Network Working Group
Request for Comments: 1958
Category: Informational

B. Carpenter, Editor
IAB

June 1996

Architectural Principles of the Internet

Status of This Memo

Abstract

The Internet and its architecture have grown in evolutionary fashion from modest beginnings, rather than from a Grand Plan. While this process of evolution is one of the main reasons for the technology's success, it nevertheless seems useful to record a snapshot of the current principles of the Internet architecture. This is intended for general guidance and general interest, and is in no way intended to be a formal or invariant reference model.

Table of Contents

1. Constant Change

In searching for Internet architectural principles, we must remember that technical change is continuous in the information technology industry. The Internet reflects this. Over the 25 years since the ARPANET started, various measures of the size of the Internet have increased by factors between 1000 (backbone speed) and 1000000 (number of hosts). In this environment, some architectural principles inevitably change. Principles that seemed inviolable a few years ago are deprecated today. Principles that seem sacred today will be deprecated tomorrow. The principle of constant change is perhaps the only principle of the Internet that should survive indefinitely.

The purpose of this document is not, therefore, to lay down dogma about how Internet protocols should be designed, or even about how they should fit together. Rather, it is to convey various guidelines that have been found useful in the past, and that may be useful to those designing new protocols or evaluating such designs.

A good analogy for the development of the Internet is that of constantly renewing the individual streets and buildings of a city, rather than razing the city and rebuilding it. The architectural principles therefore aim to provide a framework for creating cooperation and standards, as a small "spanning set" of rules that generates a large, varied and evolving space of technology.

Some current technical triggers for change include the limits to the scaling of IPv4, the fact that gigabit/second networks and multimedia present fundamentally new challenges, and the need for quality of service and security guarantees in the commercial Internet.

As Lord Kelvin stated in 1895, "Heavier-than-air flying machines are impossible." We would be foolish to imagine that the principles listed below are more than a snapshot of our current understanding.

2. Is there an Internet Architecture?

2.1 Many members of the Internet community would argue that there is no architecture, but only a tradition, which was not written down for the first 25 years (or at least not by the IAB). However, in very general terms, the community believes that the goal is connectivity, the tool is the Internet Protocol, and the intelligence is end to end rather than hidden in the network.

The current exponential growth of the network seems to show that connectivity is its own reward, and is more valuable than any individual application such as mail or the World-Wide Web. This connectivity requires technical cooperation between service providers, and flourishes in the increasingly liberal and competitive commercial telecommunications environment.

The key to global connectivity is the inter-networking layer. The key to exploiting this layer over diverse hardware providing global connectivity is the "end to end argument".

2.2 It is generally felt that in an ideal situation there should be one, and only one, protocol at the Internet level. This allows for uniform and relatively seamless operations in a competitive, multi-vendor, multi-provider public network. There can of course be multiple protocols to satisfy different requirements at other levels, and there are many successful examples of large private networks with

multiple network layer protocols in use.

In practice, there are at least two reasons why more than one network layer protocol might be in use on the public Internet. Firstly, there can be a need for gradual transition from one version of IP to another. Secondly, fundamentally new requirements might lead to a fundamentally new protocol.

The Internet level protocol must be independent of the hardware medium and hardware addressing. This approach allows the Internet to exploit any new digital transmission technology of any kind, and to decouple its addressing mechanisms from the hardware. It allows the Internet to be the easy way to interconect fundamentally different transmission media, and to offer a single platform for a wide variety of Information Infrastructure applications and services. There is a good exposition of this model, and other important fundemental issues, in [Clark].

2.3 It is also generally felt that end-to-end functions can best be realised by end-to-end protocols.

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

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

This principle has important consequences if we require applications to survive partial network failures. An end-to-end protocol design should not rely on the maintenance of state (i.e. information about the state of the end-to-end communication) inside the network. Such state should be maintained only in the endpoints, in such a way that the state can only be destroyed when the endpoint itself breaks (known as fate-sharing). An immediate consequence of this is that datagrams are better than classical virtual circuits. The network's job is to transmit datagrams as efficiently and flexibly as possible.

Everything else should be done at the fringes.

To perform its services, the network maintains some state information: routes, QoS guarantees that it makes, session information where that is used in header compression, compression histories for data compression, and the like. This state must be self-healing; adaptive procedures or protocols must exist to derive and maintain that state, and change it when the topology or activity of the network changes. The volume of this state must be minimized, and the loss of the state must not result in more than a temporary denial of service given that connectivity exists. Manually configured state must be kept to an absolute minimum.

2.4 Fortunately, nobody owns the Internet, there is no centralized control, and nobody can turn it off. Its evolution depends on rough consensus about technical proposals, and on running code. Engineering feed-back from real implementations is more important than any architectural principles.

3. General Design Issues

3.1 Heterogeneity is inevitable and must be supported by design. Multiple types of hardware must be allowed for, e.g. transmission speeds differing by at least 7 orders of magnitude, various computer word lengths, and hosts ranging from memory-starved microprocessors up to massively parallel supercomputers. Multiple types of application protocol must be allowed for, ranging from the simplest such as remote login up to the most complex such as distributed databases.

3.2 If there are several ways of doing the same thing, choose one. If a previous design, in the Internet context or elsewhere, has successfully solved the same problem, choose the same solution unless there is a good technical reason not to. Duplication of the same protocol functionality should be avoided as far as possible, without of course using this argument to reject improvements.

3.3 All designs must scale readily to very many nodes per site and to many millions of sites.

3.4 Performance and cost must be considered as well as functionality.

3.5 Keep it simple. When in doubt during design, choose the simplest solution.

3.6 Modularity is good. If you can keep things separate, do so.

3.7 In many cases it is better to adopt an almost complete solution now, rather than to wait until a perfect solution can be found.

3.8 Avoid options and parameters whenever possible. Any options and parameters should be configured or negotiated dynamically rather than manually.

3.9 Be strict when sending and tolerant when receiving. Implementations must follow specifications precisely when sending to the network, and tolerate faulty input from the network. When in doubt, discard faulty input silently, without returning an error message unless this is required by the specification.

3.10 Be parsimonious with unsolicited packets, especially multicasts and broadcasts.

3.11 Circular dependencies must be avoided.

For example, routing must not depend on look-ups in the Domain Name System (DNS), since the updating of DNS servers depends on successful routing.

3.12 Objects should be self decribing (include type and size), within reasonable limits. Only type codes and other magic numbers assigned by the Internet Assigned Numbers Authority (IANA) may be used.

3.13 All specifications should use the same terminology and notation, and the same bit- and byte-order convention.

3.14 And perhaps most important: Nothing gets standardised until there are multiple instances of running code.

4. Name and address issues

4.1 Avoid any design that requires addresses to be hard coded or stored on non-volatile storage (except of course where this is an essential requirement as in a name server or configuration server). In general, user applications should use names rather than addresses.

4.2 A single naming structure should be used.

4.3 Public (i.e. widely visible) names should be in case-independent ASCII. Specifically, this refers to DNS names, and to protocol elements that are transmitted in text format.

4.4 Addresses must be unambiguous (unique within any scope where they may appear).

4.5 Upper layer protocols must be able to identify end-points unambiguously. In practice today, this means that addresses must be the same at start and finish of transmission.

5. External Issues

5.1 Prefer unpatented technology, but if the best technology is patented and is available to all at reasonable terms, then incorporation of patented technology is acceptable.

5.2 The existence of export controls on some aspects of Internet technology is only of secondary importance in choosing which technology to adopt into the standards. All of the technology required to implement Internet standards can be fabricated in each country, so world wide deployment of Internet technology does not depend on its exportability from any particular country or countries.

5.3 Any implementation which does not include all of the required components cannot claim conformance with the standard.

5.4 Designs should be fully international, with support for localisation (adaptation to local character sets). In particular, there should be a uniform approach to character set tagging for information content.

6. Related to Confidentiality and Authentication

6.1 All designs must fit into the IP security architecture.

6.2 It is highly desirable that Internet carriers protect the privacy and authenticity of all traffic, but this is not a requirement of the architecture. Confidentiality and authentication are the responsibility of end users and must be implemented in the protocols used by the end users. Endpoints should not depend on the confidentiality or integrity of the carriers. Carriers may choose to provide some level of protection, but this is secondary to the primary responsibility of the end users to protect themselves.

6.3 Wherever a cryptographic algorithm is called for in a protocol, the protocol should be designed to permit alternative algorithms to be used and the specific algorithm employed in a particular implementation should be explicitly labeled. Official labels for algorithms are to be recorded by the IANA.

(It can be argued that this principle could be generalised beyond the security area.)

6.4 In choosing algorithms, the algorithm should be one which is widely regarded as strong enough to serve the purpose. Among alternatives all of which are strong enough, preference should be given to algorithms which have stood the test of time and which are not unnecessarily inefficient.

6.5 To ensure interoperation between endpoints making use of security services, one algorithm (or suite of algorithms) should be mandated to ensure the ability to negotiate a secure context between implementations. Without this, implementations might otherwise not have an algorithm in common and not be able to communicate securely.

Acknowledgements

This document is a collective work of the Internet community, published by the Internet Architecture Board. Special thanks to Fred Baker, Noel Chiappa, Donald Eastlake, Frank Kastenholz, Neal McBurnett, Masataka Ohta, Jeff Schiller and Lansing Sloan.

References

Note that the references have been deliberately limited to two fundamental papers on the Internet architecture.

[Clark] The Design Philosophy of the DARPA Internet Protocols, D.D.Clark, Proc SIGCOMM 88, ACM CCR Vol 18, Number 4, August 1988, pages 106-114 (reprinted in ACM CCR Vol 25, Number 1, January 1995, pages 102-111).

[Saltzer] End-To-End Arguments in System Design, J.H. Saltzer, D.P.Reed, D.D.Clark, ACM TOCS, Vol 2, Number 4, November 1984, pp 277-288.

Security Considerations

Security issues are discussed throughout this memo.

Editor's Address

Brian E. Carpenter
Group Leader, Communications Systems
Computing and Networks Division
CERN
European Laboratory for Particle Physics
1211 Geneva 23, Switzerland

Phone: +41 22 767-4967
Fax: +41 22 767-7155
EMail: brian@dxcoms.cern.ch