WIKIPEDIA

End-to-end principle

The **end-to-end principle** is a design framework in computer networking. In networks designed according to this principle, application-specific features reside in the communicating end nodes of the network, rather than in intermediary nodes, such as gateways and routers, that exist to establish the network.

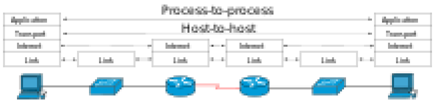
The essence of what would later be called the end-to-end principle was contained in the work of Paul Baran and Donald Davies on packet-switched networks in the 1960s. Louis Pouzin pioneered the use of the end-to-end strategy in the CYCLADES network in the 1970s.[1] The principle was first articulated explicitly in 1981 by Saltzer, Reed, and Clark.[2][a] The meaning of the end-to-end principle has been continuously reinterpreted ever since its initial articulation. Also, noteworthy formulations of the end-to-end principle can be found before the seminal 1981 Saltzer, Reed, and Clark paper.[3]

A basic premise of the principle is that the payoffs from adding features to a simple network quickly diminish, especially in cases in which the end hosts have to implement those functions only for reasons of conformance, i.e. completeness and correctness based on a specification.[b] Implementing a specific function incurs some resource penalties regardless of whether the function is used or not, and implementing a specific function *in the network* distributes these penalties among all clients.

Contents

Concept

The fundamental notion behind the end-to-end principle is that for two processes communicating with each other via some communication means, the *reliability* obtained from that means cannot be expected to be perfectly aligned with the reliability requirements of the processes. In particular, meeting or exceeding very high-reliability requirements of communicating processes separated by networks of nontrivial size is more costly than obtaining the required degree of reliability by positive end-to-end acknowledgments and retransmissions (referred to as PAR or ARQ).[c] Put differently, it is far easier to obtain reliability beyond a certain margin by mechanisms



According to the end-to-end principle, the network is only responsible for providing the terminals with connections, Any kind of intelligence should be located at the terminals.

in the *end hosts* of a network rather than in the *intermediary nodes*,[d] especially when the latter are beyond the control of, and not accountable to, the former.[e] Positive end-to-end acknowledgments with infinite retries can obtain arbitrarily high reliability from any network with a higher than zero probability of successfully transmitting data from one end to another.[f]

The end-to-end principle does not trivially extend to functions beyond end-to-end error control and correction. E.g., no straightforward end-to-end arguments can be made for communication parameters such as latency and throughput. In a 2001 paper, Blumenthal and Clark note: "[F]rom the beginning, the end-to-end arguments revolved around requirements that could be implemented correctly at the endpoints; if implementation inside the network is the only way to accomplish the requirement, then an end-to-end argument isn't appropriate in the first place."[7]:80

The end-to-end principle is closely related, and sometimes seen as a direct precursor, to the principle of net neutrality.[8]

History

In the 1960s, Paul Baran and Donald Davies, in their pre-ARPANET elaborations of networking, made brief comments about reliability that capture the essence of the later end-to-end principle. To quote from a 1964 Baran paper, "Reliability and raw error rates are secondary. The network must be built with the expectation of heavy damage anyway. Powerful error removal methods exist."[9]:5 Similarly, Davies notes on end-to-end error control, "It is thought that all users of the network will provide themselves with some kind of error control and that without difficulty this could be made to show up a missing packet. Because of this, loss of packets, if it is sufficiently rare, can be tolerated."[10]:2.3

The ARPANET was the first large-scale general-purpose packet switching network – implementing several of the basic notions previously touched on by Baran and Davies.

Davies had worked on simulation of datagram networks.[11][12] Building on this idea, Louis Pouzin's CYCLADES network was the first to make the hosts responsible for the reliable delivery of data, rather than this being a centralized service of the network itself.[1] Concepts of this network influenced later ARPANET architecture.

Applications

ARPANET

The ARPANET demonstrated several important aspects of the end-to-end principle.

Packet switching pushes some logical functions toward the communication endpoints

If the basic premise of a distributed network is packet switching, then functions such as reordering and duplicate detection inevitably have to be implemented at the logical endpoints of such a network. Consequently, the ARPANET featured two distinct levels of functionality:

1. a lower level concerned with transporting data packets between neighboring network nodes (called Interface Message Processors or IMPs), and
2. a higher level concerned with various end-to-end aspects of the data transmission.[g]

Dave Clark, one of the authors of the end-to-end principle paper, concludes: "The discovery of packets is not a consequence of the end-to-end argument. It is the success of packets that make the end-to-end argument relevant."[15]:slide 31

No arbitrarily reliable data transfer without end-to-end acknowledgment and retransmission mechanisms

The ARPANET was designed to provide reliable data transport between any two endpoints of the network – much like a simple I/O channel between a computer and a nearby peripheral device.[h] In order to remedy any potential failures of packet transmission normal ARPANET messages were handed from one node to the next node with a positive acknowledgment and retransmission scheme; after a successful handover they were then discarded,[i] no source-to-destination re-transmission in case of packet loss was catered for. However, in spite of significant efforts, perfect reliability as envisaged in the initial ARPANET specification turned out to be impossible to provide – a reality that became increasingly obvious once the ARPANET grew well beyond its initial four-node topology.[j] The ARPANET thus provided a strong case for the inherent limits of network-based hop-by-hop reliability mechanisms in pursuit of true end-to-end reliability.[k]

Trade-off between reliability, latency, and throughput

The pursuit of perfect reliability may hurt other relevant parameters of a data transmission – most importantly latency and throughput. This is particularly important for applications that value predictable throughput and low latency over reliability – the classic example being interactive real-time voice applications. This use case was catered for in the ARPANET by providing a raw message service that dispensed with various reliability measures so as to provide faster and lower latency data transmission service to the end hosts.[l]

TCP/IP

Internet Protocol (IP) is a connectionless datagram service with no delivery guarantees. On the Internet, IP is used for nearly all communications. End-to-end acknowledgment and retransmission is the responsibility of the connection-oriented Transmission Control Protocol (TCP) which sits on top of IP. The functional split between IP and TCP exemplifies the proper application of the end-to-end principle to transport protocol design.

File transfer

An example of the end-to-end principle is that of an arbitrarily reliable file transfer between two endpoints in a distributed network of a varying, nontrivial size:[3] The only way two endpoints can obtain a completely reliable transfer is by transmitting and acknowledging a checksum for the entire data stream; in such a setting, lesser checksum and acknowledgment (ACK/NACK) protocols are justified only for the purpose of optimizing performance – they are useful to the vast majority of clients, but are not enough to fulfill the reliability requirement of this particular application. A thorough checksum is hence best done at the endpoints, and the network maintains a relatively low level of complexity and reasonable performance for all clients.[3]

Limitations

The most important limitation of the end-to-end principle is that its basic premise, placing functions in the application endpoints rather than in the intermediary nodes, is not trivial to implement.

An example of the limitations of the end-to-end principle exists in mobile devices, for instance with mobile IPv6.[23] Pushing service-specific complexity to the endpoints can cause issues with mobile devices if the device has unreliable access to network channels.[24]

Further problems can be seen with a decrease in network transparency from the addition of network address translation (NAT), which IPv4 relies on to combat address exhaustion.[25] With the introduction of IPv6, users once again have unique identifiers, allowing for true end-to-end connectivity. Unique identifiers may be based on a physical address, or can be generated randomly by the host.[26]

See also

- Peer-to-peer

Notes

a. The 1981 paper[2] was published in ACM's TOCS in an updated version in 1984.[3][4]

b. The full quote from the Saltzer, Reed, Clark paper states:[3] "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. When doing so, it becomes apparent that there is a list of functions each of which might be implemented in any of several ways: by the communication subsystem, by its client, as a joint venture, or perhaps redundantly, each doing its own version. In reasoning about this choice, the requirements of the application provide the basis for the following class of arguments: The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible, and moreover, produces a performance penalty for all clients of the communication system. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.) We call this line of reasoning against low-level function implementation the end-to-end argument." (p. 278).

c. In fact, even in local area networks there is a non-zero probability of communication failure – "attention to reliability at higher levels is required regardless of the control strategy of the network".[5]

d. Put in economics terms, the marginal cost of additional reliability in the network exceeds the marginal cost of obtaining the same additional reliability by measures in the end hosts. The economically efficient level of reliability improvement inside the network depends on the specific circumstances; however, it is certainly nowhere near zero:[3] "Clearly, some effort at the lower levels to improve network reliability can have a significant effect on application performance. (p. 281)."

e. The possibility of enforceable contractual remedies notwithstanding, it is impossible for any network in which intermediary resources are shared in a non-deterministic fashion to guarantee perfect reliability. At most, it may quote statistical performance averages.

f. More precisely:[6] "THM 1: A correctly functioning PAR protocol with infinite retry count never fails to deliver, loses, or duplicates messages. COR 1A: A correctly functioning PAR protocol with finite retry count never loses or duplicates messages, and the probability of failing to deliver a message can be made arbitrarily small by the sender." (p. 3).

g. In accordance with the ARPANET RFQ[13] (pp. 47 f.) the ARPANET conceptually separated certain functions. As BBN points out in a 1977 paper:[14] "[T]he ARPA Network implementation uses the technique of breaking messages into packets to minimize the delay seen for long transmissions over many hops. The ARPA Network implementation also allows several messages to be in transit simultaneously between a given pair of Hosts. However, the several messages and the packets within the messages may arrive at the destination IMP out of order, and in the event of a broken IMP or line, there may be duplicates. The task of the ARPA Network source-to-destination transmission procedure is to reorder packets and messages at their destination, to cull duplicates, and after all the packets of a message have arrived, pass the message on to the destination Host and return an end-to-end acknowledgment. (p. 284)."

h. This requirement was spelled out in the ARPANET RFQ, "From the point of view of the ARPA contractors as users of the network, the communication subnet is a self-contained facility whose software and hardware is maintained by the network contractor. In designing Interconnection Software we should only need to use the I/0 conventions for moving data into and out of the subnet and not otherwise be involved in the details of subnet operation. Specifically, error checking, fault detection, message switching, fault recovery, line switching, carrier failures and carrier quality assessment, as required to guarantee reliable network performance, are the sole responsibility of the network contractor."[13]:25
i. Notes Walden in a 1972 paper, "Each IMP holds on to a packet until it gets a positive acknowledgment from the next IMP down the line that the packet has been properly received. If it gets the acknowledgment, all is well; the IMP knows that the next IMP now has responsibility for the packet and the transmitting IMP can discard its copy of the packet."[16]:11
j. By 1973, BBN acknowledged that the initial aim of perfect reliability inside the ARPANET was not achievable, "Initially, it was thought that the only components in the network design that were prone to errors were the communications circuits, and the modem interfaces in the IMPs are equipped with a CRC checksum to detect 'almost all' such errors. The rest of the system, including Host interfaces, IMP processors, memories, and interfaces, were all considered to be error-free. We have had to re-evaluate this position in the light of our experience.[17]:1 In fact, as Metcalfe summarizes by 1973, "there have been enough bits in error in the ARPANET to fill this quota [one undetected transmission bit error per year] for centuries."[18]:7–28 See also BBN Report 2816[19]:10 ff for additional elaboration about the experiences gained in the first years of operating the ARPANET.
k. Incidentally, the ARPANET also provides a good case for the trade-offs between the cost of end-to-end reliability mechanisms versus the benefits to be obtained thus. Note that true end-to-end reliability mechanisms would have been prohibitively costly at the time, given that the specification held that there could be up to 8 host-level messages in flight at the same time between two endpoints, each having a maximum of more than 8000 bits. The amount of memory that would have been required to keep copies of all those data for possible retransmission in case no acknowledgment came from the destination IMP was too expensive to be worthwhile. As for host-based end-to-end reliability mechanisms – those would have added considerable complexity to the common host level protocol (Host-Host Protocol). While the desirability of host-host reliability mechanisms was articulated in RFC 1 (https://datatracker.ietf.org/doc/html/rfc1), after some discussion they were dispensed with (although higher-level protocols or applications were, of course, free to implement such mechanisms themselves). For a recount of the debate at the time see Bärwolff 2010,[20] pp. 56-58 and the notes therein, especially notes 151 and 163.
l. Early experiments with packet voice date back to 1971, and by 1972 more formal ARPA research on the subject commenced. As documented in RFC 660 (https://datatracker.ietf.org/doc/html/rfc660) (p. 2),[21] in 1974 BBN introduced the raw message service (Raw Message Interface, RMI) to the ARPANET, primarily in order to allow hosts to experiment with packet voice applications, but also acknowledging the use of such facility in view of possibly internetwork communication (cf. a BBN Report 2913[22] at pp. 55 f.). See also Bärwolff 2010,[20] pp. 80-84 and the copious notes therein.

References

1. Bennett, Richard (September 2009). "Designed for Change: End-to-End Arguments, Internet Innovation, and the Net Neutrality Debate" (https://www.itif.org/files/2009-designed-for-change.pdf) (PDF). Information Technology and Innovation Foundation. pp. 7, 11. Retrieved 11 September 2017.

2. Saltzer, J. H., D. P. Reed, and D. D. Clark (1981) "End-to-End Arguments in System Design". In: Proceedings of the Second International Conference on Distributed Computing Systems. Paris, France. April 8–10, 1981. IEEE Computer Society, pp. 509-512.
3. Saltzer, J. H., D. P. Reed, and D. D. Clark (1984) "End-to-End Arguments in System Design". In: ACM Transactions on Computer Systems 2.4, pp. 277-288. (See also here (http://web.mit.edu/Saltzer/www/publications/endtoend/endtoend.pdf) for a version from Saltzer's MIT homepage.)
4. Saltzer, J. H. (1980). End-to-End Arguments in System Design. Request for Comments No. 185, MIT Laboratory for Computer Science, Computer Systems Research Division. (Online copy (http://web.mit.edu/Saltzer/www/publications/rfc/csr-rfc-185.pdf)).
5. Clark, D. D., K. T. Pogran, and D. P. Reed (1978). “An Introduction to Local Area Networks”. In: Proceedings of the IEEE 66.11, pp. 1497–1517.
6. Sunshine, C. A. (1975). Issues in Communication Protocol Design – Formal Correctness. Draft. INWG Protocol Note 5. IFIP WG 6.1 (INWG). (Copy from CBI (http://xn--brwolff-5wa.de/public/Sunshine-1975-Issues-in-Communication-Protocol-Design--Formal-Correctness--INWG-Note-5.pdf)).
7. Blumenthal, M. S. and D. D. Clark (2001). "Rethinking the Design of the Internet: The End-to-End Arguments vs. the Brave World". In: ACM Transactions on Internet Technology 1.1, pp. 70–109. (Online pre-publication version (https://dspace.mit.edu/bitstream/handle/1721.1/1519/TPRC_Clark_Blumenthal.pdf?sequence=1&origin=publication_detail)).
8. Alexis C. Madrigal & Adrienne LaFrance (25 Apr 2014). "Net Neutrality: A Guide to (and History of) a Contested Idea" (https://www.theatlantic.com/technology/archive/2014/04/the-best-writing-on-net-neutrality/361237/). The Atlantic. Retrieved 5 Jun 2014. "This idea of net neutrality... [Lawrence Lessig] used to call the principle e2e, for end to end"
9. Baran, P. (1964). "On Distributed Communications Networks". In: IEEE Transactions on Communications 12.1, pp. 1–9.
10. Davies, D. W., K. A. Bartlett, R. A. Scantlebury, and P. T. Wilkinson (1967). "A Digital Communication Network for Computers Giving Rapid Response at Remote Terminals". In: SOSP '67: Proceedings of the First ACM Symposium on Operating System Principles. Gatlinburg, TN. October 1–4, 1967. New York, NY: ACM, pp. 2.1–2.17.
11. C. Hempstead; W. Worthington (2005). *Encyclopedia of 20th-Century Technology* (https://books.google.com/books?id=2ZCNAgAAQBAJ&pg=PA574). Routledge. ISBN 9781135455514. "Simulation work on packet networks was also undertaken by the NPL group."
12. Pelkey, James. "6.3 CYCLADES Network and Louis Pouzin 1971-1972". *Entrepreneurial Capitalism and Innovation: A History of Computer Communications 1968-1988* (http://www.historyofcomputercommunications.info/Book/6/6.3-CYCLADESNetworkLouisPouzin1-72.html). "He had done some simulation of datagram networks, although he had not built any, and it looked technically viable."
13. Scheblik, T. J., D. B. Dawkins, and Advanced Research Projects Agency (1968). RFQ for ARPA Computer Network. Request for Quotations. Advanced Research Projects Agency (ARPA), Department of Defense (DoD). (Online copy (http://www.cs.utexas.edu/users/chris/DIGITAL_ARCHIVE/ARPANET/RFQ-ARPA-IMP.pdf) Archived (https://web.archive.org/web/20110815020022/http://www.cs.utexas.edu/users/chris/DIGITAL_ARCHIVE/ARPANET/RFQ-ARPA-IMP.pdf) 2011-08-15 at the Wayback Machine).
14. McQuillan, J. M. and D. C. Walden (1977). "The ARPA Network Design Decisions". In: Computer Networks 1.5, pp. 243–289. (Online copy (http://www.walden-family.com/public/whole-paper.pdf)). Based on a Crowther et al. (1975) paper, which is based on BBN Report 2918, which in turn is an extract from BBN Report 2913, both from 1974.
15. Clark, D. D. (2007). Application Design and the End-to-End Arguments. MIT Communications Futures Program Bi-Annual Meeting. Philadelphia, PA. May 30–31, 2007. Presentation slides. (Online copy (http://cfp.mit.edu/events/may07/presentations/CLARK%20Application%20Design.ppt)).

16. Walden, D. C. (1972). "The Interface Message Processor, Its Algorithms, and Their Implementation". In: AFCET Journées d'Études: Réseaux de Calculateurs (AFCET Workshop on Computer Networks). Paris, France. May 25–26, 1972. Association Française pour la Cybernétique Économique et Technique (AFCET). (Online copy (http://www.walden-family.com/public/1972-afcet-paris.pdf)).
17. McQuillan, J. M. (1973). Software Checksumming in the IMP and Network Reliability. RFC 528 (https://datatracker.ietf.org/doc/html/rfc528). Historic. NWG.
18. Metcalfe, R. M. (1973). "Packet Communication". PhD thesis. Cambridge, MA: Harvard University. Online copy (http://publications.csail.mit.edu/lcs/pubs/pdf/MIT-LCS-TR-114.pdf) (revised edition, published as MIT Laboratory for Computer Science Technical Report 114). Mostly written at MIT Project MAC and Xerox PARC.
19. Bolt, Beranek and Newman Inc. (1974). Interface Message Processors for the Arpa Computer Network. BBN Report 2816. Quarterly Technical Report No.5, 1 January 1974 to 31 March 1974. Bolt, Beranek and Newman Inc. (BBN). (Private copy, courtesy of BBN (http://xn--brwolff-5wa.de/public/BBN-1974--Interface-Message-Processors-for-the-ARPA-Computer-Network--Report-2816--Quarterly-Technical-Report-5.pdf)).
20. Bärwolff, M. (2010). "End-to-End Arguments in the Internet: Principles, Practices, and Theory". Self-published online and via Createspace/Amazon (PDF, errata, etc. (http://xn--brwolff-5wa.de/publications/2010-10-PhD-thesis.html))
21. Walden, D. C. (1974) Some Changes to the IMP and the IMP/Host Interface. RFC 660 (https://datatracker.ietf.org/doc/html/rfc660). Historic. NWG.
22. BBN (1974). Interface Message Processors for the Arpa Computer Network. BBN Report 2913. Quarterly Technical Report No. 7, 1 July 1974 to 30 September 1974. Bolt, Beranek and Newman Inc. (BBN).
23. J. Kempf; R. Austein (March 2004). *The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Archichecture* (https://tools.ietf.org/html/rfc3724). Network Working Group, IETF. doi:10.17487/RFC3724 (https://doi.org/10.17487%2FRFC3724). RFC 3724 (https://tools.ietf.org/html/rfc3724).
24. "CNF Protocol Architecture" (http://www.winlab.rutgers.edu/docs/focus/CNF/protocol%20architecture.html). *Focus Projects*. Winlab, Rutgers University. Retrieved May 23, 2016.
25. Ward, Mark (2012-09-14). "Europe hits old internet address limits" (https://www.bbc.com/news/technology-19600718). *BBC News*. Retrieved 2017-02-28.
26. Steve Deering & Bob Hinden, Co-Chairs of the IETF's IP Next Generation Working Group (November 6, 1999). "Statement on IPv6 Address Privacy" (ftp://ftp.cuhk.edu.hk/pub/doc/ipng/html/ipv6-address-privacy.html). Retrieved 2017-02-28.

Retrieved from "https://en.wikipedia.org/w/index.php?title=End-to-end_principle&oldid=1030730509"

This page was last edited on 27 June 2021, at 18:26 (UTC).