

67. *See id.*

68. Barbara van Schewick, Architecture & Innovation: The Role of the End-to-End Arguments in the Original Internet (July 21, 2004) (unpublished Ph.D. dissertation, Technische Universität Berlin), http://www.lessig.org/blog/archives/van%20Schewick%20Dissertation%2012102004.pdf.

69. BARBARA VAN SCHEWICK, INTERNET ARCHITECTURE AND INNOVATION (2010).

70. Saltzer et al., *supra* note 1, at 278 (emphasis omitted).

71. Reed et al., *supra* note 61, at 69.

72. SCHEWICK, *supra* note 69, at 5. This is the most detailed textual and economic analysis to date. Its almost Talmudic character begs the question of how important is the exact wording used by technologists who acknowledge that their own understanding of their subject has grown with time and experience.

73. Reed et al., *supra* note 61, at 70.

making the network more complex may be misplaced."[74] The 1998 paper reflects the broad acceptance of the layered-system architectural paradigm, deeper understanding of the challenges posed by system complexity as a result of technical and economic activity since the original paper, and insight into evolving views of the tension between programmability and flexibility on one hand, and specialization on the other. Specialization, or the adding of function to facilitate specific applications, can privilege specific uses and users by making what they do more efficient.[75]

The idea of trust as a fundamental tool for the analysis and application of the end-to-end argument is not original to this Article. Consistent with our discussion herein, the previously cited *Realizing the Information Future* observed that, "If the [National Information Infrastructure] is to flourish, we must provide solutions so that any end node attached to the network can mitigate its risk to an acceptable level."[76] More recently, Tim Moors examined the influence of responsibility and trust on the end-to-end argument.[77] His emphasis on the role of trust is very similar to our point of view, but his analysis focuses on lower-level functions such as congestion control.[78] He observes that in today's commercial environment (as opposed to the smaller, nonprofit community of the early Internet years) it is naïve to expect end points to behave altruistically (e.g., in terms of refraining from congestion-inducing behavior).[79] He also points out the need to identify the end nodes carefully as part of understanding "what entity is *responsible* for ensuring that service, and the extent to which that entity can *trust* other entities to maintain that service."[80]

Kempf and Austein assert that "the single most important change from the Internet of 15 years ago is the lack of trust between users,"[81] underscored by the rise of "deliberate, active attacks on the network infrastructure and end nodes."[82] They argue that that lack of trust drives

74. *Id.*

75. The companion piece by Partridge, et al., suggests that growth in understanding of complexity and programmability shift the balance toward more programmability in network management while preserving simplicity in the Internet layer to assure broad connectivity. *See* Craig Partridge et al., BBN Techs., Commentary, *Commentaries on "Active Networking and End-to-End Arguments,"* IEEE NETWORK, May/June 1998, at 67–69.

76. NRENAISSANCE, *supra* note 57, at 79.

77. Tim Moors, *A Critical Review of "End-to-End Arguments in System Design,"* 5 IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS 1214 (2002).

78. *See id.*

79. *Id.*

80. *Id.* at 1219.

81. Memorandum from James Kempf & Rob Austein, *The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture*, IETF RFC 3724, at 5 (rel. Mar. 2004), http://www.ietf.org/rfc/rfc3724.txt.

82. *Id.* at 8.

choices by application and system designers about authentication, and they observe that

> One of the most common examples of network elements interposing between end hosts are those dedicated to security . . . .[they] are designed to protect the network from unimpeded attack or to allow two end nodes whose users may have no inherent reason to trust each other to achieve some level of authentication."[83]

Those users, in turn, need to "determine which third parties they trust."[84] Third parties, such as ISPs, have their own interests (e.g., making profits) to address, and while they can serve as "trust anchors" by acting to protect end users, they can insert mechanisms to support their own policy (e.g., censorship) into the network.[85] Kempf and Austein caution against application design that creates dependencies among protocols and system layers, citing the controversy (discussed above) associated with Open Pluggable Edge Services.[86] They assert that

> the trust relationships between the network elements involved in the protocol must be defined, and boundaries must be drawn between those network elements that share a trust relationship. The trust boundaries should be used to determine what type of communication occurs between the network elements involved in the protocol and which network elements signal each other.[87]

They suggest that the right approach to decomposition allows for the end-to-end argument to apply internally to an application, and while it may not apply to the application as a whole, this approach can assure the benefits that have come to be associated with the end-to-end argument, such as innovation and robustness.[88]

VIII. CONCLUSIONS

We have argued that "trust-to-trust" is an important generalization of end-to-end. The original paper equated the end node with the trusted node, and therefore it did not elaborate on this issue. But we argue that the fundamental basis for placement of function is that it is placed where it can be trusted to carry out its function reliably. Our preference, consistent with the end-to-end argument, is that the end user should have control over the trust decisions. It is the movement of trust to the edge that is consistent with the end-to-end argument, not the placement of all function at the end node.

The inability of the end users to trust their own computers (their end

83. *Id.* at 5.
84. *Id.* at 6.
85. *Id.* at 7.
86. *See id.* at 3–5.
87. *Id.* at 8.
88. *Id.* at 10.

nodes), and uncertainty about this, is the most corrosive problem for the end-to-end argument, not the placement of services in the net, per se. Accordingly, we have highlighted the challenge of designing trustworthy end nodes.

The original reasoning about the communication subsystem remains valid. We now have a "two layer" end-to-end argument, and a more complex "the rest," where "the rest" is broken up into regions based on trust.

We have mentioned economics and the discipline of competition. We argue that the "trust architecture" is the most fundamental factor, and the economic architecture can only be understood in the context of the trust architecture. With market power, monopolists can attempt to trump trust; furthermore, governments may erode trust in other ways (but they also have ways to enhance trust). If market power is the only force undermining trust, the applications may be designed to work around this and recreate the desired trust relationship. In countries where governments make "lawful" interception pervasive, application work-arounds may remain limited, and so may the experience of communication that can be described as trust-to-trust. Depending on the regime, the notion of trust may be more or less nuanced—and that variation may be tempered by movement among governments to collaborate in combating cybercrime and related concerns.

We have identified a number of reasons why it might be beneficial to design applications so that parts of the application function are positioned, if not "in the network," then in a more broadly distributed implementation of the application—that is, at intermediate points rather than at the end point computers associated with the end users:

- The operator of an end node (the end user) may not want to go to the effort of providing the service with the desired level of reliability. It may be easier to delegate or out-source it.
- By performing the function at an intermediate point, the service may have access to more information (e.g., the state of many end users, not just one).
- By performing the function at an intermediate point, the end user can avoid the cost and overhead of transferring unwelcome traffic across the communications subsystem to the ultimate end point.
- An end machine might have a vulnerability that would allow a virus to attack it before a virus checker on the machine could detect it. Doing the check at an intermediate point can protect the machine from a vulnerability that its owner cannot rectify.
- Pre-positioning information at an intermediate point can make

the subsequent delivery more responsive as well as more reliable. Replicated intermediate points can specifically improve reliability.

For each of these reasons, of course, there is a range of further considerations, which, as in the framing of the original end-to-end argument, must be seen as a starting point for the consideration of the inevitable second-order effects, not as dogma.

All of these reasons seem to fit within the paradigm of *delegation.* That is, a service of these sorts would be deployed as part of an application because one of the end points chose to do so, based on a unilateral assessment of trust, function, and reliability. We could refer to the "trust circles" in Figure 1, and in most of the cases above we could include the server for such services unambiguously inside the circle belonging to one specific end point. This was the "trust-to-trust" model with end nodes that were distributed at the application level.

On the other hand, we stress the importance of "trusted third parties," and argue that these are going to be especially important in the context of parties that want to interact but do not trust each other. Again, if the third parties are selected by the end points, we see their presence as consistent with the end-to-end argument (or, as we have reframed it, the trust-to-trust argument).

Finally, we have posed a number of interesting design questions for application designers:

- Identify functional modules that might be usefully delegated or outsourced, and specify protocols that hook these together.
- Design these protocols so that the end node (the point where trust decisions are made) can keep control of the actual delegation.
- Design applications so that they can support several modes of communication, ranging from mutually open and trusting, to suspicious and bilaterally verified, or mediated by a trusted third party.

We have also highlighted the challenge of designing trustworthy end nodes.