

# An Introduction to Computer Networks

Stanford Univ CS144 Fall 2012

# Contents

## Articles

--WEEK ONE--	<b>1</b>
--------------	----------

Introduction	<b>2</b>
--------------	----------

Internet	2
----------	---

What the Internet is	<b>20</b>
----------------------	-----------

Internet protocol suite	20
-------------------------	----

OSI model	31
-----------	----

Internet Protocol	39
-------------------	----

Transmission Control Protocol	42
-------------------------------	----

User Datagram Protocol	59
------------------------	----

Internet Control Message Protocol	65
-----------------------------------	----

Hypertext Transfer Protocol	68
-----------------------------	----

Skype protocol	75
----------------	----

BitTorrent	81
------------	----

Architectural Principles	<b>95</b>
--------------------------	-----------

Encapsulation (networking)	95
----------------------------	----

Packet switching	96
------------------	----

Hostname	100
----------	-----

End-to-end principle	102
----------------------	-----

Finite-state machine	107
----------------------	-----

--END WEEK ONE--	<b>118</b>
------------------	------------

## References

Article Sources and Contributors	119
----------------------------------	-----

Image Sources, Licenses and Contributors	124
--	-----

## Article Licenses

License	125
---------	-----

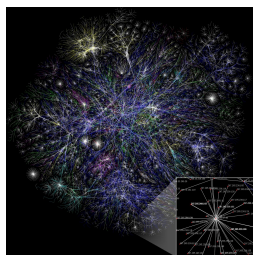
---

# --WEEK ONE--

---

# Introduction

## Internet



Routing paths through a portion of the Internet as visualized by the Opte Project

### General

Access · Censorship · Democracy

Digital divide · Digital rights

Freedom · History · Network neutrality

Phenomenon · Pioneers · Privacy

Sociology · Usage

### Internet governance

Internet Corporation for Assigned  
Names and Numbers (ICANN)

Internet Engineering Task Force (IETF)

Internet Governance Forum (IGF)

Internet Society (ISOC)

### Protocols and infrastructure

Domain Name System (DNS)

Hypertext Transfer Protocol (HTTP)

IP address Internet exchange point

Internet Protocol (IP)

Internet Protocol Suite (TCP/IP)

Internet service provider (ISP)

Simple Mail Transfer Protocol (SMTP)

### Services


Blogs · Microblogs · E-mail

Fax · File sharing · File transfer

Instant messaging · Gaming

Podcast · TV · Search

Shopping · Voice over IP (VoIP)

World Wide Web	
Guides	
Outline	
	<i>Internet portal</i>

The **Internet** is a global system of interconnected computer networks that use the standard Internet protocol suite (often called TCP/IP, although not all applications use TCP) to serve billions of users worldwide. It is a *network of networks* that consists of millions of private, public, academic, business, and government networks, of local to global scope, that are linked by a broad array of electronic, wireless and optical networking technologies. The Internet carries an extensive range of information resources and services, such as the inter-linked hypertext documents of the World Wide Web (WWW) and the infrastructure to support email.

Most traditional communications media including telephone, music, film, and television are reshaped or redefined by the Internet, giving birth to new services such as Voice over Internet Protocol (VoIP) and Internet Protocol Television (IPTV). Newspaper, book and other print publishing are adapting to Web site technology, or are reshaped into blogging and web feeds. The Internet has enabled and accelerated new forms of human interactions through instant messaging, Internet forums, and social networking. Online shopping has boomed both for major retail outlets and small artisans and traders. Business-to-business and financial services on the Internet affect supply chains across entire industries.

The origins of the Internet reach back to research of the 1960s, commissioned by the United States government in collaboration with private commercial interests to build robust, fault-tolerant, and distributed computer networks. The funding of a new U.S. backbone by the National Science Foundation in the 1980s, as well as private funding for other commercial backbones, led to worldwide participation in the development of new networking technologies, and the merger of many networks. The commercialization of what was by the 1990s an international network resulted in its popularization and incorporation into virtually every aspect of modern human life. As of 2011 more than 2.2 billion people—nearly a third of Earth's population—used the services of the Internet.<sup>[1]</sup>

The Internet has no centralized governance in either technological implementation or policies for access and usage; each constituent network sets its own standards. Only the overarching definitions of the two principal name spaces in the Internet, the Internet Protocol address space and the Domain Name System, are directed by a maintainer organization, the Internet Corporation for Assigned Names and Numbers (ICANN). The technical underpinning and standardization of the core protocols (IPv4 and IPv6) is an activity of the Internet Engineering Task Force (IETF), a non-profit organization of loosely affiliated international participants that anyone may associate with by contributing technical expertise.

## Terminology

*Internet* is a short form of the technical term internetwork,<sup>[2]</sup> the result of interconnecting computer networks with special gateways or routers. Historically the word has been used, uncapitalized, as a verb and adjective since 1883 to refer to interconnected motions. It was also used from 1974 before *the* Internet, uncapitalized, as a verb meaning to connect together, especially for networks.<sup>[3]</sup> The Internet is also often referred to as *the Net*.

The *Internet*, referring to the specific entire global system of IP networks, is a proper noun and written with an initial capital letter. In the media and common use it is often not capitalized: "the internet". Some guides specify that the word should be capitalized as a noun but not capitalized as an adjective.<sup>[4]</sup>

The terms *Internet* and *World Wide Web* are often used interchangeably in everyday speech; it is common to speak of going on the Internet when invoking a browser to view Web pages. However, the Internet is a particular global

computer network connecting millions of computing devices; the World Wide Web is just one of many services running on the Internet. The Web is a collection of interconnected documents (Web pages) and other resources, linked by hyperlinks and URLs.<sup>[5]</sup> In addition to the Web, a multitude of other services are implemented over the Internet, including e-mail, file transfer, remote computer control, newsgroups, and online games. Web (and other) services can be implemented on any intranet, accessible to network users.

## History

Research into packet switching started in the early 1960s and packet switched networks such as Mark I at NPL in the UK,<sup>[6]</sup> ARPANET, CYCLADES,<sup>[7][8]</sup> Merit Network,<sup>[9]</sup> Tymnet, and Telenet, were developed in the late 1960s and early 1970s using a variety of protocols. The ARPANET in particular led to the development of protocols for internetworking, where multiple separate networks could be joined together into a network of networks thanks to the work of British scientist Donald Davies whose ground-breaking work on Packet Switching was essential to the system.<sup>[10]</sup>

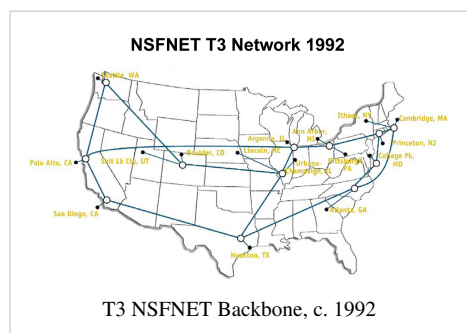
The first two nodes of what would become the ARPANET were interconnected between Leonard Kleinrock's Network Measurement Center at the UCLA's School of Engineering and Applied Science and Douglas Engelbart's NLS system at SRI International (SRI) in Menlo Park, California, on 29 October 1969.<sup>[11]</sup> The third site on the ARPANET was the Culler-Fried Interactive Mathematics center at the University of California at Santa Barbara, and the fourth was the University of Utah Graphics Department. In an early sign of future growth, there were already fifteen sites connected to the young ARPANET by the end of 1971.<sup>[12][13]</sup> These early years were documented in the 1972 film *Computer Networks: The Heralds of Resource Sharing*.



Professor Leonard Kleinrock with the first ARPANET Interface Message Processors at UCLA

Early international collaborations on ARPANET were sparse. For various political reasons, European developers were concerned with developing the X.25 networks.<sup>[14]</sup> Notable exceptions were the *Norwegian Seismic Array* (NORSAR) in 1972, followed in 1973 by Sweden with satellite links to the Tanum Earth Station and Peter T. Kirstein's research group in the UK, initially at the Institute of Computer Science, London University and later at University College London.<sup>[15]</sup>

In December 1974, *RFC 675 – Specification of Internet Transmission Control Program*, by Vinton Cerf, Yogen Dalal, and Carl Sunshine, used the term *internet*, as a shorthand for *internetworking*; later RFCs repeat this use, so the word started out as an adjective rather than the noun it is today.<sup>[16]</sup> Access to the ARPANET was expanded in 1981 when the National Science Foundation (NSF) developed the Computer Science Network (CSNET). In 1982, the Internet Protocol Suite (TCP/IP) was standardized and the concept of a world-wide network of fully interconnected TCP/IP networks called the Internet was introduced.



TCP/IP network access expanded again in 1986 when the National Science Foundation Network (NSFNET) provided access to supercomputer sites in the United States from research and education organizations, first at 56 kbit/s and later at 1.5 Mbit/s and 45 Mbit/s.<sup>[17]</sup> Commercial internet service providers (ISPs) began to emerge in the late 1980s and early 1990s. The ARPANET was decommissioned in 1990. The Internet was commercialized in 1995 when NSFNET was decommissioned, removing the last restrictions on the use of the Internet to carry commercial traffic.<sup>[18]</sup> The Internet started a rapid expansion to Europe and Australia in the mid to late 1980s<sup>[19][20]</sup> and to Asia

in the late 1980s and early 1990s.<sup>[21]</sup>

Since the mid-1990s the Internet has had a tremendous impact on culture and commerce, including the rise of near instant communication by email, instant messaging, Voice over Internet Protocol (VoIP) "phone calls", two-way interactive video calls, and the World Wide Web<sup>[22]</sup> with its discussion forums, blogs, social networking, and online shopping sites. Increasing amounts of data are transmitted at higher and higher speeds over fiber optic networks operating at 1-Gbit/s, 10-Gbit/s, or more. The Internet continues to grow, driven by ever greater amounts of online information and knowledge, commerce, entertainment and social networking.<sup>[23]</sup>

During the late 1990s, it was estimated that traffic on the public Internet grew by 100 percent per year, while the mean annual growth in the number of Internet users was thought to be between 20% and 50%.<sup>[24]</sup> This growth is often attributed to the lack of central administration, which allows organic growth of the network, as well as the non-proprietary open nature of the Internet protocols, which encourages vendor interoperability and prevents any one company from exerting too much control over the network.<sup>[25]</sup> As of 31 March 2011, the estimated total number of Internet users was 2.095 billion (30.2% of world population).<sup>[26]</sup> It is estimated that in 1993 the Internet carried only 1% of the information flowing through two-way telecommunication, by 2000 this figure had grown to 51%, and by 2007 more than 97% of all telecommunicated information was carried over the Internet.<sup>[27]</sup>



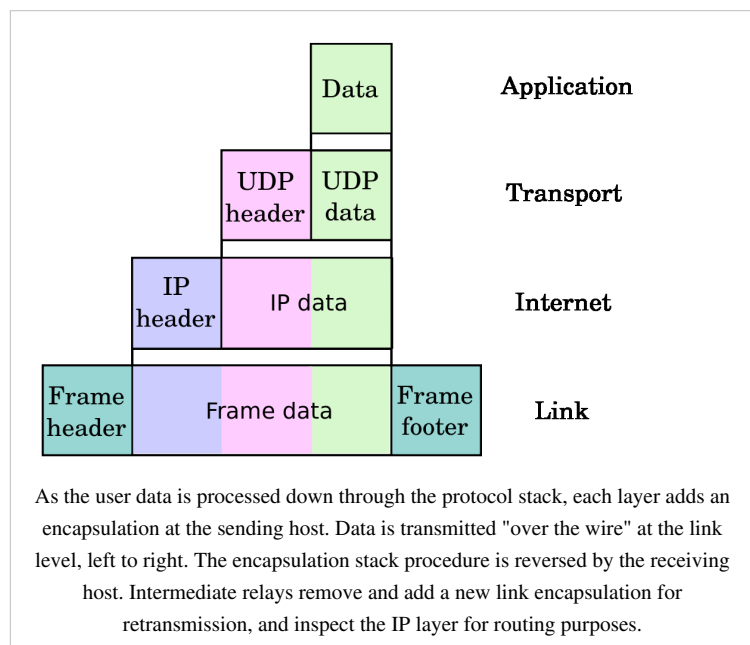
This NeXT Computer was used by Sir Tim Berners-Lee at CERN and became the world's first Web server.

## Technology

### Protocols

The communications infrastructure of the Internet consists of its hardware components and a system of software layers that control various aspects of the architecture. While the hardware can often be used to support other software systems, it is the design and the rigorous standardization process of the software architecture that characterizes the Internet and provides the foundation for its scalability and success. The responsibility for the architectural design of the Internet software systems has been delegated to the Internet Engineering Task Force (IETF).<sup>[28]</sup> The IETF conducts standard-setting work groups, open to any individual, about the various aspects of Internet architecture. Resulting discussions and final standards are

published in a series of publications, each called a Request for Comments (RFC), freely available on the IETF web site. The principal methods of networking that enable the Internet are contained in specially designated RFCs that constitute the Internet Standards. Other less rigorous documents are simply informative, experimental, or historical, or document the best current practices (BCP) when implementing Internet technologies.



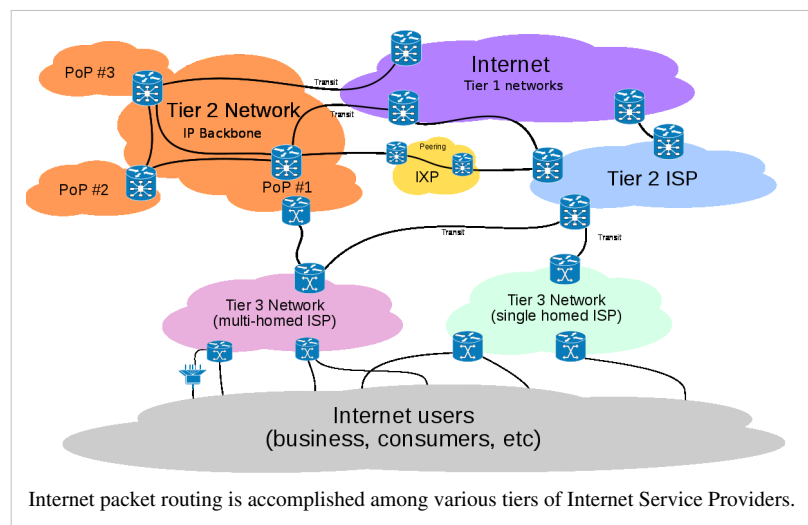
The Internet standards describe a framework known as the Internet protocol suite. This is a model architecture that divides methods into a layered system of protocols (RFC 1122, RFC 1123). The layers correspond to the environment or scope in which their services operate. At the top is the application layer, the space for the application-specific networking methods used in software applications, e.g., a web browser program. Below this top layer, the transport layer connects applications on *different hosts* via the network (e.g., client–server model) with appropriate data exchange methods. Underlying these layers are the core networking technologies, consisting of two layers. The internet layer enables computers to identify and locate each other via Internet Protocol (IP) addresses, and allows them to connect to one another via intermediate (transit) networks. Last, at the bottom of the architecture, is a software layer, the link layer, that provides connectivity between hosts on the same local network link, such as a local area network (LAN) or a dial-up connection. The model, also known as TCP/IP, is designed to be independent of the underlying hardware, which the model therefore does not concern itself with in any detail. Other models have been developed, such as the Open Systems Interconnection (OSI) model, but they are not compatible in the details of description or implementation; many similarities exist and the TCP/IP protocols are usually included in the discussion of OSI networking.

The most prominent component of the Internet model is the Internet Protocol (IP), which provides addressing systems (IP addresses) for computers on the Internet. IP enables internetworking and in essence establishes the Internet itself. IP Version 4 (IPv4) is the initial version used on the first generation of today's Internet and is still in dominant use. It was designed to address up to  $\sim 4.3$  billion ( $10^9$ ) Internet hosts. However, the explosive growth of the Internet has led to IPv4 address exhaustion, which entered its final stage in 2011,<sup>[29]</sup> when the global address allocation pool was exhausted. A new protocol version, IPv6, was developed in the mid-1990s, which provides vastly larger addressing capabilities and more efficient routing of Internet traffic. IPv6 is currently in growing deployment around the world, since Internet address registries (RIRs) began to urge all resource managers to plan rapid adoption and conversion.<sup>[30]</sup>

IPv6 is not interoperable with IPv4. In essence, it establishes a parallel version of the Internet not directly accessible with IPv4 software. This means software upgrades or translator facilities are necessary for networking devices that need to communicate on both networks. Most modern computer operating systems already support both versions of the Internet Protocol. Network infrastructures, however, are still lagging in this development. Aside from the complex array of physical connections that make up its infrastructure, the Internet is facilitated by bi- or multi-lateral commercial contracts (e.g., peering agreements), and by technical specifications or protocols that describe how to exchange data over the network. Indeed, the Internet is defined by its interconnections and routing policies.

## Routing

Internet Service Providers connect customers (thought of at the "bottom" of the routing hierarchy) to customers of other ISPs. At the "top" of the routing hierarchy are ten or so Tier 1 networks, large telecommunication companies which exchange traffic directly "across" to all other Tier 1 networks via unpaid peering agreements. Tier 2 networks buy Internet transit from other ISP to reach at least some parties on the global





Internet, though they may also engage in unpaid peering (especially for local partners of a similar size). ISPs can use a single "upstream" provider for connectivity, or use multihoming to provide protection from problems with individual links. Internet exchange points create physical connections between multiple ISPs, often hosted in buildings owned by independent third parties.

Computers and routers use routing tables to direct IP packets among locally connected machines. Tables can be constructed manually or automatically via DHCP for an individual computer or a routing protocol for routers themselves. In single-homed situations, a default route usually points "up" toward an ISP providing transit. Higher-level ISPs use the Border Gateway Protocol to sort out paths to any given range of IP addresses across the complex connections of the global Internet.

Academic institutions, large companies, governments, and other organizations can perform the same role as ISPs, engaging in peering and purchasing transit on behalf of their internal networks of individual computers. Research networks tend to interconnect into large subnetworks such as GEANT, GLORIAD, Internet2, and the UK's national research and education network, JANET. These in turn are built around smaller networks (see the list of academic computer network organizations).

Not all computer networks are connected to the Internet. For example, some classified United States websites are only accessible from separate secure networks.

## General structure

The Internet structure and its usage characteristics have been studied extensively. It has been determined that both the Internet IP routing structure and hypertext links of the World Wide Web are examples of scale-free networks.<sup>[31]</sup>

Many computer scientists describe the Internet as a "prime example of a large-scale, highly engineered, yet highly complex system".<sup>[32]</sup> The Internet is heterogeneous; for instance, data transfer rates and physical characteristics of connections vary widely. The Internet exhibits "emergent phenomena" that depend on its large-scale organization. For example, data transfer rates exhibit temporal self-similarity. The principles of the routing and addressing methods for traffic in the Internet reach back to their origins in the 1960s when the eventual scale and popularity of the network could not be anticipated. Thus, the possibility of developing alternative structures is investigated.<sup>[33]</sup> The Internet structure was found to be highly robust<sup>[34]</sup> to random failures and very vulnerable to high degree attacks.<sup>[35]</sup>

## Governance

The Internet is a globally distributed network comprising many voluntarily interconnected autonomous networks. It operates without a central governing body. However, to maintain interoperability, all technical and policy aspects of the underlying core infrastructure and the principal name spaces are administered by the Internet Corporation for Assigned Names and Numbers (ICANN), headquartered in Marina del Rey, California. ICANN is the authority that coordinates the assignment of unique identifiers for use on the Internet, including domain names, Internet Protocol (IP) addresses, application port numbers in the transport protocols, and many other parameters. Globally unified name spaces, in which names and numbers are uniquely assigned, are essential for the global reach of the Internet.

ICANN is governed by an international board of directors drawn from across the Internet technical, business, academic, and other non-commercial communities. The government of the United States continues to have the primary role in approving changes to the DNS root zone that lies at the heart of the domain name system.<sup>[36]</sup> ICANN's role in coordinating the assignment of unique identifiers distinguishes it as perhaps the only central



ICANN headquarters in Marina Del Rey,  
California, United States

coordinating body on the global Internet. On 16 November 2005, the World Summit on the Information Society, held in Tunis, established the Internet Governance Forum (IGF) to discuss Internet-related issues.

## Modern uses

The Internet allows greater flexibility in working hours and location, especially with the spread of unmetered high-speed connections. The Internet can be accessed almost anywhere by numerous means, including through mobile Internet devices. Mobile phones, datacards, handheld game consoles and cellular routers allow users to connect to the Internet wirelessly. Within the limitations imposed by small screens and other limited facilities of such pocket-sized devices, the services of the Internet, including email and the web, may be available. Service providers may restrict the services offered and mobile data charges may be significantly higher than other access methods.

Educational material at all levels from pre-school to post-doctoral is available from websites. Examples range from CBeebies, through school and high-school revision guides, virtual universities, to access to top-end scholarly literature through the likes of Google Scholar. For distance education, help with homework and other assignments, self-guided learning, whiling away spare time, or just looking up more detail on an interesting fact, it has never been easier for people to access educational information at any level from anywhere. The Internet in general and the World Wide Web in particular are important enablers of both formal and informal education.

The low cost and nearly instantaneous sharing of ideas, knowledge, and skills has made collaborative work dramatically easier, with the help of collaborative software. Not only can a group cheaply communicate and share ideas but the wide reach of the Internet allows such groups more easily to form. An example of this is the free software movement, which has produced, among other things, Linux, Mozilla Firefox, and OpenOffice.org. Internet chat, whether in the form of an IRC chat room or channel, via an instant messaging system, or a social networking website, allows colleagues to stay in touch in a very convenient way when working at their computers during the day. Messages can be exchanged even more quickly and conveniently than via email. These systems may allow files to be exchanged, drawings and images to be shared, or voice and video contact between team members.

Content management systems allow collaborating teams to work on shared sets of documents simultaneously without accidentally destroying each other's work. Business and project teams can share calendars as well as documents and other information. Such collaboration occurs in a wide variety of areas including scientific research, software development, conference planning, political activism and creative writing. Social and political collaboration is also becoming more widespread as both Internet access and computer literacy spread.

The Internet allows computer users to remotely access other computers and information stores easily, wherever they may be. They may do this with or without computer security, i.e. authentication and encryption technologies, depending on the requirements. This is encouraging new ways of working from home, collaboration and information sharing in many industries. An accountant sitting at home can audit the books of a company based in another country, on a server situated in a third country that is remotely maintained by IT specialists in a fourth. These accounts could have been created by home-working bookkeepers, in other remote locations, based on information emailed to them from offices all over the world. Some of these things were possible before the widespread use of the Internet, but the cost of private leased lines would have made many of them infeasible in practice. An office worker away from their desk, perhaps on the other side of the world on a business trip or a holiday, can access their emails, access their data using cloud computing, or open a remote desktop session into their office PC using a secure Virtual Private Network (VPN) connection on the Internet. This can give the worker complete access to all of their normal files and data, including email and other applications, while away from the office. This concept has been referred to among system administrators as the Virtual Private Nightmare,<sup>[37]</sup> because it extends the secure perimeter of a corporate network into remote locations and its employees' homes.

## Services

### World Wide Web

Many people use the terms *Internet* and *World Wide Web*, or just the *Web*, interchangeably, but the two terms are not synonymous. The World Wide Web is a global set of documents, images and other resources, logically interrelated by hyperlinks and referenced with Uniform Resource Identifiers (URIs). URIs symbolically identify services, servers, and other databases, and the documents and resources that they can provide. Hypertext Transfer Protocol (HTTP) is the main access protocol of the World Wide Web, but it is only one of the hundreds of communication protocols used on the Internet. Web services also use HTTP to allow software systems to communicate in order to share and exchange business logic and data.

World Wide Web browser software, such as Microsoft's Internet Explorer, Mozilla Firefox, Opera, Apple's Safari, and Google Chrome, lets users navigate from one web page to another via hyperlinks embedded in the documents. These documents may also contain any combination of computer data, including graphics, sounds, text, video, multimedia and interactive content that runs while the user is interacting with the page. Client-side software can include animations, games, office applications and scientific demonstrations. Through keyword-driven Internet research using search engines like Yahoo! and Google, users worldwide have easy, instant access to a vast and diverse amount of online information. Compared to printed media, books, encyclopedias and traditional libraries, the World Wide Web has enabled the decentralization of information on a large scale.

The Web has also enabled individuals and organizations to publish ideas and information to a potentially large audience online at greatly reduced expense and time delay. Publishing a web page, a blog, or building a website involves little initial cost and many cost-free services are available. Publishing and maintaining large, professional web sites with attractive, diverse and up-to-date information is still a difficult and expensive proposition, however. Many individuals and some companies and groups use *web logs* or blogs, which are largely used as easily updatable online diaries. Some commercial organizations encourage staff to communicate advice in their areas of specialization in the hope that visitors will be impressed by the expert knowledge and free information, and be attracted to the corporation as a result. One example of this practice is Microsoft, whose product developers publish their personal blogs in order to pique the public's interest in their work. Collections of personal web pages published by large service providers remain popular, and have become increasingly sophisticated. Whereas operations such as Angelfire and GeoCities have existed since the early days of the Web, newer offerings from, for example, Facebook and Twitter currently have large followings. These operations often brand themselves as social network services rather than simply as web page hosts.

Advertising on popular web pages can be lucrative, and e-commerce or the sale of products and services directly via the Web continues to grow.

When the Web began in the 1990s, a typical web page was stored in completed form on a web server, formatted in HTML, ready to be sent to a user's browser in response to a request. Over time, the process of creating and serving web pages has become more automated and more dynamic. Websites are often created using content management or wiki software with, initially, very little content. Contributors to these systems, who may be paid staff, members of a club or other organization or members of the public, fill underlying databases with content using editing pages designed for that purpose, while casual visitors view and read this content in its final HTML form. There may or may not be editorial, approval and security systems built into the process of taking newly entered content and making it available to the target visitors.

## Communication

Email is an important communications service available on the Internet. The concept of sending electronic text messages between parties in a way analogous to mailing letters or memos predates the creation of the Internet. Pictures, documents and other files are sent as email attachments. Emails can be cc-ed to multiple email addresses.

Internet telephony is another common communications service made possible by the creation of the Internet. VoIP stands for Voice-over-Internet Protocol, referring to the protocol that underlies all Internet communication. The idea began in the early 1990s with walkie-talkie-like voice applications for personal computers. In recent years many VoIP systems have become as easy to use and as convenient as a normal telephone. The benefit is that, as the Internet carries the voice traffic, VoIP can be free or cost much less than a traditional telephone call, especially over long distances and especially for those with always-on Internet connections such as cable or ADSL. VoIP is maturing into a competitive alternative to traditional telephone service. Interoperability between different providers has improved and the ability to call or receive a call from a traditional telephone is available. Simple, inexpensive VoIP network adapters are available that eliminate the need for a personal computer.

Voice quality can still vary from call to call, but is often equal to and can even exceed that of traditional calls. Remaining problems for VoIP include emergency telephone number dialing and reliability. Currently, a few VoIP providers provide an emergency service, but it is not universally available. Traditional phones are line-powered and operate during a power failure; VoIP does not do so without a backup power source for the phone equipment and the Internet access devices. VoIP has also become increasingly popular for gaming applications, as a form of communication between players. Popular VoIP clients for gaming include Ventrilo and Teamspeak. Wii, PlayStation 3, and Xbox 360 also offer VoIP chat features.

## Data transfer

File sharing is an example of transferring large amounts of data across the Internet. A computer file can be emailed to customers, colleagues and friends as an attachment. It can be uploaded to a website or FTP server for easy download by others. It can be put into a "shared location" or onto a file server for instant use by colleagues. The load of bulk downloads to many users can be eased by the use of "mirror" servers or peer-to-peer networks. In any of these cases, access to the file may be controlled by user authentication, the transit of the file over the Internet may be obscured by encryption, and money may change hands for access to the file. The price can be paid by the remote charging of funds from, for example, a credit card whose details are also passed – usually fully encrypted – across the Internet. The origin and authenticity of the file received may be checked by digital signatures or by MD5 or other message digests. These simple features of the Internet, over a worldwide basis, are changing the production, sale, and distribution of anything that can be reduced to a computer file for transmission. This includes all manner of print publications, software products, news, music, film, video, photography, graphics and the other arts. This in turn has caused seismic shifts in each of the existing industries that previously controlled the production and distribution of these products.

Streaming media is the real-time delivery of digital media for the immediate consumption or enjoyment by end users. Many radio and television broadcasters provide Internet feeds of their live audio and video productions. They may also allow time-shift viewing or listening such as Preview, Classic Clips and Listen Again features. These providers have been joined by a range of pure Internet "broadcasters" who never had on-air licenses. This means that an Internet-connected device, such as a computer or something more specific, can be used to access on-line media in much the same way as was previously possible only with a television or radio receiver. The range of available types of content is much wider, from specialized technical webcasts to on-demand popular multimedia services. Podcasting is a variation on this theme, where – usually audio – material is downloaded and played back on a computer or shifted to a portable media player to be listened to on the move. These techniques using simple equipment allow anybody, with little censorship or licensing control, to broadcast audio-visual material worldwide.

---

Digital media streaming increases the demand for network bandwidth. For example, standard image quality needs 1 Mbit/s link speed for SD 480p, HD 720p quality requires 2.5 Mbit/s, and the top-of-the-line HDX quality needs 4.5 Mbit/s for 1080p.<sup>[38]</sup>

Webcams are a low-cost extension of this phenomenon. While some webcams can give full-frame-rate video, the picture either is usually small or updates slowly. Internet users can watch animals around an African waterhole, ships in the Panama Canal, traffic at a local roundabout or monitor their own premises, live and in real time. Video chat rooms and video conferencing are also popular with many uses being found for personal webcams, with and without two-way sound. YouTube was founded on 15 February 2005 and is now the leading website for free streaming video with a vast number of users. It uses a flash-based web player to stream and show video files. Registered users may upload an unlimited amount of video and build their own personal profile. YouTube claims that its users watch hundreds of millions, and upload hundreds of thousands of videos daily.<sup>[39]</sup>

## Access

Common methods of Internet access in homes include dial-up, landline broadband (over coaxial cable, fiber optic or copper wires), Wi-Fi, satellite and 3G/4G technology cell phones. Public places to use the Internet include libraries and Internet cafes, where computers with Internet connections are available. There are also Internet access points in many public places such as airport halls and coffee shops, in some cases just for brief use while standing. Various terms are used, such as "public Internet kiosk", "public access terminal", and "Web payphone". Many hotels now also have public terminals, though these are usually fee-based. These terminals are widely accessed for various usage like ticket booking, bank deposit, online payment etc. Wi-Fi provides wireless access to computer networks, and therefore can do so to the Internet itself. Hotspots providing such access include Wi-Fi cafes, where would-be users need to bring their own wireless-enabled devices such as a laptop or PDA. These services may be free to all, free to customers only, or fee-based. A hotspot need not be limited to a confined location. A whole campus or park, or even an entire city can be enabled.

Grassroots efforts have led to wireless community networks. Commercial Wi-Fi services covering large city areas are in place in London, Vienna, Toronto, San Francisco, Philadelphia, Chicago and Pittsburgh. The Internet can then be accessed from such places as a park bench.<sup>[40]</sup> Apart from Wi-Fi, there have been experiments with proprietary mobile wireless networks like Ricochet, various high-speed data services over cellular phone networks, and fixed wireless services. High-end mobile phones such as smartphones in general come with Internet access through the phone network. Web browsers such as Opera are available on these advanced handsets, which can also run a wide variety of other Internet software. More mobile phones have Internet access than PCs, though this is not as widely used.<sup>[41]</sup> An Internet access provider and protocol matrix differentiates the methods used to get online.

An Internet blackout or outage can be caused by local signaling interruptions. Disruptions of submarine communications cables may cause blackouts or slowdowns to large areas, such as in the 2008 submarine cable disruption. Less-developed countries are more vulnerable due to a small number of high-capacity links. Land cables are also vulnerable, as in 2011 when a woman digging for scrap metal severed most connectivity for the nation of Armenia.<sup>[42]</sup> Internet blackouts affecting almost entire countries can be achieved by governments as a form of Internet censorship, as in the blockage of the Internet in Egypt, whereby approximately 93%<sup>[43]</sup> of networks were without access in 2011 in an attempt to stop mobilization for anti-government protests.<sup>[44]</sup>

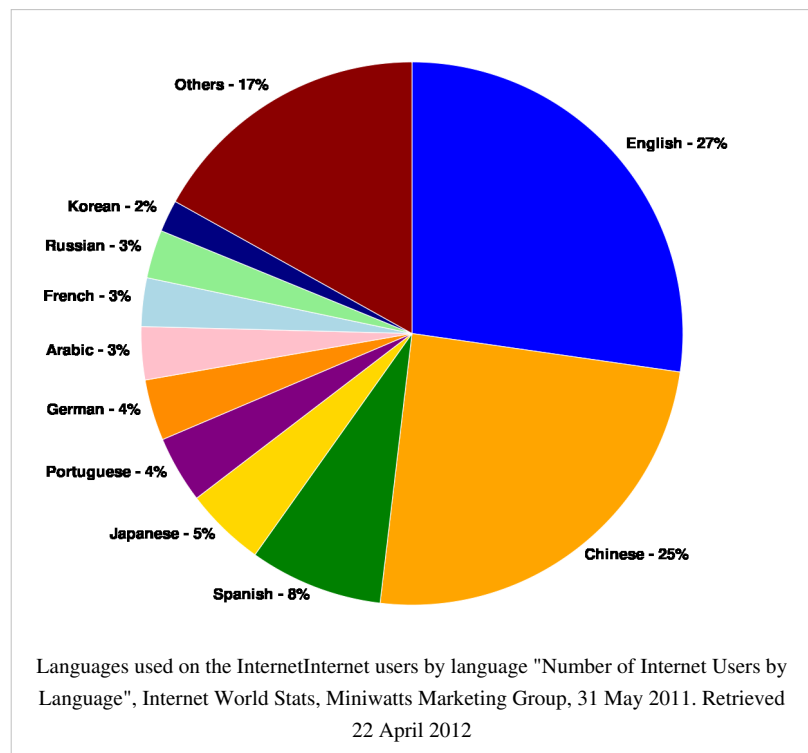
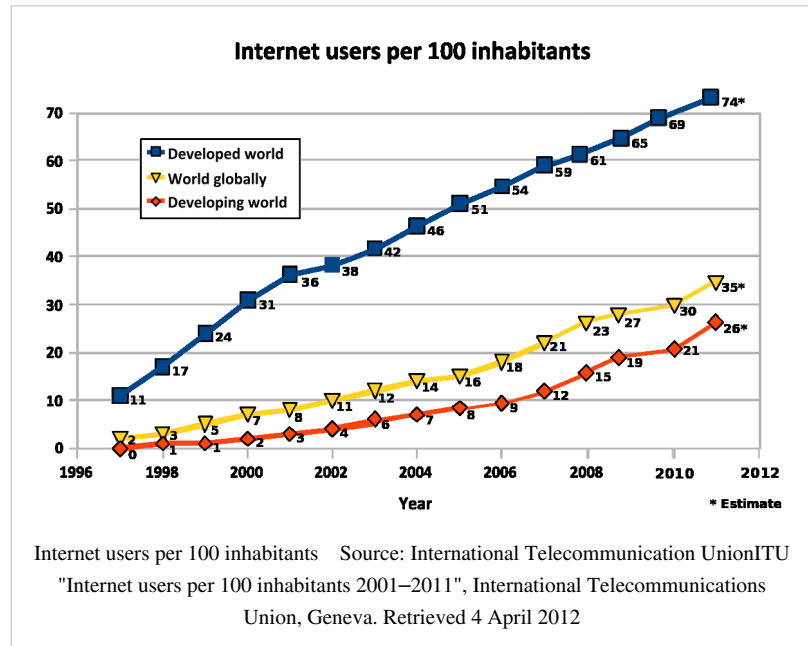
## Users

Overall Internet usage has seen tremendous growth. From 2000 to 2009, the number of Internet users globally rose from 394 million to 1.858 billion.<sup>[48]</sup> By 2010, 22 percent of the world's population had access to computers with 1 billion Google searches every day, 300 million Internet users reading blogs, and 2 billion videos viewed daily on YouTube.<sup>[49]</sup>

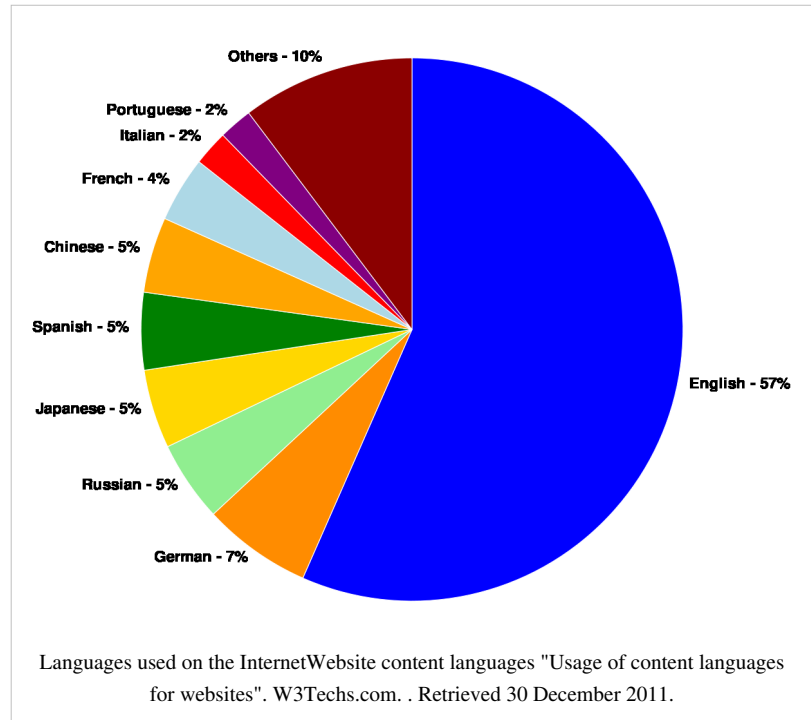
The prevalent language for communication on the Internet has been English. This may be a result of the origin of the Internet, as well as the language's role as a lingua franca. Early computer systems were limited to the characters in the American Standard Code for Information Interchange (ASCII), a subset of the Latin alphabet.

After English (27%), the most requested languages on the World Wide Web are Chinese (23%), Spanish (8%), Japanese (5%), Portuguese and German (4% each), Arabic, French and Russian (3% each), and Korean (2%).<sup>[50]</sup> By region, 42% of the world's Internet users are based in Asia, 24% in Europe, 14% in North America, 10% in Latin America and the Caribbean taken together, 6% in Africa, 3% in the Middle East and 1% in Australia/Oceania.<sup>[51]</sup> The Internet's technologies have developed enough in recent years, especially in the use of

Unicode, that good facilities are available for development and communication in the world's widely used languages. However, some glitches such as *mojibake* (incorrect display of some languages' characters) still remain.



In an American study in 2005, the percentage of men using the Internet was very slightly ahead of the percentage of women, although this difference reversed in those under 30. Men logged on more often, spent more time online, and were more likely to be broadband users, whereas women tended to make more use of opportunities to communicate (such as email). Men were more likely to use the Internet to pay bills, participate in auctions, and for recreation such as downloading music and videos. Men and women were equally likely to use the Internet for shopping and banking.<sup>[52]</sup> More recent studies indicate that in 2008, women significantly outnumbered men on



most social networking sites, such as Facebook and Myspace, although the ratios varied with age.<sup>[53]</sup> In addition, women watched more streaming content, whereas men downloaded more.<sup>[54]</sup> In terms of blogs, men were more likely to blog in the first place; among those who blog, men were more likely to have a professional blog, whereas women were more likely to have a personal blog.<sup>[55]</sup>

## Social impact

The Internet has enabled entirely new forms of social interaction, activities, and organizing, thanks to its basic features such as widespread usability and access. In the first decade of the 21st century, the first generation is raised with widespread availability of Internet connectivity, bringing consequences and concerns in areas such as personal privacy and identity, and distribution of copyrighted materials. These "digital natives" face a variety of challenges that were not present for prior generations.

## Social networking and entertainment

Many people use the World Wide Web to access news, weather and sports reports, to plan and book vacations and to find out more about their interests. People use chat, messaging and email to make and stay in touch with friends worldwide, sometimes in the same way as some previously had pen pals. The Internet has seen a growing number of Web desktops, where users can access their files and settings via the Internet.

Social networking websites such as Facebook, Twitter, and MySpace have created new ways to socialize and interact. Users of these sites are able to add a wide variety of information to pages, to pursue common interests, and to connect with others. It is also possible to find existing acquaintances, to allow communication among existing groups of people. Sites like LinkedIn foster commercial and business connections. YouTube and Flickr specialize in users' videos and photographs.

The Internet has been a major outlet for leisure activity since its inception, with entertaining social experiments such as MUDs and MOOs being conducted on university servers, and humor-related Usenet groups receiving much traffic. Today, many Internet forums have sections devoted to games and funny videos; short cartoons in the form of Flash movies are also popular. Over 6 million people use blogs or message boards as a means of communication and

for the sharing of ideas. The pornography and gambling industries have taken advantage of the World Wide Web, and often provide a significant source of advertising revenue for other websites.<sup>[56]</sup> Although many governments have attempted to restrict both industries' use of the Internet, in general this has failed to stop their widespread popularity.<sup>[57]</sup>

Another area of leisure activity on the Internet is multiplayer gaming.<sup>[58]</sup> This form of recreation creates communities, where people of all ages and origins enjoy the fast-paced world of multiplayer games. These range from MMORPG to first-person shooters, from role-playing video games to online gambling. While online gaming has been around since the 1970s, modern modes of online gaming began with subscription services such as GameSpy and MPlayer.<sup>[59]</sup> Non-subscribers were limited to certain types of game play or certain games. Many people use the Internet to access and download music, movies and other works for their enjoyment and relaxation. Free and fee-based services exist for all of these activities, using centralized servers and distributed peer-to-peer technologies. Some of these sources exercise more care with respect to the original artists' copyrights than others.

Internet usage has been correlated to users' loneliness.<sup>[60]</sup> Lonely people tend to use the Internet as an outlet for their feelings and to share their stories with others, such as in the "I am lonely will anyone speak to me" thread.

Cybersectarianism is a new organizational form which involves: "highly dispersed small groups of practitioners that may remain largely anonymous within the larger social context and operate in relative secrecy, while still linked remotely to a larger network of believers who share a set of practices and texts, and often a common devotion to a particular leader. Overseas supporters provide funding and support; domestic practitioners distribute tracts, participate in acts of resistance, and share information on the internal situation with outsiders. Collectively, members and practitioners of such sects construct viable virtual communities of faith, exchanging personal testimonies and engaging in collective study via email, on-line chat rooms and web-based message boards."<sup>[61]</sup>

Cyberslacking can become a drain on corporate resources; the average UK employee spent 57 minutes a day surfing the Web while at work, according to a 2003 study by Peninsula Business Services.<sup>[62]</sup> Internet addiction disorder is excessive computer use that interferes with daily life. Psychologist Nicolas Carr believe that Internet use has other effects on individuals, for instance improving skills of scan-reading and interfering with the deep thinking that leads to true creativity.<sup>[63]</sup>

## Politics and political revolutions

The Internet has achieved new relevance as a political tool. The presidential campaign of Howard Dean in 2004 in the United States was notable for its success in soliciting donation via the Internet. Many political groups use the Internet to achieve a new method of organizing in order to carry out their mission, having given rise to Internet activism, most notably practiced by rebels in the Arab Spring.<sup>[64][65]</sup>

The *New York Times* suggested that social media websites such as Facebook and Twitter helped people organize the political revolutions in Egypt where it helped certain classes of protesters organize protests, communicate grievances, and disseminate information.<sup>[66]</sup>

The potential of the Internet as a civic tool of communicative power was thoroughly explored by Simon R. B. Berdal in his thesis of 2004:

*As the globally evolving Internet provides ever new access points to virtual discourse forums, it also promotes new civic relations and associations within which communicative power may flow and accumulate. Thus, traditionally ... national-embedded peripheries get entangled into greater, international peripheries, with stronger combined powers... The Internet, as a consequence, changes the topology of the "centre-periphery" model, by stimulating conventional peripheries to interlink into "super-periphery" structures, which enclose and "besiege" several centres at once.*<sup>[67]</sup>

Berdal, therefore, extends the Habermasian notion of the Public sphere to the Internet, and underlines the inherent global and civic nature that intervowen Internet technologies provide. To limit the growing civic potential of the



Internet, Berdal also notes how "self-protective measures" are put in place by those threatened by it:

If we consider China's attempts to filter "unsuitable material" from the Internet, most of us would agree that this resembles a self-protective measure by the system against the growing civic potentials of the Internet. Nevertheless, both types represent limitations to "peripheral capacities". Thus, the Chinese government tries to prevent communicative power to build up and unleash (as the 1989 Tiananmen Square uprising suggests, the government may find it wise to install "upstream measures"). Even though limited, the Internet is proving to be an empowering tool also to the Chinese periphery: Analysts believe that Internet petitions have influenced policy implementation in favour of the public's online-articulated will ...<sup>[67]</sup>

## Philanthropy

The spread of low-cost internet access in developing countries has opened up new possibilities for peer-to-peer charities, which allow individuals to contribute small amounts to charitable projects for other individuals. Websites such as DonorsChoose and GlobalGiving allow small-scale donors to direct funds to individual projects of their choice.

A popular twist on internet-based philanthropy is the use of peer-to-peer lending for charitable purposes. Kiva pioneered this concept in 2005, offering the first web-based service to publish individual loan profiles for funding. Kiva raises funds for local intermediary microfinance organizations which post stories and updates on behalf of the borrowers. Lenders can contribute as little as \$25 to loans of their choice, and receive their money back as borrowers repay. Kiva falls short of being a pure peer-to-peer charity, in that loans are disbursed before being funded by lenders and borrowers do not communicate with lenders themselves.<sup>[68][69]</sup>

However, the recent spread of low cost Internet access in developing countries has made genuine international person-to-person philanthropy increasingly feasible. In 2009 the US-based nonprofit Zidisha tapped into this trend to offer the first person-to-person microfinance platform to link lenders and borrowers across international borders without intermediaries. Members can fund loans for as little as a dollar, which the borrowers then use to develop business activities that improve their families' incomes while repaying loans to the members with interest. Borrowers access the internet via public cybercafes, donated laptops in village schools, and even smart phones, then create their own profile pages through which they share photos and information about themselves and their businesses. As they repay their loans, borrowers continue to share updates and dialogue with lenders via their profile pages. This direct web-based connection allows members themselves to take on many of the communication and recording tasks traditionally performed by local organizations, bypassing geographic barriers and dramatically reducing the cost of microfinance services to the entrepreneurs.<sup>[70]</sup>

## Censorship

Some governments, such as those of Burma, Iran, North Korea, the People's Republic of China, Saudi Arabia, and the United Arab Emirates restrict what people in their countries can access on the Internet, especially political and religious content. This is accomplished through software that filters domains and content so that they may not be easily accessed or obtained without elaborate circumvention.<sup>[71]</sup>

In Norway, Denmark, Finland, and Sweden, major Internet service providers have voluntarily, possibly to avoid such an arrangement being turned into law, agreed to restrict access to sites listed by authorities. While this list of forbidden URLs is supposed to contain addresses of only known child pornography sites, the content of the list is secret.<sup>[72]</sup> Many countries, including the United States, have enacted laws against the possession or distribution of certain material, such as child pornography, via the Internet, but do not mandate filtering software. There are many free and commercially available software programs, called content-control software, with which a user can choose to block offensive websites on individual computers or networks, in order to limit a child's access to pornographic materials or depiction of violence.

## References

- [1] Internet World Stats (<http://www.internetworldstats.com/stats.htm>), updated 31 March 2011. Retrieved 8 November 2011.
- [2] "Internet, n." (<http://dictionary.oed.com/cgi/entry/00304286>). *Oxford English Dictionary* (Draft ed.). March 2009. . Retrieved 26 October 2010. "Shortened < INTERNETWORK n., perhaps influenced by similar words in -net"
- [3] Oxford English Dictionary, 2nd ed., gives nineteenth-century use and pre-Internet verb use]
- [4] "7.76 Terms like 'web' and 'Internet'" ([http://www.chicagomanualofstyle.org/16/ch07/ch07\\_sec076.html?para=](http://www.chicagomanualofstyle.org/16/ch07/ch07_sec076.html?para=)), *Chicago Manual of Style*, University of Chicago, 16th edition
- [5] "Links" (<http://www.w3.org/TR/html401/struct/links.html#h-12.1>). *HTML 4.01 Specification*. World Wide Web Consortium. HTML 4.01 Specification. . Retrieved 13 August 2008. "[T]he link (or hyperlink, or Web link) [is] the basic hypertext construct. A link is a connection from one Web resource to another. Although a simple concept, the link has been one of the primary forces driving the success of the Web."
- [6] Celebrating 40 years of the net (<http://news.bbc.co.uk/1/hi/technology/8331253.stm>), by Mark Ward, Technology correspondent, BBC News, 29 October 2009
- [7] "A Technical History of CYCLADES" (<http://www.cs.utexas.edu/users/chris/think/Cyclades/index.shtml>), *Technical Histories of the Internet & other Network Protocols*, Computer Science Department, University of Texas Austin, 11 June 2002
- [8] "The Cyclades Experience: Results and Impacts" (<http://www.informatik.uni-trier.de/~ley/db/conf/ifip/ifip1977.html#Zimmermann77>), Zimmermann, H., Proc. IFIP'77 Congress, Toronto, August 1977, pp. 465–469
- [9] *A Chronicle of Merit's Early History* (<http://www.merit.edu/about/history/article.php>), John Mulcahy, 1989, Merit Network, Ann Arbor, Michigan
- [10] "A Technical History of National Physical Laboratories (NPL) Network Architecture" (<http://www.cs.utexas.edu/users/chris/think/NPL/index.shtml>), *Technical Histories of the Internet & other Network Protocols*, Computer Science Department, University of Texas Austin, 11 June 2002
- [11] "Roads and Crossroads of Internet History" (<http://www.netvalley.com/intval.html>) by Gregory Gromov. 1995
- [12] Hafner, Katie (1998). *Where Wizards Stay Up Late: The Origins Of The Internet*. Simon & Schuster. ISBN 0-684-83267-4.
- [13] Ronda Hauben (2001). *From the ARPANET to the Internet* ([http://www.columbia.edu/~rh120/other/tcpdigest\\_paper.txt](http://www.columbia.edu/~rh120/other/tcpdigest_paper.txt)). . Retrieved 28 May 2009.
- [14] "Events in British Telecomms History" ([http://web.archive.org/web/20030405153523/http://www.sigtel.com/tel\\_hist\\_brief.html](http://web.archive.org/web/20030405153523/http://www.sigtel.com/tel_hist_brief.html)). *Events in British TelecommsHistory*. Archived from the original ([http://www.sigtel.com/tel\\_hist\\_brief.html](http://www.sigtel.com/tel_hist_brief.html)) on 5 April 2003. . Retrieved 25 November 2005.
- [15] "NORSAR and the Internet" (<http://www.norsar.no/pc-5-30-NORSAR-and-the-Internet.aspx>). NORSAR. . Retrieved 5 June 2009.
- [16] Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, Stephen Wolff (2003). *A Brief History of Internet* (<http://www.isoc.org/internet/history/brief.shtml>). . Retrieved 28 May 2009.
- [17] *NSFNET: A Partnership for High-Speed Networking, Final Report 1987-1995* ([http://www.merit.edu/about/history/pdf/NSFNET\\_final.pdf](http://www.merit.edu/about/history/pdf/NSFNET_final.pdf)), Karen D. Frazer, Merit Network, Inc., 1995
- [18] "Retiring the NSFNET Backbone Service: Chronicling the End of an Era" ([http://www.merit.edu/networkresearch/projecthistory/nsfnet/nsfnet\\_article.php](http://www.merit.edu/networkresearch/projecthistory/nsfnet/nsfnet_article.php)), Susan R. Harris and Elise Gerich, *CommeXions*, Vol. 10, No. 4, April 1996
- [19] Ben Segal (1995). *A Short History of Internet Protocols at CERN* (<http://www.cern.ch/ben/TCPHIST.html>). .
- [20] Réseaux IP Européens (RIPE)
- [21] "Internet History in Asia" (<http://www.apan.net/meetings/busan03/cs-history.htm>). *16th APAN Meetings/Advanced Network Conference in Busan*. . Retrieved 25 December 2005.
- [22] How the web went world wide (<http://news.bbc.co.uk/2/hi/science/nature/5242252.stm>), Mark Ward, Technology Correspondent, BBC News. Retrieved 24 January 2011
- [23] "Brazil, Russia, India and China to Lead Internet Growth Through 2011" (<http://clickz.com/showPage.html?page=3626274>). Clickz.com. . Retrieved 28 May 2009.
- [24] Coffman, K. G; Odlyzko, A. M. (2 October 1998) (PDF). *The size and growth rate of the Internet* (<http://www.dtc.umn.edu/~odlyzko/doc/internet.size.pdf>). AT&T Labs. . Retrieved 21 May 2007.
- [25] Comer, Douglas (2006). *The Internet book*. Prentice Hall. p. 64. ISBN 0-13-233553-0.
- [26] "World Internet Users and Population Stats" (<http://www.internetworldstats.com/stats.htm>). *Internet World Stats*. Miniwatts Marketing Group. 22 June 2011. . Retrieved 23 June 2011.
- [27] "The World's Technological Capacity to Store, Communicate, and Compute Information" (<http://www.sciencemag.org/content/suppl/2011/02/08/science.1200970.DC1/Hilbert-SOM.pdf>), Martin Hilbert and Priscila López (April 2011), *Science*, 332(6025), 60–65.
- [28] "IETF Home Page" (<http://www.ietf.org/>). Ietf.org. . Retrieved 20 June 2009.
- [29] Huston, Geoff. "IPv4 Address Report, daily generated" (<http://www.potaroo.net/tools/ipv4/index.html>). . Retrieved 20 May 2009.
- [30] "Notice of Internet Protocol version 4 (IPv4) Address Depletion" ([https://www.arin.net/knowledge/about\\_resources/ceo\\_letter.pdf](https://www.arin.net/knowledge/about_resources/ceo_letter.pdf)) (PDF). . Retrieved 7 August 2009.
- [31] A. L. Barabási, R. Albert; Barabási, Albert-László (2002). "Statistical mechanics of complex networks" ([http://rmp.aps.org/abstract/RMP/v74/i1/p47\\_1](http://rmp.aps.org/abstract/RMP/v74/i1/p47_1)). *Rev. Mod. Phys* **74**: 47–94. doi:10.1103/RevModPhys.74.47. .

- [32] Walter Willinger, Ramesh Govindan, Sugih Jamin, Vern Paxson, and Scott Shenker (2002). Scaling phenomena in the Internet ([http://www.pnas.org/cgi/content/full/99/suppl\\_1/2573](http://www.pnas.org/cgi/content/full/99/suppl_1/2573)), in *Proceedings of the National Academy of Sciences*, 99, suppl. 1, 2573–2580
- [33] Jesdanun, Anick (16 April 2007). "Internet Makeover? Some argue it's time" ([http://seattletimes.nwsources.com/html/business/technology/2003667811\\_btbuildnet16.html](http://seattletimes.nwsources.com/html/business/technology/2003667811_btbuildnet16.html)). *Seattletimes.nwsources.com*. Retrieved 8 August 2011.
- [34] R. Cohen, K. Erez, D. ben-Avraham, S. Havlin (2000). "Resilience of the Internet to random breakdowns" ([http://havlin.biu.ac.il/Publications.php?keyword=Resilience+of+the+Internet+to+random+breakdowns&year=\\*&match=all](http://havlin.biu.ac.il/Publications.php?keyword=Resilience+of+the+Internet+to+random+breakdowns&year=*&match=all)). *Phys. Rev. Lett* **85**: 4625. .
- [35] R. Cohen, K. Erez, D. ben-Avraham, S. Havlin; Erez, K; Ben-Avraham, D; Havlin, S (2001). "Breakdown of the Internet under intentional attack" ([http://havlin.biu.ac.il/Publications.php?keyword=Breakdown+of+the+Internet+under+intentional+attack&year=\\*&match=all](http://havlin.biu.ac.il/Publications.php?keyword=Breakdown+of+the+Internet+under+intentional+attack&year=*&match=all)). *Phys. Rev. Lett* **86** (16): 3682–5. doi:10.1103/PhysRevLett.86.3682. PMID 11328053. .
- [36] "Bush administration annexes internet" ([http://www.theregister.co.uk/2005/07/01/bush\\_net\\_policy/](http://www.theregister.co.uk/2005/07/01/bush_net_policy/)), Kieren McCarthy, The Register, 1 July 2005
- [37] "The Virtual Private Nightmare: VPN" (<http://librenix.com/?inode=5013>). Librenix. 4 August 2004. Retrieved 21 July 2010.
- [38] Morrison, Geoff (18 November 2010). "What to know before buying a 'connected' TV - Technology & science - Tech and gadgets - Tech Holiday Guide" ([http://www.msnbc.msn.com/id/40241749/ns/technology\\_and\\_science-tech\\_and\\_gadgets](http://www.msnbc.msn.com/id/40241749/ns/technology_and_science-tech_and_gadgets)). MSNBC. Retrieved 8 August 2011.
- [39] "YouTube Fact Sheet" (<http://www.webcitation.org/5qyMMarNd>). YouTube, LLC. Archived from the original ([http://www.youtube.com/t/fact\\_sheet](http://www.youtube.com/t/fact_sheet)) on 4 July 2010. Retrieved 20 January 2009.
- [40] Pasternak, Sean B. (7 March 2006). "Toronto Hydro to Install Wireless Network in Downtown Toronto" (<http://www.bloomberg.com/apps/news?pid=10000082&sid=aQ0ZfhMa4XGQ&refer=canada>). Bloomberg. Retrieved 8 August 2011.
- [41] "By 2013, mobile phones will overtake PCs as the most common Web access device worldwide", according a forecast in "Gartner Highlights Key Predictions for IT Organizations and Users in 2010 and Beyond" (<http://www.gartner.com/it/page.jsp?id=1278413>), Gartner, Inc., 13 January 2010
- [42] "Georgian woman cuts off web access to whole of Armenia" (<http://www.guardian.co.uk/world/2011/apr/06/georgian-woman-cuts-web-access>). The Guardian. 6 April 2011. Retrieved 11 April 2012.
- [43] Cowie, James. "Egypt Leaves the Internet" (<http://www.renesys.com/blog/2011/01/egypt-leaves-the-internet.shtml>). Renesys. Archived (<http://www.webcitation.org/5w51j0pga>) from the original on 28 January 2011. Retrieved 28 January 2011.
- [44] "Egypt severs internet connection amid growing unrest" (<http://www.bbc.co.uk/news/technology-12306041>). *BBC News*. 28 January 2011. .
- [45] "Internet users per 100 inhabitants 2001–2011" ([http://www.itu.int/ITU-D/ict/statistics/material/excel/2011/Internet\\_users\\_01-11.xls](http://www.itu.int/ITU-D/ict/statistics/material/excel/2011/Internet_users_01-11.xls)), International Telecommunications Union, Geneva. Retrieved 4 April 2012
- [46] "Number of Internet Users by Language" (<http://www.internetworldstats.com/stats7.htm>), *Internet World Stats*, Miniwatts Marketing Group, 31 May 2011. Retrieved 22 April 2012
- [47] "Usage of content languages for websites" ([http://w3techs.com/technologies/overview/content\\_language/all](http://w3techs.com/technologies/overview/content_language/all)). *W3Techs.com*. Retrieved 30 December 2011.
- [48] Internet users graphs (<http://www.itu.int/ITU-D/ict/statistics/>), Market Information and Statistics, International Telecommunications Union
- [49] <http://www.antaranews.com/en/news/71940/google-earth-demonstrates-how-technology-benefits-ris-civil-society-govt>
- [50] Internet World Stats (<http://www.internetworldstats.com/stats7.htm>), updated for 30 June 2010. Retrieved 20 February 2011.
- [51] World Internet Usage Statistics News and Population Stats (<http://www.internetworldstats.com/stats.htm>) updated for 30 June 2010. Retrieved 20 February 2011.
- [52] How men and women use the Internet Pew Research Center 28 December 2005
- [53] "Rapleaf Study on Social Network Users" ([http://business.rapleaf.com/company\\_press\\_2008\\_07\\_29.html](http://business.rapleaf.com/company_press_2008_07_29.html)). .
- [54] "Women Ahead Of Men In Online Tv, Dvr, Games, And Social Media." (<http://www.entrepreneur.com/tradejournals/article/178175272.html>). Entrepreneur.com. 1 May 2008. Retrieved 8 August 2011.
- [55] "Technorati's State of the Blogosphere" (<http://technorati.com/blogging/state-of-the-blogosphere/>). Technorati. Retrieved 8 August 2011.
- [56] "Internet Pornography Statistics" (<http://internet-filter-review.toptenreviews.com/internet-pornography-statistics.html>), Jerry Ropelato, Top Ten Reviews, 2006
- [57] "Do It Yourself! Amateur Porn Stars Make Bank" (<http://abcnews.go.com/Business/SmallBiz/story?id=4151592>), Russell Goldman, ABC News, 22 January 2008
- [58] "Top Online Game Trends of the Decade" (<http://internetgames.about.com/od/gamingnews/a/trendsdecade.htm>), Dave Spohn, About.com, 15 December 2009
- [59] "Internet Game Timeline: 1963 – 2004" (<http://internetgames.about.com/od/gamingnews/a/timeline.htm>), Dave Spohn, About.com, 2 June 2011
- [60] Carole Hughes, Boston College. "The Relationship Between Internet Use and Loneliness Among College Students" (<https://www2.bc.edu/~hughesc/abstract.html>). Boston College. Retrieved 11 August 2011.
- [61] Patricia M. Thornton, "The New Cybersects: Resistance and Repression in the Reform era." In Elizabeth Perry and Mark Selden, eds., *Chinese Society: Change, Conflict and Resistance* (second edition) (London and New York: Routledge, 2003), pp. 149–50.

- [62] "Net abuse hits small city firms" (<http://www.scotsman.com/news/net-abuse-hits-small-city-firms-1-892163>). *The Scotsman* (Edinburgh). 11 September 2003. . Retrieved 7 August 2009.
- [63] *The Shallows: What the Internet Is Doing to Our Brains* ([http://www.theshallowsbook.com/nicholascarr/Nicholas\\_Carrs\\_The\\_Shallows.html](http://www.theshallowsbook.com/nicholascarr/Nicholas_Carrs_The_Shallows.html)), Nicholas Carr, W. W. Norton, 7 June 2010, 276 pp., ISBN 0-393-07222-3, ISBN 978-0-393-07222-8
- [64] "The Arab Uprising's Cascading Effects" (<http://www.miller-mccune.com/politics/the-cascading-effects-of-the-arab-spring-28575/>). Miller-mccune.com. 23 February 2011. . Retrieved 27 February 2011.
- [65] The Role of the Internet in Democratic Transition: Case Study of the Arab Spring ([http://www.etd.ceu.hu/2011/chokoshvili\\_davit.pdf](http://www.etd.ceu.hu/2011/chokoshvili_davit.pdf)), Davit Chokoshvili, Master's Thesis, June 2011
- [66] Kirkpatrick, David D. (9 February 2011). "Wired and Shrewd, Young Egyptians Guide Revolt" (<http://www.nytimes.com/2011/02/10/world/middleeast/10youth.html>). *The New York Times*. .
- [67] Berdal, S.R.B. (2004) (PDF), *Public deliberation on the Web: A Habermasian inquiry into online discourse* (<http://www.duo.uio.no/publ/informatikk/2004/20535/SimonBerdal.pdf>), Oslo: University of Oslo,
- [68] *Kiva Is Not Quite What It Seems* ([http://blogs.cgdev.org/open\\_book/2009/10/kiva-is-not-quite-what-it-seems.php](http://blogs.cgdev.org/open_book/2009/10/kiva-is-not-quite-what-it-seems.php)), by David Roodman, Center for Global Development, 2 October 2009, as accessed 2 & 16 January 2010
- [69] *Confusion on Where Money Lent via Kiva Goes* ([http://www.nytimes.com/2009/11/09/business/global/09kiva.html?\\_r=1&scp=1&sq=Kiva&st=cse](http://www.nytimes.com/2009/11/09/business/global/09kiva.html?_r=1&scp=1&sq=Kiva&st=cse)), by Stephanie Strom, in The New York Times, 8 November 2009, as accessed 2 & 16 January 2010
- [70] "Zidisha Set to "Expand" in Peer-to-Peer Microfinance", Microfinance Focus, Feb 2010 (<http://www.microfinancefocus.com/news/2010/02/07/zidisha-set-to-expand-in-peer-to-peer-microfinance-julia-kurnia/>)
- [71] *Access Controlled: The Shaping of Power, Rights, and Rule in Cyberspace* (<http://mitpress.mit.edu/catalog/item/default.asp?tttype=2&tid=12187>), Ronald J. Deibert, John G. Palfrey, Rafal Rohozinski, and Jonathan Zittrain (eds), MIT Press, April 2010, ISBN 0-262-51435-4, ISBN 978-0-262-51435-4
- [72] "Finland censors anti-censorship site" ([http://www.theregister.co.uk/2008/02/18/finnish\\_policy\\_censor\\_activist/](http://www.theregister.co.uk/2008/02/18/finnish_policy_censor_activist/)). *The Register*. 18 February 2008. . Retrieved 19 February 2008.

## External links

### Organizations

- The Internet Society (<http://www.isoc.org/>)
- Berkman Center for Internet and Society (<http://cyber.law.harvard.edu/>)
- European Commission Information Society ([http://ec.europa.eu/information\\_society/index\\_en.htm](http://ec.europa.eu/information_society/index_en.htm))
- Living Internet (<http://www.livinginternet.com/>), Internet history and related information, including information from many creators of the Internet

### Articles, books, and journals

- First Monday* (<http://www.firstmonday.org/>), a peer-reviewed journal on the Internet established in 1996 as a Great Cities Initiative of the University Library of the University of Illinois at Chicago, ISSN: 1396-0466
- Rise of the Network Society* (<http://www.wiley.com/WileyCDA/WileyTitle/productCd-1405196866.html>), Manuel Castells, Wiley-Blackwell, 1996 (1st ed) and 2009 (2nd ed), ISBN 978-1-4051-9686-4
- "The Internet: Changing the Way We Communicate" (<http://www.nsf.gov/about/history/nsf0050/internet/internet.htm>) in *America's Investment in the Future* (<http://www.nsf.gov/about/history/nsf0050/index.jsp>), National Science Foundation, Arlington, Va. USA, 2000
- "Lessons from the History of the Internet" (<http://www.oup.com/us/catalog/general/subject/Sociology/EnvironmentTechnology/?view=usa&ci=9780199255771>), Manuel Castells, in *The Internet Galaxy*, Ch. 1, pp 9–35, Oxford University Press, 2001, ISBN 978-0-19-925577-1 ISBN10: 0-19-925577-6
- "Media Freedom Internet Cookbook" (<http://www.osce.org/fom/13836>) by the OSCE Representative on Freedom of the Media Vienna, 2004
- The Internet Explained* (<http://www.southbourne.com/articles/internet-explained>), Vincent Zegna & Mike Pepper, Sonet Digital, November 2005, Pages 1 – 7.
- "How Much Does The Internet Weigh?" (<http://discovermagazine.com/2007/jun/how-much-does-the-internet-weigh>), by Stephen Cass, *Discover*, 2007

- "The Internet spreads its tentacles" ([http://findarticles.com/p/articles/mi\\_m1200/is\\_25\\_171/ai\\_n19448170/](http://findarticles.com/p/articles/mi_m1200/is_25_171/ai_n19448170/)), Julie Rehmeyer, *Science News*, Vol. 171, No. 25, pp. 387–388, 23 June 2007
  - *Internet* (<http://www.routledge.com/books/details/9780415352277/>), Lorenzo Cantoni & Stefano Tardini, Routledge, 2006, ISBN 978-0-203-69888-4
-

# What the Internet is

## Internet protocol suite

The **Internet protocol suite** is the set of communications protocols used for the Internet and similar networks, and generally the most popular protocol stack for wide area networks. It is commonly known as **TCP/IP**, because of its most important protocols: Transmission Control Protocol (TCP) and Internet Protocol (IP), which were the first networking protocols defined in this standard. It is occasionally known as the **DoD model** due to the foundational influence of the ARPANET in the 1970s (operated by DARPA, an agency of the United States Department of Defense).

TCP/IP provides end-to-end connectivity specifying how data should be formatted, addressed, transmitted, routed and received at the destination. It has four abstraction layers, each with its own protocols.<sup>[1][2]</sup> From lowest to highest, the layers are:

1. The link layer (commonly Ethernet) contains communication technologies for a local network.
2. The internet layer (IP) connects local networks, thus establishing internetworking.
3. The transport layer (TCP) handles host-to-host communication.
4. The application layer (for example HTTP) contains all protocols for specific data communications services on a process-to-process level (for example how a web browser communicates with a web server).

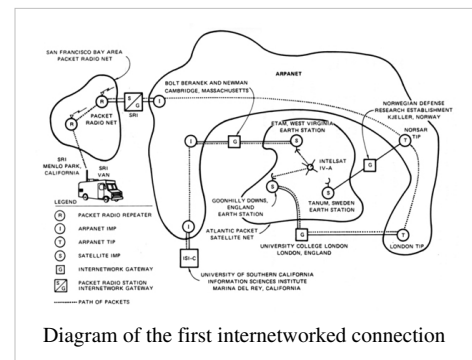
The TCP/IP model and related protocols are maintained by the (IETF) or Internet Engineering Task Force.

## History

### Early research

The Internet protocol suite resulted from research and development conducted by the Defense Advanced Research Projects Agency (DARPA) in the early 1970s. After initiating the pioneering ARPANET in 1969, DARPA started work on a number of other data transmission technologies. In 1972, Robert E. Kahn joined the DARPA Information Processing Technology Office, where he worked on both satellite packet networks and ground-based radio packet networks, and recognized the value of being able to communicate across both. In the spring of 1973, Vinton Cerf, the developer of the existing ARPANET Network Control Program (NCP) protocol, joined Kahn to work on open-architecture interconnection models with the goal of designing the next protocol generation for the ARPANET.

By the summer of 1973, Kahn and Cerf had worked out a fundamental reformulation, where the differences between network protocols were hidden by using a common internetwork protocol, and, instead of the network being responsible for reliability, as in the ARPANET, the hosts became responsible. Cerf credits Hubert Zimmerman and Louis Pouzin, designer of the CYCLADES network, with important influences on this design.



The network's design included the recognition it should provide only the functions of efficiently transmitting and routing traffic between end nodes and that all other intelligence should be located at the edge of the network, in the end nodes. Using a simple design, it became possible to connect almost any network to the ARPANET, irrespective of their local characteristics, thereby solving Kahn's initial problem. One popular expression is that TCP/IP, the eventual product of Cerf and Kahn's work, will run over "*two tin cans and a string*." As a joke, the IP over Avian Carriers formal protocol specification was created and successfully tested.



A Stanford Research Institute packet radio van, site of the first three-way internetworked transmission.

A computer called a router is provided with an interface to each network. It forwards packets back and forth between them.<sup>[3]</sup> Originally a router was called *gateway*, but the term was changed to avoid confusion with other types of gateways.

## Specification

From 1973 to 1974, Cerf's networking research group at Stanford worked out details of the idea, resulting in the first TCP specification.<sup>[4]</sup> A significant technical influence was the early networking work at Xerox PARC, which produced the PARC Universal Packet protocol suite, much of which existed around that time.

DARPA then contracted with BBN Technologies, Stanford University, and the University College London to develop operational versions of the protocol on different hardware platforms. Four versions were developed: TCP v1, TCP v2, TCP v3 and IP v3, and TCP/IP v4. The last protocol is still in use today.

In 1975, a two-network TCP/IP communications test was performed between Stanford and University College London (UCL). In November, 1977, a three-network TCP/IP test was conducted between sites in the US, the UK, and Norway. Several other TCP/IP prototypes were developed at multiple research centers between 1978 and 1983. The migration of the ARPANET to TCP/IP was officially completed on flag day January 1, 1983, when the new protocols were permanently activated.<sup>[5]</sup>

## Adoption

In March 1982, the US Department of Defense declared TCP/IP as the standard for all military computer networking.<sup>[6]</sup> In 1985, the Internet Architecture Board held a three-day workshop on TCP/IP for the computer industry, attended by 250 vendor representatives, promoting the protocol and leading to its increasing commercial use.

In 1985 the first Interop conference was held, focusing on network interoperability via further adoption of TCP/IP. It was founded by Dan Lynch, an early Internet activist. From the beginning, it was attended by large corporations, such as IBM and DEC. Interoperability conferences have been held every year since then. Every year from 1985 through 1993, the number of attendees tripled.

IBM, ATT and DEC were the first major corporations to adopt TCP/IP, despite having competing internal protocols (SNA, XNS, etc.). In IBM, from 1984, Barry Appelman's group did TCP/IP development. (Appelman later moved to AOL to be the head of all its development efforts.) They navigated the corporate politics to get a stream of TCP/IP products for various IBM systems, including MVS, VM, and OS/2. At the same time, several smaller companies began offering TCP/IP stacks for DOS and MS Windows, such as the company FTP Software, and the Wollongong Group.<sup>[7]</sup> The first VM/CMS TCP/IP stack came from the University of Wisconsin.<sup>[8]</sup>

Back then, most of these TCP/IP stacks were written single-handedly by a few talented programmers. For example, John Romkey of FTP Software was the author of the MIT PC/IP package.<sup>[9]</sup> John Romkey's PC/IP implementation

was the first IBM PC TCP/IP stack. Jay Elinsky and Oleg Vishnepolsky of IBM Research wrote TCP/IP stacks for VM/CMS and OS/2, respectively.<sup>[10]</sup>

The spread of TCP/IP was fueled further in June 1989, when AT&T agreed to put into the public domain the TCP/IP code developed for UNIX. Various vendors, including IBM, included this code in their own TCP/IP stacks. Many companies sold TCP/IP stacks for Windows until Microsoft released its own TCP/IP stack in Windows 95. This event was a little late in the evolution of the Internet, but it cemented TCP/IP's dominance over other protocols, which eventually disappeared. These protocols included IBM's SNA, OSI, Microsoft's native NetBIOS, and Xerox' XNS.

## Key architectural principles

An early architectural document, RFC 1122, emphasizes architectural principles over layering.<sup>[11]</sup>

- End-to-end principle: This principle has evolved over time. Its original expression put the maintenance of state and overall intelligence at the edges, and assumed the Internet that connected the edges retained no state and concentrated on speed and simplicity. Real-world needs for firewalls, network address translators, web content caches and the like have forced changes in this principle.<sup>[12]</sup>
- Robustness Principle: "In general, an implementation must be conservative in its sending behavior, and liberal in its receiving behavior. That is, it must be careful to send well-formed datagrams, but must accept any datagram that it can interpret (e.g., not object to technical errors where the meaning is still clear)."<sup>[13]</sup> "The second part of the principle is almost as important: software on other hosts may contain deficiencies that make it unwise to exploit legal but obscure protocol features."<sup>[14]</sup>

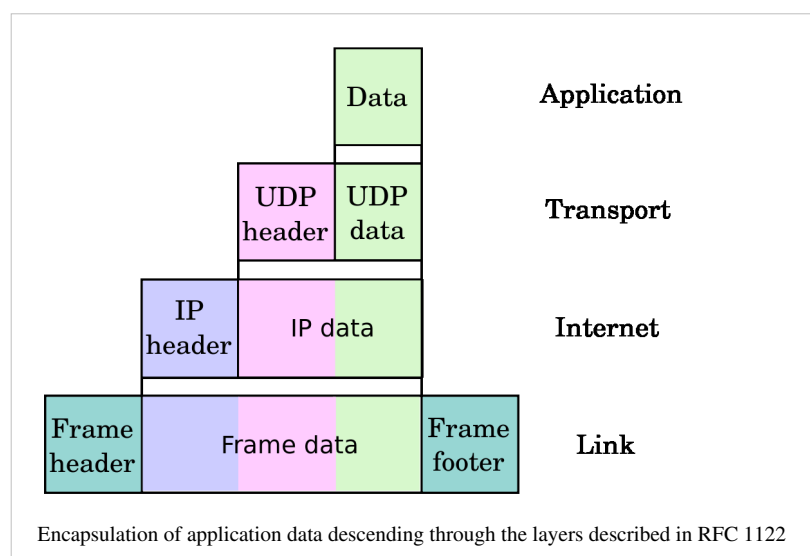
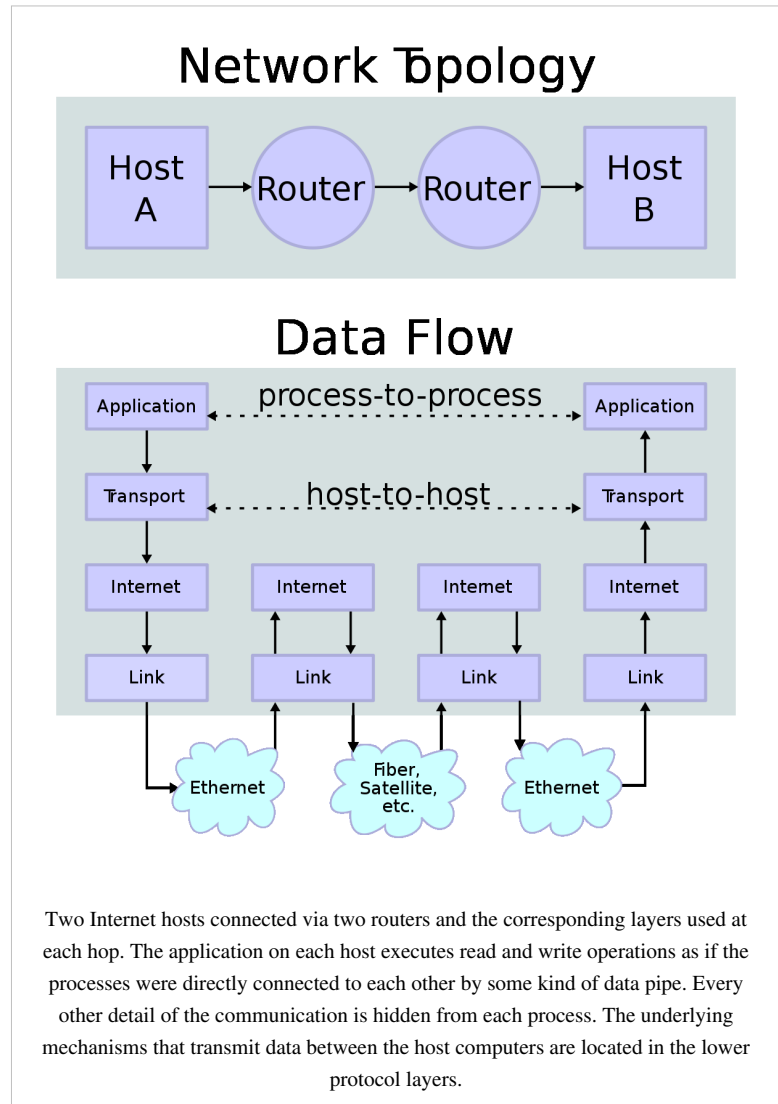


## Layers in the Internet protocol suite

The Internet protocol suite uses encapsulation to provide abstraction of protocols and services. Encapsulation is usually aligned with the division of the protocol suite into layers of general functionality. In general, an application (the highest level of the model) uses a set of protocols to send its data down the layers, being further encapsulated at each level.

The "layers" of the protocol suite near the top are logically closer to the user application, while those near the bottom are logically closer to the physical transmission of the data. Viewing layers as providing or consuming a service is a method of abstraction to isolate upper layer protocols from the nitty-gritty detail of transmitting bits over, for example, Ethernet and collision detection, while the lower layers avoid having to know the details of each and every application and its protocol.

Even when the layers are examined, the assorted architectural documents—there is no single architectural model such as ISO 7498, the Open Systems Interconnection (OSI) model—have fewer and less rigidly defined layers than the OSI model, and thus provide an easier fit for real-world protocols. In point of fact, one frequently referenced document, RFC 1958, does not contain a stack of layers. The lack of emphasis on layering is a strong difference between the IETF and OSI approaches. It only refers to the existence of the "internetworking layer" and generally to "upper layers"; this document was intended as a 1996 "snapshot" of the architecture: "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."

RFC 1122, entitled *Host Requirements*, is structured in paragraphs referring to layers, but the document refers to many other architectural principles not emphasizing layering. It loosely defines a four-layer model, with the layers having names, not numbers, as follows:

- Application layer (process-to-process): This is the scope within which applications create user data and communicate this data to other processes or applications on another or the same host. The communications partners are often called *peers*. This is where the "higher level" protocols such as SMTP, FTP, SSH, HTTP, etc. operate.
- Transport layer (host-to-host): The transport layer constitutes the networking regime between two network hosts, either on the local network or on remote networks separated by routers. The transport layer provides a uniform networking interface that hides the actual topology (layout) of the underlying network connections. This is where flow-control, error-correction, and connection protocols exist, such as TCP. This layer deals with opening and maintaining connections between Internet hosts.
- Internet layer (internetworking): The internet layer has the task of exchanging datagrams across network boundaries. It is therefore also referred to as the layer that establishes internetworking, indeed, it defines and establishes the Internet. This layer defines the addressing and routing structures used for the TCP/IP protocol suite. The primary protocol in this scope is the Internet Protocol, which defines IP addresses. Its function in routing is to transport datagrams to the next IP router that has the connectivity to a network closer to the final data destination.
- Link layer: This layer defines the networking methods within the scope of the local network link on which hosts communicate without intervening routers. This layer describes the protocols used to describe the local network topology and the interfaces needed to affect transmission of Internet layer datagrams to next-neighbor hosts. (cf. the OSI data link layer).

The Internet protocol suite and the layered protocol stack design were in use before the OSI model was established. Since then, the TCP/IP model has been compared with the OSI model in books and classrooms, which often results in confusion because the two models use different assumptions, including about the relative importance of strict layering.

This abstraction also allows upper layers to provide services that the lower layers cannot, or choose not, to provide. Again, the original OSI model was extended to include connectionless services (OSIRM CL).<sup>[15]</sup> For example, IP is not designed to be reliable and is a best effort delivery protocol. This means that all transport layer implementations must choose whether or not to provide reliability and to what degree. UDP provides data integrity (via a checksum) but does not guarantee delivery; TCP provides both data integrity and delivery guarantee (by retransmitting until the receiver acknowledges the reception of the packet).

This model lacks the formalism of the OSI model and associated documents, but the IETF does not use a formal model and does not consider this a limitation, as in the comment by David D. Clark, "We reject: kings, presidents and voting. We believe in: rough consensus and running code." Criticisms of this model, which have been made with respect to the OSI model, often do not consider ISO's later extensions to that model.

1. For multiaccess links with their own addressing systems (e.g. Ethernet) an address mapping protocol is needed. Such protocols can be considered to be below IP but above the existing link system. While the IETF does not use the terminology, this is a subnetwork dependent convergence facility according to an extension to the OSI model, the internal organization of the network layer (IONL).<sup>[16]</sup>
2. ICMP & IGMP operate on top of IP but do not transport data like UDP or TCP. Again, this functionality exists as layer management extensions to the OSI model, in its *Management Framework* (OSIRM MF)<sup>[17]</sup>

3. The SSL/TLS library operates above the transport layer (uses TCP) but below application protocols. Again, there was no intention, on the part of the designers of these protocols, to comply with OSI architecture.
4. The link is treated like a black box here. This is fine for discussing IP (since the whole point of IP is it will run over virtually anything). The IETF explicitly does not intend to discuss transmission systems, which is a less academic but practical alternative to the OSI model.

The following is a description of each layer in the TCP/IP networking model starting from the lowest level.

### Link layer

The link layer is the networking scope of the local network connection to which a host is attached. This regime is called the *link* in Internet literature. This is the lowest component layer of the Internet protocols, as TCP/IP is designed to be hardware independent. As a result TCP/IP is able to be implemented on top of virtually any hardware networking technology.

The link layer is used to move packets between the Internet layer interfaces of two different hosts on the same link. The processes of transmitting and receiving packets on a given link can be controlled both in the software device driver for the network card, as well as on firmware or specialized chipsets. These will perform data link functions such as adding a packet header to prepare it for transmission, then actually transmit the frame over a physical medium. The TCP/IP model includes specifications of translating the network addressing methods used in the Internet Protocol to data link addressing, such as Media Access Control (MAC), however all other aspects below that level are implicitly assumed to exist in the link layer, but are not explicitly defined.

This is also the layer where packets may be selected to be sent over a virtual private network or other networking tunnel. In this scenario, the link layer data may be considered application data which traverses another instantiation of the IP stack for transmission or reception over another IP connection. Such a connection, or virtual link, may be established with a transport protocol or even an application scope protocol that serves as a tunnel in the link layer of the protocol stack. Thus, the TCP/IP model does not dictate a strict hierarchical encapsulation sequence.

### Internet layer

The internet layer has the responsibility of sending packets across potentially multiple networks. Internetworking requires sending data from the source network to the destination network. This process is called routing.<sup>[18]</sup>

In the Internet protocol suite, the Internet Protocol performs two basic functions:

- *Host addressing and identification*: This is accomplished with a hierarchical addressing system (see IP address).
- *Packet routing*: This is the basic task of sending packets of data (datagrams) from source to destination by sending them to the next network node (router) closer to the final destination.

The internet layer is not only agnostic of application data structures as the transport layer, but it also does not distinguish between operation of the various transport layer protocols. So, IP can carry data for a variety of different upper layer protocols. These protocols are each identified by a unique protocol number: for example, Internet Control Message Protocol (ICMP) and Internet Group Management Protocol (IGMP) are protocols 1 and 2, respectively.

Some of the protocols carried by IP, such as ICMP (used to transmit diagnostic information about IP transmission) and IGMP (used to manage IP Multicast data) are layered on top of IP but perform internetworking functions. This illustrates the differences in the architecture of the TCP/IP stack of the Internet and the OSI model.

The internet layer only provides an unreliable datagram transmission facility between hosts located on potentially different IP networks by forwarding the transport layer datagrams to an appropriate next-hop router for further relaying to its destination. With this functionality, the internet layer makes possible internetworking, the interworking of different IP networks, and it essentially establishes the Internet. The Internet Protocol is the principal component of the internet layer, and it defines two addressing systems to identify network hosts computers, and to

locate them on the network. The original address system of the ARPANET and its successor, the Internet, is Internet Protocol version 4 (IPv4). It uses a 32-bit IP address and is therefore capable of identifying approximately four billion hosts. This limitation was eliminated by the standardization of Internet Protocol version 6 (IPv6) in 1998, and beginning production implementations in approximately 2006.

## Transport layer

The transport layer establishes host-to-host connectivity, meaning it handles the details of data transmission that are independent of the structure of user data and the logistics of exchanging information for any particular specific purpose. Its responsibility includes end-to-end message transfer independent of the underlying network, along with error control, segmentation, flow control, congestion control, and application addressing (port numbers). End to end message transmission or connecting applications at the transport layer can be categorized as either connection-oriented, implemented in TCP, or connectionless, implemented in UDP.

The transport layer can be thought of as a transport mechanism, e.g., a vehicle with the responsibility to make sure that its contents (passengers/goods) reach their destination safely and soundly, unless another protocol layer is responsible for safe delivery. The layer simply establishes a basic data channel that an application uses in its task-specific data exchange.

For this purpose the layer establishes the concept of the port, a numbered logical construct allocated specifically for each of the communication channels an application needs. For many types of services, these port numbers have been standardized so that client computers may address specific services of a server computer without the involvement of service announcements or directory services.

Since IP provides only a best effort delivery, the transport layer is the first layer of the TCP/IP stack to offer reliability. IP can run over a reliable data link protocol such as the High-Level Data Link Control (HDLC).

For example, the TCP is a connection-oriented protocol that addresses numerous reliability issues to provide a reliable byte stream:

- data arrives in-order
- data has minimal error (i.e. correctness)
- duplicate data is discarded
- lost/discarded packets are resent
- includes traffic congestion control

The newer Stream Control Transmission Protocol (SCTP) is also a reliable, connection-oriented transport mechanism. It is message-stream-oriented — not byte-stream-oriented like TCP — and provides multiple streams multiplexed over a single connection. It also provides multi-homing support, in which a connection end can be represented by multiple IP addresses (representing multiple physical interfaces), such that if one fails, the connection is not interrupted. It was developed initially for telephony applications (to transport SS7 over IP), but can also be used for other applications.

User Datagram Protocol is a connectionless datagram protocol. Like IP, it is a best effort, "unreliable" protocol. Reliability is addressed through error detection using a weak checksum algorithm. UDP is typically used for applications such as streaming media (audio, video, Voice over IP etc.) where on-time arrival is more important than reliability, or for simple query/response applications like DNS lookups, where the overhead of setting up a reliable connection is disproportionately large. Real-time Transport Protocol (RTP) is a datagram protocol that is designed for real-time data such as streaming audio and video.

The applications at any given network address are distinguished by their TCP or UDP port. By convention certain *well known ports* are associated with specific applications. (*See List of TCP and UDP port numbers.*)

## Application layer

The application layer contains the higher-level protocols used by most applications for network communication. Examples of application layer protocols include the File Transfer Protocol (FTP) and the Simple Mail Transfer Protocol (SMTP).<sup>[19]</sup> Data coded according to application layer protocols are then encapsulated into one or (occasionally) more transport layer protocols (such as TCP or UDP), which in turn use lower layer protocols to effect actual data transfer.

Since the IP stack defines no layers between the application and transport layers, the application layer must include any protocols that act like the OSI's presentation and session layer protocols. This is usually done through libraries.

Application layer protocols generally treat the transport layer (and lower) protocols as black boxes which provide a stable network connection across which to communicate, although the applications are usually aware of key qualities of the transport layer connection such as the end point IP addresses and port numbers. As noted above, layers are not necessarily clearly defined in the Internet protocol suite. Application layer protocols are most often associated with client–server applications, and the commoner servers have specific ports assigned to them by the IANA: HTTP has port 80; Telnet has port 23; etc. Clients, on the other hand, tend to use ephemeral ports, i.e. port numbers assigned at random from a range set aside for the purpose.

Transport and lower level layers are largely unconcerned with the specifics of application layer protocols. Routers and switches do not typically "look inside" the encapsulated traffic to see what kind of application protocol it represents, rather they just provide a conduit for it. However, some firewall and bandwidth throttling applications do try to determine what's inside, as with the Resource Reservation Protocol (RSVP). It's also sometimes necessary for Network Address Translation (NAT) facilities to take account of the needs of particular application layer protocols. (NAT allows hosts on private networks to communicate with the outside world via a single visible IP address using port forwarding, and is an almost ubiquitous feature of modern domestic broadband routers).

## Layer names and number of layers in the literature

The following table shows various networking models. The number of layers varies between three and seven.

Kurose, <sup>[20]</sup> Forouzan <sup>[21]</sup>	Comer, <sup>[22]</sup> Kozierok <sup>[23]</sup>	Stallings <sup>[24]</sup>	Tanenbaum <sup>[25]</sup>	RFC 1122, Internet STD 3 (1989)	Cisco Academy <sup>[26]</sup>	Mike Padlipsky's 1982 "Arpanet Reference Model" (RFC 871)	OSI model
<i>Five layers</i>	<i>Four+one layers</i>	<i>Five layers</i>	<i>Five layers</i>	<i>Four layers</i>	<i>Four layers</i>	<i>Three layers</i>	<i>Seven layers</i>
"Five-layer Internet model" or "TCP/IP protocol suite"	"TCP/IP 5-layer reference model"	"TCP/IP model"	"TCP/IP 5-layer reference model"	"Internet model"	"Internet model"	"Arpanet reference model"	OSI model
Application	Application	Application	Application	Application	Application	Application/Process	Application Presentation Session
Transport	Transport	Host-to-host or transport	Transport	Transport	Transport	Host-to-host	Transport
Network	Internet	Internet	Internet	Internet	Internetwork		Network
Data link	Data link (Network interface)	Network access	Data link	Link	Network interface	Network interface	Data link
Physical	(Hardware)	Physical	Physical				Physical

Some of the networking models are from textbooks, which are secondary sources that may contravene the intent of RFC 1122 and other IETF primary sources.<sup>[27]</sup>

## OSI and TCP/IP layering differences

The three top layers in the OSI model—the application layer, the presentation layer and the session layer—are not distinguished separately in the TCP/IP model where it is just the application layer. While some pure OSI protocol applications, such as X.400, also combined them, there is no requirement that a TCP/IP protocol stack must impose monolithic architecture above the transport layer. For example, the NFS application protocol runs over the eXternal Data Representation (XDR) presentation protocol, which, in turn, runs over a protocol called Remote Procedure Call (RPC). RPC provides reliable record transmission, so it can run safely over the best-effort UDP transport.

Different authors have interpreted the RFCs differently, about whether the link layer (and the TCP/IP model) covers OSI model layer 1 (physical layer) issues, or if a hardware layer is assumed below the link layer.

Several authors have attempted to incorporate the OSI model's layers 1 and 2 into the TCP/IP model, since these are commonly referred to in modern standards (for example, by IEEE and ITU). This often results in a model with five layers, where the link layer or network access layer is split into the OSI model's layers 1 and 2.

The session layer roughly corresponds to the Telnet virtual terminal functionality, which is part of text based protocols such as the HTTP and SMTP TCP/IP model application layer protocols. It also corresponds to TCP and UDP port numbering, which is considered as part of the transport layer in the TCP/IP model. Some functions that would have been performed by an OSI presentation layer are realized at the Internet application layer using the MIME standard, which is used in application layer protocols such as HTTP and SMTP.

The IETF protocol development effort is not concerned with strict layering. Some of its protocols may not fit cleanly into the OSI model, although RFCs sometimes refer to it and often use the old OSI layer numbers. The IETF has repeatedly stated that Internet protocol and architecture development is not intended to be OSI-compliant. RFC 3439, addressing Internet architecture, contains a section entitled: "Layering Considered Harmful".<sup>[27]</sup>

Conflicts are apparent also in the original OSI model, ISO 7498, when not considering the annexes to this model (e.g., ISO 7498/4 Management Framework), or the ISO 8648 Internal Organization of the Network layer (IONL). When the IONL and Management Framework documents are considered, the ICMP and IGMP are neatly defined as layer management protocols for the network layer. In like manner, the IONL provides a structure for "subnetwork dependent convergence facilities" such as ARP and RARP.

IETF protocols can be encapsulated recursively, as demonstrated by tunneling protocols such as Generic Routing Encapsulation (GRE). GRE uses the same mechanism that OSI uses for tunneling at the network layer.

## Implementations

No specific hardware or software implementation is required by the protocols or the layered model, so there are many. Most computer operating systems in use today, including all consumer-targeted systems, include a TCP/IP implementation.

A minimally acceptable implementation includes the following protocols, listed from most essential to least essential: IP, ARP, ICMP, UDP, TCP and sometimes IGMP. In principle, it is possible to support only one transport protocol, such as UDP, but this is rarely done, because it limits usage of the whole implementation. IPv6, beyond its own version of ARP (NDP), ICMP (ICMPv6) and IGMP (IGMPv6), has some additional required functions, and often is accompanied by an integrated IPsec security layer. Other protocols could be easily added later (possibly being implemented entirely in userspace), such as DNS for resolving domain names to IP addresses, or DHCP for automatically configuring network interfaces.

Normally, application programmers are concerned only with interfaces in the application layer and often also in the transport layer, while the layers below are services provided by the TCP/IP stack in the operating system. Most IP

implementations are accessible to programmers through sockets and APIs.

Unique implementations include Lightweight TCP/IP, an open source stack designed for embedded systems, and KA9Q NOS, a stack and associated protocols for amateur packet radio systems and personal computers connected via serial lines.

Microcontroller firmware in the network adapter typically handles link issues, supported by driver software in the operational system. Non-programmable analog and digital electronics are normally in charge of the physical components below the link layer, typically using an application-specific integrated circuit (ASIC) chipset for each network interface or other physical standard. High-performance routers are to a large extent based on fast non-programmable digital electronics, carrying out link level switching.

## References

- [1] RFC 1122, *Requirements for Internet Hosts – Communication Layers*, R. Braden (ed.), October 1989
- [2] RFC 1123, *Requirements for Internet Hosts – Application and Support*, R. Braden (ed.), October 1989
- [3] RFC 1812, *Requirements for IP Version 4 Routers*, F. Baker (June 1995)
- [4] RFC 675, *Specification of Internet Transmission Control Protocol*, V. Cerf et al. (December 1974)
- [5] Internet History (<http://www.livinginternet.com/i/ii.htm>)
- [6] Ronda Hauben. "From the ARPANET to the Internet" ([http://www.columbia.edu/~rh120/other/tcpdigest\\_paper.txt](http://www.columbia.edu/~rh120/other/tcpdigest_paper.txt)). TCP Digest (UUCP). . Retrieved 2007-07-05.
- [7] Wollongong (<http://support.microsoft.com/kb/108007>)
- [8] A Short History of Internet Protocols at CERN (<http://www.weblab.isti.cnr.it/education/ssfs/lezioni/slides/archives/cern.htm>)
- [9] About I "romkey" (<http://www.romkey.com/about/>)
- [10] Barry Appelman
- [11] Architectural Principles of the Internet (<ftp://ftp.rfc-editor.org/in-notes/rfc1958.txt>), RFC 1958, B. Carpenter, June 1996
- [12] Rethinking the design of the Internet: The end to end arguments vs. the brave new world ([http://www.csd.uoc.gr/~hy558/papers/Rethinking\\_2001.pdf](http://www.csd.uoc.gr/~hy558/papers/Rethinking_2001.pdf)), Marjory S. Blumenthal, David D. Clark, August 2001
- [13] p.23 INTERNET PROTOCOL DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION September 1981 Jon Postel Editor (<http://www.ietf.org/rfc/rfc0791.txt?number=791>)
- [14] Requirements for Internet Hosts -- Communication Layers p.13 October 1989 R. Braden, Editor (<http://tools.ietf.org/html/rfc1122#page-12>)
- [15] [ OSI: Reference Model Addendum 1: Connectionless-mode Transmission,ISO7498/AD1],ISO7498/AD1, May 1986
- [16] "Information processing systems -- Open Systems Interconnection -- Internal organization of the Network Layer" ([http://www.iso.org/iso/home/store/catalogue\\_tc/catalogue\\_detail.htm?csnumber=16011](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=16011)), ISO 8648:1988.
- [17] "Information processing systems -- Open Systems Interconnection -- Basic Reference Model -- Part 4: Management framework" ([http://www.iso.org/iso/home/store/catalogue\\_tc/catalogue\\_detail.htm?csnumber=14258](http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=14258)), ISO 7498-4:1989.
- [18] IP Packet Structure (<http://www.comsci.us/datacom/ippacket.html>)
- [19] *TCP/IP Illustrated: the protocols* (<http://www.kohala.com/start/tcpipv1.html>), ISBN 0-201-63346-9, W. Richard Stevens, February 1994
- [20] James F. Kurose, Keith W. Ross, Computer Networking: A Top-Down Approach, 2008, ISBN 0-321-49770-8 ([http://www.pearsonhighered.com/educator/academic/product/0,,0321497708,00+en-USS\\_01DBC.html](http://www.pearsonhighered.com/educator/academic/product/0,,0321497708,00+en-USS_01DBC.html))
- [21] Behrouz A. Forouzan, Data Communications and Networking, 2003 ([http://books.google.com/books?id=U3Gcf65Pu9IC&printsec=frontcover&dq=forouzan+\"computer+networks\"&ei=RPZ9SOCvMofctAO02di0AQ&hl=en&sig=ACfU3U2Hh\\_n83pPtf5uCreCih0HnWvNcxg#PPA29,M1](http://books.google.com/books?id=U3Gcf65Pu9IC&printsec=frontcover&dq=forouzan+\))
- [22] Douglas E. Comer, Internetworking with TCP/IP: Principles, Protocols and Architecture, Pearson Prentice Hall 2005, ISBN 0-13-187671-6 ([http://books.google.com/books?id=jonyuTASbWAC&pg=PA155&hl=sv&source=gbs\\_toc\\_r&cad=0\\_0&sig=ACfU3U18gHAia1pU\\_Pxn-rhkCnH1v70M6Q#PPA161,M1](http://books.google.com/books?id=jonyuTASbWAC&pg=PA155&hl=sv&source=gbs_toc_r&cad=0_0&sig=ACfU3U18gHAia1pU_Pxn-rhkCnH1v70M6Q#PPA161,M1))
- [23] Charles M. Kozierok, "The TCP/IP Guide", No Starch Press 2005 ([http://books.google.com/books?id=Pm4RgYV2w4YC&pg=PA131&dq=TCP/IP+model+layers\"&lr=&hl=sv&sig=ACfU3U3ofMwYAbZfGz1BmAXc2oNNFC2b8A#PPA129,M1](http://books.google.com/books?id=Pm4RgYV2w4YC&pg=PA131&dq=TCP/IP+model+layers\))
- [24] William Stallings, Data and Computer Communications, Prentice Hall 2006, ISBN 0-13-243310-9 ([http://books.google.com/books?id=c\\_AWmhkovR0C&pg=PA35&dq=internet+layer+\"network+access+layer\"&ei=O99SI3EJo32sgOQpPTThDw&hl=en&sig=ACfU3U38aXznzeAnQdbLcPFxfCgxAd4IFg](http://books.google.com/books?id=c_AWmhkovR0C&pg=PA35&dq=internet+layer+\))
- [25] Andrew S. Tanenbaum, Computer Networks, Prentice Hall 2002, ISBN 0-13-066102-3 ([http://books.google.com/books?id=Pd-z64SJRBA&pg=PA42&vq=internet+layer&dq=networks&hl=sv&source=gbs\\_search\\_s&sig=ACfU3U3DHANelz0sOsd5NK4VXSrgNFYVAw#PPA42,M1](http://books.google.com/books?id=Pd-z64SJRBA&pg=PA42&vq=internet+layer&dq=networks&hl=sv&source=gbs_search_s&sig=ACfU3U3DHANelz0sOsd5NK4VXSrgNFYVAw#PPA42,M1))
- [26] Mark Dye, Mark A. Dye, Wendell, Network Fundamentals: CCNA Exploration Companion Guide, 2007, ISBN 1-58713-208-7

- [27] R. Bush; D. Meyer (December 2002), *Some Internet Architectural Guidelines and Philosophy* (<http://www.ietf.org/rfc/rfc3439.txt>), Internet Engineering Task Force,

## Further reading

- Douglas E. Comer. *Internetworking with TCP/IP - Principles, Protocols and Architecture*. ISBN 86-7991-142-9
- Joseph G. Davies and Thomas F. Lee. *Microsoft Windows Server 2003 TCP/IP Protocols and Services*. ISBN 0-7356-1291-9
- Forouzan, Behrouz A. (2003). *TCP/IP Protocol Suite* (2nd ed.). McGraw-Hill. ISBN 0-07-246060-1.
- Craig Hunt *TCP/IP Network Administration*. O'Reilly (1998) ISBN 1-56592-322-7
- Maufer, Thomas A. (1999). *IP Fundamentals*. Prentice Hall. ISBN 0-13-975483-0.
- Ian McLean. *Windows(R) 2000 TCP/IP Black Book*. ISBN 1-57610-687-X
- Ajit Mungale *Pro .NET 1.1 Network Programming*. ISBN 1-59059-345-6
- W. Richard Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. ISBN 0-201-63346-9
- W. Richard Stevens and Gary R. Wright. *TCP/IP Illustrated, Volume 2: The Implementation*. ISBN 0-201-63354-X
- W. Richard Stevens. *TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols*. ISBN 0-201-63495-3
- Andrew S. Tanenbaum. *Computer Networks*. ISBN 0-13-066102-3
- Clark, D. (1988). "The Design Philosophy of the DARPA Internet Protocols" (<http://www.cs.princeton.edu/~jrex/teaching/spring2005/reading/clark88.pdf>). *SIGCOMM '88 Symposium proceedings on Communications architectures and protocols* (ACM): 106–114. doi:10.1145/52324.52336. Retrieved 2011-10-16.

## External links

- Internet History (<http://www.livinginternet.com/i/ii.htm>)—Pages on Robert Kahn, Vinton Cerf, and TCP/IP (reviewed by Cerf and Kahn).
- RFC 675 (<http://www.ietf.org/rfc/rfc0675.txt>) - Specification of Internet Transmission Control Program, December 1974 Version
- TCP/IP State Transition Diagram ([http://www.night-ray.com/TCPIP\\_State\\_Transition\\_Diagram.pdf](http://www.night-ray.com/TCPIP_State_Transition_Diagram.pdf)) (PDF)
- RFC 1180 A TCP/IP Tutorial - from the Internet Engineering Task Force (January 1991)
- TCP/IP FAQ (<http://www.itprc.com/tcpipfaq/>)
- The TCP/IP Guide (<http://www.tcpipguide.com/free/>) - A comprehensive look at the protocols and the procedures/processes involved
- A Study of the ARPANET TCP/IP Digest ([http://www.columbia.edu/~rh120/other/tcpdigest\\_paper.txt](http://www.columbia.edu/~rh120/other/tcpdigest_paper.txt))
- TCP/IP Sequence Diagrams (<http://www.eventhelix.com/RealtimeMantra/Networking/>)
- The Internet in Practice (<http://www.searchandgo.com/articles/internet/internet-practice-4.php>)
- TCP/IP - Directory & Informational Resource (<http://softtechinfo.com/network/tcpip.html>)
- Daryl's TCP/IP Primer (<http://www.ipprimer.com/>) - Intro to TCP/IP LAN administration, conversational style
- Introduction to TCP/IP (<http://www.linux-tutorial.info/MContent-142>)
- TCP/IP commands from command prompt (<http://blog.webgk.com/2007/10/dns-tcpip-commands-from-command-prompt.html>)
- cIPS (<http://sourceforge.net/projects/cipsuite/>) — Robust TCP/IP stack for embedded devices without an Operating System



# OSI model

The **Open Systems Interconnection (OSI) model** (ISO/IEC 7498-1) is a product of the Open Systems Interconnection effort at the International Organization for Standardization. It is a prescription of characterizing and standardizing the functions of a communications system in terms of abstraction layers. Similar communication functions are grouped into logical layers. A layer serves the layer above it and is served by the layer below it.

For example, a layer that provides error-free communications across a network provides the path needed by applications above it, while it calls the next lower layer to send and receive packets that make up the contents of that path. Two instances at one layer are connected by a horizontal connection on that layer.

## History

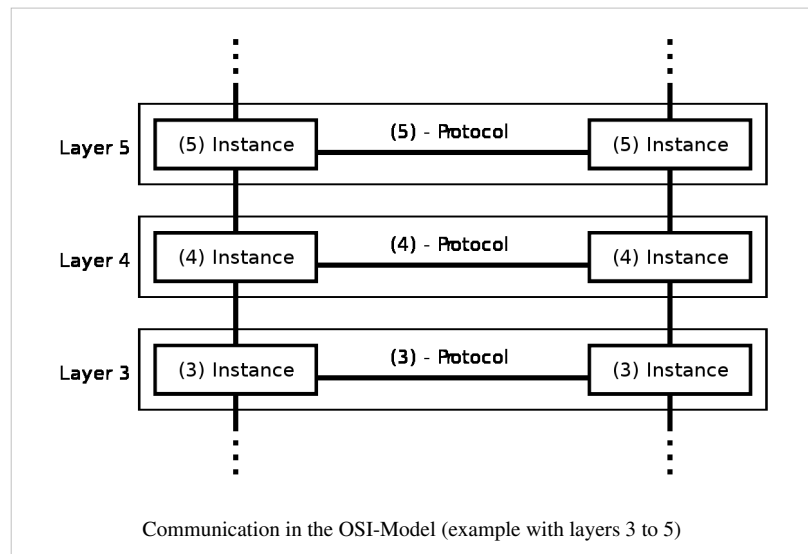
Work on a layered model of network architecture was started and the International Organization for Standardization (ISO) began to develop its OSI framework architecture. OSI had two major components: an *abstract model* of networking, called the Basic Reference Model or seven-layer model, and a set of specific protocols.

The concept of a seven-layer model was provided by the work of Charles Bachman, Honeywell Information

Services. Various aspects of OSI design evolved from experiences with the ARPANET, the fledgling Internet, NPLNET, EIN, CYCLADES network and the work in IFIP WG6.1. The new design was documented in ISO 7498 and its various addenda. In this model, a networking system was divided into layers. Within each layer, one or more entities implement its functionality. Each entity interacted directly only with the layer immediately beneath it, and provided facilities for use by the layer above it.

Protocols enabled an entity in one host to interact with a corresponding entity at the same layer in another host. Service definitions abstractly described the functionality provided to an (N)-layer by an (N-1) layer, where N was one of the seven layers of protocols operating in the local host.

The OSI standards documents are available from the ITU-T as the X.200-series of recommendations.<sup>[1]</sup> Some of the protocol specifications were also available as part of the ITU-T X series. The equivalent ISO and ISO/IEC standards for the OSI model were available from ISO, but only some of them without fees.<sup>[2]</sup>



## Description of OSI layers

According to recommendation X.200, there are seven layers, labeled 1 to 7, with layer 1 at the bottom. Each layer is generically known as an N layer. An "N+1 entity" (at layer N+1) requests services from an "N entity" (at layer N).

At each level, two entities (N-entity peers) interact by means of the N protocol by transmitting protocol data units (PDU).

A Service Data Unit (SDU) is a specific unit of data that has been passed down from an OSI layer to a lower layer, and which the lower layer has not yet encapsulated into a protocol data unit (PDU). An SDU is a set of data that is sent by a user of the services of a given layer, and is transmitted semantically unchanged to a peer service user.

The PDU at a layer N is the SDU of layer N-1. In effect the SDU is the 'payload' of a given PDU. That is, the process of changing an SDU to a PDU, consists of an encapsulation process, performed by the lower layer. All the data contained in the SDU becomes encapsulated within the PDU. The layer N-1 adds headers or footers, or both, to the SDU, transforming it into a PDU of layer N. The added headers or footers are part of the process used to make it possible to get data from a source to a destination.

OSI Model			
	Data unit	Layer	Function
Host layers	Data	7. Application	Network process to application
		6. Presentation	Data representation, encryption and decryption, convert machine dependent data to machine independent data
		5. Session	Interhost communication, managing sessions between applications
	Segments	4. Transport	End-to-end connections, reliability and flow control
Media layers	Packet/Datagram	3. Network	Path determination and logical addressing
	Frame	2. Data link	Physical addressing
	Bit	1. Physical	Media, signal and binary transmission

Some orthogonal aspects, such as management and security, involve every layer.

Security services are not related to a specific layer: they can be related by a number of layers, as defined by ITU-T X.800 Recommendation.<sup>[3]</sup>

These services are aimed to improve the CIA triad (confidentiality, integrity, and availability) of transmitted data. Actually the availability of communication service is determined by network design and/or network management protocols. Appropriate choices for these are needed to protect against denial of service.

### Layer 1: physical layer

The physical layer defines electrical and physical specifications for devices. In particular, it defines the relationship between a device and a transmission medium, such as a copper or fiber optical cable. This includes the layout of pins, voltages, line impedance, cable specifications, signal timing, hubs, repeaters, network adapters, host bus adapters (HBA used in storage area networks) and more.

The major functions and services performed by the physical layer are:

- Establishment and termination of a connection to a communications medium.
- Participation in the process whereby the communication resources are effectively shared among multiple users. For example, contention resolution and flow control.
- Modulation or conversion between the representation of digital data in user equipment and the corresponding signals transmitted over a communications channel. These are signals operating over the physical cabling (such as copper and optical fiber) or over a radio link.

Parallel SCSI buses operate in this layer, although it must be remembered that the logical SCSI protocol is a transport layer protocol that runs over this bus. Various physical-layer Ethernet standards are also in this layer; Ethernet incorporates both this layer and the data link layer. The same applies to other local-area networks, such as token ring, FDDI, ITU-T G.hn and IEEE 802.11, as well as personal area networks such as Bluetooth and IEEE 802.15.4.

## **Layer 2: data link layer**

The data link layer provides the functional and procedural means to transfer data between network entities and to detect and possibly correct errors that may occur in the physical layer. Originally, this layer was intended for point-to-point and point-to-multipoint media, characteristic of wide area media in the telephone system. Local area network architecture, which included broadcast-capable multi-access media, was developed independently of the ISO work in IEEE Project 802. IEEE work assumed sublayer-ing and management functions not required for WAN use. In modern practice, only error detection, not flow control using sliding window, is present in data link protocols such as Point-to-Point Protocol (PPP), and, on local area networks, the IEEE 802.2 LLC layer is not used for most protocols on the Ethernet, and on other local area networks, its flow control and acknowledgment mechanisms are rarely used. Sliding window flow control and acknowledgment is used at the transport layer by protocols such as TCP, but is still used in niches where X.25 offers performance advantages.

The ITU-T G.hn standard, which provides high-speed local area networking over existing wires (power lines, phone lines and coaxial cables), includes a complete data link layer which provides both error correction and flow control by means of a selective repeat Sliding Window Protocol.

Both WAN and LAN service arrange bits, from the physical layer, into logical sequences called frames. Not all physical layer bits necessarily go into frames, as some of these bits are purely intended for physical layer functions. For example, every fifth bit of the FDDI bit stream is not used by the layer.

### **WAN protocol architecture**

Connection-oriented WAN data link protocols, in addition to framing, detect and may correct errors. They are also capable of controlling the rate of transmission. A WAN data link layer might implement a sliding window flow control and acknowledgment mechanism to provide reliable delivery of frames; that is the case for Synchronous Data Link Control (SDLC) and HDLC, and derivatives of HDLC such as LAPB and LAPD.

### **IEEE 802 LAN architecture**

Practical, connectionless LANs began with the pre-IEEE Ethernet specification, which is the ancestor of IEEE 802.3. This layer manages the interaction of devices with a shared medium, which is the function of a media access control (MAC) sublayer. Above this MAC sublayer is the media-independent IEEE 802.2 Logical Link Control (LLC) sublayer, which deals with addressing and multiplexing on multi-access media.

While IEEE 802.3 is the dominant wired LAN protocol and IEEE 802.11 the wireless LAN protocol, obsolete MAC layers include Token Ring and FDDI. The MAC sublayer detects but does not correct errors.

## **Layer 3: network layer**

The network layer provides the functional and procedural means of transferring variable length data sequences from a source host on one network to a destination host on a different network (in contrast to the data link layer which connects hosts within the same network), while maintaining the quality of service requested by the transport layer. The network layer performs network routing functions, and might also perform fragmentation and reassembly, and report delivery errors. Routers operate at this layer, sending data throughout the extended network and making the Internet possible. This is a logical addressing scheme – values are chosen by the network engineer. The addressing scheme is not hierarchical.

---

The network layer may be divided into three sublayers:

1. Subnetwork access – that considers protocols that deal with the interface to networks, such as X.25;
2. Subnetwork-dependent convergence – when it is necessary to bring the level of a transit network up to the level of networks on either side
3. Subnetwork-independent convergence – handles transfer across multiple networks.

An example of this latter case is CLNP, or IPv6 ISO 8473. It manages the connectionless transfer of data one hop at a time, from end system to ingress router, router to router, and from egress router to destination end system. It is not responsible for reliable delivery to a next hop, but only for the detection of erroneous packets so they may be discarded. In this scheme, IPv4 and IPv6 would have to be classed with X.25 as subnet access protocols because they carry interface addresses rather than node addresses.

A number of layer-management protocols, a function defined in the Management Annex, ISO 7498/4, belong to the network layer. These include routing protocols, multicast group management, network-layer information and error, and network-layer address assignment. It is the function of the payload that makes these belong to the network layer, not the protocol that carries them.

## Layer 4: transport layer

The transport layer provides transparent transfer of data between end users, providing reliable data transfer services to the upper layers. The transport layer controls the reliability of a given link through flow control, segmentation/desegmentation, and error control. Some protocols are state- and connection-oriented. This means that the transport layer can keep track of the segments and retransmit those that fail. The transport layer also provides the acknowledgement of the successful data transmission and sends the next data if no errors occurred.

OSI defines five classes of connection-mode transport protocols ranging from class 0 (which is also known as TP0 and provides the least features) to class 4 (TP4, designed for less reliable networks, similar to the Internet). Class 0 contains no error recovery, and was designed for use on network layers that provide error-free connections. Class 4 is closest to TCP, although TCP contains functions, such as the graceful close, which OSI assigns to the session layer. Also, all OSI TP connection-mode protocol classes provide expedited data and preservation of record boundaries. Detailed characteristics of TP0-4 classes are shown in the following table.<sup>[4]</sup>

Feature Name	TP0	TP1	TP2	TP3	TP4
Connection oriented network	Yes	Yes	Yes	Yes	Yes
Connectionless network	No	No	No	No	Yes
Concatenation and separation	No	Yes	Yes	Yes	Yes
Segmentation and reassembly	Yes	Yes	Yes	Yes	Yes
Error Recovery	No	Yes	Yes	Yes	Yes
Reinitiate connection (if an excessive number of PDUs are unacknowledged)	No	Yes	No	Yes	No
Multiplexing and demultiplexing over a single virtual circuit	No	No	Yes	Yes	Yes
Explicit flow control	No	No	Yes	Yes	Yes
Retransmission on timeout	No	No	No	No	Yes
Reliable Transport Service	No	Yes	No	Yes	Yes

An easy way to visualize the transport layer is to compare it with a Post Office, which deals with the dispatch and classification of mail and parcels sent. Do remember, however, that a post office manages the outer envelope of mail. Higher layers may have the equivalent of double envelopes, such as cryptographic presentation services that can be read by the addressee only. Roughly speaking, tunneling protocols operate at the transport layer, such as carrying non-IP protocols such as IBM's SNA or Novell's IPX over an IP network, or end-to-end encryption with IPsec.

While Generic Routing Encapsulation (GRE) might seem to be a network-layer protocol, if the encapsulation of the payload takes place only at endpoint, GRE becomes closer to a transport protocol that uses IP headers but contains complete frames or packets to deliver to an endpoint. L2TP carries PPP frames inside transport packet.

Although not developed under the OSI Reference Model and not strictly conforming to the OSI definition of the transport layer, the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) of the Internet Protocol Suite are commonly categorized as layer-4 protocols within OSI.

### **Layer 5: session layer**

The session layer controls the dialogues (connections) between computers. It establishes, manages and terminates the connections between the local and remote application. It provides for full-duplex, half-duplex, or simplex operation, and establishes checkpointing, adjournment, termination, and restart procedures. The OSI model made this layer responsible for graceful close of sessions, which is a property of the Transmission Control Protocol, and also for session checkpointing and recovery, which is not usually used in the Internet Protocol Suite. The session layer is commonly implemented explicitly in application environments that use remote procedure calls. On this level, Inter-Process communication happen (SIGHUP, SIGKILL, End Process, etc.).

### **Layer 6: presentation layer**

The presentation layer establishes context between application-layer entities, in which the higher-layer entities may use different syntax and semantics if the presentation service provides a mapping between them. If a mapping is available, presentation service data units are encapsulated into session protocol data units, and passed down the stack.

This layer provides independence from data representation (e.g., encryption) by translating between application and network formats. The presentation layer transforms data into the form that the application accepts. This layer formats and encrypts data to be sent across a network. It is sometimes called the syntax layer.<sup>[5]</sup>

The original presentation structure used the Basic Encoding Rules of Abstract Syntax Notation One (ASN.1), with capabilities such as converting an EBCDIC-coded text file to an ASCII-coded file, or serialization of objects and other data structures from and to XML.

### **Layer 7: application layer**

The application layer is the OSI layer closest to the end user, which means that both the OSI application layer and the user interact directly with the software application. This layer interacts with software applications that implement a communicating component. Such application programs fall outside the scope of the OSI model. Application-layer functions typically include identifying communication partners, determining resource availability, and synchronizing communication. When identifying communication partners, the application layer determines the identity and availability of communication partners for an application with data to transmit. When determining resource availability, the application layer must decide whether sufficient network or the requested communication exist. In synchronizing communication, all communication between applications requires cooperation that is managed by the application layer. Some examples of application-layer implementations also include:

- On OSI stack:
  - FTAM File Transfer and Access Management Protocol
  - X.400 Mail
  - Common Management Information Protocol (CMIP)
- On TCP/IP stack:
  - Hypertext Transfer Protocol (HTTP),
  - File Transfer Protocol (FTP),

- Simple Mail Transfer Protocol (SMTP)
- Simple Network Management Protocol (SNMP).

## Cross-layer functions

There are some functions or services that are not tied to a given layer, but they can affect more than one layer. Examples include the following:

- security service (telecommunication)<sup>[3]</sup> as defined by ITU-T X.800 Recommendation.
- management functions, i.e. functions that permit to configure, instantiate, monitor, terminate the communications of two or more entities: there is a specific application layer protocol, common management information protocol (CMIP) and its corresponding service, common management information service (CMIS), they need to interact with every layer in order to deal with their instances.
- Multiprotocol Label Switching (MPLS) operates at an OSI-model layer that is generally considered to lie between traditional definitions of layer 2 (data link layer) and layer 3 (network layer), and thus is often referred to as a "layer-2.5" protocol. It was designed to provide a unified data-carrying service for both circuit-based clients and packet-switching clients which provide a datagram service model. It can be used to carry many different kinds of traffic, including IP packets, as well as native ATM, SONET, and Ethernet frames.
- ARP is used to translate IPv4 addresses (OSI layer 3) into Ethernet MAC addresses (OSI layer 2).

## Interfaces

Neither the OSI Reference Model nor OSI protocols specify any programming interfaces, other than as deliberately abstract service specifications. Protocol specifications precisely define the interfaces between different computers, but the software interfaces inside computers, known as network sockets are implementation-specific.

For example Microsoft Windows' Winsock, and Unix's Berkeley sockets and System V Transport Layer Interface, are interfaces between applications (layer 5 and above) and the transport (layer 4). NDIS and ODI are interfaces between the media (layer 2) and the network protocol (layer 3).

Interface standards, except for the physical layer to media, are approximate implementations of OSI service specifications.

## Examples

Layer		OSI protocols	TCP/IP protocols	Signaling System 7 <sup>[6]</sup>	AppleTalk	IPX	SNA	UMTS	Misc. examples
#	Name								
7	Application	FTAM, X.400, X.500, DAP, ROSE, RTSE, ACSE <sup>[7]</sup> , CMIP <sup>[8]</sup>	NNTP, SIP, SSI, DNS, FTP, Gopher, HTTP, NFS, NTP, DHCP, SMPP, SMTP, SNMP, Telnet, RIP, BGP	INAP, MAP, TCAP, ISUP, TUP	AFP, ZIP, RTMP, NBP	RIP, SAP	APPC		HL7, Modbus
6	Presentation	ISO/IEC 8823, X.226, ISO/IEC 9576-1, X.236	MIME, SSL, TLS, XDR		AFP				TDI, ASCII, EBCDIC, MIDI, MPEG

5	Session	ISO/IEC 8327, X.225, ISO/IEC 9548-1, X.235	Sockets. Session establishment in TCP, RTP		ASP, ADSP, PAP	NWLink	DLC?		Named pipes, NetBIOS, SAP, half duplex, full duplex, simplex, RPC, SOCKS
4	Transport	ISO/IEC 8073, TP0, TP1, TP2, TP3, TP4 (X.224), ISO/IEC 8602, X.234	TCP, UDP, SCTP, DCCP			DDP, SPX			NBF
3	Network	ISO/IEC 8208, X.25 (PLP), ISO/IEC 8878, X.223, ISO/IEC 8473-1, CLNP X.233.	IP, IPsec, ICMP, IGMP, OSPF	SCCP, MTP	ATP (TokenTalk or EtherTalk)	IPX		RRC (Radio Resource Control) and BMC (Broadcast/Multicast Control)	NBF, Q.931, NDP ARP (maps layer 3 to layer 2 address), IS-IS
2	Data Link	ISO/IEC 7666, X.25 (LAPB), Token Bus, X.222, ISO/IEC 8802-2 LLC Type 1 and 2 <sup>[9]</sup>	PPP, SBT, SLIP, PPTP	MTP, Q.710	LocalTalk, AppleTalk Remote Access, PPP	IEEE 802.3 framing, Ethernet II framing	SDLC	Packet Data Convergence Protocol (PDCP) <sup>[10]</sup> , LLC (Logical Link Control), MAC (Media Access Control)	802.3 (Ethernet), 802.11a/b/g/n MAC/LLC, 802.1Q (VLAN), ATM, HDLC, FDDI, Fibre Channel, Frame Relay, HDLC, ISL, PPP, Q.921, Token Ring, CDP, ITU-T G.hn DLL CRC, Bit stuffing, ARQ, Data Over Cable Service Interface Specification (DOCSIS), interface bonding
1	Physical	X.25 (X.21bis, EIA/TIA-232, EIA/TIA-449, EIA-530, G.703) <sup>[9]</sup>		MTP, Q.710	RS-232, RS-422, STP, PhoneNet		Twinax	UMTS Physical layer or L1	RS-232, Full duplex, RJ45, V.35, V.34, I.430, I.431, T1, E1, 10BASE-T, 100BASE-TX, 1000BASE-T, POTS, SONET, SDH, DSL, 802.11a/b/g/n PHY, ITU-T G.hn PHY, Controller Area Network, Data Over Cable Service Interface Specification (DOCSIS)

## Comparison with TCP/IP model

In the TCP/IP model of the Internet, protocols are deliberately not as rigidly designed into strict layers as in the OSI model.<sup>[11]</sup> RFC 3439 contains a section entitled "Layering considered harmful (section link here [12])." However, TCP/IP does recognize four broad layers of functionality which are derived from the operating scope of their contained protocols, namely the scope of the software application, the end-to-end transport connection, the internetworking range, and the scope of the direct links to other nodes on the local network.

Even though the concept is different from the OSI model, these layers are nevertheless often compared with the OSI layering scheme in the following way: The Internet application layer includes the OSI application layer, presentation layer, and most of the session layer. Its end-to-end transport layer includes the graceful close function of the OSI session layer as well as the OSI transport layer. The internetworking layer (Internet layer) is a subset of the OSI network layer (see above), while the link layer includes the OSI data link and physical layers, as well as parts of OSI's network layer. These comparisons are based on the original seven-layer protocol model as defined in ISO 7498, rather than refinements in such things as the internal organization of the network layer document.

The presumably strict peer layering of the OSI model as it is usually described does not present contradictions in TCP/IP, as it is permissible that protocol usage does not follow the hierarchy implied in a layered model. Such examples exist in some routing protocols (e.g., OSPF), or in the description of tunneling protocols, which provide a link layer for an application, although the tunnel host protocol may well be a transport or even an application layer protocol in its own right.

## References

- [1] ITU-T X-Series Recommendations (<http://www.itu.int/rec/T-REC-X/en>)
- [2] "Publicly Available Standards" (<http://standards.iso.org/ittf/PubliclyAvailableStandards/index.html>). Standards.iso.org. 2010-07-30. . Retrieved 2010-09-11.
- [3] X.800 : Security architecture for Open Systems Interconnection for CCITT applications (<http://www.itu.int/rec/T-REC-X.800-199103-I/>)
- [4] "ITU-T Recommendation X.224 (11/1995) ISO/IEC 8073" (<http://www.itu.int/rec/T-REC-X.224-199511-I/en/>). .
- [5] Grigonis, Richard (2000). *Computer telephony encyclopedia* (<http://books.google.com/books?id=cUYk0ZhOxpEC&printsec=frontcover&dq=computer+telephony+encyclopedia&ct=result#v=onepage&q&f=false>). CMP. pp. 331. .
- [6] ITU-T Recommendation Q.1400 (03/1993) (<http://www.itu.int/rec/T-REC-Q.1400/en/>), *Architecture framework for the development of signaling and OA&M protocols using OSI concepts*, pp 4, 7.
- [7] ITU Rec. X.227 (ISO 8650), X.217 (ISO 8649)
- [8] X.700 series of recommendations from the ITU-T (in particular X.711), and ISO 9596
- [9] CISCO Cisco Systems, Inc. Internetworking Technology Handbook OSI Model Physical layer (<http://www.cisco.com/en/US/docs/internetworking/technology/handbook/Intro-to-Internet.html#wp1020669>)
- [10] 3GPP TS 36.300 : E-UTRA and E-UTRAN Overall Description, Stage 2, Release 11 (<http://www.3gpp.org/ftp/Specs/html-info/36300.htm>)
- [11] RFC 3439
- [12] <http://tools.ietf.org/html/rfc3439#section-3>

## External links

- ISO/IEC standard 7498-1:1994 ([http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269\\_ISO\\_IEC\\_7498-1\\_1994\(E\).zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/s020269_ISO_IEC_7498-1_1994(E).zip)) (PDF document inside ZIP archive) (requires HTTP cookies in order to accept licence agreement)
- ITU-T X.200 (the same contents as from ISO) ([http://www.itu.int/rec/dologin\\_pub.asp?lang=e&id=T-REC-X.200-199407-I!!PDF-E&type=items](http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.200-199407-I!!PDF-E&type=items))
- The ISO OSI Reference Model , Beluga graph of data units and groups of layers (<http://infchg.appspot.com/usr?at=1263939371>)
- Zimmermann, Hubert (April 1980). "OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection". *IEEE Transactions on Communications* **28** (4): 425–432. CiteSeerX: 10.1.1.136.9497 (<http://>



[citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.136.9497](http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.136.9497)).

- Cisco Systems Internetworking Technology Handbook ([http://docwiki.cisco.com/wiki/Internetworking\\_Technology\\_Handbook](http://docwiki.cisco.com/wiki/Internetworking_Technology_Handbook))
- Collection of animations and videos concerning computer networks (<http://www.khurramtanvir.com/cs460demos.php>)

## Internet Protocol

The **Internet Protocol (IP)** is the principal communications protocol used for relaying datagrams (also known as network packets) across an internetwork using the Internet Protocol Suite. Responsible for routing packets across network boundaries, it is the primary protocol that establishes the Internet.

IP is the primary protocol in the Internet Layer of the Internet Protocol Suite and has the task of delivering datagrams from the source host to the destination host solely based on the addresses. For this purpose, IP defines datagram structures that encapsulate the data to be delivered. It also defines addressing methods that are used to label the datagram source and destination.

Historically, IP was the connectionless datagram service in the original Transmission Control Program introduced by Vint Cerf and Bob Kahn in 1974, the other being the connection-oriented Transmission Control Protocol (TCP). The Internet Protocol Suite is therefore often referred to as TCP/IP.

The first major version of IP, Internet Protocol Version 4 (IPv4), is the dominant protocol of the internet. Its successor is Internet Protocol Version 6 (IPv6), which is increasing in use.

### Function

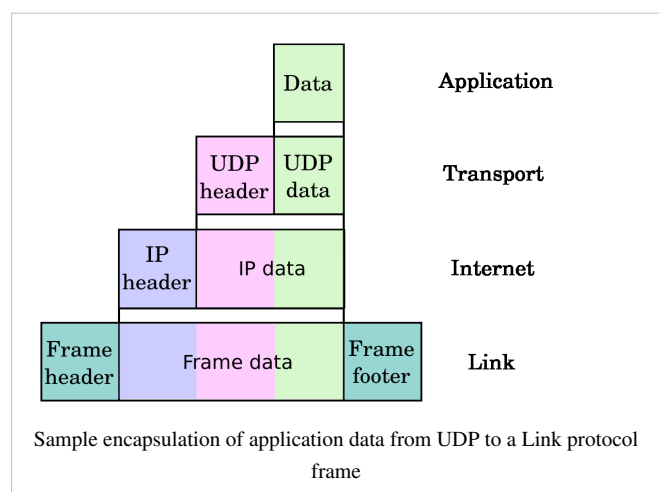
The Internet Protocol is responsible for addressing hosts and routing datagrams (packets) from a source host to the destination host across one or more IP networks. For this purpose the Internet Protocol defines an addressing system that has two functions: identifying hosts and providing a logical location service. This is accomplished by defining standard datagrams and a standard addressing system.

### Datagram construction

Each datagram has two components, a header and a payload. The IP header is tagged with the source IP address, destination IP address, and other meta-data needed to route and deliver the datagram. The payload is the data to be transported. This process of nesting data payloads in a packet with a header is called encapsulation.

### IP addressing and routing

Perhaps the most complex aspects of IP are IP addressing and routing. Addressing refers to how end hosts are assigned IP addresses and how subnetworks of IP host addresses are divided and grouped. IP routing is performed by all hosts, but most importantly by routers, which typically use either interior gateway protocols (IGPs) or external gateway protocols (EGPs) to decide how to move datagrams among networks.



IP routing is also common in local networks. For example, Ethernet switches sold today support IP multicast.<sup>[1]</sup> These switches use IP addresses and Internet Group Management Protocol for control of the multicast routing but use MAC addresses for the actual routing.

## Reliability

The design principles of the Internet protocols assume that the network infrastructure is inherently unreliable at any single network element or transmission medium and that it is dynamic in terms of availability of links and nodes. No central monitoring or performance measurement facility exists that tracks or maintains the state of the network. For the benefit of reducing network complexity, the intelligence in the network is purposely mostly located in the end nodes of each data transmission, cf. end-to-end principle. Routers in the transmission path simply forward packets to the next known local gateway matching the routing prefix for the destination address.

As a consequence of this design, the Internet Protocol only provides best effort delivery and its service is characterized as *unreliable*. In network architectural language it is a *connection-less* protocol, in contrast to so-called connection-oriented modes of transmission. The lack of reliability permits various error conditions, such as data corruption, packet loss and duplication, as well as out-of-order packet delivery. Since routing is dynamic for every packet and the network maintains no state of the path of prior packets, it is possible that some packets are routed on a longer path to their destination, resulting in improper sequencing at the receiver.

The only assistance that IPv4 provides regarding unreliability is to ensure that the IP packet header is error-free. A routing node calculates a checksum for a packet. If the checksum is bad, the routing node discards the packet. The routing node does not have to notify either end node, although the Internet Control Message Protocol (ICMP) allows such notification. In contrast, IPv6 abandons checksums in favor of faster routing.

Upper layer protocols are responsible for resolving reliability issues. For example, an upper layer protocol may cache data to make sure that it is in the correct order, before giving the data to an application.

In addition to issues of reliability, the dynamic nature and the diversity of the Internet and its components provide no guarantee that any particular path is actually capable of, or suitable for, performing the data transmission requested, even if the path is available and reliable. One of the technical constraints is the size of data packets allowed on a given link. An application must assure that it uses proper transmission characteristics. Some of this responsibility lies also in the upper layer protocols between application and IP. Facilities exist to examine the maximum transmission unit (MTU) size of the local link, as well as for the entire projected path to the destination when using IPv6. The IPv4 internetworking layer has the capability to automatically fragment the original datagram into smaller units for transmission. In this case, IP does provide re-ordering of fragments delivered out-of-order.<sup>[2]</sup>

Transmission Control Protocol (TCP) is an example of a protocol that will adjust its segment size to be smaller than the MTU. User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP) disregard MTU size, thereby forcing IP to fragment oversized datagrams.<sup>[3]</sup>

## Version history

In May 1974, the Institute of Electrical and Electronic Engineers (IEEE) published a paper entitled "A Protocol for Packet Network Intercommunication."<sup>[4]</sup> The paper's authors, Vint Cerf and Bob Kahn, described an internetworking protocol for sharing resources using packet-switching among the nodes. A central control component of this model was the "Transmission Control Program" (TCP) that incorporated both connection-oriented links and datagram services between hosts. The monolithic Transmission Control Program was later divided into a modular architecture consisting of the Transmission Control Protocol at the connection-oriented layer and the Internet Protocol at the internetworking (datagram) layer. The model became known informally as TCP/IP, although formally referenced as the Internet Protocol Suite.

---

The Internet Protocol is one of the elements that define the Internet. The dominant internetworking protocol in the Internet Layer in use today is IPv4; the number 4 is the protocol version number carried in every IP datagram. IPv4 is described in RFC 791 (1981).

The successor to IPv4 is IPv6. Its most prominent modification from version 4 is the addressing system. IPv4 uses 32-bit addresses (c. 4 billion, or  $43 \times 10^9$ , addresses) while IPv6 uses 128-bit addresses (c. 340 undecillion, or  $34 \times 10^{38}$  addresses). Although adoption of IPv6 has been slow, as of June 2008, all United States government systems have demonstrated basic infrastructure support for IPv6 (if only at the backbone level).<sup>[5]</sup>

IP versions 0 to 3 were development versions of IPv4 and were used between 1977 and 1979. Version 5 was used by the Internet Stream Protocol, an experimental streaming protocol. Version numbers 6 through 9 were proposed for various protocol models designed to replace IPv4: SIPP (Simple Internet Protocol Plus, known now as IPv6), TP/IX (RFC 1475), PIP (RFC 1621) and TUBA (TCP and UDP with Bigger Addresses, RFC 1347).

Other protocol proposals named *IPv9* and *IPv8* briefly surfaced, but have no support.<sup>[6]</sup>

On April 1, 1994, the IETF published an April Fool's Day joke about IPv9.<sup>[7]</sup>

## Vulnerabilities

The Internet Protocol is vulnerable to a variety of attacks. A thorough vulnerability assessment, along with proposed mitigations, was published in 2008,<sup>[8]</sup> and is currently being pursued within the IETF.<sup>[9]</sup>

## References

- [1] Netgear ProSafe XSM7224S reference manual
- [2] Siyan, Karanjit. *Inside TCP/IP*, New Riders Publishing, 1997. ISBN 1-56205-714-6
- [3] Basic Journey of a Packet (<http://www.securityfocus.com/infocus/1870>)
- [4] Vinton G. Cerf, Robert E. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. 22, No. 5, May 1974 pp. 637-648
- [5] CIO council adds to IPv6 transition primer ([http://www.gcn.com/print/25\\_16/41051-1.html](http://www.gcn.com/print/25_16/41051-1.html)), gcn.com
- [6] Theregister.com ([http://www.theregister.co.uk/2004/07/06/ipv9\\_hype\\_dismissed/](http://www.theregister.co.uk/2004/07/06/ipv9_hype_dismissed/))
- [7] RFC 1606: *A Historical Perspective On The Usage Of IP Version 9*. April 1, 1994.
- [8] Security Assessment of the Internet Protocol (IP)(archived version) (<http://web.archive.org/web/20100211145721/http://www.cpni.gov.uk/Docs/InternetProtocol.pdf>)
- [9] Security Assessment of the Internet Protocol version 4 (IPv4) (<http://tools.ietf.org/html/draft-ietf-opsec-ip-security>)

## External links

- Internet Protocol (<http://www.dmoz.org/Computers/Internet/Protocols/>) at the Open Directory Project
- RFC 791
- Data Communication Lectures of Manfred Lindner - Part IP Technology Basics ([http://www.ict.tuwien.ac.at/lva/384.081/infobase/L30-IP\\_Technology\\_Basics\\_v4-6.pdf](http://www.ict.tuwien.ac.at/lva/384.081/infobase/L30-IP_Technology_Basics_v4-6.pdf))
- Data Communication Lectures of Manfred Lindner - Part IP Technology Details ([http://www.ict.tuwien.ac.at/lva/384.081/infobase/L31-IP\\_Technology\\_Details\\_v4-7.pdf](http://www.ict.tuwien.ac.at/lva/384.081/infobase/L31-IP_Technology_Details_v4-7.pdf))
- Data Communication Lectures of Manfred Lindner - Part IPv6 ([http://www.ict.tuwien.ac.at/lva/384.081/infobase/L80-IPv6\\_v4-6.pdf](http://www.ict.tuwien.ac.at/lva/384.081/infobase/L80-IPv6_v4-6.pdf))
- IPv6.com - Knowledge Center for Next Generation Internet IPv6 (<http://www.ipv6.com>)

---

# Transmission Control Protocol

---

The **Transmission Control Protocol (TCP)** is one of the core protocols of the Internet Protocol Suite. TCP is one of the two original components of the suite, complementing the Internet Protocol (IP), and therefore the entire suite is commonly referred to as *TCP/IP*. TCP provides reliable, ordered delivery of a stream of octets from a program on one computer to another program on another computer. TCP is the protocol used by major Internet applications such as the World Wide Web, email, remote administration and file transfer. Other applications, which do not require reliable data stream service, may use the User Datagram Protocol (UDP), which provides a datagram service that emphasizes reduced latency over reliability.

## Historical origin

In May 1974 the Institute of Electrical and Electronic Engineers (IEEE) published a paper entitled "*A Protocol for Packet Network Intercommunication*."<sup>[1]</sup> The paper's authors, Vint Cerf and Bob Kahn, described an internetworking protocol for sharing resources using packet-switching among the nodes. A central control component of this model was the *Transmission Control Program* that incorporated both connection-oriented links and datagram services between hosts. The monolithic Transmission Control Program was later divided into a modular architecture consisting of the *Transmission Control Protocol* at the connection-oriented layer and the *Internet Protocol* at the internetworking (datagram) layer. The model became known informally as *TCP/IP*, although formally it was henceforth called the *Internet Protocol Suite*.

## Network function

The protocol corresponds to the transport layer of TCP/IP suite. TCP provides a communication service at an intermediate level between an application program and the Internet Protocol (IP). That is, when an application program desires to send a large chunk of data across the Internet using IP, instead of breaking the data into IP-sized pieces and issuing a series of IP requests, the software can issue a single request to TCP and let TCP handle the IP details.

IP works by exchanging pieces of information called packets. A packet is a sequence of octets and consists of a *header* followed by a *body*. The header describes the packet's destination and, optionally, the routers to use for forwarding until it arrives at its destination. The body contains the data IP is transmitting.

Due to network congestion, traffic load balancing, or other unpredictable network behavior, IP packets can be lost, duplicated, or delivered out of order. TCP detects these problems, requests retransmission of lost data, rearranges out-of-order data, and even helps minimize network congestion to reduce the occurrence of the other problems. Once the TCP receiver has reassembled the sequence of octets originally transmitted, it passes them to the application program. Thus, TCP abstracts the application's communication from the underlying networking details.

TCP is utilized extensively by many of the Internet's most popular applications, including the World Wide Web (WWW), E-mail, File Transfer Protocol, Secure Shell, peer-to-peer file sharing, and some streaming media applications.

TCP is optimized for accurate delivery rather than timely delivery, and therefore, TCP sometimes incurs relatively long delays (in the order of seconds) while waiting for out-of-order messages or retransmissions of lost messages. It is not particularly suitable for real-time applications such as Voice over IP. For such applications, protocols like the Real-time Transport Protocol (RTP) running over the User Datagram Protocol (UDP) are usually recommended instead.<sup>[2]</sup>

TCP is a reliable stream delivery service that guarantees that all bytes received will be identical with bytes sent and in the correct order. Since packet transfer is not reliable, a technique known as positive acknowledgment with retransmission is used to guarantee reliability of packet transfers. This fundamental technique requires the receiver to

---

respond with an acknowledgment message as it receives the data. The sender keeps a record of each packet it sends. The sender also keeps a timer from when the packet was sent, and retransmits a packet if the timer expires before the message has been acknowledged. The timer is needed in case a packet gets lost or corrupted.<sup>[2]</sup>

TCP consists of a set of rules: for the protocol, that are used with the Internet Protocol, and for the IP, to send data "in a form of message units" between computers over the Internet. While IP handles actual delivery of the data, TCP keeps track of the individual units of data transmission, called *segments*, that a message is divided into for efficient routing through the network. For example, when an HTML file is sent from a Web server, the TCP software layer of that server divides the sequence of octets of the file into segments and forwards them individually to the IP software layer (Internet Layer). The Internet Layer encapsulates each TCP segment into an IP packet by adding a header that includes (among other data) the destination IP address. Even though every packet has the same destination address, they can be routed on different paths through the network. When the client program on the destination computer receives them, the TCP layer (Transport Layer) reassembles the individual segments and ensures they are correctly ordered and error free as it streams them to an application.

## TCP segment structure

Transmission Control Protocol accepts data from a data stream, segments it into chunks, and adds a TCP header creating a TCP segment. The TCP segment is then encapsulated into an Internet Protocol (IP) datagram. A TCP segment is "the packet of information that TCP uses to exchange data with its peers."<sup>[3]</sup>

The term *TCP packet*, though sometimes informally used, is not in line with current terminology, where *segment* refers to the TCP PDU, *datagram*<sup>[4]</sup> to the IP PDU and *frame* to the data link layer PDU:

Processes transmit data by calling on the TCP and passing buffers of data as arguments. The TCP packages the data from these buffers into segments and calls on the internet module [e.g. IP] to transmit each segment to the destination TCP.<sup>[5]</sup>

A TCP segment consists of a segment *header* and a *data* section. The TCP header contains 10 mandatory fields, and an optional extension field (*Options*, orange background in table).

The data section follows the header. Its contents are the payload data carried for the application. The length of the data section is not specified in the TCP segment header. It can be calculated by subtracting the combined length of the TCP header and the encapsulating IP header from the total IP datagram length (specified in the IP header).

### TCP Header

Offsets	Octet	0								1								2								3																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																									
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																		
0	0	Source port																Destination port																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
4	32	Sequence number																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
8	64	Acknowledgment number (if ACK set)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	
12	96	Data offset				Reserved 0 0 0			N	C	E	U	A	P	R	S	F	Window Size																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																	

- Source port (16 bits) – identifies the sending port
- Destination port (16 bits) – identifies the receiving port
- Sequence number (32 bits) – has a dual role:

- If the `SYN` flag is set (1), then this is the initial sequence number. The sequence number of the actual first data byte and the acknowledged number in the corresponding `ACK` are then this sequence number plus 1.
- If the `SYN` flag is clear (0), then this is the accumulated sequence number of the first data byte of this packet for the current session.
- Acknowledgment number (32 bits) – if the `ACK` flag is set then the value of this field is the next sequence number that the receiver is expecting. This acknowledges receipt of all prior bytes (if any). The first `ACK` sent by each end acknowledges the other end's initial sequence number itself, but no data.
- Data offset (4 bits) – specifies the size of the TCP header in 32-bit words. The minimum size header is 5 words and the maximum is 15 words thus giving the minimum size of 20 bytes and maximum of 60 bytes, allowing for up to 40 bytes of options in the header. This field gets its name from the fact that it is also the offset from the start of the TCP segment to the actual data.
- Reserved (3 bits) – for future use and should be set to zero
- Flags (9 bits) (aka Control bits) – contains 9 1-bit flags
  - `NS` (1 bit) – ECN-nonce concealment protection (added to header by RFC 3540).
  - `CWR` (1 bit) – Congestion Window Reduced (`CWR`) flag is set by the sending host to indicate that it received a TCP segment with the `ECE` flag set and had responded in congestion control mechanism (added to header by RFC 3168).
  - `ECE` (1 bit) – ECN-Echo indicates
    - If the `SYN` flag is set (1), that the TCP peer is ECN capable.
    - If the `SYN` flag is clear (0), that a packet with Congestion Experienced flag in IP header set is received during normal transmission (added to header by RFC 3168).
  - `URG` (1 bit) – indicates that the Urgent pointer field is significant
  - `ACK` (1 bit) – indicates that the Acknowledgment field is significant. All packets after the initial `SYN` packet sent by the client should have this flag set.
  - `PSH` (1 bit) – Push function. Asks to push the buffered data to the receiving application.
  - `RST` (1 bit) – Reset the connection
  - `SYN` (1 bit) – Synchronize sequence numbers. Only the first packet sent from each end should have this flag set. Some other flags change meaning based on this flag, and some are only valid for when it is set, and others when it is clear.
  - `FIN` (1 bit) – No more data from sender
- Window size (16 bits) – the size of the *receive window*, which specifies the number of bytes (beyond the sequence number in the acknowledgment field) that the sender of this segment is currently willing to receive (*see Flow control and Window Scaling*)
- Checksum (16 bits) – The 16-bit checksum field is used for error-checking of the header and data
- Urgent pointer (16 bits) – if the `URG` flag is set, then this 16-bit field is an offset from the sequence number indicating the last urgent data byte
- Options (Variable 0–320 bits, divisible by 32) – The length of this field is determined by the data offset field. Options have up to three fields: Option-Kind (1 byte), Option-Length (1 byte), Option-Data (variable). The Option-Kind field indicates the type of option, and is the only field that is not optional. Depending on what kind of option we are dealing with, the next two fields may be set: the Option-Length field indicates the total length of the option, and the Option-Data field contains the value of the option, if applicable. For example, an Option-Kind byte of 0x01 indicates that this is a No-Op option used only for padding, and does not have an Option-Length or Option-Data byte following it. An Option-Kind byte of 0 is the End Of Options option, and is also only one byte. An Option-Kind byte of 0x02 indicates that this is the Maximum Segment Size option, and will be followed by a byte specifying the length of the MSS field (should be 0x04). Note that this length is the total length of the given options field, including Option-Kind and Option-Length bytes. So while the MSS value is typically expressed in

two bytes, the length of the field will be 4 bytes (+2 bytes of kind and length). In short, an MSS option field with a value of 0x05B4 will show up as (0x02 0x04 0x05B4) in the TCP options section.

- Padding – The TCP header padding is used to ensure that the TCP header ends and data begins on a 32 bit boundary. The padding is composed of zeros.<sup>[6]</sup>

Some options may only be sent when SYN is set; they are indicated below as <sup>[SYN]</sup>. Option-Kind and standard lengths given as (Option-Kind,Option-Length).

- 0 (8 bits) – End of options list
- 1 (8 bits) – No operation (NOP, Padding) This may be used to align option fields on 32-bit boundaries for better performance.
- 2,4,SS (32 bits) – Maximum segment size (*see maximum segment size*) <sup>[SYN]</sup>
- 3,3,S (24 bits) – Window scale (*see window scaling for details*) <sup>[SYN]</sup><sup>[7]</sup>
- 4,2 (16 bits) – Selective Acknowledgement permitted. <sup>[SYN]</sup> (*See selective acknowledgments for details*)<sup>[8]</sup>
- 5,N,BBBB,EEEE,... (variable bits, N is either 10, 18, 26, or 34)- Selective ACKnowledgement (SACK)<sup>[9]</sup>  
These first two bytes are followed by a list of 1–4 blocks being selectively acknowledged, specified as 32-bit begin/end pointers.
- 8,10,TTTT,EEEE (80 bits)- Timestamp and echo of previous timestamp (*see TCP timestamps for details*)<sup>[10]</sup>
- 14,3,S (24 bits) – TCP Alternate Checksum Request. <sup>[SYN]</sup><sup>[11]</sup>
- 15,N,... (variable bits) – TCP Alternate Checksum Data.

(The remaining options are obsolete, experimental, not yet standardized, or unassigned)

## Protocol operation

TCP protocol operations may be divided into three phases. Connections must be properly established in a multi-step handshake process (*connection establishment*) before entering the *data transfer* phase. After data transmission is completed, the *connection termination* closes established virtual circuits and releases all allocated resources.

A TCP connection is managed by an operating system through a programming interface that represents the local end-point for communications, the *Internet socket*. During the lifetime of a TCP connection the local end-point undergoes a series of state changes:<sup>[13]</sup>

### LISTEN

(server) represents waiting for a connection request from any remote TCP and port.

### SYN-SENT

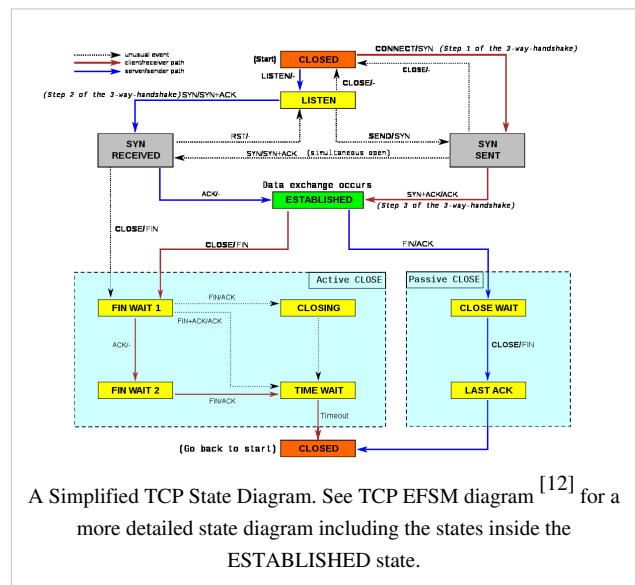
(client) represents waiting for a matching connection request after having sent a connection request.

### SYN-RECEIVED

(server) represents waiting for a confirming connection request acknowledgment after having both received and sent a connection request.

### ESTABLISHED

(both server and client) represents an open connection, data received can be delivered to the user. The normal state for the data transfer phase of the connection.



**FIN-WAIT-1**

(both server and client) represents waiting for a connection termination request from the remote TCP, or an acknowledgment of the connection termination request previously sent.

**FIN-WAIT-2**

(both server and client) represents waiting for a connection termination request from the remote TCP.

**CLOSE-WAIT**

(both server and client) represents waiting for a connection termination request from the local user.

**CLOSING**

(both server and client) represents waiting for a connection termination request acknowledgment from the remote TCP.

**LAST-ACK**

(both server and client) represents waiting for an acknowledgment of the connection termination request previously sent to the remote TCP (which includes an acknowledgment of its connection termination request).

**TIME-WAIT**

(either server or client) represents waiting for enough time to pass to be sure the remote TCP received the acknowledgment of its connection termination request. [According to RFC 793 a connection can stay in TIME-WAIT for a maximum of four minutes known as a MSL (maximum segment lifetime).]

**CLOSED**

(both server and client) represents no connection state at all.

**Connection establishment**

To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open. To establish a connection, the three-way (or 3-step) handshake occurs:

1. **SYN:** The active open is performed by the client sending a SYN to the server. The client sets the segment's sequence number to a random value A.
2. **SYN-ACK:** In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number ( $A + 1$ ), and the sequence number that the server chooses for the packet is another random number, B.
3. **ACK:** Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e.  $A + 1$ , and the acknowledgement number is set to one more than the received sequence number i.e.  $B + 1$ .

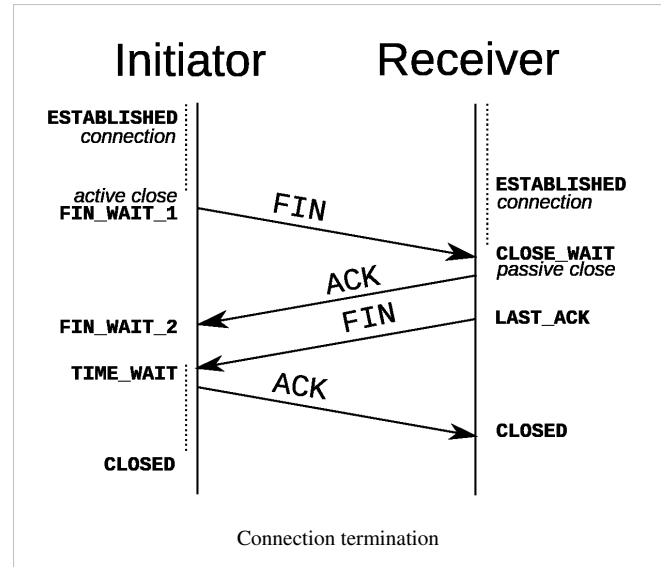
At this point, both the client and server have received an acknowledgment of the connection. The steps 1, 2 establish the connection parameter (sequence number) for one direction and it is acknowledged. The steps 2, 3 establish the connection parameter (sequence number) for the other direction and it is acknowledged. With these, a full-duplex communication is established.

---



## Connection termination

The connection termination phase uses a four-way handshake, with each side of the connection terminating independently. When an endpoint wishes to stop its half of the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. Therefore, a typical tear-down requires a pair of FIN and ACK segments from each TCP endpoint. After both FIN/ACK exchanges are concluded, the side which sent the first FIN before receiving one waits for a timeout before finally closing the connection, during which time the local port is unavailable for new connections; this prevents confusion due to delayed packets being delivered during subsequent connections.



A connection can be "half-open", in which case one side has terminated its end, but the other has not. The side that has terminated can no longer send any data into the connection, but the other side can. The terminating side should continue reading the data until the other side terminates as well.

It is also possible to terminate the connection by a 3-way handshake, when host A sends a FIN and host B replies with a FIN & ACK (merely combines 2 steps into one) and host A replies with an ACK.<sup>[14]</sup> This is perhaps the most common method.

It is possible for both hosts to send FINs simultaneously then both just have to ACK. This could possibly be considered a 2-way handshake since the FIN/ACK sequence is done in parallel for both directions.

Some host TCP stacks may implement a half-duplex close sequence, as Linux or HP-UX do. If such a host actively closes a connection but still has not read all the incoming data the stack already received from the link, this host sends a RST instead of a FIN (Section 4.2.2.13 in RFC 1122<sup>[15]</sup>). This allows a TCP application to be sure the remote application has read all the data the former sent—waiting the FIN from the remote side, when it actively closes the connection. However, the remote TCP stack cannot distinguish between a *Connection Aborting RST* and this *Data Loss RST*. Both cause the remote stack to throw away all the data it received, but that the application still didn't read.

Some application protocols may violate the OSI model layers, using the TCP open/close handshaking for the application protocol open/close handshaking — these may find the RST problem on active close. As an example:

```
s = connect(remote);
send(s, data);
close(s);
```

For a usual program flow like above, a TCP/IP stack like that described above does not guarantee that all the data arrives to the other application.

## Resource usage

Most implementations allocate an entry in a table that maps a session to a running operating system process. Because TCP packets do not include a session identifier, both endpoints identify the session using the client's address and port. Whenever a packet is received, the TCP implementation must perform a lookup on this table to find the destination process.

The number of sessions in the server side is limited only by memory and can grow as new connections arrive, but the client must allocate a random port before sending the first SYN to the server. This port remains allocated during the whole conversation, and effectively limits the number of outgoing connections from each of the client's IP addresses. If an application fails to properly close unrequired connections, a client can run out of resources and become unable to establish new TCP connections, even from other applications.

Both endpoints must also allocate space for unacknowledged packets and received (but unread) data.

## Data transfer

There are a few key features that set TCP apart from User Datagram Protocol:

- Ordered data transfer — the destination host rearranges according to sequence number<sup>[2]</sup>
- Retransmission of lost packets — any cumulative stream not acknowledged is retransmitted<sup>[2]</sup>
- Error-free data transfer<sup>[16]</sup>
- Flow control — limits the rate a sender transfers data to guarantee reliable delivery. The receiver continually hints the sender on how much data can be received (controlled by the sliding window). When the receiving host's buffer fills, the next acknowledgment contains a 0 in the window size, to stop transfer and allow the data in the buffer to be processed.<sup>[2]</sup>
- Congestion control<sup>[2]</sup>

## Reliable transmission

TCP uses a *sequence number* to identify each byte of data. The sequence number identifies the order of the bytes sent from each computer so that the data can be reconstructed in order, regardless of any fragmentation, disordering, or packet loss that may occur during transmission. For every payload byte transmitted, the sequence number must be incremented. In the first two steps of the 3-way handshake, both computers exchange an initial sequence number (ISN). This number can be arbitrary, and should in fact be unpredictable to defend against TCP Sequence Prediction Attacks.

TCP primarily uses a *cumulative acknowledgment* scheme, where the receiver sends an acknowledgment signifying that the receiver has received all data preceding the acknowledged sequence number. The sender sets the sequence number field to the sequence number of the first payload byte in the segment's data field, and the receiver sends an acknowledgment specifying the sequence number of the next byte they expect to receive. For example, if a sending computer sends a packet containing four payload bytes with a sequence number field of 100, then the sequence numbers of the four payload bytes are 100, 101, 102 and 103. When this packet arrives at the receiving computer, it would send back an acknowledgment number of 104 since that is the sequence number of the next byte it expects to receive in the next packet.

In addition to cumulative acknowledgments, TCP receivers can also send selective acknowledgments to provide further information.

If the sender infers that data has been lost in the network, it retransmits the data.

## Error detection

Sequence numbers and acknowledgments cover discarding duplicate packets, retransmission of lost packets, and ordered-data transfer. To assure correctness a checksum field is included (*see TCP segment structure for details on checksumming*).

The TCP checksum is a weak check by modern standards. Data Link Layers with high bit error rates may require additional link error correction/detection capabilities. The weak checksum is partially compensated for by the common use of a CRC or better integrity check at layer 2, below both TCP and IP, such as is used in PPP or the Ethernet frame. However, this does not mean that the 16-bit TCP checksum is redundant: remarkably, introduction of errors in packets between CRC-protected hops is common, but the end-to-end 16-bit TCP checksum catches most of these simple errors.<sup>[17]</sup> This is the end-to-end principle at work.

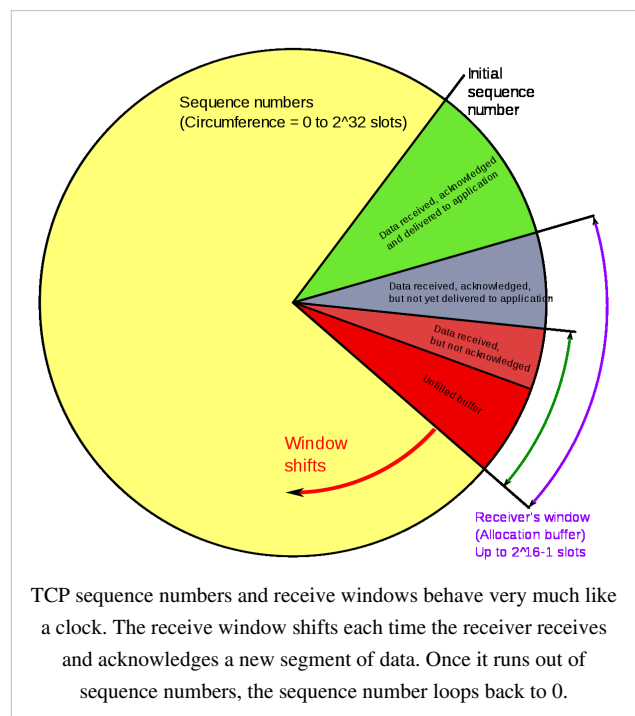
## Flow control

TCP uses an end-to-end flow control protocol to avoid having the sender send data too fast for the TCP receiver to receive and process it reliably. Having a mechanism for flow control is essential in an environment where machines of diverse network speeds communicate. For example, if a PC sends data to a smartphone that is slowly processing received data, the smartphone must regulate the data flow so as not to be overwhelmed.<sup>[2]</sup>

TCP uses a sliding window flow control protocol. In each TCP segment, the receiver specifies in the *receive window* field the amount of additionally received data (in bytes) that it is willing to buffer for the connection. The sending host can send only up to that amount of data before it must wait for an acknowledgment and window update from the receiving host.

When a receiver advertises a window size of 0, the sender stops sending data and starts the *persist timer*. The persist timer is used to protect TCP from a deadlock situation that could arise if a subsequent window size update from the receiver is lost, and the sender cannot send more data until receiving a new window size update from the receiver. When the persist timer expires, the TCP sender attempts recovery by sending a small packet so that the receiver responds by sending another acknowledgement containing the new window size.

If a receiver is processing incoming data in small increments, it may repeatedly advertise a small receive window. This is referred to as the silly window syndrome, since it is inefficient to send only a few bytes of data in a TCP segment, given the relatively large overhead of the TCP header. TCP senders and receivers typically employ flow control logic to specifically avoid repeatedly sending small segments. The sender-side silly window syndrome avoidance logic is referred to as Nagle's algorithm.



## Congestion control

The final main aspect of TCP is congestion control. TCP uses a number of mechanisms to achieve high performance and avoid congestion collapse, where network performance can fall by several orders of magnitude. These mechanisms control the rate of data entering the network, keeping the data flow below a rate that would trigger collapse. They also yield an approximately max-min fair allocation between flows.

Acknowledgments for data sent, or lack of acknowledgments, are used by senders to infer network conditions between the TCP sender and receiver. Coupled with timers, TCP senders and receivers can alter the behavior of the flow of data. This is more generally referred to as congestion control and/or network congestion avoidance.

Modern implementations of TCP contain four intertwined algorithms: Slow-start, congestion avoidance, fast retransmit, and fast recovery (RFC 5681).

In addition, senders employ a *retransmission timeout* (RTO) that is based on the estimated round-trip time (or RTT) between the sender and receiver, as well as the variance in this round trip time. The behavior of this timer is specified in RFC 6298. There are subtleties in the estimation of RTT. For example, senders must be careful when calculating RTT samples for retransmitted packets; typically they use Karn's Algorithm or TCP timestamps (see RFC 1323). These individual RTT samples are then averaged over time to create a Smoothed Round Trip Time (SRTT) using Jacobson's algorithm. This SRTT value is what is finally used as the round-trip time estimate.

Enhancing TCP to reliably handle loss, minimize errors, manage congestion and go fast in very high-speed environments are ongoing areas of research and standards development. As a result, there are a number of TCP congestion avoidance algorithm variations.

## Maximum segment size

The maximum segment size (MSS) is the largest amount of data, specified in bytes, that TCP is willing to receive in a single segment. For best performance, the MSS should be set small enough to avoid IP fragmentation, which can lead to packet loss and excessive retransmissions. To try to accomplish this, typically the MSS is announced by each side using the MSS option when the TCP connection is established, in which case it is derived from the maximum transmission unit (MTU) size of the data link layer of the networks to which the sender and receiver are directly attached. Furthermore, TCP senders can use path MTU discovery to infer the minimum MTU along the network path between the sender and receiver, and use this to dynamically adjust the MSS to avoid IP fragmentation within the network.

MSS announcement is also often called "MSS negotiation". Strictly speaking, the MSS is not "negotiated" between the originator and the receiver, because that would imply that both originator and receiver will negotiate and agree upon a single, unified MSS that applies to all communication in both directions of the connection. In fact, two completely independent values of MSS are permitted for the two directions of data flow in a TCP connection.<sup>[18]</sup> This situation may arise, for example, if one of the devices participating in a connection has an extremely limited amount of memory reserved (perhaps even smaller than the overall discovered Path MTU) for processing incoming TCP segments.

## Selective acknowledgments

Relying purely on the cumulative acknowledgment scheme employed by the original TCP protocol can lead to inefficiencies when packets are lost. For example, suppose 10,000 bytes are sent in 10 different TCP packets, and the first packet is lost during transmission. In a pure cumulative acknowledgment protocol, the receiver cannot say that it received bytes 1,000 to 9,999 successfully, but failed to receive the first packet, containing bytes 0 to 999. Thus the sender may then have to resend all 10,000 bytes.

To solve this problem TCP employs the *selective acknowledgment* (SACK) option, defined in RFC 2018, which allows the receiver to acknowledge discontinuous blocks of packets that were received correctly, in addition to the

sequence number of the last contiguous byte received successively, as in the basic TCP acknowledgment. The acknowledgement can specify a number of *SACK blocks*, where each SACK block is conveyed by the starting and ending sequence numbers of a contiguous range that the receiver correctly received. In the example above, the receiver would send SACK with sequence numbers 1000 and 9999. The sender thus retransmits only the first packet, bytes 0 to 999.

An extension to the SACK option is the duplicate-SACK option, defined in RFC 2883. An out-of-order packet delivery can often falsely indicate the TCP sender of lost packet and, in turn, the TCP sender retransmits the suspected-to-be-lost packet and slow down the data delivery to prevent network congestion. The TCP sender undoes the action of slow-down, that is a recovery of the original pace of data transmission, upon receiving a D-SACK that indicates the retransmitted packet is duplicate.

The SACK option is not mandatory and it is used only if both parties support it. This is negotiated when connection is established. SACK uses the optional part of the TCP header (*see TCP segment structure for details*). The use of SACK is widespread — all popular TCP stacks support it. Selective acknowledgment is also used in Stream Control Transmission Protocol (SCTP).

## Window scaling

For more efficient use of high bandwidth networks, a larger TCP window size may be used. The TCP window size field controls the flow of data and its value is limited to between 2 and 65,535 bytes.

Since the size field cannot be expanded, a scaling factor is used. The TCP window scale option, as defined in RFC 1323, is an option used to increase the maximum window size from 65,535 bytes to 1 gigabyte. Scaling up to larger window sizes is a part of what is necessary for TCP Tuning.

The window scale option is used only during the TCP 3-way handshake. The window scale value represents the number of bits to left-shift the 16-bit window size field. The window scale value can be set from 0 (no shift) to 14 for each direction independently. Both sides must send the option in their SYN segments to enable window scaling in either direction.

Some routers and packet firewalls rewrite the window scaling factor during a transmission. This causes sending and receiving sides to assume different TCP window sizes. The result is non-stable traffic that may be very slow. The problem is visible on some sending and receiving sites behind the path of defective routers.<sup>[19]</sup>

## TCP timestamps

TCP timestamps, defined in RFC 1323, can help TCP determine in which order packets were sent. TCP timestamps are not normally aligned to the system clock and start at some random value. Many operating systems will increment the timestamp for every elapsed milisecond; however the RFC only states that the ticks should be proportional.

There are two timestamp fields:

```
a 4-byte sender timestamp value (my timestamp)
a 4-byte echo reply timestamp value (the most recent timestamp received from you).
```

TCP timestamps are used in an algorithm known as *Protection Against Wrapped Sequence* numbers, or *PAWS* (see RFC 1323 for details). PAWS is used when the TCP window size exceeds the possible numbers of sequence numbers ( $2^{32}$  or 4 billion/gig). In the case where a packet was potentially retransmitted it answers the question: "Is this sequence number in the first 4 GB or the second?" And the timestamp is used to break the tie.

RFC 1323 incorrectly states in section 2.2 that the window scale must be limited to  $2^{14}$  to remain under 1 GB (which is correct, but the sequence number limit is 4 GB); however a scale of 16 and a window size of 65535 would be 65536 less than the  $2^{32}$  possible sequence numbers and thus an acceptable yet excessive value. Because of this error many systems have limited the max scale to  $2^{14}$  to "follow the RFC".

Also, the Eifel detection algorithm (RFC 3522) uses TCP timestamps to determine if retransmissions are occurring because packets are lost or simply out of order.

## Out of band data

One is able to interrupt or abort the queued stream instead of waiting for the stream to finish. This is done by specifying the data as *urgent*. This tells the receiving program to process it immediately, along with the rest of the urgent data. When finished, TCP informs the application and resumes back to the stream queue. An example is when TCP is used for a remote login session, the user can send a keyboard sequence that interrupts or aborts the program at the other end. These signals are most often needed when a program on the remote machine fails to operate correctly. The signals must be sent without waiting for the program to finish its current transfer.<sup>[2]</sup>

TCP OOB data was not designed for the modern Internet. The *urgent* pointer only alters the processing on the remote host and doesn't expedite any processing on the network itself. When it gets to the remote host there are two slightly different interpretations of the protocol, which means only single bytes of OOB data are reliable. This is assuming it is reliable at all as it is one of the least commonly used protocol elements and tends to be poorly implemented.<sup>[20][21]</sup>

## Forcing data delivery

Normally, TCP waits for 200 ms or for a full packet of data to send (Nagle's Algorithm = tries to group small messages into a single packet). This creates minor, but potentially serious delays if repeated constantly during a file transfer. For example a typical send block would be 4 KB, a typical MSS is 1460, so 2 packets go out on a 10 Mbit/s ethernet taking ~1.2 ms each followed by a third carrying the remaining 1176 after a 197 ms pause because TCP is waiting for a full buffer.

In the case of telnet, each user keystroke is echoed back by the server before the user can see it on the screen. This delay would become very annoying.

Setting the socket option `TCP_NODELAY` overrides the default 200 ms send delay. Application programs use this socket option to force output to be sent after writing a character or line of characters.

The RFC defines the `PSH` push bit as "a message to the receiving TCP stack to send this data immediately up to the receiving application".<sup>[2]</sup> There is no way to indicate or control it in User space using Berkeley sockets and it is controlled by Protocol stack only.<sup>[22]</sup>

## Vulnerabilities

TCP may be attacked in a variety of ways. The results of a thorough security assessment of TCP, along with possible mitigations for the identified issues, was published in 2009,<sup>[23]</sup> and is currently being pursued within the IETF.<sup>[24]</sup>

## Denial of service

By using a spoofed IP address and repeatedly sending purposely assembled SYN packets, attackers can cause the server to consume large amounts of resources keeping track of the bogus connections. This is known as a SYN flood attack. Proposed solutions to this problem include SYN cookies and cryptographic puzzles. Sockstress is a similar attack, that might be mitigated with system resource management.<sup>[25]</sup> An advanced DoS attack involving the exploitation of the TCP Persist Timer was analyzed at Phrack #66.<sup>[26]</sup>

## Connection hijacking

An attacker who is able to eavesdrop a TCP session and redirect packets can hijack a TCP connection. To do so, the attacker learns the sequence number from the ongoing communication and forges a false segment that looks like the next segment in the stream. Such a simple hijack can result in one packet being erroneously accepted at one end. When the receiving host acknowledges the extra segment to the other side of the connection, synchronization is lost. Hijacking might be combined with ARP or routing attacks that allow taking control of the packet flow, so as to get permanent control of the hijacked TCP connection.<sup>[27]</sup>

Impersonating a different IP address was not difficult prior to RFC 1948, when the initial *sequence number* was easily guessable. That allowed an attacker to blindly send a sequence of packets that the receiver would believe to come from a different IP address, without the need to deploy ARP or routing attacks: it is enough to ensure that the legitimate host of the impersonated IP address is down, or bring it to that condition using denial of service attacks. This is why the initial sequence number is now chosen at random.

## TCP ports

TCP uses port numbers to identify sending and receiving application end-points on a host, or *Internet sockets*. Each side of a TCP connection has an associated 16-bit unsigned port number (0-65535) reserved by the sending or receiving application. Arriving TCP data packets are identified as belonging to a specific TCP connection by its sockets, that is, the combination of source host address, source port, destination host address, and destination port. This means that a server computer can provide several clients with several services simultaneously, as long as a client takes care of initiating any simultaneous connections to one destination port from different source ports.

Port numbers are categorized into three basic categories: well-known, registered, and dynamic/private. The well-known ports are assigned by the Internet Assigned Numbers Authority (IANA) and are typically used by system-level or root processes. Well-known applications running as servers and passively listening for connections typically use these ports. Some examples include: FTP (20 and 21), SSH (22), TELNET (23), SMTP (25), SSL (443) and HTTP (80). Registered ports are typically used by end user applications as ephemeral source ports when contacting servers, but they can also identify named services that have been registered by a third party. Dynamic/private ports can also be used by end user applications, but are less commonly so. Dynamic/private ports do not contain any meaning outside of any particular TCP connection.

## Development

TCP is a complex protocol. However, while significant enhancements have been made and proposed over the years, its most basic operation has not changed significantly since its first specification RFC 675 in 1974, and the v4 specification RFC 793, published in September 1981. RFC 1122, Host Requirements for Internet Hosts, clarified a number of TCP protocol implementation requirements. RFC 2581, TCP Congestion Control, one of the most important TCP-related RFCs in recent years, describes updated algorithms that avoid undue congestion. In 2001, RFC 3168 was written to describe explicit congestion notification (ECN), a congestion avoidance signaling mechanism.

The original TCP congestion avoidance algorithm was known as "TCP Tahoe", but many alternative algorithms have since been proposed (including TCP Reno, TCP Vegas, FAST TCP, TCP New Reno, and TCP Hybla).

TCP Interactive (iTCP)<sup>[28]</sup> is a research effort into TCP extensions that allows applications to subscribe to TCP events and register handler components that can launch applications for various purposes, including application-assisted congestion control.

Multipath TCP (MPTCP)<sup>[29][30]</sup> is an ongoing effort within the IETF that aims at allowing a TCP connection to use multiple paths to maximise resource usage and increase redundancy. The redundancy offered by Multipath TCP in the context of wireless networks<sup>[31]</sup> enables statistical multiplexing of resources, and thus increases TCP throughput

dramatically. Multipath TCP also brings performance benefits in datacenter environments.<sup>[32]</sup> The reference implementation<sup>[33]</sup> of Multipath TCP is being developed in the Linux kernel.<sup>[34]</sup>

TCP Cookie Transactions (TCPCT) is an extension proposed in December 2009 to secure servers against denial-of-service attacks. Unlike SYN cookies, TCPCT does not conflict with other TCP extensions such as window scaling. TCPCT was designed due to necessities of DNSSEC, where servers have to handle large numbers of short-lived TCP connections.

tcpcrypt is an extension proposed in July 2010 to provide transport-level encryption directly in TCP itself. It is designed to work transparently and not require any configuration. Unlike TLS (SSL), tcpcrypt itself does not provide authentication, but provides simple primitives down to the application to do that. As of 2010, the first tcpcrypt IETF draft has been published and implementations exist for several major platforms.

TCP Fast Open is an extension to speed up the opening of successive TCP connections between two endpoints. It works by skipping the three-way handshake using a cryptographic "cookie". It is similar to an earlier proposal called T/TCP, which was not widely adopted due to security issues.<sup>[35]</sup> As of July 2012, it is an IETF Internet draft.<sup>[36]</sup>

## TCP over wireless networks

TCP has been optimized for wired networks. Any packet loss is considered to be the result of network congestion and the congestion window size is reduced dramatically as a precaution. However, wireless links are known to experience sporadic and usually temporary losses due to fading, shadowing, hand off, and other radio effects, that cannot be considered congestion. After the (erroneous) back-off of the congestion window size, due to wireless packet loss, there can be a congestion avoidance phase with a conservative decrease in window size. This causes the radio link to be underutilized. Extensive research has been done on the subject of how to combat these harmful effects. Suggested solutions can be categorized as end-to-end solutions (which require modifications at the client or server),<sup>[37]</sup> link layer solutions (such as RLP in cellular networks), or proxy based solutions (which require some changes in the network without modifying end nodes).<sup>[37][38]</sup>

A number of alternative congestion control algorithms have been proposed to help solve the wireless problem, such as Vegas, Westwood, Veno and Santa Cruz.

## Hardware implementations

One way to overcome the processing power requirements of TCP is to build hardware implementations of it, widely known as TCP Offload Engines (TOE). The main problem of TOEs is that they are hard to integrate into computing systems, requiring extensive changes in the operating system of the computer or device. One company to develop such a device was Alacritech.

## Debugging

A packet sniffer, which intercepts TCP traffic on a network link, can be useful in debugging networks, network stacks and applications that use TCP by showing the user what packets are passing through a link. Some networking stacks support the SO\_DEBUG socket option, which can be enabled on the socket using setsockopt. That option dumps all the packets, TCP states, and events on that socket, which is helpful in debugging. Netstat is another utility that can be used for debugging.



## Alternatives

For many applications TCP is not appropriate. One problem (at least with normal implementations) is that the application cannot access the packets coming after a lost packet until the retransmitted copy of the lost packet is received. This causes problems for real-time applications such as streaming media, real-time multiplayer games and voice over IP (VoIP) where it is generally more useful to get most of the data in a timely fashion than it is to get all of the data in order.

For both historical and performance reasons, most storage area networks (SANs) prefer to use Fibre Channel protocol (FCP) instead of TCP/IP.

Also, for embedded systems, network booting, and servers that serve simple requests from huge numbers of clients (e.g. DNS servers) the complexity of TCP can be a problem. Finally, some tricks such as transmitting data between two hosts that are both behind NAT (using STUN or similar systems) are far simpler without a relatively complex protocol like TCP in the way.

Generally, where TCP is unsuitable, the User Datagram Protocol (UDP) is used. This provides the application multiplexing and checksums that TCP does, but does not handle streams or retransmission, giving the application developer the ability to code them in a way suitable for the situation, or to replace them with other methods like forward error correction or interpolation.

SCTP is another IP protocol that provides reliable stream oriented services similar to TCP. It is newer and considerably more complex than TCP, and has not yet seen widespread deployment. However, it is especially designed to be used in situations where reliability and near-real-time considerations are important.

Venturi Transport Protocol (VTP) is a patented proprietary protocol that is designed to replace TCP transparently to overcome perceived inefficiencies related to wireless data transport.

TCP also has issues in high bandwidth environments. The TCP congestion avoidance algorithm works very well for ad-hoc environments where the data sender is not known in advance, but if the environment is predictable, a timing based protocol such as Asynchronous Transfer Mode (ATM) can avoid TCP's retransmits overhead.

Multipurpose Transaction Protocol (MTP/IP) is patented proprietary software that is designed to adaptively achieve high throughput and transaction performance in a wide variety of network conditions, particularly those where TCP is perceived to be inefficient.

## Checksum computation

### TCP checksum for IPv4

When TCP runs over IPv4, the method used to compute the checksum is defined in RFC 793:

*The checksum field is the 16 bit one's complement of the one's complement sum of all 16-bit words in the header and text. If a segment contains an odd number of header and text octets to be checksummed, the last octet is padded on the right with zeros to form a 16-bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.*

In other words, after appropriate padding, all 16-bit words are added using one's complement arithmetic. The sum is then bitwise complemented and inserted as the checksum field. A pseudo-header that mimics the IPv4 packet header used in the checksum computation is shown in the table below.

### TCP pseudo-header for checksum computation (IPv4)

Bit offset	0–3	4–7	8–15	16–31
0	Source address			
32	Destination address			
64	Zeros		Protocol	TCP length
96	Source port			Destination port
128	Sequence number			
160	Acknowledgement number			
192	Data offset	Reserved	Flags	Window
224	Checksum			Urgent pointer
256	Options (optional)			
256/288+	Data			

The source and destination addresses are those of the IPv4 header. The protocol value is 6 for TCP (cf. List of IP protocol numbers). The TCP length field is the length of the TCP header and data.

### TCP checksum for IPv6

When TCP runs over IPv6, the method used to compute the checksum is changed, as per RFC 2460:

*Any transport or other upper-layer protocol that includes the addresses from the IP header in its checksum computation must be modified for use over IPv6, to include the 128-bit IPv6 addresses instead of 32-bit IPv4 addresses.*

A pseudo-header that mimics the IPv6 header for computation of the checksum is shown below.

### TCP pseudo-header for checksum computation (IPv6)

Bit offset	0–7		8–15	16–23	24–31
0	Source address				
32					
64					
96					
128	Destination address				
160					
192					
224					
256	TCP length				
288	Zeros				Next header
320	Source port			Destination port	
352	Sequence number				
384	Acknowledgement number				
416	Data offset	Reserved	Flags	Window	
448	Checksum			Urgent pointer	

480	Options (optional)
480/512+	Data

- Source address – the one in the IPv6 header
- Destination address – the final destination; if the IPv6 packet doesn't contain a Routing header, TCP uses the destination address in the IPv6 header, otherwise, at the originating node, it uses the address in the last element of the Routing header, and, at the receiving node, it uses the destination address in the IPv6 header.
- TCP length – the length of the TCP header and data
- Next Header – the protocol value for TCP

## Checksum offload

Many TCP/IP software stack implementations provide options to use hardware assistance to automatically compute the checksum in the network adapter prior to transmission onto the network or upon reception from the network for validation. This may relieve the OS from using precious CPU cycles calculating the checksum. Hence, overall network performance is increased.

This feature may cause packet analyzers detecting outbound network traffic upstream of the network adapter and unaware or uncertain about the use of checksum offload to report invalid checksum in outbound packets.

## References

- [1] Vinton G. Cerf, Robert E. Kahn, (May 1974). "A Protocol for Packet Network Intercommunication" (<http://ece.ut.ac.ir/Classpages/F84/PrincipleofNetworkDesign/Papers/CK74.pdf>). *IEEE Transactions on Communications* **22** (5): 637–648. .
- [2] Comer, Douglas E. (2006). *Internetworking with TCP/IP: Principles, Protocols, and Architecture*. **1** (5th ed.). Prentice Hall. ISBN 0-13-187671-6.
- [3] TCP (Linktionary term) (<http://www.linktionary.com/t/tcp.html>)
- [4] RFC 791 – section 2.1 (<http://tools.ietf.org/html/rfc791#section-2.1>)
- [5] RFC 793 (<http://tools.ietf.org/html/rfc793>)
- [6] RFC 793 section 3.1
- [7] RFC 1323, TCP Extensions for High Performance, Section 2.2 (<http://tools.ietf.org/html/rfc1323#page-9>)
- [8] RFC 2018, TCP Selective Acknowledgement Options, Section 2 (<http://tools.ietf.org/html/rfc2018#section-2>)
- [9] RFC 2018, TCP Selective Acknowledgement Options, Section 3 (<http://tools.ietf.org/html/rfc2018#section-3>)
- [10] RFC 1323, TCP Extensions for High Performance, Section 3.2 (<http://tools.ietf.org/html/rfc1323#page-11>)
- [11] RFC 1146, TCP Alternate Checksum Options (<http://tools.ietf.org/html/rfc1146#page-2>)
- [12] <http://www.medianet.kent.edu/techreports/TR2005-07-22-tcp-EFSM.pdf>
- [13] RFC 793 Section 3.2
- [14] Tanenbaum, Andrew S. (2003-03-17). *Computer Networks* (Fourth ed.). Prentice Hall. ISBN 0-13-066102-3.
- [15] <http://tools.ietf.org/html/rfc1122>
- [16] "TCP Definition" (<http://www.linfo.org/tcp.html>). . Retrieved 2011-03-12.
- [17] Stone; Partridge (2000). "When The CRC and TCP Checksum Disagree" (<http://citeseer.ist.psu.edu/stone00when.html>). *Sigcomm*.
- [18] RFC 879 (<http://www.faqs.org/rfcs/rfc879.html>)
- [19] TCP window scaling and broken routers [LWN.net] (<http://lwn.net/Articles/92727/>)
- [20] Gont, Fernando (2008-11). "On the implementation of TCP urgent data" (<http://www.gont.com.ar/talks/IETF73/ietf73-tcpm-urgent-data.ppt>). 73rd IETF meeting. . Retrieved 2009-01-04.
- [21] Peterson, Larry (2003). *Computer Networks*. Morgan Kaufmann. p. 401. ISBN 1-55860-832-X.
- [22] Richard W. Stevens (2006). *TCP/IP Illustrated. Vol. 1, The protocols* (<http://books.google.com/books?id=b2elQwAACAAJ>). Addison-Wesley. pp. Chapter 20. ISBN 978-0-201-63346-7. . Retrieved 14 November 2011.
- [23] Security Assessment of the Transmission Control Protocol (TCP) (<http://www.cpni.gov.uk/Docs/tn-03-09-security-assessment-TCP.pdf>)
- [24] Security Assessment of the Transmission Control Protocol (TCP) (<http://tools.ietf.org/html/draft-ietf-tcpm-tcp-security>)
- [25] Some insights about the recent TCP DoS (Denial of Service) vulnerabilities (<http://www.gont.com.ar/talks/hacklu2009/fgont-hacklu2009-tcp-security.pdf>)
- [26] Exploiting TCP and the Persist Timer Infiniteness (<http://phrack.org/issues.html?issue=66&id=9#article>)
- [27] Laurent Joncheray, *Simple Active Attack Against TCP*, 1995 (<http://www.usenix.org/publications/library/proceedings/security95/joncheray.html>)

- [28] TCP Interactive (iTCP) ([http://www.medianet.kent.edu/projects\\_files/projectITCP.html](http://www.medianet.kent.edu/projects_files/projectITCP.html))
- [29] RFC 6182
- [30] "TCP Extensions for Multipath Operation with Multiple Addresses draft-ietf-mptcp-multiaddressed-09" (<http://tools.ietf.org/html/draft-ietf-mptcp-multiaddressed-09>). [tools.ietf.org](http://tools.ietf.org/html/draft-ietf-mptcp-multiaddressed-09). Archived from the original (<http://datatracker.ietf.org/doc/draft-ietf-mptcp-multiaddressed/>) on June 6, 2012. .
- [31] TCP with feed-forward source coding for wireless downlink networks (<http://portal.acm.org/citation.cfm?id=1794199>)
- [32] Raiciu; Barre; Pluntke; Greenhalgh; Wischik; Handley (2011). "Improving datacenter performance and robustness with multipath TCP" (<http://inl.info.ucl.ac.be/publications/improving-datacenter-performance-and-robustness-multipath-tcp>). *Sigcomm*.
- [33] MultiPath TCP - Linux Kernel implementation (<http://mptcp.info.ucl.ac.be/>)
- [34] Barre; Paasch; Bonaventure (2011). "MultiPath TCP: From Theory to Practice" (<http://inl.info.ucl.ac.be/publications/multipath-tcp-theory-practice>). *IFIP Networking*.
- [35] Michael Kerrisk (2012-08-01). "TCP Fast Open: expediting web services" (<https://lwn.net/SubscriberLink/508865/39212876277c174c/>). LWN.net. .
- [36] Y. Cheng, J. Chu, S. Radhakrishnan, A. Jain (2012-07-16). *TCP Fast Open* (<https://tools.ietf.org/html/draft-ietf-tcpm-fastopen-01>). IETF. I-D draft-ietf-tcpm-fastopen-01. .
- [37] "TCP performance over CDMA2000 RLP" (<http://academic.research.microsoft.com/Paper/3352358.aspx>). . Retrieved 2010-08-30
- [38] Muhammad Adeel & Ahmad Ali Iqbal (2004). "TCP Congestion Window Optimization for CDMA2000 Packet Data Networks" (<http://www.computer.org/portal/web/csdl/doi/10.1109/ITNG.2007.190>). *International Conference on Information Technology (ITNG'07)*: 31–35. doi:10.1109/ITNG.2007.190. ISBN 978-0-7695-2776-5. .

## Further reading

- W. Richard Stevens. TCP/IP Illustrated, Volume 1: The Protocols. ISBN 0-201-63346-9
- W. Richard Stevens and Gary R. Wright. TCP/IP Illustrated, Volume 2: The Implementation. ISBN 0-201-63354-X
- W. Richard Stevens. TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP, and the UNIX Domain Protocols. ISBN 0-201-63495-3

## External links

### RFC

- RFC 675 – Specification of Internet Transmission Control Program, December 1974 Version
- RFC 793 – TCP v4
- RFC 1122 – includes some error corrections for TCP
- RFC 1323 – TCP-Extensions
- RFC 1379 – Extending TCP for Transactions—Concepts
- RFC 1948 – Defending Against Sequence Number Attacks
- RFC 2018 – TCP Selective Acknowledgment Options
- RFC 4614 – A Roadmap for TCP Specification Documents
- RFC 5681 – TCP Congestion Control
- RFC 6298 – Computing TCP's Retransmission Timer

### Others

- Oral history interview with Robert E. Kahn (<http://purl.umn.edu/107387>), Charles Babbage Institute, University of Minnesota, Minneapolis. Focuses on Kahn's role in the development of computer networking from 1967 through the early 1980s. Beginning with his work at Bolt Beranek and Newman (BBN), Kahn discusses his involvement as the ARPANET proposal was being written, his decision to become active in its implementation, and his role in the public demonstration of the ARPANET. The interview continues into Kahn's involvement with networking when he moves to IPTO in 1972, where he was responsible for the administrative and technical evolution of the ARPANET, including programs in packet radio, the development of a new network protocol (TCP/IP), and the switch to TCP/IP to connect multiple networks.

- IANA Port Assignments (<http://www.iana.org/assignments/port-numbers>)
- John Kristoff's Overview of TCP (Fundamental concepts behind TCP and how it is used to transport data between two endpoints) (<http://condor.depaul.edu/~jkristof/technotes/tcp.html>)
- TCP fast retransmit simulation animated: slow start, sliding window, duplicated Ack, congestion window ([http://www.visualland.net/tcp\\_history.php?simu=tcp\\_fast\\_retransmit&protocol=TCP&title=4.Fast transmit&ctype=1](http://www.visualland.net/tcp_history.php?simu=tcp_fast_retransmit&protocol=TCP&title=4.Fast%20transmit&ctype=1))
- TCP, Transmission Control Protocol (<http://www.networksorcery.com/enp/protocol/tcp.htm>)
- Checksum example (<http://mathforum.org/library/drmath/view/54379.html>)
- Engineer Francesco Buffa's page about Transmission Control Protocol (<http://www.ilmondodelletelecomunicazioni.it/english/telematics/protocols.html>)
- TCP tutorial (<http://www.ssfnet.org/Exchange/tcp/tcpTutorialNotes.html>)
- Linktionary on TCP segments ([http://www.linktionary.com/s/segment\\_tcp.html](http://www.linktionary.com/s/segment_tcp.html))
- TCP Sliding Window simulation animated (ns2) ([http://www.visualland.net/tcp\\_history.php?simu=tcp\\_swnd&protocol=TCP&title=2.Sliding Window&ctype=1](http://www.visualland.net/tcp_history.php?simu=tcp_swnd&protocol=TCP&title=2.Sliding%20Window&ctype=1))
- Multipath TCP in Linux kernel (<http://inl.info.ucl.ac.be/mptcp>)

## User Datagram Protocol

---

The **User Datagram Protocol (UDP)** is one of the core members of the Internet protocol suite, the set of network protocols used for the Internet. With UDP, computer applications can send messages, in this case referred to as *datagrams*, to other hosts on an Internet Protocol (IP) network without requiring prior communications to set up special transmission channels or data paths. The protocol was designed by David P. Reed in 1980 and formally defined in RFC 768.

UDP uses a simple transmission model with a minimum of protocol mechanism.<sup>[1]</sup> It has no handshaking dialogues, and thus exposes any unreliability of the underlying network protocol to the user's program. As this is normally IP over unreliable media, there is no guarantee of delivery, ordering or duplicate protection. UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram.

UDP is suitable for purposes where error checking and correction is either not necessary or performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system.<sup>[2]</sup> If error correction facilities are needed at the network interface level, an application may use the Transmission Control Protocol (TCP) or Stream Control Transmission Protocol (SCTP) which are designed for this purpose.

A number of UDP's attributes make it especially suited for certain applications.

- It is **transaction-oriented**, suitable for simple query-response protocols such as the Domain Name System or the Network Time Protocol.
  - It provides **datagrams**, suitable for modeling other protocols such as in IP tunneling or Remote Procedure Call and the Network File System.
  - It is **simple**, suitable for bootstrapping or other purposes without a full protocol stack, such as the DHCP and Trivial File Transfer Protocol.
  - It is **stateless**, suitable for very large numbers of clients, such as in streaming media applications for example IPTV
  - The **lack of retransmission delays** makes it suitable for real-time applications such as Voice over IP, online games, and many protocols built on top of the Real Time Streaming Protocol.
  - Works well in **unidirectional** communication, suitable for broadcast information such as in many kinds of service discovery and shared information such as broadcast time or Routing Information Protocol
-

## Service ports

UDP applications use datagram sockets to establish host-to-host communications. An application binds a socket to its endpoint of data transmission, which is a combination of an IP address and a service port. A port is a software structure that is identified by the port number, a 16 bit integer value, allowing for port numbers between 0 and 65535. Port 0 is reserved, but is a permissible source port value if the sending process does not expect messages in response.

The Internet Assigned Numbers Authority has divided port numbers into three ranges.<sup>[3]</sup> Port numbers 0 through 1023 are used for common, well-known services. On Unix-like operating systems, using one of these ports requires superuser operating permission. Port numbers 1024 through 49151 are the registered ports used for IANA-registered services. Ports 49152 through 65535 are dynamic ports that are not officially designated for any specific service, and can be used for any purpose. They are also used as ephemeral ports, from which software running on the host may randomly choose a port in order to define itself.<sup>[3]</sup> In effect, they are used as temporary ports primarily by clients when communicating with servers.

## Packet structure

UDP is a minimal message-oriented Transport Layer protocol that is documented in IETF RFC 768.

UDP provides no guarantees to the upper layer protocol for message delivery and the UDP protocol layer retains no state of UDP messages once sent. For this reason, UDP is sometimes referred to as *Unreliable* Datagram Protocol.<sup>[4]</sup>

UDP provides application multiplexing (via port numbers) and integrity verification (via checksum) of the header and payload.<sup>[5]</sup> If transmission reliability is desired, it must be implemented in the user's application.

Offset (bits)	Field
0	Source Port Number
16	Destination Port Number
32	Length
48	Checksum
64+	Data □

The UDP header consists of 4 fields, each of which is 2 bytes (16 bits).<sup>[2]</sup> The use of two of those is optional in IPv4 (pink background in table). In IPv6 only the source port is optional (see below).

### Source port number

This field identifies the sender's port when meaningful and should be assumed to be the port to reply to if needed. If not used, then it should be zero. If the source host is the client, the port number is likely to be an ephemeral port number. If the source host is the server, the port number is likely to be a well-known port number.<sup>[3]</sup>

### Destination port number

This field identifies the receiver's port and is required. Similar to source port number, if the client is the destination host then the port number will likely be an ephemeral port number and if the destination host is the server then the port number will likely be a well-known port number.<sup>[3]</sup>

### Length

A field that specifies the length in bytes of the entire datagram: header and data. The minimum length is 8 bytes since that's the length of the header. The field size sets a theoretical limit of 65,535 bytes (8 byte header + 65,527 bytes of data) for a UDP datagram. The practical limit for the data length which is imposed by the

underlying IPv4 protocol is 65,507 bytes (65,535 – 8 byte UDP header – 20 byte IP header).<sup>[3]</sup>

In IPv6 Jumbograms it is possible to have UDP packets of size greater than 65,535 bytes.<sup>[6]</sup> This allows for a maximum length value of 4,294,967,295 bytes ( $2^{32} - 1$ ) with 8 bytes representing the header and 4,294,967,287 bytes for data.

### Checksum

The checksum field is used for error-checking of the header *and* data. If no checksum is generated by the transmitter, the field uses the value all-zeros.<sup>[7]</sup> This field is not optional for IPv6.<sup>[8]</sup>

## Checksum computation

The method used to compute the checksum is defined in RFC 768:

*Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.*<sup>[7]</sup>

In other words, all 16-bit words are summed using one's complement arithmetic. The sum is then one's complemented to yield the value of the UDP checksum field.

If the checksum calculation results in the value zero (all 16 bits 0) it should be sent as the one's complement (all 1s).

The difference between IPv4 and IPv6 is in the data used to compute the checksum.

## Pseudo-Headers

### IPv4 Pseudo Header

When UDP runs over IPv4, the checksum is computed using a "pseudo header"<sup>[9]</sup> that contains some of the same information from the real IPv4 header. The pseudo header is not the real IPv4 header used to send an IP packet, it used only for the checksum calculation.

bits	0 – 7	8 – 15	16 – 23	24 – 31
0	Source address			
32	Destination address			
64	Zeros	Protocol	UDP length	
96	Source Port		Destination Port	
128	Length		Checksum	
160+	Data			

The source and destination addresses are those in the IPv4 header. The protocol is that for UDP (see *List of IP protocol numbers*): 17 (0x11). The UDP length field is the length of the UDP header and data.

UDP checksum computation is optional for IPv4. If a checksum is not used it should be set to the value zero.

## IPv6 Pseudo Header

When UDP runs over IPv6, the checksum is mandatory. The method used to compute it is changed as documented in RFC 2460:

*Any transport or other upper-layer protocol that includes the addresses from the IP header in its checksum computation must be modified for use over IPv6 to include the 128-bit IPv6 addresses.<sup>[8]</sup>*

When computing the checksum, again a pseudo header is used that mimics the real IPv6 header:

bits	0 – 7	8 – 15	16 – 23	24 – 31
0	Source address			
32				
64				
96				
128	Destination address			
160				
192				
224				
256	UDP length			
288	Zeros			Next Header
320	Source Port		Destination Port	
352	Length		Checksum	
384+	Data			

The source address is the one in the IPv6 header. The destination address is the final destination; if the IPv6 packet does not contain a Routing header, that will be the destination address in the IPv6 header; otherwise, at the originating node, it will be the address in the last element of the Routing header, and, at the receiving node, it will be the destination address in the IPv6 header. The value of the Next Header field is the protocol value for UDP: 17. The UDP length field is the length of the UDP header and data.

## Reliability and congestion control solutions

Lacking reliability, UDP applications must generally be willing to accept some loss, errors or duplication. Some applications such as TFTP may add rudimentary reliability mechanisms into the application layer as needed.<sup>[3]</sup>

Most often, UDP applications do not employ reliability mechanisms and may even be hindered by them. Streaming media, real-time multiplayer games and voice over IP (VoIP) are examples of applications that often use UDP. In these particular applications, loss of packets is not usually a fatal problem. If an application requires a high degree of reliability, a protocol such as the Transmission Control Protocol may be used instead.

Potentially more seriously, unlike TCP, UDP-based applications don't necessarily have good congestion avoidance and control mechanisms. Congestion insensitive UDP applications that consume a large fraction of available bandwidth could endanger the stability of the internet, as they frequently give a bandwidth load that is inelastic. Network-based mechanisms have been proposed to minimize potential congestion collapse effects of uncontrolled, high rate UDP traffic loads. Network-based elements such as routers using packet queuing and dropping techniques are often the only tool available to slow down excessive UDP traffic. The Datagram Congestion Control Protocol (DCCP) is being designed as a partial solution to this potential problem by adding end host TCP-friendly congestion control behavior to high-rate UDP streams such as streaming media.



## Applications

Numerous key Internet applications use UDP, including: the Domain Name System (DNS), where queries must be fast and only consist of a single request followed by a single reply packet, the Simple Network Management Protocol (SNMP), the Routing Information Protocol (RIP)<sup>[2]</sup> and the Dynamic Host Configuration Protocol (DHCP).

Voice and video traffic is generally transmitted using UDP. Real-time video and audio streaming protocols are designed to handle occasional lost packets, so only slight degradation in quality occurs, rather than large delays if lost packets were retransmitted. Because both TCP and UDP run over the same network, many businesses are finding that a recent increase in UDP traffic from these real-time applications is hindering the performance of applications using TCP, such as point of sale, accounting, and database systems. When TCP detects packet loss, it will throttle back its data rate usage. Since both real-time and business applications are important to businesses, developing quality of service solutions is seen as crucial by some.<sup>[10]</sup>

## Comparison of UDP and TCP

Transmission Control Protocol is a connection-oriented protocol, which means that it requires handshaking to set up end-to-end communications. Once a connection is set up user data may be sent bi-directionally over the connection.

- *Reliable* – TCP manages message acknowledgment, retransmission and timeout. Multiple attempts to deliver the message are made. If it gets lost along the way, the server will re-request the lost part. In TCP, there's either no missing data, or, in case of multiple timeouts, the connection is dropped.
- *Ordered* – if two messages are sent over a connection in sequence, the first message will reach the receiving application first. When data segments arrive in the wrong order, TCP buffers the out-of-order data until all data can be properly re-ordered and delivered to the application.
- *Heavyweight* – TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.
- *Streaming* – Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries.

UDP is a simpler message-based connectionless protocol. Connectionless protocols do not set up a dedicated end-to-end connection. Communication is achieved by transmitting information in one direction from source to destination without verifying the readiness or state of the receiver. However, one primary benefit of UDP over TCP is the application to voice over internet protocol (VoIP) where any handshaking would hinder clear voice communication. It is assumed in VoIP UDP that the end users provide any necessary real time confirmation that the message has been received.

- *Unreliable* – When a message is sent, it cannot be known if it will reach its destination; it could get lost along the way. There is no concept of acknowledgment, retransmission, or timeout.
  - *Not ordered* – If two messages are sent to the same recipient, the order in which they arrive cannot be predicted.
  - *Lightweight* – There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
  - *Datagrams* – Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent.
  - *No congestion control* – UDP itself does not avoid congestion, and it's possible for high bandwidth applications to trigger congestion collapse, unless they implement congestion control measures at the application level.
-

## References

- [1] RFC 768 p1
- [2] Kurose, J. F.; Ross, K. W. (2010). *Computer Networking: A Top-Down Approach* (5th ed.). Boston, MA: Pearson Education. ISBN 978-0-13-136548-3.
- [3] Forouzan, B.A. (2000). *TCP/IP: Protocol Suite, 1st ed.* New Delhi, India: Tata McGraw-Hill Publishing Company Limited.
- [4] content@ipv6.com. "UDP Protocol Overview" (<http://archive.is/20120710/http://ipv6.com/articles/general/User-Datagram-Protocol.htm>). Ipv6.com. Archived from the original (<http://ipv6.com/articles/general/User-Datagram-Protocol.htm>) on 2012-07-10. . Retrieved 17 August 2011.
- [5] Clark, M.P. (2003). *Data Networks IP and the Internet, 1st ed.* West Sussex, England: John Wiley & Sons Ltd.
- [6] RFC 2675
- [7] "Postel, J. (August 1980). RFC 768: User Datagram Protocol. *Internet Engineering Task Force*. Retrieved from" (<http://archive.is/20120722/http://tools.ietf.org/html/rfc768>). Archived from the original (<http://tools.ietf.org/html/rfc768>) on 2012-07-22. .
- [8] "Deering S. & Hinden R. (December 1998). RFC 2460: Internet Protocol, Version 6 (IPv6) Specification. *Internet Engineering Task Force*. Retrieved from" (<http://archive.is/20120915/http://tools.ietf.org/html/rfc2460>). Archived from the original (<http://tools.ietf.org/html/rfc2460>) on 2012-09-15. .
- [9] RFC 768, p2
- [10] "The impact of UDP on Data Applications" ([http://www.networkperformancedaily.com/2007/08/whiteboard\\_series\\_nice\\_guys\\_fi.html](http://www.networkperformancedaily.com/2007/08/whiteboard_series_nice_guys_fi.html)). Networkperformancedaily.com. . Retrieved 17 August 2011.

## RFC references

- RFC 768 – User Datagram Protocol
- RFC 2460 – Internet Protocol, Version 6 (IPv6) Specification
- RFC 2675 - IPv6 Jumbograms
- RFC 4113 – Management Information Base for the UDP
- RFC 5405 – Unicast UDP Usage Guidelines for Application Designers

## External links

- IANA Port Assignments (<http://www.iana.org/assignments/port-numbers>)
  - The Trouble with UDP Scanning (PDF) (<http://condor.depaul.edu/~jkristof/papers/udpscanning.pdf>)
  - Breakdown of UDP frame (<http://www.networksorcery.com/enp/protocol/udp.htm>)
  - UDP on MSDN Magazine Sockets and WCF (<http://msdn.microsoft.com/en-us/magazine/cc163648.aspx>)
  - UDP connections (<http://www.faqs.org/docs/iptables/udpconnections.html>)
-

---

# Internet Control Message Protocol

---

The **Internet Control Message Protocol (ICMP)** is one of the core protocols of the Internet Protocol Suite. It is chiefly used by the operating systems of networked computers to send error messages indicating, for example, that a requested service is not available or that a host or router could not be reached. ICMP can also be used to relay query messages.<sup>[1]</sup> It is assigned protocol number 1.<sup>[2]</sup>

ICMP<sup>[3]</sup> differs from transport protocols such as TCP and UDP in that it is not typically used to exchange data between systems, nor is it regularly employed by end-user network applications (with the exception of some diagnostic tools like ping and traceroute).

ICMP for Internet Protocol version 4 (IPv4) is also known as ICMPv4. IPv6 has a similar protocol, ICMPv6.

## Technical details

The Internet Control Message Protocol is part of the Internet Protocol Suite, as defined in RFC 792. ICMP messages are typically used for diagnostic or control purposes or generated in response to errors in IP operations (as specified in RFC 1122). ICMP errors are directed to the source IP address of the originating packet.<sup>[1]</sup>

For example, every device (such as an intermediate router) forwarding an IP datagram first decrements the time to live (TTL) field in the IP header by one. If the resulting TTL is 0, the packet is discarded and an ICMP Time To Live exceeded in transit message is sent to the datagram's source address.

Although ICMP messages are contained within standard IP datagrams, ICMP messages are usually processed as a special case, distinguished from normal IP processing, rather than processed as a normal sub-protocol of IP. In many cases, it is necessary to inspect the contents of the ICMP message and deliver the appropriate error message to the application that generated the original IP packet, the one that prompted the sending of the ICMP message.

Many commonly used network utilities are based on ICMP messages. The `tracert` (traceroute), `Pathping` commands are implemented by transmitting UDP datagrams with specially set IP TTL header fields, and looking for ICMP Time to live exceeded in transit (above) and "Destination unreachable" messages generated in response. The related ping utility is implemented using the ICMP "Echo request" and "Echo reply" messages.

## ICMP segment structure

### Header

The ICMP header starts after the IPv4 header. All ICMP packets will have an 8-byte header and variable-sized data section. The first 4 bytes of the header will be consistent. The first byte is for the ICMP type. The second byte is for the ICMP code. The third and fourth bytes are a checksum of the entire ICMP message. The contents of the remaining 4 bytes of the header will vary based on the ICMP type and code.<sup>[1]</sup>

ICMP error messages contain a data section that includes the entire IP header plus the first 8 bytes of data from the IP datagram that caused the error message. The ICMP datagram is then encapsulated in a new IP datagram.<sup>[1]</sup>

---

Bits	0-7	8-15	16-23	24-31
0	Type	Code	Checksum	
32	Rest of Header			

- **Type** – ICMP type as specified below.
- **Code** – Subtype to the given type.
- **Checksum** – Error checking data. Calculated from the ICMP header+data, with value 0 for this field. The checksum algorithm is specified in RFC 1071 <sup>[4]</sup>.
- **Rest of Header** – Four byte field. Will vary based on the ICMP type and code.

### List of permitted control messages (incomplete list)

Type	Code	Description
0 – Echo Reply <sup>[5]</sup>	0	Echo reply (used to ping)
1 and 2		<i>Reserved</i>
3 – Destination Unreachable <sup>[6]</sup>	0	Destination network unreachable
	1	Destination host unreachable
	2	Destination protocol unreachable
	3	Destination port unreachable
	4	Fragmentation required, and DF flag set
	5	Source route failed
	6	Destination network unknown
	7	Destination host unknown
	8	Source host isolated
	9	Network administratively prohibited
	10	Host administratively prohibited
	11	Network unreachable for TOS
	12	Host unreachable for TOS
	13	Communication administratively prohibited
	14	Host Precedence Violation
	15	Precedence cutoff in effect
4 – Source Quench	0	Source quench (congestion control)
5 – Redirect Message	0	Redirect Datagram for the Network
	1	Redirect Datagram for the Host
	2	Redirect Datagram for the TOS & network
	3	Redirect Datagram for the TOS & host
6		Alternate Host Address
7		<i>Reserved</i>
8 – Echo Request	0	Echo request (used to ping)
9 – Router Advertisement	0	Router Advertisement
10 – Router Solicitation	0	Router discovery/selection/solicitation

11 – Time Exceeded <sup>[7]</sup>	0	TTL expired in transit
	1	Fragment reassembly time exceeded
12 – Parameter Problem: Bad IP header	0	Pointer indicates the error
	1	Missing a required option
	2	Bad length
13 – Timestamp	0	Timestamp
14 – Timestamp Reply	0	Timestamp reply
15 – Information Request	0	Information Request
16 – Information Reply	0	Information Reply
17 – Address Mask Request	0	Address Mask Request
18 – Address Mask Reply	0	Address Mask Reply
19		<i>Reserved</i> for security
20 through 29		<i>Reserved</i> for robustness experiment
30 – Traceroute	0	Information Request
31		Datagram Conversion Error
32		Mobile Host Redirect
33		Where-Are-You (originally meant for IPv6)
34		Here-I-Am (originally meant for IPv6)
35		Mobile Registration Request
36		Mobile Registration Reply
37		Domain Name Request
38		Domain Name Reply
39		SKIP Algorithm Discovery Protocol, Simple Key-Management for Internet Protocol
40		Photuris, Security failures
41		ICMP for experimental mobility protocols such as Seamoby [RFC4065]
42 through 255		<i>Reserved</i>

(Sources: IANA ICMP Parameters<sup>[8]</sup> [9] and *Computer Networking – A Top-Down Approach* by Kurose and Ross)  
 //

## References

- [1] Forouzan, Behrouz A. (2007). *Data Communications And Networking* (Fourth ed.). Boston: McGraw-Hill. pp. 621–630. ISBN 0-07-296775-7.
- [2] "Protocol Numbers" (<http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml>). . Internet Assigned Numbers Authority. . Retrieved 2011-06-23.
- [3] Postel, J. (September 1981). *Internet Control Message Protocol* (<https://tools.ietf.org/html/rfc792>). IETF. RFC 792. .
- [4] <http://tools.ietf.org/html/rfc1071>
- [5] <http://tools.ietf.org/html/rfc792#page-14>
- [6] <http://tools.ietf.org/html/rfc792#page-4>
- [7] <http://tools.ietf.org/html/rfc792#page-6>
- [8] <http://www.iana.org/assignments/icmp-parameters>
- [9] [http://freebie.fatpipe.org/~mjb/Drawings/UDP\\_ICMP\\_Headers.png](http://freebie.fatpipe.org/~mjb/Drawings/UDP_ICMP_Headers.png)

## External links

- RFCs
  - RFC 792, *Internet Control Message Protocol*
  - RFC 1122, *Requirements for Internet Hosts – Communication Layers*
  - RFC 1716, *Towards Requirements for IP Router*
  - RFC 1812, *Requirements for IP Version 4 Routers*
- IANA ICMP parameters (<http://www.iana.org/assignments/icmp-parameters>)
- IANA protocol numbers (<http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml>)

# Hypertext Transfer Protocol

---

The **Hypertext Transfer Protocol (HTTP)** is an application protocol for distributed, collaborative, hypermedia information systems.<sup>[1]</sup> HTTP is the foundation of data communication for the World Wide Web.

Hypertext is a multi-linear set of objects, building a network by using logical links (the so-called hyperlinks) between the nodes (e.g. text or words). HTTP is the protocol to exchange or transfer hypertext.

The standards development of HTTP was coordinated by the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C), culminating in the publication of a series of Requests for Comments (RFCs), most notably RFC 2616 (June 1999), which defines HTTP/1.1, the version of HTTP in common use.

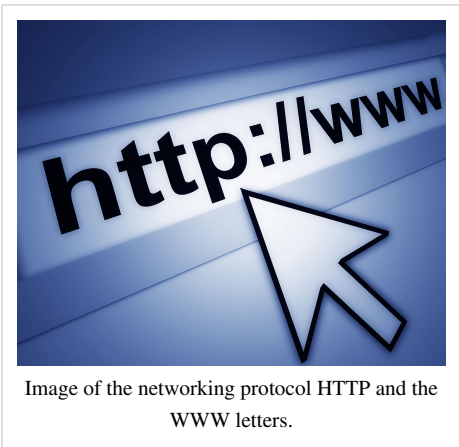
## Technical overview

HTTP functions as a request-response protocol in the client-server computing model. A web browser, for example, may be the *client* and an application running on a computer hosting a web site may be the *server*. The client submits an HTTP *request* message to the server. The server, which provides *resources* such as HTML files and other content, or performs other functions on behalf of the client, returns a *response* message to the client. The response contains completion status information about the request and may also contain requested content in its message body.

A web browser is an example of a *user agent* (UA). Other types of user agent include the indexing software used by search providers (web crawlers), voice browsers, mobile apps and other software that accesses, consumes or displays web content.

HTTP is designed to permit intermediate network elements to improve or enable communications between clients and servers. High-traffic websites often benefit from web cache servers that deliver content on behalf of upstream servers to improve response time. Web browsers cache previously accessed web resources and reuse them when possible to reduce network traffic. HTTP proxy servers at private network boundaries can facilitate communication for clients without a globally routable address, by relaying messages with external servers.

HTTP is an application layer protocol designed within the framework of the Internet Protocol Suite. Its definition presumes an underlying and reliable transport layer protocol,<sup>[2]</sup> and Transmission Control Protocol (TCP) predominates for this purpose. However HTTP can use unreliable protocols such as the User Datagram Protocol (UDP), for example in Simple Service Discovery Protocol (SSDP).



HTTP resources are identified and located on the network by Uniform Resource Identifiers (URIs)—or, more specifically, Uniform Resource Locators (URLs)—using the `http` or `https` URI schemes. URIs and hyperlinks in Hypertext Markup Language (HTML) documents form *webs* of inter-linked hypertext documents. On the Internet the World Wide Web was established in 1990 by English computer scientist and innovator Tim Berners-Lee.

HTTP/1.1 is a revision of the original HTTP (HTTP/1.0). In HTTP/1.0 a separate connection to the same server is made for every resource request. HTTP/1.1 can reuse a connection multiple times to download images, scripts, stylesheets *et cetera* after the page has been delivered. HTTP/1.1 communications therefore experience less latency as the establishment of TCP connections presents considerable overhead.

## History

The term HyperText was coined by Ted Nelson who in turn was inspired by Vannevar Bush's microfilm-based "memex". Tim Berners-Lee first proposed the "WorldWideWeb" project — now known as the World Wide Web. Berners-Lee and his team are credited with inventing the original HTTP along with HTML and the associated technology for a web server and a text-based web browser. The first version of the protocol had only one method, namely GET, which would request a page from a server.<sup>[3]</sup> The response from the server was always an HTML page.<sup>[4]</sup>



Tim Berners-Lee

The first documented version of HTTP was **HTTP V0.9**<sup>[5]</sup> (1991).

Dave Raggett led the HTTP Working Group (HTTP WG) in 1995 and wanted to expand the protocol with extended operations, extended negotiation, richer meta-information, tied with a security protocol which became more efficient by adding additional methods and header fields.<sup>[6][7]</sup> RFC 1945 officially introduced and recognized HTTP V1.0 in 1996.

The HTTP WG planned to publish new standards in December 1995<sup>[8]</sup> and the support for pre-standard HTTP/1.1 based on the then developing RFC 2068 (called HTTP-NG) was rapidly adopted by the major browser developers in early 1996. By March 1996, pre-standard HTTP/1.1 was supported in Arena,<sup>[9]</sup> Netscape 2.0,<sup>[9]</sup> Netscape Navigator Gold 2.01,<sup>[9]</sup> Mosaic 2.7, Lynx 2.5, and in Internet Explorer 2.0. End-user adoption of the new browsers was rapid. In March 1996, one web hosting company reported that over 40% of browsers in use on the Internet were HTTP 1.1 compliant. That same web hosting company reported that by June 1996, 65% of all browsers accessing their servers were HTTP/1.1 compliant.<sup>[10]</sup> The HTTP/1.1 standard as defined in RFC 2068 was officially released in January 1997. Improvements and updates to the HTTP/1.1 standard were released under RFC 2616 in June 1999.

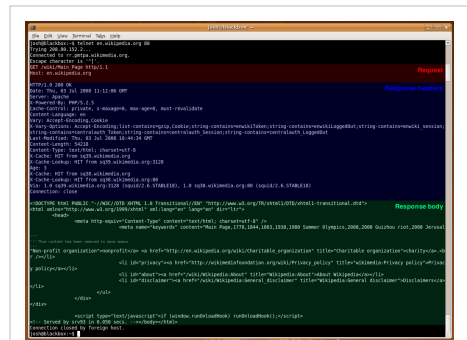
## HTTP session

An HTTP session is a sequence of network request-response transactions. An HTTP client initiates a request by establishing a Transmission Control Protocol (TCP) connection to a particular port on a server (typically port 80; see List of TCP and UDP port numbers). An HTTP server listening on that port waits for a client's request message. Upon receiving the request, the server sends back a status line, such as "HTTP/1.1 200 OK", and a message of its own. The body of this message is typically the requested resource, although an error message or other information may also be returned.<sup>[1]</sup>

## Request methods

HTTP defines methods (sometimes referred to as "verbs") to indicate the desired action to be performed on the identified **resource**. What this resource represents, whether pre-existing data or data that is generated dynamically, depends on the implementation of the server. Often, the resource corresponds to a file or the output of an executable residing on the server.

The HTTP/1.0 specification<sup>[11]</sup>:section 8 defined the GET, POST and HEAD methods and the HTTP/1.1 specification<sup>[1]</sup>:section 9 added 5 new methods: OPTIONS, PUT, DELETE, TRACE and CONNECT. By being specified in these documents their semantics is known and can be depended upon. Any client can use any method that they want and the server can choose to support any method it wants. If a method is unknown to an intermediate it will be treated as an un-safe and non-idempotent method. There is no limit to the number of methods that can be defined and this allows for future methods to be specified without breaking existing infrastructure. For example WebDAV defined 7 new methods and RFC5789 specified the PATCH method.



An HTTP request made using telnet. The request, response headers and response body are highlighted.

### HEAD

Asks for the response identical to the one that would correspond to a GET request, but without the response body. This is useful for retrieving meta-information written in response headers, without having to transport the entire content.

### GET

Requests a representation of the specified resource. Requests using GET should only retrieve data and should have no other effect. (This is also true of some other HTTP methods.)<sup>[1]</sup> The W3C has published guidance principles on this distinction, saying, "Web application design should be informed by the above principles, but also by the relevant limitations."<sup>[12]</sup> See safe methods below.

### POST

Submits data to be processed (e.g., from an HTML form) to the identified resource. The data is included in the body of the request. This may result in the creation of a new resource or the updates of existing resources or both.

### PUT

Uploads a representation of the specified resource.

### DELETE

Deletes the specified resource.

### TRACE

Echoes back the received request, so that a client can see what (if any) changes or additions have been made by intermediate servers.

### OPTIONS

Returns the HTTP methods that the server supports for specified URL. This can be used to check the functionality of a web server by requesting '\*' instead of a specific resource.

### CONNECT

Converts the request connection to a transparent TCP/IP tunnel, usually to facilitate SSL-encrypted communication (HTTPS) through an unencrypted HTTP proxy.<sup>[13]</sup>

### PATCH



Is used to apply partial modifications to a resource.<sup>[14]</sup>

HTTP servers are required to implement at least the GET and HEAD methods<sup>[15]</sup> and, whenever possible, also the OPTIONS method.

## Safe methods

Some methods (for example, HEAD, GET, OPTIONS and TRACE) are defined as *safe*, which means they are intended only for information retrieval and should not change the state of the server. In other words, they should not have side effects, beyond relatively harmless effects such as logging, caching, the serving of banner advertisements or incrementing a web counter. Making arbitrary GET requests without regard to the context of the application's state should therefore be considered safe.

By contrast, methods such as POST, PUT and DELETE are intended for actions that may cause side effects either on the server, or external side effects such as financial transactions or transmission of email. Such methods are therefore not usually used by conforming web robots or web crawlers; some that do not conform tend to make requests without regard to context or consequences.

Despite the prescribed safety of *GET* requests, in practice their handling by the server is not technically limited in any way. Therefore, careless or deliberate programming can cause non-trivial changes on the server. This is discouraged, because it can cause problems for Web caching, search engines and other automated agents, which can make unintended changes on the server.

## Idempotent methods and web applications

Methods PUT and DELETE are defined to be idempotent, meaning that multiple identical requests should have the same effect as a single request (Note that idempotence refers to the state of the system after the request has completed, so while the action the server takes (e.g. deleting a record) or the response code it returns may be different on subsequent requests, the system state will be the same every time). Methods GET, HEAD, OPTIONS and TRACE, being prescribed as safe, should also be idempotent, as HTTP is a stateless protocol.<sup>[1]</sup>

In contrast, the POST method is not necessarily idempotent, and therefore sending an identical POST request multiple times may further affect state or cause further side effects (such as financial transactions). In some cases this may be desirable, but in other cases this could be due to an accident, such as when a user does not realize that their action will result in sending another request, or they did not receive adequate feedback that their first request was successful. While web browsers may show alert dialog boxes to warn users in some cases where reloading a page may re-submit a POST request, it is generally up to the web application to handle cases where a POST request should not be submitted more than once.

Note that whether a method is idempotent is not enforced by the protocol or web server. It is perfectly possible to write a web application in which (for example) a database insert or other non-idempotent action is triggered by a GET or other request. Ignoring this recommendation, however, may result in undesirable consequences, if a user agent assumes that repeating the same request is safe when it isn't.

## Security

Implementing methods such as TRACE, TRACK and DEBUG is considered potentially insecure by some security professionals, because they can be used by attackers to gather information or bypass security controls during attacks. Security software tools such as "Tenable Nessus" and "Microsoft UrlScan Security Tool" report on the presence of these methods as being security issues.<sup>[16]</sup>

## Status codes

In HTTP/1.0 and since, the first line of the HTTP response is called the *status line* and includes a numeric *status code* (such as "404") and a textual *reason phrase* (such as "Not Found"). The way the user agent handles the response primarily depends on the code and secondarily on the response headers. Custom status codes can be used since, if the user agent encounters a code it does not recognize, it can use the first digit of the code to determine the general class of the response.<sup>[17]</sup>

Also, the standard *reason phrases* are only recommendations and can be replaced with "local equivalents" at the web developer's discretion. If the status code indicated a problem, the user agent might display the *reason phrase* to the user to provide further information about the nature of the problem. The standard also allows the user agent to attempt to interpret the *reason phrase*, though this might be unwise since the standard explicitly specifies that status codes are machine-readable and *reason phrases* are human-readable.

## Persistent connections

In HTTP/0.9 and 1.0, the connection is closed after a single request/response pair. In HTTP/1.1 a keep-alive-mechanism was introduced, where a connection could be reused for more than one request. Such *persistent connections* reduce request latency perceptibly, because the client does not need to re-negotiate the TCP connection after the first request has been sent. Another positive side effect is that in general the connection becomes faster with time due to TCP's slow-start-mechanism.

Version 1.1 of the protocol also made bandwidth optimization improvements to HTTP/1.0. For example, HTTP/1.1 introduced chunked transfer encoding to allow content on persistent connections to be streamed rather than buffered. HTTP pipelining further reduces lag time, allowing clients to send multiple requests before waiting for each response. Another improvement to the protocol was byte serving, where a server transmits just the portion of a resource explicitly requested by a client.

## HTTP session state

HTTP is a stateless protocol. A stateless protocol does not require the server to retain information or status about each user for the duration of multiple requests.

But some web applications may have to track the user's progress from page to page, for example when a web server is required to customize the content of a web page for a user. Solutions for these cases include:

- the use of HTTP cookies.
- server side sessions,
- hidden variables (when the current page contains a form), and
- URL-rewriting using URI-encoded parameters, e.g.,  
/index.php?session\_id=some\_unique\_session\_code.

## Secure HTTP

There are three methods of establishing a secure HTTP connection: HTTP Secure, Secure Hypertext Transfer Protocol and the HTTP/1.1 Upgrade header. Browser support for the latter two is, however, nearly non-existent, so HTTP Secure is the dominant method of establishing a secure HTTP connection.

## Request message

The request message consists of the following:

- A request line, for example `GET /images/logo.png HTTP/1.1`, which requests a resource called `/images/logo.png` from the server.
- Headers, such as `Accept-Language: en`
- An empty line.
- An optional message body.

The request line and headers must all end with `<CR><LF>` (that is, a carriage return character followed by a line feed character). The empty line must consist of only `<CR><LF>` and no other whitespace.<sup>[18]</sup> In the HTTP/1.1 protocol, all headers except `Host` are optional.

A request line containing only the path name is accepted by servers to maintain compatibility with HTTP clients before the HTTP/1.0 specification in RFC 1945.<sup>[19]</sup>

## Response message

The response message consists of the following:

- A Status-Line (for example `HTTP/1.1 200 OK`, which indicates that the client's request succeeded)
- Headers, such as `Content-Type: text/html`
- An empty line
- An optional message body

The Status-Line and headers must all end with `<CR><LF>` (a carriage return followed by a line feed). The empty line must consist of only `<CR><LF>` and no other whitespace.<sup>[18]</sup>

## Example session

Below is a sample conversation between an HTTP client and an HTTP server running on `www.example.com`, port 80.

### Client request

```
GET /index.html HTTP/1.1\r\n
Host: www.example.com\r\n
\r\n
```

A client request (consisting in this case of the request line and only one header) is followed by a blank line, so that the request ends with a double newline, each in the form of a carriage return followed by a line feed. The "Host" header distinguishes between various DNS names sharing a single IP address, allowing name-based virtual hosting. While optional in HTTP/1.0, it is mandatory in HTTP/1.1.

## Server response

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Etag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: none
Content-Length: 438
Connection: close
Content-Type: text/html; charset=UTF-8
```

The ETag (entity tag) header is used to determine if a cached version of the requested resource is identical to the current version of the resource on the server. *Content-Type* specifies the Internet media type of the data conveyed by the HTTP message, while *Content-Length* indicates its length in bytes. The HTTP/1.1 webserver publishes its ability to respond to requests for certain byte ranges of the document by setting the header *Accept-Ranges: bytes*. This is useful, if the client needs to have only certain portions<sup>[20]</sup> of a resource sent by the server, which is called byte serving. When *Connection: close* is sent in a header, it means that the web server will close the TCP connection immediately after the transfer of this response.

Most of the header lines are optional. When *Content-Length* is missing the length is determined in other ways. Chunked transfer encoding uses a chunk size of 0 to mark the end of the content. *Identity* encoding without *Content-Length* reads content until the socket is closed.

A *Content-Encoding* like *gzip* can be used to compress the transmitted data.

## References

- [1] Fielding, Roy T.; Gettys, James; Mogul, Jeffrey C.; Nielsen, Henrik Frystyk; Masinter, Larry; Leach, Paul J.; Berners-Lee (June 1999). "RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1" (<http://tools.ietf.org/html/rfc2616>). .
- [2] Fielding, et al. Internet RFC 2616." (<http://www.w3.org/Protocols/rfc2616/rfc2616-sec1.html#sec1.4>), section 1.4. Retrieved on January 21, 2009.
- [3] Berners-Lee, Tim. "HyperText Transfer Protocol" (<http://www.w3.org/History/19921103-hypertext/hypertext/WWW/Protocols/HTTP.html>). World Wide Web Consortium. . Retrieved 31 August 2010.
- [4] Tim Berners-Lee. "The Original HTTP as defined in 1991" (<http://www.w3.org/Protocols/HTTP/AsImplemented.html>). World Wide Web Consortium. . Retrieved 24 July 2010.
- [5] <http://www.w3.org/pub/WWW/Protocols/HTTP/AsImplemented.html>
- [6] Raggett, Dave. "Dave Raggett's Bio" (<http://www.w3.org/People/Raggett/profile.html>). World Wide Web Consortium. . Retrieved 11 June 2010.
- [7] Raggett, Dave; Berners-Lee, Tim. "Hypertext Transfer Protocol Working Group" (<http://www.w3.org/Arena/webworld/httpwgcharter.html>). World Wide Web Consortium. . Retrieved 29 September 2010.
- [8] Raggett, Dave. "HTTP WG Plans" (<http://www.w3.org/Arena/webworld/httpwgplans.html>). World Wide Web Consortium. . Retrieved 29 September 2010.
- [9] Simon Spero. "Progress on HTTP-NG" (<http://www.w3.org/Protocols/HTTP-NG/http-ng-status.html>). World Wide Web Consortium. . Retrieved 11 June 2010.
- [10] "HTTP/1.1" (<http://www.webcom.com/glossary/http1.1.shtml>). *Webcom.com Glossary entry*. . Retrieved 2009-05-29.
- [11] Berners-Lee, Tim; Fielding, Roy T.; Nielsen, Henrik Frystyk. *RFC 1945: Hypertext Transfer Protocol -- HTTP/1.0* (<http://tools.ietf.org/html/rfc1945>). .
- [12] Jacobs, Ian (2004). "URIs, Addressability, and the use of HTTP GET and POST" (<http://www.w3.org/2001/tag/doc/whenToUseGet.html#checklist>). *Technical Architecture Group finding*. W3C. . Retrieved 26 September 2010.
- [13] "Vulnerability Note VU#150227: HTTP proxy default configurations allow arbitrary TCP connections" (<http://www.kb.cert.org/vuls/id/150227>). US-CERT. 2002-05-17. . Retrieved 2007-05-10.
- [14] Dusseault, Lisa; Snell, James M.. "RFC 5789: PATCH Method for HTTP" (<http://tools.ietf.org/html/rfc5789>). .
- [15] "HTTP 1.1 Section 5.1.1" (<http://tools.ietf.org/html/rfc2616#section-5.1.1>). Tools.ietf.org. . Retrieved 2010-08-01.
- [16] "UrlScan Security Tool" (<http://technet.microsoft.com/en-us/security/cc242650.aspx>). *Security TechCenter*. Microsoft. . Retrieved 15 Jul 2012.

- [17] "6.1 Status-Line" (<http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html#sec6.1>). W3.org. . Retrieved 2010-08-01.
- [18] Fielding (June 1999). "HTTP/1.1" (<http://www.w3.org/Protocols/rfc2616/rfc2616-sec4.html>). IETF. . Retrieved 19 January 2012.
- [19] "Apache Week. HTTP/1.1" (<http://www.apacheweek.com/features/http11>). . 090502 apacheweek.com
- [20] Tools.ietf.org (<http://tools.ietf.org/html/draft-ietf-http-range-retrieval-00>), Byte Range Retrieval Extension to HTTP

## Further reading

- HTTP 0.9 - As Implemented in 1991 (<http://www.w3.org/Protocols/HTTP/AsImplemented.html>)

## External links

- "Change History for HTTP" (<http://www.w3.org/Protocols/History.html>). W3.org. Retrieved 2010-08-01. A detailed technical history of HTTP.
- "Design Issues for HTTP" (<http://www.w3.org/Protocols/DesignIssues.html>). W3.org. Retrieved 2010-08-01. Design Issues by Berners-Lee when he was designing the protocol.
- "Classic HTTP Documents" (<http://www.w3.org/Protocols/Classic.html>). W3.org. 1998-05-14. Retrieved 2010-08-01. list of other classic documents recounting the early protocol history

# Skype protocol

---

The **Skype protocol** is a proprietary Internet telephony network based on peer-to-peer architecture, used by Skype. The protocol's specifications have not been made publicly available by Skype and official applications using the protocol are closed-source.

The Skype network is not interoperable with most other VoIP networks without proper licensing from Skype. Digium, the main sponsor of Asterisk PBX released a driver licensed by Skype dubbed 'Skype for Asterisk' to interface as a client to the Skype network, however this still remains closed source.<sup>[1]</sup> Numerous attempts to study and/or reverse engineer the protocol have been undertaken to reveal the protocol, investigate security or to allow unofficial clients.

## Peer-to-peer architecture

Skype was the first peer-to-peer IP telephony network,<sup>[2]</sup> requiring minimal centralized infrastructure. The Skype user directory is decentralized and distributed among the clients, or nodes, in the network.

The network contains three types of entities: *supernodes*, ordinary nodes, and the login server. Each client maintains a *host cache* with the IP address and port numbers of reachable supernodes.

Any client with good bandwidth, no restriction due to firewall or NAT, and adequate processing power can become a supernode. This puts an extra burden on those who connect to the Internet without NAT, as Skype may use their computers and Internet connections as third party for UDP hole punching (to directly connect two clients both behind NAT) or to completely relay other users' calls. Skype does not choose to supply server power with associated bandwidth required to provide the relay service for every client who needs it, instead it uses the resource of Skype clients.<sup>[3]</sup>

Supernodes relay communications on behalf of two other clients, both of which are behind firewalls or "one to many" Network address translation. The reason that relaying is required is that without relaying clients with firewall or NAT difficulties, the two clients would be unable to make or receive calls from other. Skype tries to get the two ends to negotiate the connection details directly, but what can happen is that the sum of problems at both ends can mean that two cannot establish direct conversation.

The problems with firewalls and NAT can be

- The external port numbers or IP address are not derivable, because NAT rewrites them,
- The firewall and NAT in use prevents the session being received
- UDP is not usable due to NAT issues, such as timeout
- firewalls block many ports
- TCP through many to one NAT is always "outward only" by default - Adding Port Forwarding settings to the NAT router can allow receiving TCP sessions

Supernodes are grouped into *slots* (9-10 supernodes), and slots are grouped into *blocks* (8 slots).

## Protocol

Signaling is encrypted using RC4; however, the method only obfuscates the traffic as the key can be recovered from the packet. Voice data is encrypted with AES.<sup>[4]</sup>

The Skype client's application programming interface (API) opens the network to software developers. The Skype API allows other programs to use the Skype network to get "white pages" information and manage calls.

The Skype code is closed source, and the protocol is not standardized.<sup>[5]</sup> Parts of the client use Internet Direct (Indy), an open source socket communication library.

8 July 2012, a searcher from Benin, Ouanilo Medegan, released articles and a proof of concept client source code, results of *reverse engineering* on the Skype client<sup>[6]</sup>.

## Protocol detection

Many networking and security companies claim to detect and control Skype's protocol for enterprise and carrier applications. While the specific detection methods used by these companies are often proprietary, Pearson's chi-squared test and stochastic characterization with Naive Bayes classifiers are two approaches that were published in 2007.<sup>[7]</sup>

## Preliminaries

Abbreviations that are used:

- SN: Skype network
- SC: Skype client
- HC: host cache

## Skype client

The main functions of a Skype client are:

- login
  - user search
  - start and end calls
  - media transfer
  - presence messages
  - video conference
-

## Login

A Skype client authenticates the user with the login server, advertises its presence to other peers, determines the type of NAT and firewall it is behind and discovers nodes that have public IP addresses.

To connect to the Skype network, the host cache must contain a valid entry. A TCP connection must be established (i.e. to a supernode) otherwise the login will fail.

```
1.  start
2.  send UDP packet(s) to HC
3.  if no response within 5 seconds then
4.      attempt TCP connection with HC
5.      if not connected then
6.          attempt TCP connection with HC on port 80 (HTTP)
7.          if not connected then
8.              attempt TCP connection with HC on port 443 (HTTPS)
9.              if not connected then
10.                  attempts++
11.                  if attempts==5 then
12.                      fail
13.                  else
14.                      wait 6 seconds
15.                      goto step 2
16.  Success
```

After a Skype client is connected it must authenticate the username and password with the Skype login server. There are many different Skype login servers using different ports. An obfuscated list of servers is hardcoded in the Skype executable.

Skype servers are:

- dir1.sd.skype.net:9010
- dir2.sd.skype.net:9010
- dir3.sd.skype.net:9010
- dir4.sd.skype.net:9010
- dir5.sd.skype.net:9010
- dir6.sd.skype.net:9010
- dir7.sd.skype.net:9010
- dir8.sd.skype.net:9010
- http1.sd.skype.net:80
- http2.sd.skype.net:80
- http3.sd.skype.net:80
- http4.sd.skype.net:80
- http5.sd.skype.net:80
- http6.sd.skype.net:80
- http7.sd.skype.net:80
- http8.sd.skype.net:80

Skype-SW connects randomly to 1-8.

On each login session, Skype generates a session key from 192 random bits. The session key is encrypted with the hard-coded login server's 1536-bit RSA key to form an encrypted session key. Skype also generates a 1024-bit private/public RSA key pair. An MD5 hash of a concatenation of the user name, constant string ("nSkyper\n") and

password is used as a shared secret with the login server. The plain session key is hashed into a 256-bit AES key that is used to encrypt the session's public RSA key and the shared secret. The encrypted session key and the AES encrypted value are sent to the login server.

On the login server side, the plain session key is obtained by decrypting the encrypted session key using the login server's private RSA key. The plain session key is then used to decrypt the session's public RSA key and the shared secret. If the shared secret match, the login server will sign the user's public RSA key with its private key. The signed data is dispatched to the super nodes.

Upon searching for a buddy, a super node will return the buddy's public key signed by Skype. The SC will authenticate the buddy and agree on a session key by using the mentioned RSA key.

## UDP

UDP packets:

```
IP
UDP
Skype SoF
Skype Crypted Data01
```

The Start of Frame (SoF) consists of:

1. frame ID number (2 bytes)
2. payload type (1 byte)
  - obfuscated payload
  - Ack/NAck packet
  - payload forwarding packet
  - payload resending packet
  - other

## Obfuscation Layer

The RC4 encryption algorithm is used to obfuscate the payload of datagrams.

1. The CRC32 of public source and destination IP, Skype's packet ID are taken
2. Skype obfuscation layer's initialization vector (IV).

The XOR of these two 32-bit values is transformed to an 80-byte RC4 key using an unknown key engine.

A notable misuse of RC4 in Skype can be found on TCP streams (UDP is unaffected). The first 14 bytes (10 of which are known) are XOR-ed with the RC4 stream. Then, the cipher is reinitialized to encrypt the rest of the TCP stream.<sup>[8]</sup>



## TCP

TCP packets:

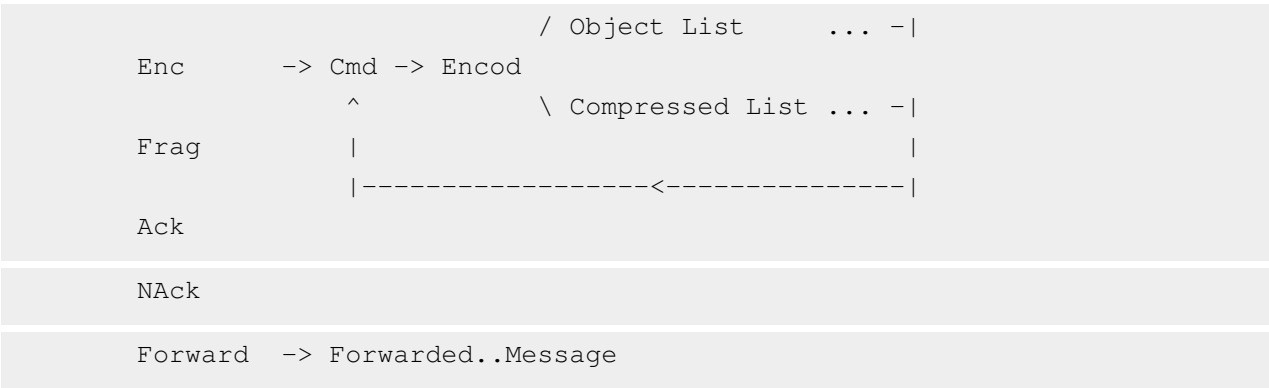
```
TCP
Skype Init TCP packet
```

The Skype Init TCP packet contains

- the seed (4 bytes)
- init\_str string 00 01 00 00 01 00 00 00 01/03

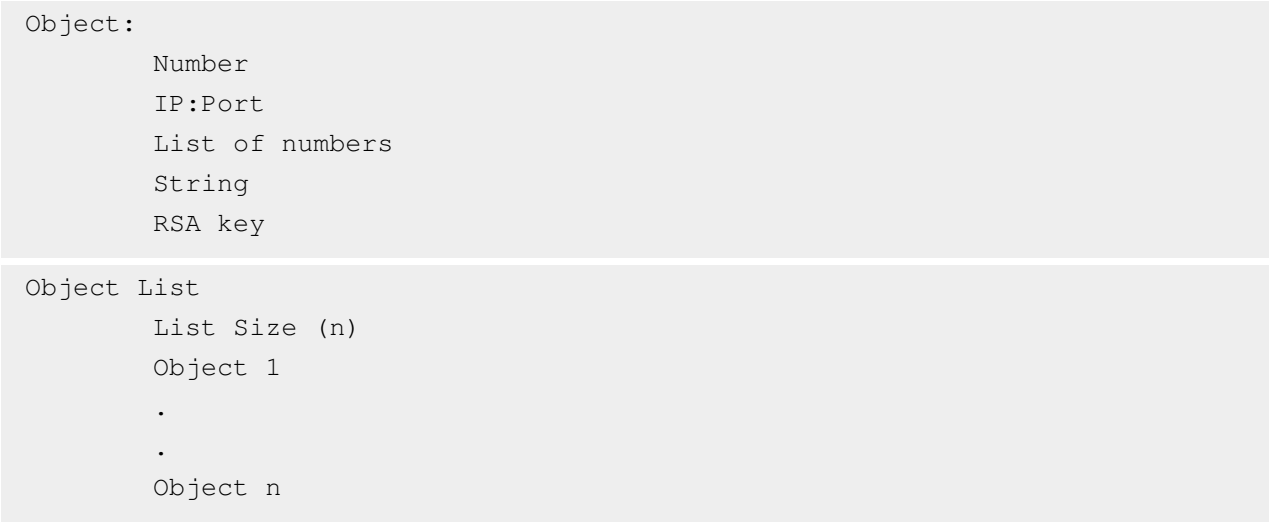
## Low-level datagrams

Almost all traffic is ciphered. Each command has its parameters appended in an object list. The object list can be compressed.



## Object Lists

An object can be a number, string, an IP:port, or even another object list. Each object has an ID. This ID identifies which command parameter the object is.



## Packet compression

Packets can be compressed. The algorithm is a variation of arithmetic compression that uses reals instead of bits.

## Legal issues

Reverse engineering of the Skype protocol by inspecting/disassembling binaries is prohibited by the terms and conditions of Skype's license agreement. However there are legal precedents when the reverse-engineering is aimed at interoperability of file formats and protocols.<sup>[9][10][11]</sup> In the United States, the Digital Millennium Copyright Act grants a safe harbor to reverse engineer software for the purposes of interoperability with other software.<sup>[12][13]</sup> In addition, many countries specifically permit a program to be copied for the purposes of reverse engineering.<sup>[14]</sup>

## Notes

- [1] Skype for Asterisk – Production Released! (<http://blogs.digium.com/2009/08/31/skype-for-asterisk-production-released/>), By pengler, August 31st, 2009, Digium - The Asterisk Company
- [2] Page 11 in Salman A. Baset; Henning Schulzrinne (2004). "An analysis of the Skype peer-to-peer Internet telephony protocol". arXiv:cs/0412017v1 [cs.NI].
- [3] Skype "3.3 Utilization of Your Computer" ([http://www.skype.com/intl/en/legal/eula/#you\\_expect](http://www.skype.com/intl/en/legal/eula/#you_expect)), *End User License Agreement*, August 2010
- [4] Introduction Skype analysis Enforcing anti-Skype policies ([http://www.ossir.org/windows/supports/2005/2005-11-07/EADS-CCR\\_Fabrice\\_Skype.pdf](http://www.ossir.org/windows/supports/2005/2005-11-07/EADS-CCR_Fabrice_Skype.pdf)), Skype uncovered Security study of Skype, Desclaux Fabrice, 7/11/2005, EADS CCR/STI/C
- [5] [http://support.skype.com/en\\_US/faq/FA153/Which-protocols-does-Skype-use](http://support.skype.com/en_US/faq/FA153/Which-protocols-does-Skype-use)
- [6] <http://www.oklabs.net/category/skype-reverse/>
- [7] Dario Bonfiglio et al. "Revealing Skype Traffic: When Randomness Plays with You," ACM SIGCOMM Computer Communication Review, Volume 37:4 (SIGCOMM 2007), p. 37-48 (<https://www.dpacket.org/articles/revealing-skype-traffic-when-randomness-plays-you>)
- [8] Fabrice Desclaux, Kostya Kortchinsky (2006-06-17). "Vanilla Skype part 2" (<http://www.recon.cx/en/f/vskype-part2.pdf>). *RECON2006*.
- [9] Sega vs Accolade, 1992
- [10] Sony vs Connectix, 2000
- [11] Pamela Samuelson and Suzanne Scotchmer, "The Law and Economics of Reverse Engineering", 111 *Yale Law Journal* 1575-1663 (May 2002) (<http://www.yalelawjournal.org/pdf/111-7/SamuelsonFINAL.pdf>)
- [12] 17 U.S.C. Sec. 1201(f).
- [13] WIPO Copyright and Performances and Phonograms Treaties Implementation Act
- [14] In the French "intellectual property" law set, there is an exception that allows any software user to reverse engineer it. See *code de la propriété intellectuelle* (<http://legifrance.gouv.fr/affichCodeArticle.do?cidTexte=LEGITEXT000006069414&idArticle=LEGIARTI000006278920&dateTexte=20080329&categorieLien=cid>) (**French**). This law is the national implementation of a piece of EU legislation: Council Directive 91/250/EEC (<http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31991L0250:EN:NOT>), since then repealed by Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (<http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32009L0024:EN:NOT>) which also has a very similar provision allowing reverse engineering/decompilation for the purposes of development and testing of independent but inter-operating programs).

## References

- Salman A. Baset; Henning Schulzrinne (2004). "An analysis of the Skype peer-to-peer Internet telephony protocol". arXiv:cs/0412017v1 [cs.NI].
- P. Biondi and F. Desclaux (March 3, 2006). "Silver Needle in the Skype" (<http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-biondi/bh-eu-06-biondi-up.pdf>).
- F. Desclaux and K. Kortchinsky (June 6, 2006). "Vanilla Skype - part 1" (<http://www.recon.cx/en/f/vskype-part1.pdf>).
- F. Desclaux and K. Kortchinsky (June 17, 2006). "Vanilla Skype - part 2" (<http://www.recon.cx/en/f/vskype-part2.pdf>).
- L. De Cicco, S. Mascolo, V. Palmisano (May 2007). "An Experimental Investigation of the Congestion Control Used by Skype VoIP." ([http://c3lab.poliba.it/images/d/d2/Skype\\_wwic07.pdf](http://c3lab.poliba.it/images/d/d2/Skype_wwic07.pdf)). *WWIC 07*. Springer.

- L. De Cicco, S. Mascolo, V. Palmisano (December 9–11, 2008). "A Mathematical Model of the Skype VoIP Congestion Control Algorithm." ([http://c3lab.poliba.it/images/2/22/Skype\\_voip\\_model.pdf](http://c3lab.poliba.it/images/2/22/Skype_voip_model.pdf)). *Proc. of IEEE Conference on Decision and Control 2008*.
- Dario Bonfiglio, Marco Melia, Michela Meo, Dario Rossi, Paolo Tofanelli (August 27–31, 2007). "Revealing Skype Traffic: When Randomness Plays With You" (<https://www.dpacket.org/articles/revealing-skype-traffic-when-randomness-plays-you>). ACM SIGCOMM Computer Communication Review.

## External links

- Repository of articles on Skype analysis (<http://www1.cs.columbia.edu/~salman/skype/>)

# BitTorrent

---

**BitTorrent** is a protocol that underpins the practice of peer-to-peer file sharing and is used for distributing large amounts of data over the Internet. BitTorrent is one of the most common protocols for transferring large files and it has been estimated that, collectively, peer-to-peer networks have accounted for approximately 43% to 70% of all Internet traffic (depending on geographical location) as of February 2009.<sup>[1]</sup>

Programmer Bram Cohen designed the protocol in April 2001 and released the first available version on July 2, 2001.<sup>[2]</sup> Currently, numerous BitTorrent clients are available for a variety of computing platforms.

As of January 2012, BitTorrent is utilized by 150 million active users (according to BitTorrent, Inc.). Based on this figure, the total number of monthly BitTorrent users can be estimated at more than a quarter of a billion.<sup>[3]</sup> At any given instant, BitTorrent has, on average, more active users than YouTube and Facebook combined (this refers to the number of active users at any instant and not to the total number of unique users).<sup>[4][5]</sup> Research has also shown that, since 2010, more than 200,000 users of the protocol have been sued.<sup>[6]</sup>

## Description

The BitTorrent protocol can be used to reduce the server and network impact of distributing large files. Rather than downloading a file from a single source server, the BitTorrent protocol allows users to join a "swarm" of hosts to download and upload from each other simultaneously. The protocol is an alternative to the older single source, multiple mirror sources technique for distributing data, and can work over networks with lower bandwidth so many small computers, like mobile phones, are able to efficiently distribute files to many recipients.

A user who wants to upload a file first creates a small *torrent* descriptor file that they distribute by conventional means (web, email, etc.). They then make the file itself available through a BitTorrent node acting as a *seed*. Those with the torrent descriptor file can give it to their own BitTorrent nodes which, acting as *peers* or *leechers*, download it by connecting to the seed and/or other peers.

The file being distributed is divided into segments called *pieces*. As each peer receives a new piece of the file it becomes a source (of that piece) for other peers, relieving the original seed from having to send that piece to every computer or user wishing a copy. With BitTorrent, the task of distributing the file is shared by those who want it; it is entirely possible for the seed to send only a single copy of the file itself and eventually distribute to an unlimited number of peers.

Each piece is protected by a cryptographic hash contained in the torrent descriptor.<sup>[7]</sup> This ensures that any modification of the piece can be reliably detected, and thus prevents both accidental and malicious modifications of any of the pieces received at other nodes. If a node starts with an authentic copy of the torrent descriptor, it can verify the authenticity of the entire file it receives.

---

Pieces are typically downloaded non-sequentially and are rearranged into the correct order by the BitTorrent Client, which monitors which pieces it needs, and which pieces it has and can upload to other peers. Pieces are of the same size throughout a single download (for example a 10 MB file may be transmitted as ten 1 MB Pieces or as forty 256 KB Pieces). Due to the nature of this approach, the download of any file can be halted at any time and be resumed at a later date, without the loss of previously downloaded information, which in turn makes BitTorrent particularly useful in the transfer of larger files. This also enables the client to seek out readily available pieces and download them immediately, rather than halting the download and waiting for the next (and possibly unavailable) piece in line, which typically reduces the overall length of the download.

When a peer completely downloads a file, it becomes an additional seed. This eventual shift from peers to seeders determines the overall "health" of the file (as determined by the number of times a file is available in its complete form).

The distributed nature of BitTorrent can lead to a flood like spreading of a file throughout many peer computer nodes. As more peers join the swarm, the likelihood of a complete successful download by any particular node increases. Relative to traditional Internet distribution schemes, this permits a significant reduction in the original distributor's hardware and bandwidth resource costs.

Distributed downloading protocols in general provide redundancy against system problems, reduces dependence on the original distributor<sup>[8]</sup> and provides sources for the file which are generally transient and therefore harder to trace by those who would block distribution compared to the situation provided by limiting availability of the file to a fixed host machine (or even several).

One such example of BitTorrent being used to reduce the distribution cost of file transmission is in the BOINC Client-Server system. If a BOINC distributed computing application needs to be updated (or merely sent to a user) it can be done so with little impact on the BOINC Server.

## Operation

A BitTorrent client is any program that implements the BitTorrent protocol. Each client is capable of preparing, requesting, and transmitting any type of computer file over a network, using the protocol. A peer is any computer running an instance of a client.

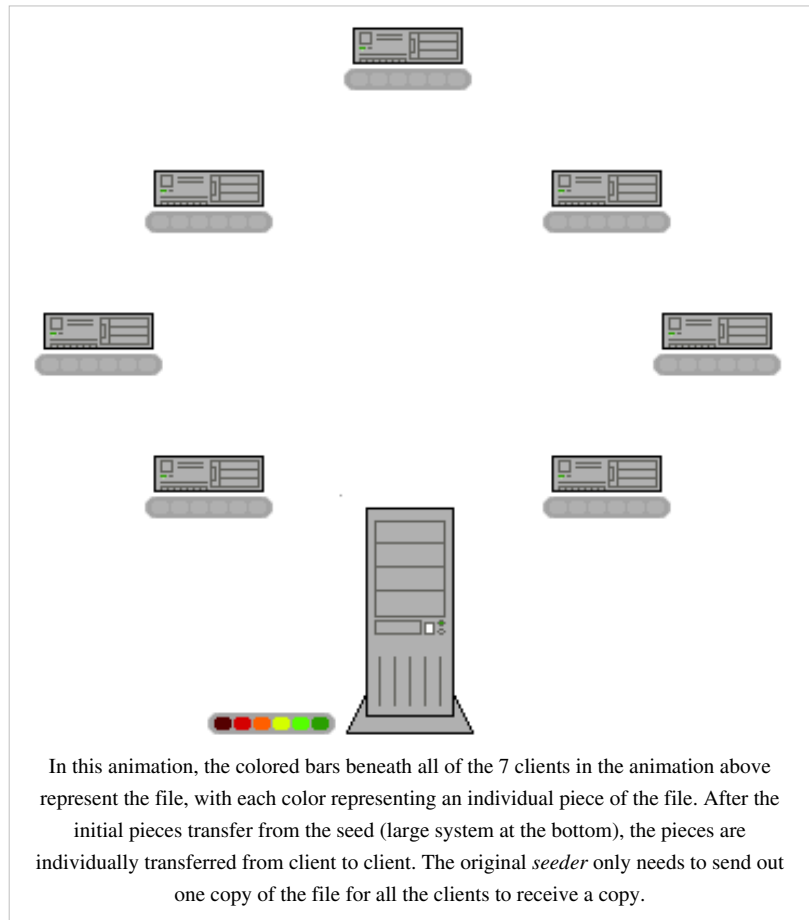
To share a file or group of files, a peer first creates a small file called a "torrent" (e.g. MyFile.torrent). This file contains metadata about the files to be shared and about the tracker, the computer that coordinates the file distribution. Peers that want to download the file must first obtain a torrent file for it and connect to the specified tracker, which tells them from which other peers to download the pieces of the file.

Though both ultimately transfer files over a network, a BitTorrent download differs from a classic download (as is typical with an HTTP or FTP request, for example) in several fundamental ways:

- BitTorrent makes many small data requests over different TCP connections to different machines, while classic downloading is typically made via a single TCP connection to a single machine.
- BitTorrent downloads in a random or in a "rarest-first"<sup>[9]</sup> approach that ensures high availability, while classic downloads are sequential.

Taken together, these differences allow BitTorrent to achieve much lower cost to the content provider, much higher redundancy, and much greater resistance to abuse or to "flash crowds" than regular server software. However, this protection, theoretically, comes at a cost: downloads can take time to rise to full speed because it may take time for enough peer connections to be established, and it may take time for a node to receive sufficient data to become an effective uploader. This contrasts with regular downloads (such as from an HTTP server, for example) that, while more vulnerable to overload and abuse, rise to full speed very quickly and maintain this speed throughout.

In general, BitTorrent's non-contiguous download methods have prevented it from supporting "progressive downloads" or "streaming playback". However, comments made by Bram Cohen in January 2007<sup>[10]</sup> suggest that streaming torrent downloads will soon be commonplace and ad supported streaming<sup>[11]</sup> appears to be the result of those comments. In January 2011 Cohen demonstrated an early version of BitTorrent streaming, saying the feature was projected to be available by summer 2011.<sup>[9]</sup>



## Creating and publishing torrents

The peer distributing a data file treats the file as a number of identically sized pieces, usually with byte sizes of a power of 2, and typically between 32 kB and 16 MB each. The peer creates a hash for each piece, using the SHA-1 hash function, and records it in the torrent file. Pieces with sizes greater than 512 kB will reduce the size of a torrent file for a very large payload, but is claimed to reduce the efficiency of the protocol.<sup>[12]</sup> When another peer later receives a particular piece, the hash of the piece is compared to the recorded hash to test that the piece is error-free.<sup>[13]</sup> Peers that provide a complete file are called seeders, and the peer providing the initial copy is called the initial seeder.

The exact information contained in the torrent file depends on the version of the BitTorrent protocol. By convention, the name of a torrent file has the suffix `.torrent`. Torrent files have an "announce" section, which specifies the URL of the tracker, and an "info" section, containing (suggested) names for the files, their lengths, the piece length used, and a SHA-1 hash code for each piece, all of which are used by clients to verify the integrity of the data they receive.

Torrent files are typically published on websites or elsewhere, and registered with at least one tracker. The tracker maintains lists of the clients currently participating in the torrent.<sup>[13]</sup> Alternatively, in a *trackerless system* (decentralized tracking) every peer acts as a tracker. Azureus was the first BitTorrent client to implement such a system through the distributed hash table (DHT) method. An alternative and incompatible DHT system, known as Mainline DHT, was later developed and adopted by the BitTorrent (Mainline),  $\mu$ Torrent, Transmission, rTorrent, KTorrent, BitComet, and Deluge clients.

After the DHT was adopted, a "private" flag — analogous to the broadcast flag — was unofficially introduced, telling clients to restrict the use of decentralized tracking regardless of the user's desires.<sup>[14]</sup> The flag is intentionally placed in the info section of the torrent so that it cannot be disabled or removed without changing the identity of the torrent. The purpose of the flag is to prevent torrents from being shared with clients that do not have access to the tracker. The flag was requested for inclusion in the official specification in August, 2008, but has not been accepted yet.<sup>[15]</sup> Clients that have ignored the private flag were banned by many trackers, discouraging the practice.<sup>[16]</sup>

## Downloading torrents and sharing files

Users find a torrent of interest, by browsing the web or by other means, download it, and open it with a BitTorrent client. The client connects to the tracker(s) specified in the torrent file, from which it receives a list of peers currently transferring pieces of the file(s) specified in the torrent. The client connects to those peers to obtain the various pieces. If the swarm contains only the initial seeder, the client connects directly to it and begins to request pieces.

Clients incorporate mechanisms to optimize their download and upload rates; for example they download pieces in a random order to increase the opportunity to exchange data, which is only possible if two peers have different pieces of the file.

The effectiveness of this data exchange depends largely on the policies that clients use to determine to whom to send data. Clients may prefer to send data to peers that send data back to them (a tit for tat scheme), which encourages fair trading. But strict policies often result in suboptimal situations, such as when newly joined peers are unable to receive any data because they don't have any pieces yet to trade themselves or when two peers with a good connection between them do not exchange data simply because neither of them takes the initiative. To counter these effects, the official BitTorrent client program uses a mechanism called "optimistic unchoking", whereby the client reserves a portion of its available bandwidth for sending pieces to random peers (not necessarily known good partners, so called preferred peers) in hopes of discovering even better partners and to ensure that newcomers get a chance to join the swarm.<sup>[17]</sup>

Although swarming scales well to tolerate flash crowds for popular content, it is less useful for unpopular content. Peers arriving after the initial rush might find the content unavailable and need to wait for the arrival of a seed in

order to complete their downloads. The seed arrival, in turn, may take long to happen (this is termed the seeder promotion problem). Since maintaining seeds for unpopular content entails high bandwidth and administrative costs, this runs counter to the goals of publishers that value BitTorrent as a cheap alternative to a client-server approach. This occurs on a huge scale; measurements have shown that 38% of all new torrents become unavailable within the first month.<sup>[18]</sup> A strategy adopted by many publishers which significantly increases availability of unpopular content consists of bundling multiple files in a single swarm.<sup>[19]</sup> More sophisticated solutions have also been proposed; generally, these use cross-torrent mechanisms through which multiple torrents can cooperate to better share content.<sup>[20]</sup>

BitTorrent does not offer its users anonymity. It is possible to obtain the IP addresses of all current and possibly previous participants in a swarm from the tracker. This may expose users with insecure systems to attacks.<sup>[17]</sup> It may also expose users to the risk of being sued, if they are distributing files without permission from the copyright holder(s). However, there are ways to promote anonymity; for example, the OneSwarm project layers privacy-preserving sharing mechanisms on top of the original BitTorrent protocol.

## Adoption

A growing number of individuals and organizations are using BitTorrent to distribute their own or licensed material. Independent adopters report that without using BitTorrent technology and its dramatically reduced demands on their private networking hardware and bandwidth, they could not afford to distribute their files.<sup>[21]</sup>

## Film, video, and music

- BitTorrent Inc. has obtained a number of licenses from Hollywood studios for distributing popular content from their websites.
- Sub Pop Records releases tracks and videos via BitTorrent Inc.<sup>[22]</sup> to distribute its 1000+ albums. Babyshambles and The Libertines (both bands associated with Pete Doherty) have extensively used torrents to distribute hundreds of demos and live videos. US industrial rock band Nine Inch Nails frequently distributes albums via BitTorrent.
- Podcasting software is starting to integrate BitTorrent to help podcasters deal with the download demands of their MP3 "radio" programs. Specifically, Juice and Miro (formerly known as Democracy Player) support automatic processing of .torrent files from RSS feeds. Similarly, some BitTorrent clients, such as µTorrent, are able to process web feeds and automatically download content found within them.
- DGM Live purchases are provided via BitTorrent.<sup>[23]</sup>
- Vodo, a service which distributes "free-to-share" movies and TV show BitTorrent.<sup>[24][25][26]</sup>

## Broadcasters

- In 2008, the CBC became the first public broadcaster in North America to make a full show (*Canada's Next Great Prime Minister*) available for download using BitTorrent.<sup>[27]</sup>
  - The Norwegian Broadcasting Corporation (NRK) has since March 2008 experimented with bittorrent distribution, available online.<sup>[28]</sup> Only selected material in which NRK owns all royalties are published. Responses have been very positive, and NRK is planning to offer more content.
  - The Dutch VPRO broadcasting organization released four documentaries under a Creative Commons license using the content distribution feature of the Mininova tracker.<sup>[29]</sup>
-

## Personal material

- The Amazon S3 "Simple Storage Service" is a scalable Internet-based storage service with a simple web service interface, equipped with built-in BitTorrent support.
- Blog Torrent offers a simplified BitTorrent tracker to enable bloggers and non-technical users to host a tracker on their site. Blog Torrent also allows visitors to download a "stub" loader, which acts as a BitTorrent client to download the desired file, allowing users without BitTorrent software to use the protocol.<sup>[30]</sup> This is similar to the concept of a self-extracting archive.

## Software

- Blizzard Entertainment uses BitTorrent (via a proprietary client called the "Blizzard Downloader") to distribute content and patches for Diablo III, StarCraft II and World of Warcraft, including the games themselves.<sup>[31]</sup>
- Many software games, especially those whose large size makes them difficult to host due to bandwidth limits, extremely frequent downloads, and unpredictable changes in network traffic, will distribute instead a specialized, stripped down bittorrent client with enough functionality to download the game from the other running clients and the primary server (which is maintained in case not enough peers are available).
- Many major open source and free software projects encourage BitTorrent as well as conventional downloads of their products (via HTTP, FTP etc.) to increase availability and to reduce load on their own servers, especially when dealing with larger files.<sup>[32]</sup>

## Government

- The UK government used BitTorrent to distribute details about how the tax money of UK citizens was spent.<sup>[33][34]</sup>

## Education

- The Florida State University uses BitTorrent to distribute large scientific data sets to its researchers.<sup>[35]</sup>
- Many universities that have BOINC distributed computing projects have used the BitTorrent functionality of the client-server system to reduce the bandwidth costs of distributing the client side applications used to process the scientific data.

## Others

- Facebook uses BitTorrent to distribute updates to Facebook servers.<sup>[36]</sup>
- Twitter uses BitTorrent to distribute updates to Twitter servers.<sup>[37][38]</sup>
- The Internet Archive added Bittorrent to its file download options for over 1.3 million existing files, and all newly uploaded files, in August 2012.<sup>[39][40]</sup> This method is the fastest means of downloading media from the Archive.<sup>[39][41]</sup>

As of 2011 BitTorrent has 100 million users and a greater share of network bandwidth than Netflix and Hulu combined.<sup>[4][42]</sup>

At any given instant of time BitTorrent has, on average, more active users than YouTube and Facebook combined. (This refers to the number of active users at any instant and not to the total number of registered users.)<sup>[4][5]</sup>

CableLabs, the research organization of the North American cable industry, estimates that BitTorrent represents 18% of all broadband traffic.<sup>[43]</sup> In 2004, CacheLogic put that number at roughly 35% of all traffic on the Internet.<sup>[44]</sup> The discrepancies in these numbers are caused by differences in the method used to measure P2P traffic on the Internet.<sup>[45]</sup>

Routers that use network address translation (NAT) must maintain tables of source and destination IP addresses and ports. Typical home routers are limited to about 2000 table entries while some more expensive routers have larger



table capacities. BitTorrent frequently contacts 20–30 servers per second, rapidly filling the NAT tables. This is a common cause of home routers locking up.<sup>[46]</sup>

## Indexing

The BitTorrent protocol provides no way to index torrent files. As a result, a comparatively small number of websites have hosted a large majority of torrents, many linking to copyrighted material without the authorization of copyright holders, rendering those sites especially vulnerable to lawsuits.<sup>[47]</sup> Several types of websites support the discovery and distribution of data on the BitTorrent network.

Public torrent-hosting sites such as The Pirate Bay allow users to search and download from their collection of torrent files. Users can typically also upload torrent files for content they wish to distribute. Often, these sites also run BitTorrent trackers for their hosted torrent files, but these two functions are not mutually dependent: a torrent file could be hosted on one site and tracked by another, unrelated site.

Private host/tracker sites operate like public ones except that they restrict access to registered users and keep track of the amount of data each user uploads and downloads, in an attempt to reduce leeching.

Search engines allow the discovery of torrent files that are hosted and tracked on other sites; examples include Mininova, BTDigg, BTJunkie, Torrentz, The Pirate Bay, Eztorrent, and isoHunt. These sites allow the user to ask for content meeting specific criteria (such as containing a given word or phrase) and retrieve a list of links to torrent files matching those criteria. This list can often be sorted with respect to several criteria, relevance (seeders-leechers ratio) being one of the most popular and useful (due to the way the protocol behaves, the download bandwidth achievable is very sensitive to this value). Bram Cohen launched a BitTorrent search engine on <http://www.bittorrent.com/search> that co-mingles licensed content with search results.<sup>[48]</sup> Metasearch engines allow one to search several BitTorrent indices and search engines at once. DHT search engines monitors the DHT network and indexes torrents via metadata exchange from peers.

However, recently some P2P, decentralized alternatives to Torrent search engines have emerged, see decentralized keyword search further down the page.

## Technologies built on BitTorrent

The BitTorrent protocol is still under development and therefore may still acquire new features and other enhancements such as improved efficiency.

### Distributed trackers

On May 2, 2005, Azureus 2.3.0.0 (now known as Vuze) was released,<sup>[49]</sup> introducing support for "trackerless" torrents through a system called the "distributed database." This system is a DHT implementation which allows the client to use torrents that do not have a working BitTorrent tracker. The following month, BitTorrent, Inc. released version 4.2.0 of the Mainline BitTorrent client, which supported an alternative DHT implementation (popularly known as "Mainline DHT", outlined in a draft<sup>[50]</sup> on their website) that is incompatible with that of Azureus.

Current versions of the official BitTorrent client, µTorrent, BitComet, Transmission and BitSpirit all share compatibility with Mainline DHT. Both DHT implementations are based on Kademlia.<sup>[51]</sup> As of version 3.0.5.0, Azureus also supports Mainline DHT in addition to its own distributed database through use of an optional application plugin.<sup>[52]</sup> This potentially allows the Azureus client to reach a bigger swarm.

Another idea that has surfaced in Vuze is that of *virtual torrents*. This idea is based on the distributed tracker approach and is used to describe some web resource. Currently, it is used for instant messaging. It is implemented using a special messaging protocol and requires an appropriate plugin. Anatomic P2P is another approach, which uses a decentralized network of nodes that route traffic to dynamic trackers.

Most BitTorrent clients also use Peer exchange (PEX) to gather peers in addition to trackers and DHT. Peer exchange checks with known peers to see if they know of any other peers. With the 3.0.5.0 release of Vuze, all major BitTorrent clients now have compatible peer exchange.

## Web seeding

Web seeding was implemented in 2006 as the ability of BitTorrent clients to download torrent pieces from an HTTP source in addition to the swarm. The advantage of this feature is that a website may distribute a torrent for a particular file or batch of files and make those files available for download from that same web server; this can simplify long-term seeding and load balancing through the use of existing, cheap, web hosting setups. In theory, this would make using BitTorrent almost as easy for a web publisher as creating a direct HTTP download. In addition, it would allow the "web seed" to be disabled if the swarm becomes too popular while still allowing the file to be readily available.

This feature has two distinct and incompatible specifications.

The first was created by John "TheSHAD0W" Hoffman, who created BitTornado.<sup>[53][54]</sup> From version 5.0 onward, the Mainline BitTorrent client also supports web seeds, and the BitTorrent web site had<sup>[55]</sup> a simple publishing tool that creates web seeded torrents.<sup>[56]</sup> µTorrent added support for web seeds in version 1.7. BitComet added support for web seeds in version 1.14. This first specification requires running a web service that serves content by info-hash and piece number, rather than filename.

The other specification is created by GetRight authors and can rely on a basic HTTP download space (using byte serving).<sup>[57][58]</sup>

In September 2010, a new service named Burnbit was launched which generates a torrent from any URL using webseeding.<sup>[59]</sup>

There exist server-side solutions that provide initial seeding of the file from the webserver via standard BitTorrent protocol and when the number of external seeders reach a limit, they stop serving the file from the original source.<sup>[60]</sup>

## RSS feeds

A technique called *broadcatching* combines RSS with the BitTorrent protocol to create a content delivery system, further simplifying and automating content distribution. Steve Gillmor explained the concept in a column for Ziff-Davis in December, 2003.<sup>[61]</sup> The discussion spread quickly among bloggers (Ernest Miller,<sup>[62]</sup> Chris Pirillo, etc.). In an article entitled *Broadcatching with BitTorrent*, Scott Raymond explained:

I want RSS feeds of BitTorrent files. A script would periodically check the feed for new items, and use them to start the download. Then, I could find a trusted publisher of an Alias RSS feed, and "subscribe" to all new episodes of the show, which would then start downloading automatically — like the "season pass" feature of the TiVo.

—Scott Raymond, [scottraymond.net](http://scottraymond.net)<sup>[63]</sup>

The RSS feed will track the content, while BitTorrent ensures content integrity with cryptographic hashing of all data, so feed subscribers will receive uncorrupted content.

One of the first and popular software clients (free and open source) for *broadcatching* is Miro. Other free software clients such as PenguinTV and KatchTV are also now supporting broadcatching.

The BitTorrent web-service MoveDigital had the ability to make torrents available to any web application capable of parsing XML through its standard REST-based interface,<sup>[64]</sup> although this has since been discontinued. Additionally, Torrenthut is developing a similar torrent API that will provide the same features, as well as further intuition to help bring the torrent community to Web 2.0 standards. Alongside this release is a first PHP application built using the API called PEP, which will parse any Really Simple Syndication (RSS 2.0) feed and automatically create and seed a

torrent for each enclosure found in that feed.<sup>[65]</sup>

## Throttling and encryption

Since BitTorrent makes up a large proportion of total traffic, some ISPs have chosen to throttle (slow down) BitTorrent transfers to ensure network capacity remains available for other uses. For this reason, methods have been developed to disguise BitTorrent traffic in an attempt to thwart these efforts.<sup>[66]</sup>

Protocol header encrypt (PHE) and Message stream encryption/Protocol encryption (MSE/PE) are features of some BitTorrent clients that attempt to make BitTorrent hard to detect and throttle. At the moment Vuze, Bitcomet, KTorrent, Transmission, Deluge, µTorrent, MooPolice, Halite, rTorrent and the latest official BitTorrent client (v6) support MSE/PE encryption.

In September 2006 it was reported that some software could detect and throttle BitTorrent traffic masquerading as HTTP traffic.<sup>[67]</sup>

Reports in August 2007 indicated that Comcast was preventing BitTorrent seeding by monitoring and interfering with the communication between peers. Protection against these efforts is provided by proxying the client-tracker traffic via an encrypted tunnel to a point outside of the Comcast network.<sup>[68]</sup> Comcast has more recently called a "truce" with BitTorrent, Inc. with the intention of shaping traffic in a protocol-agnostic manner.<sup>[69]</sup> Questions about the ethics and legality of Comcast's behavior have led to renewed debate about net neutrality in the United States.<sup>[70]</sup>

In general, although encryption can make it difficult to determine *what* is being shared, BitTorrent is vulnerable to traffic analysis. Thus, even with MSE/PE, it may be possible for an ISP to recognize BitTorrent and also to determine that a system is no longer downloading but only uploading data, and terminate its connection by injecting TCP RST (reset flag) packets.

## Multitracker

Another unofficial feature is an extension to the BitTorrent metadata format proposed by John Hoffman<sup>[71]</sup> and implemented by several indexing websites. It allows the use of multiple trackers per file, so if one tracker fails, others can continue to support file transfer. It is implemented in several clients, such as BitComet, BitTornado, BitTorrent, KTorrent, Transmission, Deluge, µTorrent, rtorrent, Vuze, Frostwire. Trackers are placed in groups, or tiers, with a tracker randomly chosen from the top tier and tried, moving to the next tier if all the trackers in the top tier fail.

Torrents with multiple trackers<sup>[72]</sup> can decrease the time it takes to download a file, but also has a few consequences:

- Poorly implemented<sup>[73]</sup> clients may contact multiple trackers, leading to more overhead-traffic.
- Torrents from closed trackers suddenly become downloadable by non-members, as they can connect to a seed via an open tracker.

## Decentralized keyword search

Even with distributed trackers, a third party is still required to find a specific torrent. This is usually done in the form of a hyperlink from the website of the content owner or through indexing websites like isoHunt, Torrentz, BTDigg or The Pirate Bay.

The Tribler BitTorrent client is the first to incorporate decentralized search capabilities. With Tribler, users can find .torrent files that are hosted among other peers, instead of on a centralized index sites. It adds such an ability to the BitTorrent protocol using a gossip protocol, somewhat similar to the eXeem network which was shut down in 2005. The software includes the ability to recommend content as well. After a dozen downloads the Tribler software can roughly estimate the download taste of the user and recommend additional content.<sup>[74]</sup>

In May 2007 Cornell University published a paper proposing a new approach to searching a peer-to-peer network for inexact strings,<sup>[75]</sup> which could replace the functionality of a central indexing site. A year later, the same team

implemented the system as a plugin for Vuze called Cubit<sup>[76]</sup> and published a follow-up paper reporting its success.<sup>[77]</sup>

A somewhat similar facility but with a slightly different approach is provided by the BitComet client through its "Torrent Exchange"<sup>[78]</sup> feature. Whenever two peers using BitComet (with Torrent Exchange enabled) connect to each other they exchange lists of all the torrents (name and info-hash) they have in the Torrent Share storage (torrent files which were previously downloaded and for which the user chose to enable sharing by Torrent Exchange).

Thus each client builds up a list of all the torrents shared by the peers it connected to in the current session (or it can even maintain the list between sessions if instructed). At any time the user can search into that Torrent Collection list for a certain torrent and sort the list by categories. When the user chooses to download a torrent from that list, the .torrent file is automatically searched for (by info-hash value) in the DHT Network and when found it is downloaded by the querying client which can after that create and initiate a downloading task.

## Implementations

The BitTorrent specification is free to use and many clients are open source, so BitTorrent clients have been created for all common operating systems using a variety of programming languages. The official BitTorrent client, µTorrent, Xunlei, Vuze and BitComet are some of the most popular clients.<sup>[79]</sup>

Some BitTorrent implementations such as MLDonkey and Torrentflux are designed to run as servers. For example, this can be used to centralize file sharing on a single dedicated server which users share access to on the network.<sup>[80]</sup> Server-oriented BitTorrent implementations can also be hosted by hosting providers at co-located facilities with high bandwidth Internet connectivity (e.g., a datacenter) which can provide dramatic speed benefits over using BitTorrent from a regular home broadband connection.

Services such as ImageShack can download files on BitTorrent for the user, allowing them to download the entire file by HTTP once it is finished.

The Opera web browser supports BitTorrent,<sup>[81]</sup> as does Wyzo. BitLet allows users to download Torrents directly from their browser using a Java applet. An increasing number of hardware devices are being made to support BitTorrent. These include routers and NAS devices containing BitTorrent-capable firmware like OpenWrt.

Proprietary versions of the protocol which implement DRM, encryption, and authentication are found within managed clients such as Pando.

## Development

An unimplemented (as of February 2008) unofficial feature is Similarity Enhanced Transfer (SET), a technique for improving the speed at which peer-to-peer file sharing and content distribution systems can share data. SET, proposed by researchers Pucha, Andersen, and Kaminsky, works by spotting chunks of identical data in files that are an exact or near match to the one needed and transferring these data to the client if the "exact" data are not present. Their experiments suggested that SET will help greatly with less popular files, but not as much for popular data, where many peers are already downloading it.<sup>[82]</sup> Andersen believes that this technique could be immediately used by developers with the BitTorrent file sharing system.<sup>[83]</sup>

As of December 2008, BitTorrent, Inc. is working with Oversi on new Policy Discover Protocols that query the ISP for capabilities and network architecture information. Oversi's ISP hosted NetEnhancer box is designed to "improve peer selection" by helping peers find local nodes, improving download speeds while reducing the loads into and out of the ISP's network.<sup>[84]</sup>

---

## Legal issues

There has been much controversy over the use of BitTorrent trackers. BitTorrent metafiles themselves do not store file contents. Whether the publishers of BitTorrent metafiles violate copyrights by linking to copyrighted material without the authorization of copyright holders is controversial.

Various jurisdictions have pursued legal action against websites that host BitTorrent trackers. High-profile examples include the closing of Suprnova.org, TorrentSpy, LokiTorrent, BTJunkie, Mininova, Demonoid and Oink's Pink Palace. The Pirate Bay torrent website, formed by a Swedish group, is noted for the "legal" section of its website in which letters and replies on the subject of alleged copyright infringements are publicly displayed. On 31 May 2006, The Pirate Bay's servers in Sweden were raided by Swedish police on allegations by the MPAA of copyright infringement;<sup>[85]</sup> however, the tracker was up and running again three days later.

In the study used to value NBC Universal in its merger with Comcast, Envisional found that all of the top 10,000 torrents on the BitTorrent network violated copyright.<sup>[86]</sup>

Between 2010 and 2012, 200,000 people have been sued by copyright trolls for uploading and downloading copyrighted content through BitTorrent.<sup>[6]</sup>

In 2011, 18.8% of North American internet traffic was used by peer-to-peer networks which equates to 132 billion music file transfers and 11 billion movie file transfers on the BitTorrent network.<sup>[87]</sup>

On April 30, 2012 the UK High Court ordered five ISPs to block BitTorrent search engine The Pirate Bay.<sup>[88]</sup>

## BitTorrent and malware

Several studies on BitTorrent have indicated that a large portion of files available for download via BitTorrent contain malware. In particular, one small sample<sup>[89]</sup> indicated that 18% of all executable programs available for download contained malware. Another study<sup>[90]</sup> claims that as much as 14.5% of BitTorrent downloads contain zero-day malware, and that BitTorrent was used as the distribution mechanism for 47% of all zero-day malware they have found.

## References

- [1] Schulze, Hendrik; Klaus Mochalski (2009). "Internet Study 2008/2009" (<http://www.ipoque.com/sites/default/files/mediafiles/documents/internet-study-2008-2009.pdf>). Leipzig, Germany: ipoque. . Retrieved 3 Oct 2011. "Peer-to-peer file sharing (P2P) still generates by far the most traffic in all monitored regions – ranging from 43 percent in Northern Africa to 70 percent in Eastern Europe."
- [2] Cohen, Bram (2001-07-02). "BitTorrent — a new P2P app" (<http://finance.groups.yahoo.com/group/decentralization/message/3160>). Yahoo eGroups. . Retrieved 2007-04-15.
- [3] "BitTorrent and µTorrent Software Surpass 150 Million User Milestone" ([http://www.bittorrent.com/intl/es/company/about/ces\\_2012\\_150m\\_users](http://www.bittorrent.com/intl/es/company/about/ces_2012_150m_users)). Bittorrent.com. 2012-01-09. . Retrieved 2012-07-09.
- [4] "fastcompany.com" (<http://www.fastcompany.com/1714001/bittorrent-swells-to-100-million-users>). fastcompany.com. . Retrieved 2012-07-09.
- [5] "comscore.com" ([http://www.comscore.com/Press\\_Events/Press\\_Releases/2010/9/comScore\\_Releases\\_August\\_2010\\_U.S.\\_Online\\_Video\\_Rankings](http://www.comscore.com/Press_Events/Press_Releases/2010/9/comScore_Releases_August_2010_U.S._Online_Video_Rankings)). comscore.com. 2010-09-30. . Retrieved 2012-07-09.
- [6] Jacobsson Purewal, Sarah (2011-08-09). "Copyright Trolls: 200,000 BitTorrent Users Sued Since 2010" ([http://www.pcworld.com/article/237593/copyright\\_trolls\\_200000\\_bittorrent\\_users\\_sued\\_since\\_2010.html](http://www.pcworld.com/article/237593/copyright_trolls_200000_bittorrent_users_sued_since_2010.html)). PC World. . Retrieved 2012-05-06.
- [7] Bram Cohen (10-Jan-2008). "The BitTorrent Protocol Specification" ([http://www.bittorrent.org/beps/bep\\_0003.html](http://www.bittorrent.org/beps/bep_0003.html)). *BitTorrent.org*. . Retrieved 20 November 2010.
- [8] Estimating Self-Sustainability in Peer-to-Peer Swarming Systems (<http://arxiv4.library.cornell.edu/abs/1004.0395v2>) by D. Menasche, A. Rocha, E. de Souza e Silva, R. M. Leao, D. Towsley, A. Venkataramani
- [9] Urvoy-Keller (December 2006). "Rarest First and Choke Algorithms Are Enough" (<http://conferences.sigcomm.org/imc/2006/papers/p20-logout.pdf>) (PDF). SIGCOMM. . Retrieved 2012-03-09.
- [10] <http://torrentfreak.com/interview-with-bram-cohen-the-inventor-of-bittorrent>
- [11] <http://torrentfreak.com/bittorrent-launches-ad-supported-streaming-071218>
- [12] "Theory.org" (<http://wiki.theory.org/index.php/BitTorrentSpecification>). Wiki.theory.org. . Retrieved 2012-07-09.
- [13] Cohen, Bram (October 2002). "BitTorrent Protocol 1.0" ([http://www.bittorrent.org/beps/bep\\_0003.html](http://www.bittorrent.org/beps/bep_0003.html)). BitTorrent.org. . Retrieved 2008-10-27.

- [14] "Unofficial BitTorrent Protocol Specification v1.0" ([http://wiki.theory.org/BitTorrentSpecification#Info\\_Dictionary](http://wiki.theory.org/BitTorrentSpecification#Info_Dictionary)). . Retrieved 2009-10-04.
- [15] "Private Torrents" ([http://bittorrent.org/beps/bep\\_0027.html](http://bittorrent.org/beps/bep_0027.html)). Bittorrent.org. . Retrieved 2009-10-04.
- [16] "BitComet Banned From Growing Number of Private Trackers" (<http://www.slyck.com/news.php?story=1021>). . Retrieved 2009-10-04.
- [17] Tamilmani, Karthik (2003-10-25). "Studying and enhancing the BitTorrent protocol" ([http://web.archive.org/web/20041119150847/http://mnl.cs.stonybrook.edu/home/karthik/BitTorrent/Robustness\\_of\\_BT.doc](http://web.archive.org/web/20041119150847/http://mnl.cs.stonybrook.edu/home/karthik/BitTorrent/Robustness_of_BT.doc)) (DOC). Stony Brook University. Archived from the original ([http://mnl.cs.stonybrook.edu/home/karthik/BitTorrent/Robustness\\_of\\_BT.doc](http://mnl.cs.stonybrook.edu/home/karthik/BitTorrent/Robustness_of_BT.doc)) on 2004-11-19. . Retrieved 2006-05-06.
- [18] Unraveling BitTorrent's File Unavailability: Measurements and Analysis (<http://eprints.comp.lancs.ac.uk/2281/1/P2P10.pdf>) by Sebastian Kaune, Ruben Cuevas Rumin, Gareth Tyson, Andreas Mauthe, Ralf Steinmetz
- [19] Content Availability and Bundling in Swarming Systems (<http://conferences.sigcomm.org/co-next/2009/papers/Menasche.pdf>) by D. Menasche, A. Rocha, B. Li, D. Towsley, A. Venkataramani
- [20] The Seeder Promotion Problem: Measurements, Analysis and Solution Space (<http://www.dcs.kcl.ac.uk/staff/tysong/files/ICCCN09.pdf>) by Sebastian Kaune, Gareth Tyson, Konstantin Pussep, Andreas Mauthe, Aleksandra Kovacevic and Ralf Steinmetz
- [21] See, for example, Why Bit Torrent (<http://tasvideos.org/WhyBitTorrent.html>) at <http://tasvideos.org> tasvideos.org.
- [22] "Sub Pop page on BitTorrent.com" (<http://www.bittorrent.com/users/subpoprecords/>). . Retrieved 2006-12-13.
- [23] "DGMLive.com" (<http://www.dgmlive.com/help.htm#whatisbittorrent>). DGMLive.com. . Retrieved 2012-07-09.
- [24] VODO - About. . URL:<http://vo.do/about>. Accessed: 2012-04-15. (Archived by WebCite® at <http://www.webcitation.org/66wxu53jV>)
- [25] Cory Doctorow. Vodo: a filesharing service for film-makers - Boing Boing. Happy Mutants LLC.. . URL:<http://boingboing.net/2009/10/15/vodo-a-filesharing-s.html>. Accessed: 2012-04-15. (Archived by WebCite® at <http://www.webcitation.org/66wy0PFq1>)
- [26] Ernesto. Pioneer One, The BitTorrent Exclusive TV-Series Continues. TorrentFreak. . URL:<https://torrentfreak.com/pioneer-one-the-bittorrent-exclusive-tv-series-continues-101215/>. Accessed: 2012-04-15. (Archived by WebCite® at <http://www.webcitation.org/66wyOriB>)
- [27] "CBC to BitTorrent Canada's Next Great Prime Minister:" ([http://www.cbc.ca/nextprimeminister/blog/2008/03/canadas\\_next\\_great\\_prime\\_minis.html](http://www.cbc.ca/nextprimeminister/blog/2008/03/canadas_next_great_prime_minis.html)). *CBC News*. 19 March 2008. . Retrieved 2008-03-19.
- [28] "Bittorrent" (<http://nrkbeta.no/bittorrent/>). Nrkbeta.no. . Retrieved 2012-07-09.
- [29] "Torrents uploaded by EeuwvandeStad" (<http://www.mininova.org/user/EeuwvandeStad>). .
- [30] Rustad, Roger E. (26 August 2004). "Blog Torrent and Participatory Culture" (<http://grep.law.harvard.edu/article.pl?sid=04/08/26/0236209>). Grep Law. . Retrieved 2006-05-09.
- [31] "Blizzard Downloader" ([http://www.wowpedia.org/Blizzard\\_Downloader](http://www.wowpedia.org/Blizzard_Downloader)). Curse Inc.. 4 November 2010. . Retrieved 2010-11-04.
- [32] "Complete Download Options List – BitTorrent" (<http://www.ubuntu.com/getubuntu/downloadmirrors#bt>). . Retrieved 2009-05-07.
- [33] HM Government (4). "omined Online Information System" (<http://data.gov.uk/dataset/coins>). *Data.Gov.Uk Beta*. Controller of Her Majesty's Stationery Office. . Retrieved 7 September 2012.
- [34] Ernesto (4). "UK Government Uses BitTorrent to Share Public Spending Data" (<http://torrentfreak.com/uk-government-uses-bittorrent-to-share-public-spending-data-100604/>). *TorrentFreak*. TorrentFreak. . Retrieved 7 September 2012.
- [35] "HPC Data Repository" ([http://www.hpc.fsu.edu/index.php?option=com\\_wrapper&view=wrapper&Itemid=80](http://www.hpc.fsu.edu/index.php?option=com_wrapper&view=wrapper&Itemid=80)). .
- [36] Ernesto (25). "Facebook Uses BitTorrent, and They Love It" (<http://torrentfreak.com/facebook-uses-bittorrent-and-they-love-it-100625/>). *Torrent Freak*. Torrent Freak. . Retrieved 7 September 2012.
- [37] Ernesto (10). "Twitter Uses BitTorrent For Server Deployment" (<http://torrentfreak.com/twitter-uses-bittorrent-for-server-deployment-100210/>). *Torrent Freak*. Torrent Freak. . Retrieved 7 September 2012.
- [38] Ernesto (16). "BitTorrent Makes Twitter's Server Deployment 75x Faster" (<http://torrentfreak.com/bittorrent-makes-twitters-server-deployment-75-faster-100716/>). *Torrent Freak*. Torrent Freak. . Retrieved 7 September 2012.
- [39] "Internet Archive Starts Seeding 1,398,875 Torrents" (<https://torrentfreak.com/internet-archive-starts-seeding-1398635-torrents-120807/>). TorrentFreak. August 7, 2012. . Retrieved August 7, 2012.
- [40] Hot List for bt1.us.archive.org (Updated August 7 2012, 7:31 pm PDT) (<http://bt1.archive.org/hotlist.php>). Archive.org.
- [41] "Welcome to Archive torrents" (<http://archive.org/details/bittorrent>). Archive.org.
- [42] "financialpost.com" (<http://business.financialpost.com/2011/07/01/bittorrent-turns-ten/>). Business.financialpost.com. . Retrieved 2012-07-09.
- [43] Ellis, Leslie (8 May 2006). "BitTorrent's Swarms Have a Deadly Bite On Broadband Nets" (<http://www.multichannel.com/article/CA6332098.html>). Multichannel News. . Retrieved 2006-05-08.
- [44] Pasick, Adam (4 November 2004). "LiveWire — File-sharing network thrives beneath the radar" (<http://www.interesting-people.org/archives/interesting-people/200411/msg00078.html>). Yahoo! News. . Retrieved 2006-05-09.
- [45] "RCCSP" ([http://www.the-resource-center.com/index/telecom\\_seminars.htm](http://www.the-resource-center.com/index/telecom_seminars.htm)). The-resource-center.com. 2012-06-11. . Retrieved 2012-07-09.
- [46] "uTorrent's FAQ page" ([http://www.utorrent.com/faq.php#Modems\\_routers\\_that\\_are\\_known\\_to\\_have\\_problems\\_with\\_P2P](http://www.utorrent.com/faq.php#Modems_routers_that_are_known_to_have_problems_with_P2P)). .
- [47] "PublicBT Tracker Set To Patch BitTorrent' Achilles' Heel" (<http://torrentfreak.com/publicbt-tracker-set-to-patch-bittorrents-achilles-heel-090712/>). 12 July 2009. . Retrieved 14 July 2009.
- [48] Worthington, David; Nate Mook (25 May 2015). "BitTorrent Creator Opens Online Search" ([http://www.betanews.com/article/BitTorrent\\_Creator\\_Opens\\_Online\\_Search/1117065427](http://www.betanews.com/article/BitTorrent_Creator_Opens_Online_Search/1117065427)). BetaNews. . Retrieved 2006-05-09.
- [49] "Vuze Changelog" (<http://azureus.sourceforge.net/changelog.php>). Azureus.sourceforge.net. .

- [50] [http://bittorrent.org/beps/bep\\_0005.html](http://bittorrent.org/beps/bep_0005.html)
- [51] "Khashmir.Sourceforge.net" (<http://khashmir.sourceforge.net/>). Khashmir.Sourceforge.net. . Retrieved 2012-07-09.
- [52] "Azureus.sourceforge.net" ([http://azureus.sourceforge.net/plugin\\_details.php?plugin=mlDHT](http://azureus.sourceforge.net/plugin_details.php?plugin=mlDHT)). Azureus.sourceforge.net. . Retrieved 2012-07-09.
- [53] "HTTP-Based Seeding Specification" (<http://bittornado.com/docs/webseed-spec.txt>) (TXT). . Retrieved 2006-05-09.
- [54] John Hoffman, DeHackEd (2008-02-25). "*HTTP Seeding* – BitTorrent Enhancement Proposal № 17" ([http://www.bittorrent.org/beps/bep\\_0017.html](http://www.bittorrent.org/beps/bep_0017.html)). . Retrieved 2012-02-17.
- [55] "The Torrent Entertainment Network has closed" (<http://www.bittorrent.com/btusers/nowplaying/>). .
- [56] "Publish – BitTorrent" (<http://web.archive.org/web/20070526065412/http://www.bittorrent.com/publish>). Archived from the original (<http://www.bittorrent.com/publish>) on 2007-05-26. . (archived page from May 26, 2007, web.archive.org)
- [57] "HTTP/FTP Seeding for BitTorrent" (<http://www.getright.com/seedtorrent.html>). . Retrieved 2010-03-18.
- [58] Michael Burford (2008-02-25). "*WebSeed - HTTP/FTP Seeding (GetRight style)* – BitTorrent Enhancement Proposal № 19" ([http://www.bittorrent.org/beps/bep\\_0019.html](http://www.bittorrent.org/beps/bep_0019.html)). . Retrieved 2012-02-17.
- [59] "Burn Any Web-Hosted File into a Torrent With Burnbit" (<http://torrentfreak.com/burn-any-web-hosted-file-into-a-torrent-with-burnbit-100913/>). TorrentFreak. 2010-09-13. . Retrieved 2012-07-09.
- [60] "PHP based torrent file creator, tracker and seed server" (<http://php-tracker.org/>). PHPTracker. . Retrieved 2012-07-09.
- [61] Gillmore, Steve. BitTorrent and RSS Create Disruptive Revolution (<http://www.eweek.com/article2/0,1895,1413403,00.asp>) *EWeek.com*, 13 December 2003. Retrieved on 22 April 2007.
- [62] Corante.com (<http://www.corante.com/importance/>)
- [63] Raymond, Scott: Broadcatching with BitTorrent (<http://web.archive.org/web/20040213093750/http://scottraymond.net/archive/4745>). *scottraymond.net*: 16 December 2003.
- [64] "Move Digital REST API" ([http://www.movedigital.com/docs/index.php/MoveDigital\\_API](http://www.movedigital.com/docs/index.php/MoveDigital_API)). Move Digital. . Retrieved 2006-05-09. Documentation.
- [65] "Prodigem Enclosure Puller(pep.txt)" (<http://web.archive.org/web/20060526130219/http://prodigem.com/code/pep/pep.txt>) (TXT). Prodigem.com. Archived from the original (<http://prodigem.com/code/pep/pep.txt>) on 2006-05-26. . Retrieved 2006-05-09. via Internet Wayback Machine
- [66] "Encrypting Bittorrent to take out traffic shapers" (<http://torrentfreak.com/encrypting-bittorrent-to-take-out-traffic-shapers/>). Torrentfreak.com. 2006-02-05. . Retrieved 2006-05-09.
- [67] Sales, Ben (September 2006). "ResTech solves network issues" (<http://www.studlife.com/archives/News/2006/09/27/ResTechsolvesnetworkissues/>). studlife.com. .
- [68] Comcast Throttles BitTorrent Traffic, Seeding Impossible (<http://torrentfreak.com/comcast-throttles-bittorrent-traffic-seeding-impossible/>), *TorrentFreak*, 17 August 2007
- [69] Broache, Anne (2008-03-27). "Comcast and Bittorrent Agree to Collaborate" ([http://www.news.com/8301-10784\\_3-9904494-7.html](http://www.news.com/8301-10784_3-9904494-7.html)). News.com. . Retrieved 2012-07-09.
- [70] Soghoian, Chris (2007-09-04). "Is Comcast's BitTorrent filtering violating the law?" ([http://www.cnet.com/8301-13739\\_1-9769645-46.html](http://www.cnet.com/8301-13739_1-9769645-46.html)). Cnet.com. . Retrieved 2012-07-09.
- [71] "Multitracker Metadata Entry Specification" (<http://www.bittornado.com/docs/multitracker-spec.txt>) (TXT). Bittornado.com. . Retrieved 2006-05-09.
- [72] Called MultiTorrents by indexing website myBittorrent.com (<http://www.mybittorrent.com/>)
- [73] "P2P:Protocol:Specifications:Multitracker" ([http://wiki.depthstrike.com/index.php/P2P:Protocol:Specifications:Multitracker#Bad\\_Implementations](http://wiki.depthstrike.com/index.php/P2P:Protocol:Specifications:Multitracker#Bad_Implementations)). wiki.depthstrike.com. . Retrieved 2009-11-13.
- [74] "DecentralizedRecommendation –" (<https://www.tribler.org/DecentralizedRecommendation>). Tribler.org. . Retrieved 2012-07-09.
- [75] "Hyperspaces for Object Clustering and Approximate Matching in Peer-to-Peer Overlays" (<http://www.cs.cornell.edu/People/egs/papers/hyperspaces.pdf>) (PDF). Cornell University. . Retrieved 2008-05-26.
- [76] "Cubit: Approximate Matching for Peer-to-Peer Overlays" (<http://www.cs.cornell.edu/~bwong/cubit/index.html>). Cornell University. . Retrieved 2008-05-26.
- [77] "Approximate Matching for Peer-to-Peer Overlays with Cubit" (<http://www.cs.cornell.edu/~bwong/cubit/tr-cubit.pdf>) (PDF). Cornell University. . Retrieved 2008-05-26.
- [78] Torrent Exchange ([http://wiki.bitcomet.com/Torrent\\_Exchange](http://wiki.bitcomet.com/Torrent_Exchange)). The torrent sharing feature of BitComet. Retrieved 2010-01-31.
- [79] Van Der Sar, Ernesto. "Thunder Blasts uTorrent's Market Share Away" (<http://torrentfreak.com/thunder-blasts-utorrents-market-share-away-091204/>). TorrentFreak. Archived (<http://www.webcitation.org/61iuRToZn>) from the original on 2011-09-15. . Retrieved 2011-09-15.
- [80] "Torrent Server combines a file server with P2P file sharing" (<http://www.turnkeylinux.org/torrentserver>). Turnkeylinux.org. . Retrieved 2012-07-09.
- [81] Anderson, Nate (1 February 2007). "Does network neutrality mean an end to BitTorrent throttling?" (<http://arstechnica.com/news.ars/post/20070201-8750.html>). Ars Technica, LLC. . Retrieved 2007-02-09.
- [82] Himabindu Pucha, David G. Andersen, Michael Kaminsky (April 2007). "Exploiting Similarity for Multi-Source Downloads Using File Handprints" (<http://www.cs.cmu.edu/~dga/papers/nsdi2007-set/>). Purdue University, Carnegie Mellon University, Intel Research Pittsburgh. . Retrieved 2007-04-15.

- [83] "Speed boost plan for file-sharing" (<http://news.bbc.co.uk/2/hi/technology/6544919.stm>). BBC News. 12 April 2007. . Retrieved 2007-04-21.
- [84] Johnston, Casey (2008-12-09). "Arstechnica.com" (<http://arstechnica.com/news.ars/post/20081209-bittorrent-has-new-plan-to-shape-up-p2p-behavior.html>). Arstechnica.com. . Retrieved 2012-07-09.
- [85] "The Piratebay is Down: Raided by the Swedish Police" (<http://torrentfreak.com/the-piratebay-is-down-raided-by-the-swedish-police/>). TorrentFreak. 31.05.2006. . Retrieved 2007-05-20.
- [86] "Technical report: An Estimate of Infringing Use of the Internet" ([http://documents.envisional.com/docs/Envisional-Internet\\_Usage-Jan2011.pdf](http://documents.envisional.com/docs/Envisional-Internet_Usage-Jan2011.pdf)). Envisional. 2011-01-01. . Retrieved 2012-05-06.
- [87] "Piracy Volume in 2011" (<http://ethicalfan.com/?p=78>). Ethical Fan. 2012-04-29. . Retrieved 2012-05-06.
- [88] Albanesius, Chloe (2012-04-30). "U.K. High Court Orders ISPs to Block The Pirate Bay" (<http://www.pcmag.com/article2/0,2817,2403749,00.asp>). PC Magazine. . Retrieved 2012-05-06.
- [89] "Searching for Malware in Bit Torrent" (<http://www.docstoc.com/docs/14960461/Searching-for-Malware-in-Bit-Torrent>). .
- [90] Håvard Vegge, Finn Michael Halvorsen and Rune Walsø Nergård (2009), *Where Only Fools Dare to Tread: An Empirical Study on the Prevalence of Zero-day Malware*, 2009 Fourth International Conference on Internet Monitoring and Protection

## Further reading

- Pouwelse, Johan; *et al.* (2005). "The Bittorrent P2P File-Sharing System: Measurements and Analysis" ([http://books.google.com/books?id=Dnw7E8xzQUMC&lpg=PA205&dq=The Bittorrent P2P File-Sharing System: Measurements and Analysis Johan Pouwelse , Pawel Garbacki, Dick Epema,&pg=PA205#v=onepage&q&f=false](http://books.google.com/books?id=Dnw7E8xzQUMC&lpg=PA205&dq=The+Bittorrent+P2P+File-Sharing+System:+Measurements+and+Analysis+Johan+Pouwelse,+Pawel+Garbacki,+Dick+Epema,&pg=PA205#v=onepage&q&f=false)). *Peer-to-Peer Systems IV*. Berlin: Springer. pp. 205–216. doi:10.1007/11558989\_19. ISBN 978-3-540-29068-1. Retrieved September 4, 2011.

## External links

- Official BitTorrent website (<http://www.bittorrent.com/>)
- Official BitTorrent Specification ([http://www.bittorrent.org/beps/bep\\_0003.html](http://www.bittorrent.org/beps/bep_0003.html))
- BitTorrent ([http://www.dmoz.org/Computers/Internet/File\\_Sharing/BitTorrent/](http://www.dmoz.org/Computers/Internet/File_Sharing/BitTorrent/)) at the Open Directory Project
- Interview with chief executive Ashwin Navin ([http://streaming.scmp.com/podcasting/upload/News\\_BitTorrent\\_june15.mp3](http://streaming.scmp.com/podcasting/upload/News_BitTorrent_june15.mp3))
- Unofficial BitTorrent Protocol Specification v1.0 (<http://wiki.theory.org/BitTorrentSpecification>) at [wiki.theory.org](http://wiki.theory.org)
- Unofficial BitTorrent Location-aware Protocol 1.0 Specification ([http://wiki.theory.org/BitTorrent\\_Location-aware\\_Protocol\\_1.0\\_Specification](http://wiki.theory.org/BitTorrent_Location-aware_Protocol_1.0_Specification)) at [wiki.theory.org](http://wiki.theory.org)
- Michal Czerniawski, Responsibility of Bittorrent Search Engines for Copyright Infringements ([http://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=1540913](http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1540913)), at SSRN (December 2009)
- Under the hood of BitTorrent (<http://www.stanford.edu/class/ee380/Abstracts/050216.html>) — lecture given by BitTorrent protocol designer, Bram Cohen at Stanford University ( video archive (<http://stanford-online.stanford.edu/courses/ee380/050216-ee380-100.asx>)).
- Tiny perl script to view contents inside torrent files (<http://wiki.gotux.net/downloads/btview>)

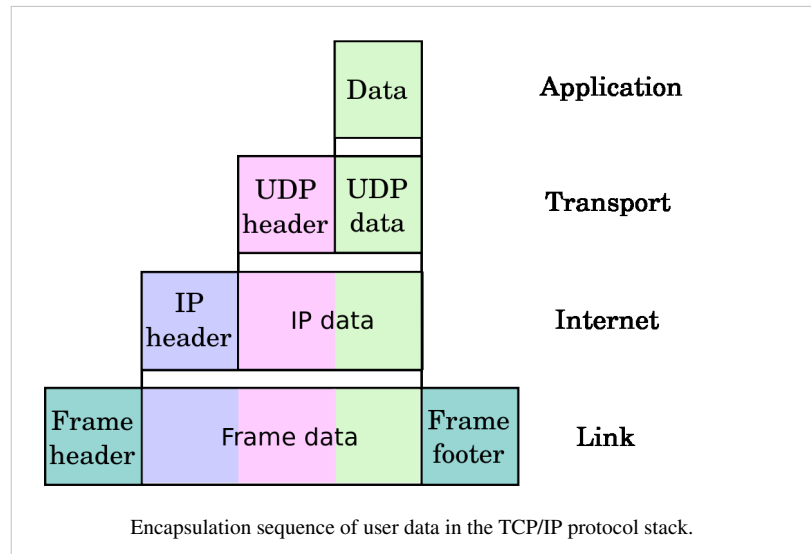


# Architectural Principles

## Encapsulation (networking)

In computer networking, **encapsulation** is a method of designing modular communication protocols in which logically separate functions in the network are abstracted from their underlying structures by inclusion or information hiding within higher level objects.

The physical layer is responsible for physical transmission of the data. Link encapsulation allows local area networking and IP provides global addressing of individual computers; UDP adds application or process selection, i.e., the port specifies the service such as a Web or TFTP server.



In discussions of encapsulation, the more abstract layer is often called the upper layer protocol while the more specific layer is called the lower layer protocol. Sometimes, however, the terms upper layer protocols and lower layer protocols are used to describe the layers above and below IP, respectively.

Encapsulation is a characteristic feature of most networking models, including the OSI Model and TCP/IP suite of protocols.

### External links

- How Encapsulation Works Within the TCP/IP Model <sup>[1]</sup>
- Animation of router encapsulation and decapsulation <sup>[2]</sup>

### References

- [1] <http://learn-networking.com/tcp-ip/how-encapsulation-works-within-the-tcpip-model>
- [2] <http://www.visualland.net/view.php?cid=1028&protocol=Encapsulation&title=2.%20Router%20encap&ctype=1>

# Packet switching

---

**Packet switching** is a digital networking communications method that groups all transmitted data – regardless of content, type, or structure – into suitably sized blocks, called *packets*. Packet switching features delivery of variable-bit-rate data streams (sequences of packets) over a shared network. When traversing network adapters, switches, routers and other network nodes, packets are buffered and queued, resulting in variable delay and throughput depending on the traffic load in the network.

Packet switching contrasts with another principal networking paradigm, circuit switching, a method which sets up a limited number of dedicated connections of constant bit rate and constant delay between nodes for exclusive use during the communication session. In case of traffic fees (as opposed to flat rate), for example in cellular communication services, circuit switching is characterized by a fee per time unit of connection time, even when no data is transferred, while packet switching is characterized by a fee per unit of information.

Two major packet switching modes exist; (1) connectionless packet switching, also known as datagram switching, and (2) connection-oriented packet switching, also known as virtual circuit switching. In the first case each packet includes complete addressing or routing information. The packets are routed individually, sometimes resulting in different paths and out-of-order delivery. In the second case a connection is defined and preallocated in each involved node during a connection phase before any packet is transferred. The packets include a connection identifier rather than address information, and are delivered in order. See below.

**Packet mode** communication may be utilized with or without intermediate forwarding nodes (packet switches or routers). In all packet mode communication, network resources are managed by statistical multiplexing or dynamic bandwidth allocation in which a communication channel is effectively divided into an arbitrary number of logical variable-bit-rate channels or data streams. Statistical multiplexing, packet switching and other store-and-forward buffering introduces varying latency and throughput in the transmission. Each logical stream consists of a sequence of packets, which normally are forwarded by the multiplexers and intermediate network nodes asynchronously using first-in, first-out buffering. Alternatively, the packets may be forwarded according to some scheduling discipline for fair queuing, traffic shaping or for differentiated or guaranteed quality of service, such as weighted fair queuing or leaky bucket. In case of a shared physical medium, the packets may be delivered according to some packet-mode multiple access scheme.

## History

The concept of switching small blocks of data was first explored by Paul Baran in the early 1960s. Independently, Donald Davies at the National Physical Laboratory (NPL) in the UK had developed the same ideas a few years later (Abbate, 2000).

Leonard Kleinrock conducted early research in queueing theory which would be important in packet switching, and published a book in the related field of digital message switching (without the packets) in 1961; he also later played a leading role in building and management of the world's first packet-switched network, the ARPANET.

Baran developed the concept of message block switching during his research at the RAND Corporation for the US Air Force into survivable communications networks, first presented to the Air Force in the summer of 1961 as briefing B-265<sup>[1]</sup> then published as RAND Paper P-2626<sup>[2]</sup> in 1962 and then including and expanding somewhat within a series of eleven papers titled On Distributed Communications<sup>[3]</sup> in 1964. Baran's P-2626 paper described a general architecture for a large-scale, distributed, survivable communications network. The paper focuses on three key ideas: first, use of a decentralized network with multiple paths between any two points; and second, dividing complete user messages into what he called *message blocks* (later called packets); then third, delivery of these messages by store and forward switching.

---

Baran's study made its way to Robert Taylor and J.C.R. Licklider at the Information Processing Technology Office, both wide-area network evangelists, and it helped influence Lawrence Roberts to adopt the technology when Taylor put him in charge of development of the ARPANET.

Baran's work was similar to the research performed independently by Donald Davies at the National Physical Laboratory, UK. In 1965, Davies developed the concept of packet-switched networks and proposed development of a UK wide network. He gave a talk on the proposal in 1966, after which a person from the Ministry of Defence (MoD) told him about Baran's work. A member of Davies' team met Lawrence Roberts at the 1967 ACM Symposium on Operating System Principles, bringing the two groups together.

Interestingly, Davies had chosen some of the same parameters for his original network design as Baran, such as a packet size of 1024 bits. In 1966 Davies proposed that a network should be built at the laboratory to serve the needs of NPL and prove the feasibility of packet switching. The NPL Data Communications Network entered service in 1970. Roberts and the ARPANET team took the name "packet switching" itself from Davies's work.

The first computer network and packet switching network deployed for computer resource sharing was the Octopus Network at the Lawrence Livermore National Laboratory that began connecting four Control Data 6600 computers to several shared storage devices (including an IBM 2321 Data Cell<sup>[4]</sup> in 1968 and an IBM Photostore<sup>[5]</sup> in 1970) and to several hundred Teletype Model 33 ASR terminals for time sharing use starting in 1968.<sup>[6]</sup>

In 1973 Vint Cerf and Bob Kahn wrote the specifications for Transmission Control Protocol (TCP), an internetworking protocol for sharing resources using packet-switching among the nodes.

## Connectionless and connection-oriented packet switching

The service actually provided to the user by networks using packet switching nodes can be either connectionless (based on datagram messages), or virtual circuit switching (also known as connection oriented). Some connectionless protocols are Ethernet, IP, and UDP; connection oriented packet-switching protocols include X.25, Frame relay, Multiprotocol Label Switching (MPLS), and TCP.

In connection-oriented networks, each packet is labeled with a connection ID rather than an address. Address information is only transferred to each node during a connection set-up phase, when the route to the destination is discovered and an entry is added to the switching table in each network node through which the connection passes. The signalling protocols used allow the application to specify its requirements and the network to specify what capacity etc. is available, and acceptable values for service parameters to be negotiated. Routing a packet is very simple, as it just requires the node to look up the ID in the table. The packet header can be small, as it only needs to contain the ID and any information (such as length, timestamp, or sequence number) which is different for different packets.

In connectionless networks, each packet is labeled with a destination address, source address, and port numbers; it may also be labeled with the sequence number of the packet. This precludes the need for a dedicated path to help the packet find its way to its destination, but means that much more information is needed in the packet header, which is therefore larger, and this information needs to be looked up in power-hungry content-addressable memory. Each packet is dispatched and may go via different routes; potentially, the system has to do as much work for every packet as the connection-oriented system has to do in connection set-up, but with less information as to the application's requirements. At the destination, the original message/data is reassembled in the correct order, based on the packet sequence number. Thus a virtual connection, also known as a virtual circuit or byte stream is provided to the end-user by a transport layer protocol, although intermediate network nodes only provides a connectionless network layer service.

## Packet switching in networks

Packet switching is used to optimize the use of the channel capacity available in digital telecommunication networks such as computer networks, to minimize the transmission latency (the time it takes for data to pass across the network), and to increase robustness of communication.

The best-known use of packet switching is the Internet and most local area networks. The Internet is implemented by the Internet Protocol Suite using a variety of Link Layer technologies. For example, Ethernet and Frame Relay are common. Newer mobile phone technologies (e.g., GPRS, I-mode) also use packet switching.

X.25 is a notable use of packet switching in that, despite being based on packet switching methods, it provided virtual circuits to the user. These virtual circuits carry variable-length packets. In 1978, X.25 provided the first international and commercial packet switching network, the International Packet Switched Service (IPSS). Asynchronous Transfer Mode (ATM) also is a virtual circuit technology, which uses fixed-length cell relay connection oriented packet switching.

Datagram packet switching is also called connectionless networking because no connections are established. Technologies such as Multiprotocol Label Switching (MPLS) and the resource reservation protocol (RSVP) create virtual circuits on top of datagram networks. Virtual circuits are especially useful in building robust failover mechanisms and allocating bandwidth for delay-sensitive applications.

MPLS and its predecessors, as well as ATM, have been called "fast packet" technologies. MPLS, indeed, has been called "ATM without cells".<sup>[7]</sup> Modern routers, however, do not require these technologies to be able to forward variable-length packets at multigigabit speeds across the network.

## X.25 vs. Frame Relay packet switching

Both X.25 and Frame Relay provide connection-oriented packet switching, also known as virtual circuit switching. A major difference between X.25 and Frame Relay packet switching is that X.25 is a reliable protocol, based on node-to-node automatic repeat request, while Frame Relay is a non-reliable protocol, maximum packet length is 1000 bytes. Any retransmissions must be carried out by higher layer protocols. The X.25 protocol is a network layer protocol, and is part of the X.25 protocol suite, also known as the OSI protocol suite. It was widely used in switching networks during the 1980s and early 1990s, for example as an alternative to circuit mode terminal switching, and for automated teller machines. Frame relay is a further development of X.25. The simplicity of Frame Relay made it considerably faster and more cost effective than X.25 packet switching. Frame relay is a data link layer protocol, and does not provide logical addresses and routing. It is only used for "semi-permanent" connections, while X.25 connections also can be established for each communication session. Frame Relay was used to interconnect LANs or LAN segments, mainly in the 1990s by large companies that had a requirement to handle heavy telecommunications traffic across wide area networks.<sup>[8]:250</sup> Despite the benefits of frame relay packet switching, many international companies are staying with the X.25 standard. In the United States, X.25 packet switching was used heavily in government and financial networks that use mainframe applications. Many companies did not intend to cross over to Frame Relay packet switching because it is more cost effective to use X.25 on slower networks. In certain parts of the world, particularly in Asia-Pacific and South America regions, X.25 was the only technology available.<sup>[9]</sup>

## References

- [1] Stewart, Bill (2000-01-07). "Paul Baran Invents Packet Switching" ([http://www.livinginternet.com/i/ii\\_rand.htm](http://www.livinginternet.com/i/ii_rand.htm)). *Living Internet*. . Retrieved 2008-05-08.
- [2] <http://www.rand.org/pubs/papers/P2626/>
- [3] [http://www.rand.org/pubs/research\\_memoranda/RM3420/index.html](http://www.rand.org/pubs/research_memoranda/RM3420/index.html)
- [4] The IBM 2321 Data Cell Drive (<http://www.columbia.edu/acis/history/datacell.html>), Columbia University Computing History
- [5] The IBM 1360 Photostore (<http://www.computer-history.info/Page4.dir/pages/Photostore.dir/index.html>), Lawrence Livermore Laboratory Computing History
- [6] Mendicino, Samuel (1970-11-30). "Octopus: The Lawrence Radiation Laboratory Network" (<http://www.rogerdmoore.ca/PS/OCTOA/OCTO.html>). . Retrieved 2009-05-06.
- [7] Interview with the author (of an MPLS-based VPN article) ([http://www.certificationzone.com/cisco/newsletter/SL/interview\\_08-12-03.html](http://www.certificationzone.com/cisco/newsletter/SL/interview_08-12-03.html)), G. Pildush
- [8] O'Brien, J. A. & Marakas, G. M. (2009). *Management Information Systems* (9th ed.). New York: McGraw-Hill/Irwin.
- [9] Girard, K. (1997, January). X.25 users remaining loyal despite frame-relay hype. *Computerworld*, 31(4), 16. Retrieved March 6, 2009, from ABI/INFORM Global database. (Document ID: 10946641).

## Bibliography

- Leonard Kleinrock, *Information Flow in Large Communication Nets* ([http://www.lk.cs.ucla.edu/bibliography-public\\_reports.html](http://www.lk.cs.ucla.edu/bibliography-public_reports.html)), (MIT, Cambridge, May 31, 1961) Proposal for a Ph.D. Thesis
- Leonard Kleinrock. *Information Flow in Large Communication Nets* (RLE Quarterly Progress Report, July 1961)
- Leonard Kleinrock. *Communication Nets: Stochastic Message Flow and Delay* (McGraw-Hill, New York, 1964)
- Paul Baran et al., *On Distributed Communications, Volumes I-XI* (<http://www.rand.org/about/history/baran-list.html>) (RAND Corporation Research Documents, August, 1964)
  - Paul Baran, *On Distributed Communications: I Introduction to Distributed Communications Network* (<http://www.rand.org/publications/RM/RM3420/>) (RAND Memorandum RM-3420-PR. August 1964)
- Paul Baran, *On Distributed Communications Networks* (<http://www.cs.ucla.edu/classes/cs217/Baran64.pdf>), (IEEE Transactions on Communications Systems, Vol. CS-12 No. 1, pp. 1-9, March 1964)
- D. W. Davies, K. A. Bartlett, R. A. Scantlebury, and P. T. Wilkinson, *A digital communications network for computers giving rapid response at remote terminals* (ACM Symposium on Operating Systems Principles. October 1967)
- R. A. Scantlebury, P. T. Wilkinson, and K. A. Bartlett, *The design of a message switching Centre for a digital communication network* (IFIP 1968)
- Larry Roberts and Tom Merrill, *Toward a Cooperative Network of Time-Shared Computers* (<http://www.packet.cc/files/toward-coop-net.html>) (Fall AFIPS Conference. October 1966)
- Lawrence Roberts, *The Evolution of Packet Switching* (<http://www.packet.cc/files/ev-packet-sw.html>) (Proceedings of the IEEE, November, 1978)

## Further reading

- Katie Hafner, *Where Wizards Stay Up Late* (Simon and Schuster, 1996) pp 52–67
- Janet Abbate, *Inventing the Internet* (MIT Press, 2000) ISBN 0-262-51115-0
- Arthur Norberg, Judy E. O'Neill, *Transforming Computer Technology: Information Processing for the Pentagon, 1962-1982* (Johns Hopkins University, 1996)

## External links

- Oral history interview with Paul Baran (<http://purl.umn.edu/107101>). Charles Babbage Institute University of Minnesota, Minneapolis. Baran describes his working environment at RAND, as well as his initial interest in survivable communications, and the evolution, writing and distribution of his eleven-volume work, "On Distributed Communications." Baran discusses his interaction with the group at ARPA who were responsible for

the later development of the ARPANET.

- Packet Switching History and Design ([http://www.livinginternet.com/i/iw\\_packet.htm](http://www.livinginternet.com/i/iw_packet.htm)), site reviewed by Baran, Roberts, and Kleinrock
- Paul Baran and the Origins of the Internet (<http://www.rand.org/about/history/baran.html>)
- A Brief History of the Internet (<http://www.isoc.org/internet/history/brief.shtml>)

This article is based on material taken from the Free On-line Dictionary of Computing prior to 1 November 2008 and incorporated under the "relicensing" terms of the GFDL, version 1.3 or later.

## Hostname

---

A **hostname** is a label that is assigned to a device connected to a computer network and that is used to identify the device in various forms of electronic communication such as the World Wide Web, e-mail or Usenet. Hostnames may be simple names consisting of a single word or phrase, or they may have appended a domain name, which is a name in a Domain Name System (DNS), separated from the host specific label by a period (dot). In the latter form, the hostname is also called a domain name. If the domain name is completely specified including a top-level domain of the Internet, then the hostname is said to be a fully qualified domain name (FQDN).

Example: 'saturn' and 'jupiter' could be the hostnames of a couple of devices connected to a network called Alpha. Within Alpha the devices are reached by their hostname. Alpha could be configured so that its domain name is 'alpha' (in lower case letters). In that case, the mentioned devices' hostnames, including the domain-name, would be: 'saturn.alpha' and 'jupiter.alpha', respectively. With these names they could be reached in a private network of networks, each with its own domain name. If alpha is registered and can be reached as alpha.net on the Internet, then the fully qualified domain names for the devices would be: 'saturn.alpha.net' and 'jupiter.alpha.net'

Hostnames that include DNS domains are often stored in the Domain Name System together with the IP addresses of the host they represent for the purpose of mapping the hostname to an address, or the reverse process.

### Overview

Hostnames are human-readable nicknames that correspond to the address of a device connected to a network. They are used by various naming systems, e.g., Network Information Service (NIS), Domain Name System (DNS), Server Message Block (SMB), and the meaning of *hostname* will vary according to the naming system used. A hostname meaningful to a Microsoft NetBIOS workgroup may be an invalid Internet hostname. When presented with a hostname without any context, it is usually safe to assume that the network is the Internet and the hostname's naming system is the DNS.

Host names are typically used in an administrative capacity and may appear in computer browser lists, active directory lists, IP address to hostname resolutions, email headers, etc.

### Internet hostnames

On the Internet, a hostname is a domain name assigned to a host computer. This is usually a combination of the host's local name with its parent domain's name. For example, *en.wikipedia.org* consists of a local hostname (*en*) and the domain name *wikipedia.org*. This kind of hostname is translated into an IP address via the local hosts file, or the Domain Name System (DNS) resolver. It is possible for a single host computer to have several hostnames; but generally the operating system of the host prefers to have one hostname that the host uses for itself.

Any domain name can also be a hostname, as long as the restrictions mentioned below are followed. So, for example, both *en.wikipedia.org* and *wikipedia.org* are hostnames because they both have IP addresses assigned to them. The domain name *pmtpa.wikimedia.org* is not a hostname since it does not have an IP address (as of now), but

*rr.pmtpa.wikimedia.org* is a hostname. A hostname may be a domain name, if it is properly organized into the domain name system. A domain name may be a hostname if it has been assigned to an Internet host and associated with the host's IP address.

## Restrictions on valid host names

Hostnames are composed of series of labels concatenated with dots, as are all domain names. For example, "en.wikipedia.org" is a hostname. Each label must be between 1 and 63 characters long,<sup>[1]</sup> and the entire hostname (including the delimiting dots) has a maximum of 255 characters.

The Internet standards (Request for Comments) for protocols mandate that component hostname labels may contain only the ASCII letters 'a' through 'z' (in a case-insensitive manner), the digits '0' through '9', and the hyphen ('-'). The original specification of hostnames in RFC 952, mandated that labels could not start with a digit or with a hyphen, and must not end with a hyphen. However, a subsequent specification (RFC 1123) permitted hostname labels to start with digits. No other symbols, punctuation characters, or white space are permitted.

While a hostname may not contain other characters, such as the underscore character (`_`), other DNS names may contain the underscore.<sup>[2]</sup> Systems such as DomainKeys and service records use the underscore as a means to assure that their special character is not confused with hostnames. For example, `_http._sctp.www.example.com` specifies a service pointer for an SCTP capable webserver host (www) in the domain example.com. Note that some applications (e.g. IE) won't work correctly if any part of the hostname will contain an underscore character<sup>[3]</sup>.

One common cause of non-compliance with this specification is that neither Microsoft Windows nor Android smartphones enforce the rule against using an underscore in hostnames. Since some systems will reject invalid hostnames while others will not, the use of invalid hostname characters may cause subtle problems in systems that connect to standards-based services. For example, RFC-compliant mail servers will refuse to deliver mail for MS Windows computers with names containing underscores.

The hostname *en.wikipedia.org* is composed of the DNS labels *en* (hostname or leaf domain), *wikipedia* (second-level domain) and *org* (top-level domain). Labels such as *2600* and *3abc* may be used in hostnames, but *-hi-* and *\*hi\** are invalid.

A hostname is considered to be a fully qualified domain name (FQDN) if all the labels up to and including the top-level domain name (TLD) are specified. The hostname `en.wikipedia.org` terminates with the top-level domain `org` and is thus fully qualified. Depending on the operating system DNS software implementation, an unqualified hostname such as *csail* or *wikipedia* may be automatically combined with default domain names configured into the system, in order to determine the fully qualified domain name. As an example, a student at MIT may be able to send mail to "joe@csail" and have it automatically qualified by the mail system to be sent to *joe@csail.mit.edu*.

General guidelines on choosing good hostnames are outlined in RFC 1178.

## References

- [1] RFC 1034, Section 3.1 "Name space specifications and terminology" (<http://tools.ietf.org/html/rfc1034#section-3.1>)
- [2] "Underscores in DNS" (<http://domainkeys.sourceforge.net/underscore.html>). . Retrieved 2011-07-20.
- [3] Internet Explorer Cookie Internals (FAQ) (<http://blogs.msdn.com/b/ieinternals/archive/2009/08/20/wininet-ie-cookie-internals-faq.aspx>)

# End-to-end principle

---

The **end-to-end principle** is a classic design principle of computer networking,<sup>[1]</sup> first explicitly articulated in a 1981 conference paper by Saltzer, Reed, and Clark.<sup>[2] [3]</sup>

The end-to-end principle states that application-specific functions ought to reside in the *end hosts* of a network rather than in *intermediary nodes* – provided they can be implemented "completely and correctly" in the end hosts. Going back to Baran's work on obtaining reliability from unreliable parts in the early 1960s, the basic intuition behind the original principle is that the payoffs from adding functions to the *network* quickly diminish, especially in those cases where the *end hosts* will have to re-implement functions for reasons of "completeness and correctness" anyway (regardless of the efforts of the network).<sup>[4]</sup> Moreover, there is an unfair performance penalty paid by all network clients when application functions of just a few clients are pushed into the intermediate nodes of a network.

The canonical example for the end-to-end principle is that of arbitrarily reliable file transfer between two communication end points in a distributed network of nontrivial size<sup>[5]</sup>. The only way two end points can obtain perfect reliability for this file transfer is by positive acknowledgment of end-to-end checksums over the final file in the destination storage locations on the destination machine. In such a system, lesser checksum and acknowledgment (ACK/NACK) protocols are justified only as a performance optimization, useful to the vast majority of clients, but are incapable of anticipating the reliability requirement of the transfer application itself (because said requirements may be arbitrarily high).

As an example of the end-to-end principle in practice, to reach Six Sigma reliability, file servers made by NetApp were forced to implement end-to-end checksums (making the disk drive checksums redundant) because the checksums used by certain disk makers were insufficient to meet a guarantee of Six Sigma reliability.

In debates about network neutrality a common interpretation of the end-to-end principle is that it implies a neutral or "dumb" network.

## Basic content of the principle

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 acknowledgements and retransmissions (referred to as PAR or ARQ).<sup>[6]</sup> Put differently, it is far easier and more tractable to obtain reliability beyond a certain margin by mechanisms in the *end hosts* of a network rather than in the *intermediary nodes*,<sup>[7]</sup> especially when the latter are beyond the control of and accountability to the former.<sup>[8]</sup> An end-to-end PAR protocol 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.<sup>[9]</sup>

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. Based on a personal communication with Saltzer (lead author of the original end-to-end paper<sup>[5]</sup>) Blumenthal and Clark in a 2001 paper note:<sup>[10]</sup>

[F]rom the beginning, the end-to-end arguments revolved around requirements that could be implemented correctly at the end-points; 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. (p. 80)



## History

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 prior to the seminal 1981 Saltzer, Reed, and Clark paper.<sup>[5]</sup>

### The basic notion: reliability from unreliable parts

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:<sup>[11]</sup>

Reliability and raw error rates are secondary. The network must be built with the expectation of heavy damage anyway. Powerful error removal methods exist. (p. 5)

Similarly, Davies notes on end-to-end error control:<sup>[12]</sup>

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. (p. 2.3)

### Early trade-offs: experiences in the Arpanet

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, and demonstrating a number of important aspects to the end-to-end principle:

Packet switching pushes some logical functions toward the communication end points

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 end points of such network. Consequently, the Arpanet featured two distinct levels of functionality – (1) a lower level concerned with transporting data packets between neighboring network nodes (called IMPs), and (2) a higher level concerned with various end-to-end aspects of the data transmission.<sup>[13]</sup> Dave Clark, one of the authors of the end-to-end principle paper, concludes:<sup>[14]</sup> "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" (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 end points of the network – much like a simple I/O channel between a computer and a nearby peripheral device.<sup>[15]</sup> 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,<sup>[16]</sup> no source to destination retransmission 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.<sup>[17]</sup> 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.<sup>[18]</sup>

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 require no perfect reliability, but rather value predictable throughput and low latency – 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.<sup>[19]</sup>

---

## The canonical case: TCP/IP

In the Internet the IP protocol – a connectionless datagram service with no delivery guarantees and effectively no QoS parameters – is used for nearly all communications. Arbitrary protocols may sit on top of IP. It turns out that some applications (such as voice, in many cases) do not need reliable retransmission, and so the only reliability in IP is in the checksum of the IP header (which is necessary to prevent bit errors from sending packets on wild routing paths.) End-to-end acknowledgment and retransmission is relegated to the connection-oriented TCP which sits on top of IP. The functional split between IP and TCP exemplifies proper application of the end-to-end principle to transport protocol design. In addition, to function properly networks must also have methods for shedding or rejecting loads that would cause the network to thrash and collapse (think "busy signal" on a telephone network.) The vast majority of applications on the Internet use TCP for communications. It was surprising that it took fully 7 years after TCP was standardized for Van Jacobsen and Karels to invent end-to-end congestion control algorithms for TCP, which adaptively and in a distributed fashion, scale back transmission rates to shed load from an overloaded Internet.

## Limitations of the principle

The most important limitation of the end-to-end principle is that its basic conclusion – put functions in the application end points rather than the intermediary nodes – is not trivial to operationalize. Specifically:

- it assumes a notion of distinct application end points as opposed to intermediary nodes that makes little sense when considering the structure of distributed applications;
- it assumes a dichotomy between non-application-specific and application-specific functions (the former to be part of the operations between application end points and the latter to be implemented by the application end points themselves) while arguably no function to be performed in a network is fully orthogonal to all possible application needs;
- it remains silent on functions that may not be implemented "completely and correctly" in the application end points and places no upper bound on the amount of application specific functions that may be placed with intermediary nodes on grounds of performance considerations, economic trade-offs, etc.

## Notes

[1] See Denning's Great Principles of Computing

[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] The 1981 paper was published in ACM's TOCS in an updated version in 1984. 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>)).

[4] The full quote from the Saltzer, Reed, Clark paper reads:

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)

- [5] 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.)
- [6] 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".
- [7] 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:

Clearly, some effort at the lower levels to improve network reliability can have a significant effect on application performance. (p. 281)

- [8] 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.
- [9] More precisely:

A correctly functioning PAR protocol with infinite retry count never loses or duplicates messages. [Corollary:] 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)

- [10] Blumenthal, M. S. and D. D. Clark (2001). "Rethinking the Design of the Internet: The End-to-End Arguments vs. the Brave New World". In: ACM Transactions on Internet Technology 1.1, pp. 70–109. ( Online pre-publication version (<http://mia.ece.uic.edu/~papers/Networking/pdf00002.pdf>)).
- [11] Baran, P. (1964). "On Distributed Communications Networks". In: IEEE Transactions on Communications 12.1, pp. 1–9.
- [12] 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.
- [13] In accordance with the Arpanet RFQ (pp. 47 f.) the Arpanet conceptually separated certain functions. As BBN point out in a 1977 paper:
 

[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)
- [14] 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 Application Design.ppt](http://cfp.mit.edu/events/may07/presentations/CLARK%20Application%20Design.ppt))).
- [15] 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/O 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. (p. 25)

- [16] 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. It is 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. (p. 11)

- [17] 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. (p. 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" (p. 7-28). See also BBN Report 2816 (pp. 10 ff.) for additional elaboration about the experiences gained in the first years of operating the Arpanet.

- [18] 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 thusly. 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 end points, 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, 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, pp. 56-58 and the notes therein, especially notes 151 and 163.
- [19] 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 (p. 2), 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 at pp. 55 f.). See also Bärwolff 2010, pp. 80-84 and the copious notes therein.

## References

### External links

- MIT homepage of Jerome H. Saltzer (<http://web.mit.edu/Saltzer/>) featuring publication list, working papers, biography, etc.
- Personal homepage of David P. Reed (<http://reed.com/>) featuring publication list, blog, biography, etc.
- MIT homepage of David D. Clark (<http://groups.csail.mit.edu/ana/People/Clark.html>) featuring publication list, working papers, biography, etc.

# Finite-state machine

A **finite-state machine** (FSM) or **finite-state automaton** (plural: *automata*), or simply a **state machine**, is a mathematical model of computation used to design both computer programs and sequential logic circuits. It is conceived as an abstract machine that can be in one of a finite number of *states*. The machine is in only one state at a time; the state it is in at any given time is called the *current state*. It can change from one state to another when initiated by a triggering event or condition, this is called a *transition*. A particular FSM is defined by a list of its states, and the triggering condition for each transition.

The behavior of state machines can be observed in many devices in modern society which perform a predetermined sequence of actions depending on a sequence of events they are presented with. Simple examples are vending machines which dispense products when the proper combination of coins are deposited, elevators which drop riders off at upper floors before going down, traffic lights which change sequence when cars are waiting, and combination locks which require the input of combination numbers in the proper order.

Finite-state machines can model a large number of problems, among which are electronic design automation, communication protocol design, language parsing and other engineering applications. In biology and artificial intelligence research, state machines or hierarchies of state machines have been used to describe neurological systems and in linguistics—to describe the grammars of natural languages.

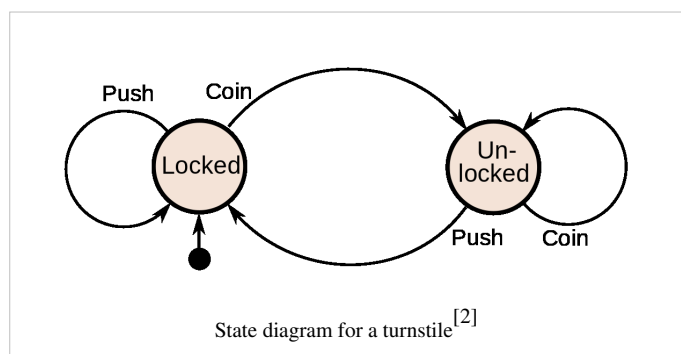
Considered as an abstract model of computation, the finite state machine is weak; it has less computational power than some other models of computation such as the Turing machine.<sup>[1]</sup> That is, there are tasks which no FSM can do but a Turing machine can do. This is because the FSM has limited memory. The memory is limited by the number of states.

FSMs are studied in the more general field of automata theory.

## Example: a turnstile

An example of a very simple mechanism that can be modeled by a state machine is a turnstile.<sup>[2][3]</sup> A turnstile, used to control access to subways and amusement park rides, is a gate with three rotating arms at waist height, one across the entryway. Initially the arms are locked, barring the entry, preventing customers from passing through. Depositing a coin or token in a slot on the turnstile unlocks the arms, allowing them to rotate by one-third of a complete turn, allowing a single customer to push through. After the customer passes through, the arms are locked again until another coin is inserted.

The turnstile has two states: **Locked** and **Unlocked**.<sup>[2]</sup> There are two inputs that affect its state: putting a coin in the slot (*coin*) and pushing the arm (*push*). In the locked state, pushing on the arm has no effect; no matter how many times the input *push* is given it stays in the locked state.



Putting a coin in, that is giving the machine a *coin* input, shifts the state from **Locked** to **Unlocked**. In the unlocked state, putting additional coins in has no effect; that is, giving additional *coin* inputs does not change the state. However, a customer pushing through the arms, giving a *push* input, shifts the state back to **Locked**.

The turnstile state machine can be represented by a state transition table, showing for each state the new state and the output (action) resulting from each input

Current State	Input	Next State	Output
<b>Locked</b>	coin	Unlocked	Release turnstile so customer can push through
	push	Locked	None
<b>Unlocked</b>	coin	Unlocked	None
	push	Locked	When customer has pushed through lock turnstile

It can also be represented by a directed graph called a state diagram (*above*). Each of the states is represented by a node (*circle*). Edges (*arrows*) show the transitions from one state to another. Each arrow is labeled with the input that triggers that transition. Inputs that don't cause a change of state (such as a *coin* input in the **Unlocked** state) are represented by a circular arrow returning to the original state. The arrow into the **Locked** node from the black dot indicates it is the initial state.

## Concepts and vocabulary

A *state* is a description of the status of a system that is waiting to execute a *transition*. A transition is a set of actions to be executed when a condition is fulfilled or when an event is received. For example, when using an audio system to listen to the radio (the system is in the "radio" state), receiving a "next" stimulus results in moving to the next station. When the system is in the "CD" state, the "next" stimulus results in moving to the next track. Identical stimuli trigger different actions depending on the current state.

In some finite-state machine representations, it is also possible to associate actions with a state:

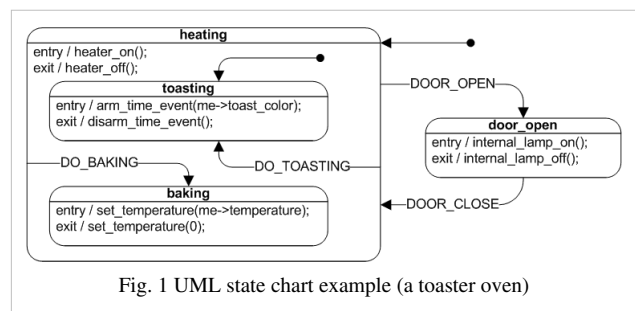
- Entry action: performed *when entering* the state,
- Exit action: performed *when exiting* the state.

## Representations

For an introduction, see *State diagram*.

### State/Event table

Several state transition table types are used. The most common representation is shown below: the combination of current state (e.g. B) and input (e.g. Y) shows the next state (e.g. C). The complete actions information is not directly described in the table and can only be added using footnotes. A FSM definition including the full actions information is possible using state tables (see also VFSM).



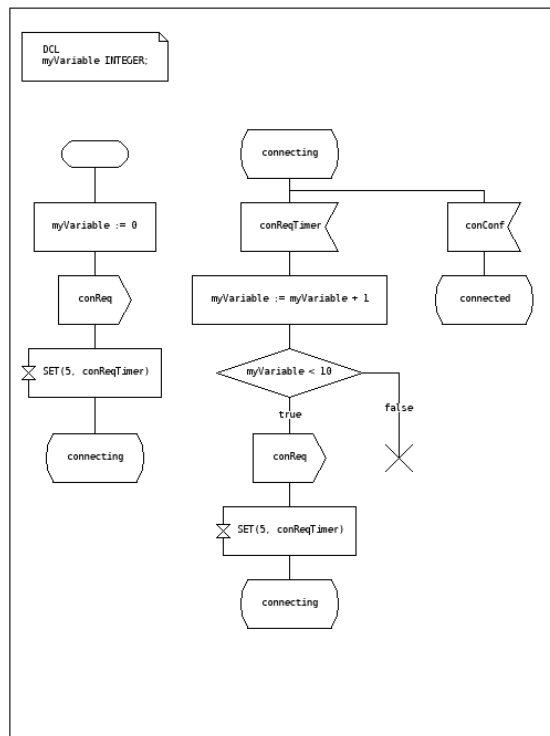


Fig. 2 SDL state machine example

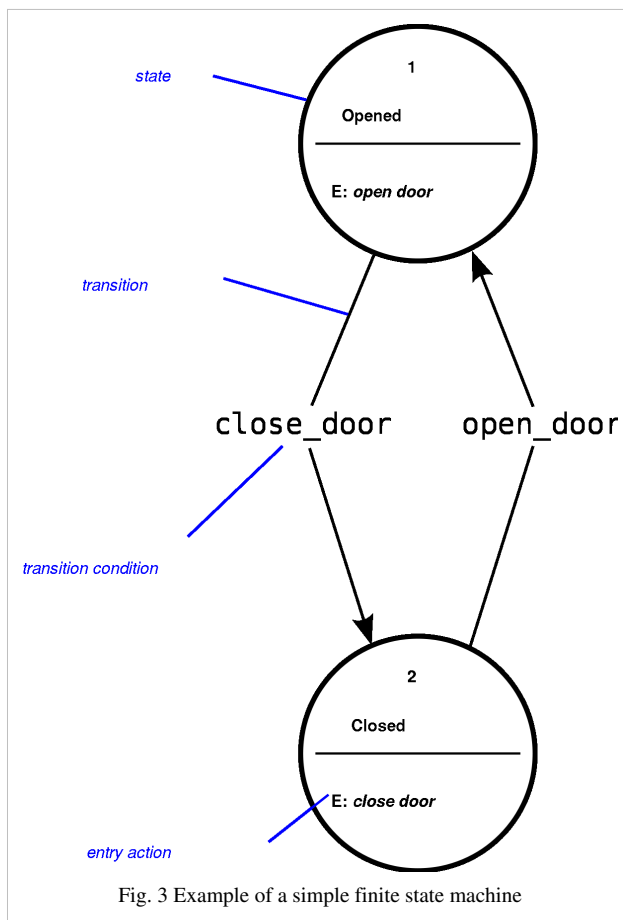


Fig. 3 Example of a simple finite state machine

**State transition table**

Current state → Input ↓	State A	State B	State C
Input X	...	...	...
Input Y	...	State C	...
Input Z	...	...	...

### UML state machines

The Unified Modeling Language has a notation for describing state machines. UML state machines overcome the limitations of traditional finite state machines while retaining their main benefits. UML state machines introduce the new concepts of hierarchically nested states and orthogonal regions, while extending the notion of actions. UML state machines have the characteristics of both Mealy machines and Moore machines. They support actions that depend on both the state of the system and the triggering event, as in Mealy machines, as well as entry and exit actions, which are associated with states rather than transitions, as in Moore machines.

### SDL state machines

The Specification and Description Language is a standard from ITU which includes graphical symbols to describe actions in the transition:

- send an event
- receive an event
- start a timer
- cancel a timer
- start another concurrent state machine
- decision

SDL embeds basic data types called Abstract Data Types, an action language, and an execution semantic in order to make the finite state machine executable.

### Other state diagrams

There are a large number of variants to represent an FSM such as the one in figure 3.

### Usage

In addition to their use in modeling reactive systems presented here, finite state automata are significant in many different areas, including electrical engineering, linguistics, computer science, philosophy, biology, mathematics, and logic. Finite state machines are a class of automata studied in automata theory and the theory of computation. In computer science, finite state machines are widely used in modeling of application behavior, design of hardware digital systems, software engineering, compilers, network protocols, and the study of computation and languages.



## Classification

There are two different groups of state machines: Acceptors/Recognizers and Transducers.

### Acceptors and recognizers

**Acceptors** and **recognizers** (also **sequence detectors**) produce a binary output, saying either *yes* or *no* to answer whether the input is accepted by the machine or not. All states of the FSM are said to be either accepting or not accepting. At the time when all input is processed, if the current state is an accepting state, the input is accepted; otherwise it is rejected. As a rule the input are symbols (characters); actions are not used. The example in figure 4 shows a finite state machine which accepts the word "nice". In this FSM the only accepting state is number 7.

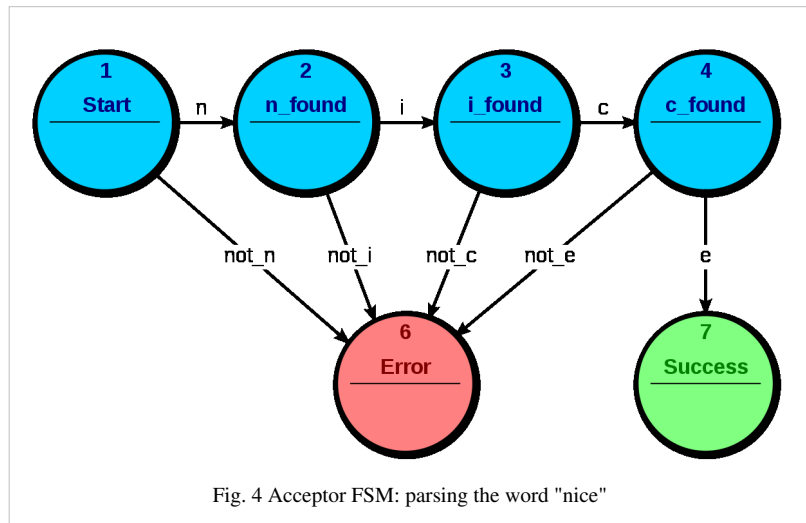


Fig. 4 Acceptor FSM: parsing the word "nice"

The machine can also be described as defining a language, which would contain every word accepted by the machine but none of the rejected ones; we say then that the language is *accepted* by the machine. By definition, the languages accepted by FSMs are the regular languages—that is, a language is regular if there is some FSM that accepts it.

### Start state

The start state is usually shown drawn with an arrow "pointing to it from any where" (Sipser (2006) p. 34).

### Accept (or final) states

**Accept states** (also referred to as **accepting** or **final** states) are those at which the machine reports that the input string, as processed so far, is a member of the language it accepts. It is usually represented by a double circle.

An example of an accepting state appears in the diagram to the right: a deterministic finite automaton (DFA) that detects whether the binary input string contains an even number of 0's.

$S_1$  (which is also the start state) indicates the state at which an even number of 0's has been input.  $S_1$  is therefore an accepting state. This machine will finish in an accept state, if the binary string contains an even number of 0's (including any binary string containing no 0's). Examples of strings accepted by this DFA are epsilon (the empty string), 1, 11, 11..., 00, 010, 1010, 10110, etc...

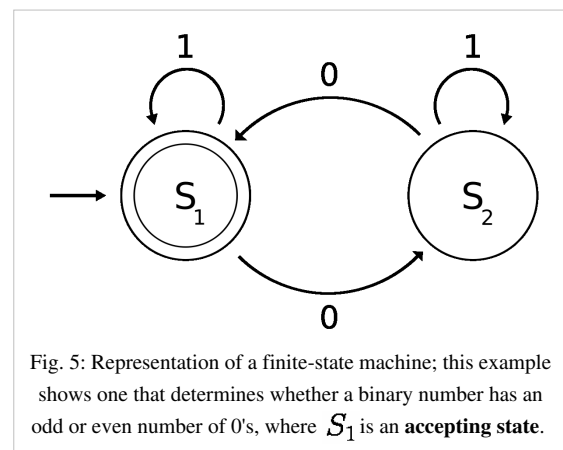


Fig. 5: Representation of a finite-state machine; this example shows one that determines whether a binary number has an odd or even number of 0's, where  $S_1$  is an **accepting state**.

## Transducers

Transducers generate output based on a given input and/or a state using actions. They are used for control applications and in the field of computational linguistics. Here two types are distinguished:

### Moore machine

The FSM uses only entry actions, i.e., output depends only on the state. The advantage of the Moore model is a simplification of the behaviour. Consider an elevator door. The state machine recognizes two commands: "command\_open" and "command\_close" which trigger state changes. The entry action (E:) in state "Opening" starts a motor opening the door, the entry action in state "Closing" starts a motor in the other direction closing the door. States "Opened" and "Closed" stop the motor when fully opened or closed. They signal to the outside world (e.g., to other state machines) the situation: "door is open" or "door is closed".

### Mealy machine

The FSM uses only input actions, i.e., output depends on input and state. The use of a Mealy FSM leads often to a reduction of the number of states. The example in figure 7 shows a Mealy FSM implementing the same behaviour as in the Moore

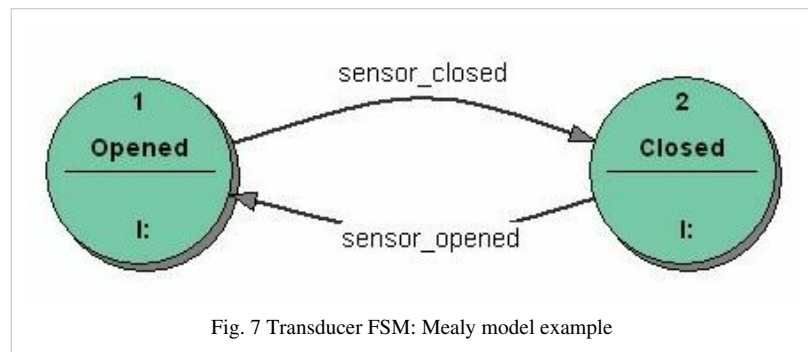


Fig. 7 Transducer FSM: Mealy model example

example (the behaviour depends on the implemented FSM execution model and will work, e.g., for virtual FSM but not for event driven FSM). There are two input actions (I:): "start motor to close the door if command\_close arrives" and "start motor in the other direction to open the door if command\_open arrives". The "opening" and "closing" intermediate states are not shown.

In practice<sup>[of what?]</sup> mixed models are often used.

More details about the differences and usage of Moore and Mealy models, including an executable example, can be found in the external technical note "Moore or Mealy model?"<sup>[4]</sup>

## Determinism

A further distinction is between **deterministic** (DFA) and **non-deterministic** (NFA, GNFA) automata. In deterministic automata, every state has exactly one transition for each possible input. In non-deterministic automata, an input can lead to one, more than one or no transition for a given state. This distinction is relevant in practice, but not in theory, as there exists an algorithm (the powerset construction) which can transform any NFA into a more complex DFA with identical functionality.

The FSM with only one state is called a combinatorial FSM and uses only input actions. This concept is useful in cases where a number of FSM are required to work together, and where it is convenient to consider a purely combinatorial part as a form of FSM to suit the design tools.

## Alternative semantics

There are other sets of semantics available to represent state machines. For example, there are tools for modeling and designing logic for embedded controllers.<sup>[5]</sup> They combine hierarchical state machines, flow graphs, and truth tables into one language, resulting in a different formalism and set of semantics.<sup>[6]</sup> Figure 8 illustrates this mix of state machines and flow graphs with a set of states to represent the state of a stopwatch and a flow graph to control the ticks of the watch. These charts, like Harel's original state machines,<sup>[7]</sup> support hierarchically nested states,

orthogonal regions, state actions, and transition actions.<sup>[8]</sup>

## FSM logic

The next state and output of an FSM is a function of the input and of the current state. The FSM logic is shown in Figure 8.

## Mathematical model

In accordance with the general classification, the following formal definitions are found:

- A *deterministic finite state machine* or *acceptor deterministic finite state machine* is a quintuple  $(\Sigma, S, s_0, \delta, F)$ , where:
  - $\Sigma$  is the input alphabet (a finite, non-empty set of symbols).
  - $S$  is a finite, non-empty set of states.
  - $s_0$  is an initial state, an element of  $S$ .
  - $\delta$  is the state-transition function:  $\delta : S \times \Sigma \rightarrow S$  (in a nondeterministic finite automaton it would be  $\delta : S \times \Sigma \rightarrow \mathcal{P}(S)$ , i.e.,  $\delta$  would return a set of states).
  - $F$  is the set of final states, a (possibly empty) subset of  $S$ .

For both deterministic and non-deterministic FSMs, it is conventional to allow  $\delta$  to be a partial function, i.e.  $\delta(q, x)$  does not have to be defined for every combination of  $q \in S$  and  $x \in \Sigma$ . If an FSM  $M$  is in a state  $q$ , the next symbol is  $x$  and  $\delta(q, x)$  is not defined, then  $M$  can announce an error (i.e. reject the input). This is useful in definitions of general state machines, but less useful when transforming the machine.

Some algorithms in their default form may require total functions. A finite-state machine is a restricted Turing machine where the head can only perform "read" operations, and always moves from left to right.<sup>[9]</sup>

A finite-state machine is a restricted Turing machine where the head can only perform "read" operations, and always moves from left to right.<sup>[9]</sup>

- A *finite state transducer* is a sextuple  $(\Sigma, \Gamma, S, s_0, \delta, \omega)$ , where:
  - $\Sigma$  is the input alphabet (a finite non empty set of symbols).
  - $\Gamma$  is the output alphabet (a finite, non-empty set of symbols).
  - $S$  is a finite, non-empty set of states.
  - $s_0$  is the initial state, an element of  $S$ . In a nondeterministic finite automaton,  $s_0$  is a set of initial states.
  - $\delta$  is the state-transition function:  $\delta : S \times \Sigma \rightarrow S$ .
  - $\omega$  is the output function.

If the output function is a function of a state and input alphabet ( $\omega : S \times \Sigma \rightarrow \Gamma$ ) that definition corresponds to the **Mealy model**, and can be modelled as a Mealy machine. If the output function depends only on a state ( $\omega : S \rightarrow \Gamma$ ) that definition corresponds to the **Moore model**, and can be modelled as a Moore machine. A finite-state machine with no output function at all is known as a semiautomaton or transition system.

If we disregard the first output symbol of a Moore machine,  $\omega(s_0)$ , then it can be readily converted to an output-equivalent Mealy machine by setting the output function of every Mealy transition (i.e. labeling every edge) with the output symbol given of the destination Moore state. The converse transformation is less straightforward because a Mealy machine state may have different output labels on its incoming transitions (edges). Every such state needs to be split in multiple Moore machine states, one for every incident output symbol.<sup>[10]</sup>

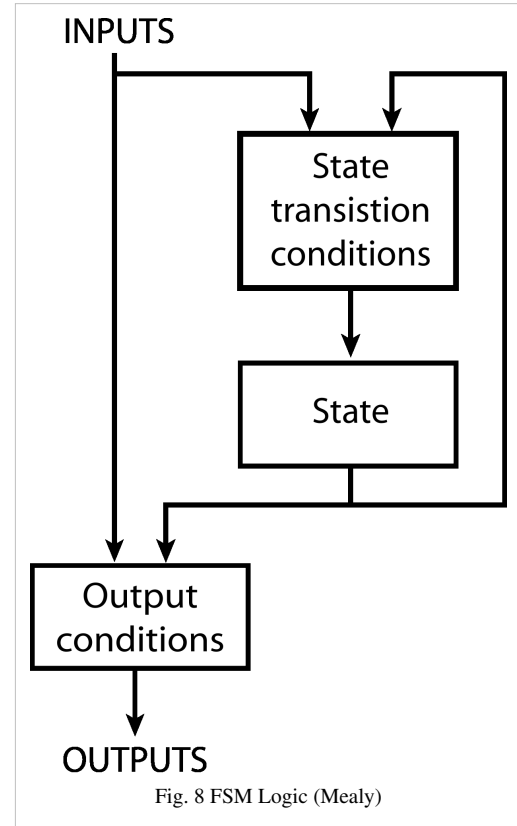


Fig. 8 FSM Logic (Mealy)

## Optimization

Optimizing an FSM means finding the machine with the minimum number of states that performs the same function. The fastest known algorithm doing this is the Hopcroft minimization algorithm.<sup>[11][12]</sup> Other techniques include using an implication table, or the Moore reduction procedure. Additionally, acyclic FSAs can be minimized in linear time Revuz (1992).<sup>[13]</sup>

## Implementation

### Hardware applications

In a digital circuit, an FSM may be built using a programmable logic device, a programmable logic controller, logic gates and flip flops or relays. More specifically, a hardware implementation requires a register to store state variables, a block of combinational logic which determines the state transition, and a second block of combinational logic that determines the output of an FSM. One of the classic hardware implementations is the Richards controller.

Mealy and Moore machines produce logic with asynchronous output, because there is a propagation delay between the flip-flop and output. This causes slower operating frequencies in FSM. A Mealy or Moore machine can be convertible to a FSM which output is directly from a flip-flop, which makes the FSM run at higher frequencies. This kind of FSM is sometimes called Medvedev FSM.<sup>[14]</sup> A counter is the simplest form of this kind of FSM.

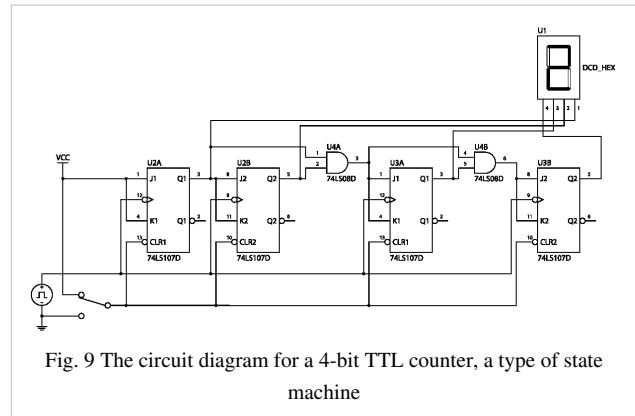


Fig. 9 The circuit diagram for a 4-bit TTL counter, a type of state machine

### Software applications

The following concepts are commonly used to build software applications with finite state machines:

- Automata-based programming
- Event driven FSM
- Virtual FSM (VFSM)

## References

- [1] Belzer, Jack; Albert George Holzman, Allen Kent (1975). *Encyclopedia of Computer Science and Technology*, Vol. 25 ([http://books.google.com/books?id=W2YLBIdELIEC&printsec=frontcover&dq="finite+state+machine"&source=bl&ots=7lcYvqsRvY&sig=19Ez1HGwvAm2HDaWYmbKDvod3Ec&hl=en&sa=X&ei=auwAUIWOFoezqQGglNGxDA&ved=0CFsQ6AEwBg#v=onepage&q="finite+state+machine"&f=false](http://books.google.com/books?id=W2YLBIdELIEC&printsec=frontcover&dq=)). USA: CRC Press. pp. 73. ISBN 0824722752. .
- [2] Koshy, Thomas (2004). *Discrete Mathematics With Applications* ([http://books.google.com/books?id=90KApidK5NwC&pg=PA762&dq=state+machine+turnstile&hl=en&sa=X&ei=4s70T\\_HtKIj1qAGr1cjbAw&ved=0CFgQ6AEwBA#v=onepage&q=state+machine+turnstile&f=false](http://books.google.com/books?id=90KApidK5NwC&pg=PA762&dq=state+machine+turnstile&hl=en&sa=X&ei=4s70T_HtKIj1qAGr1cjbAw&ved=0CFgQ6AEwBA#v=onepage&q=state+machine+turnstile&f=false)). Academic Press. pp. 762. ISBN 0124211801. .
- [3] Wright, David R. (2005). "Finite State Machines" (<http://www4.ncsu.edu/~drwright3/docs/courses/csc216/fsm-notes.pdf>). *CSC215 Class Notes*. Prof. David R. Wright website, N. Carolina State Univ.. Retrieved July 14, 2012.
- [4] <http://www.stateworks.com/technology/TN10-Moore-Or-Mealy-Model/>
- [5] Tiwari, A. (2002). Formal Semantics and Analysis Methods for Simulink Stateflow Models. (<http://www.csl.sri.com/users/tiwari/papers/stateflow.pdf>)
- [6] Hamon, G. (2005). "A Denotational Semantics for Stateflow". International Conference on Embedded Software. Jersey City, NJ: ACM. pp. 164–172. CiteSeerX: 10.1.1.89.8817 (<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.89.8817>).
- [7] Harel, D. (1987). A Visual Formalism for Complex Systems. *Science of Computer Programming* , 231–274. (<http://www.fceia.unr.edu.ar/asist/harel01.pdf>)

- [8] Alur, R., Kanade, A., Ramesh, S., & Shashidhar, K. C. (2008). Symbolic analysis for improving simulation coverage of Simulink/Stateflow models. *International Conference on Embedded Software* (pp. 89–98). Atlanta, GA: ACM. ([http://drona.csa.iisc.ernet.in/~kanade/publications/symbolic\\_analysis\\_for\\_improving\\_simulation\\_coverage\\_of\\_simulink\\_stateflow\\_models.pdf](http://drona.csa.iisc.ernet.in/~kanade/publications/symbolic_analysis_for_improving_simulation_coverage_of_simulink_stateflow_models.pdf))
- [9] Black, Paul E (12 May 2008). "Finite State Machine" (<http://www.nist.gov/dads/HTML/finiteStateMachine.html>). *Dictionary of Algorithms and Data Structures* (U.S. National Institute of Standards and Technology). .
- [10] James Andrew Anderson; Thomas J. Head (2006). *Automata theory with modern applications* (<http://books.google.com/books?id=ikS8BLdLDxIC&pg=PA105>). Cambridge University Press. pp. 105–108. ISBN 978-0-521-84887-9. .
- [11] Hopcroft, John E (1971). An  $n \log n$  algorithm for minimizing states in a finite automaton (<ftp://reports.stanford.edu/pub/ctr/reports/cs/tr/71/190/CS-TR-71-190.pdf>)
- [12] Almeida, Marco; Moreira, Nelma; Reis, Rogerio (2007). On the performance of automata minimization algorithms. (<http://www.dcc.fc.up.pt/dcc/Pubs/TRreports/TR07/dcc-2007-03.pdf>)
- [13] Revuz D. Minimization of Acyclic automata in Linear Time. *Theoretical Computer Science* 92 (1992) 181-189 Elsevier
- [14] "FSM: Medvedev" ([http://www.vhdl-online.de/tutorial/deutsch/ct\\_226.htm](http://www.vhdl-online.de/tutorial/deutsch/ct_226.htm)). .

## Further reading

### General

- Wagner, F., "Modeling Software with Finite State Machines: A Practical Approach", Auerbach Publications, 2006, ISBN 0-8493-8086-3.
- ITU-T, *Recommendation Z.100 Specification and Description Language (SDL)* (<http://www.itu.int/rec/T-REC-Z.100-200711-I/en>)
- Samek, M., *Practical Statecharts in C/C++* (<http://www.state-machine.com/psicc/index.php>), CMP Books, 2002, ISBN 1-57820-110-1.
- Samek, M., *Practical UML Statecharts in C/C++, 2nd Edition* (<http://www.state-machine.com/psicc2/index.php>), Newnes, 2008, ISBN 0-7506-8706-1.
- Gardner, T., *Advanced State Management* (<http://www.troyworks.com/cogs/>), 2007
- Cassandras, C., Lafortune, S., "Introduction to Discrete Event Systems". Kluwer, 1999, ISBN 0-7923-8609-4.
- Timothy Kam, *Synthesis of Finite State Machines: Functional Optimization*. Kluwer Academic Publishers, Boston 1997, ISBN 0-7923-9842-4
- Tiziano Villa, *Synthesis of Finite State Machines: Logic Optimization*. Kluwer Academic Publishers, Boston 1997, ISBN 0-7923-9892-0
- Carroll, J., Long, D., *Theory of Finite Automata with an Introduction to Formal Languages*. Prentice Hall, Englewood Cliffs, 1989.
- Kohavi, Z., *Switching and Finite Automata Theory*. McGraw-Hill, 1978.
- Gill, A., *Introduction to the Theory of Finite-state Machines*. McGraw-Hill, 1962.
- Ginsburg, S., *An Introduction to Mathematical Machine Theory*. Addison-Wesley, 1962.

### Finite state machines (automata theory) in theoretical computer science

- Arbib, Michael A. (1969). *Theories of Abstract Automata* (1st ed.). Englewood Cliffs, N.J.: Prentice-Hall, Inc.. ISBN 0-13-913368-2.
- Bobrow, Leonard S.; Michael A. Arbib (1974). *Discrete Mathematics: Applied Algebra for Computer and Information Science* (1st ed.). Philadelphia: W. B. Saunders Company, Inc.. ISBN 0-7216-1768-9.
- Booth, Taylor L. (1967). *Sequential Machines and Automata Theory* (1st ed.). New York: John Wiley and Sons, Inc.. Library of Congress Card Catalog Number 67-25924.
- Boolos, George; Richard Jeffrey (1989, 1999). *Computability and Logic* (3rd ed.). Cambridge, England: Cambridge University Press. ISBN 0-521-20402-X.
- Brookshear, J. Glenn (1989). *Theory of Computation: Formal Languages, Automata, and Complexity*. Redwood City, California: Benjamin/Cummings Publish Company, Inc.. ISBN 0-8053-0143-7.

- Davis, Martin; Ron Sigal, Elaine J. Weyuker (1994). *Computability, Complexity, and Languages and Logic: Fundamentals of Theoretical Computer Science* (2nd ed.). San Diego: Academic Press, Harcourt, Brace & Company. ISBN 0-12-206382-1.
- Hopcroft, John; Jeffrey Ullman (1979). *Introduction to Automata Theory, Languages, and Computation* (1st ed.). Reading Mass: Addison-Wesley. ISBN 0-201-02988-X.
- Hopcroft, John E.; Rajeev Motwani, Jeffrey D. Ullman (2001). *Introduction to Automata Theory, Languages, and Computation* (2nd ed.). Reading Mass: Addison-Wesley. ISBN 0-201-44124-1.
- Hopkin, David; Barbara Moss (1976). *Automata*. New York: Elsevier North-Holland. ISBN 0-444-00249-9.
- Kozen, Dexter C. (1997). *Automata and Computability* (1st ed.). New York: Springer-Verlag. ISBN 0-387-94907-0.
- Lewis, Harry R.; Christos H. Papadimitriou (1998). *Elements of the Theory of Computation* (2nd ed.). Upper Saddle River, New Jersey: Prentice-Hall. ISBN 0-13-262478-8.
- Linz, Peter (2006). *Formal Languages and Automata* (4th ed.). Sudbury, MA: Jones and Bartlett. ISBN 978-0-7637-3798-6.
- Minsky, Marvin (1967). *Computation: Finite and Infinite Machines* (1st ed.). New Jersey: Prentice-Hall.
- Christos Papadimitriou (1993). *Computational Complexity* (1st ed.). Addison Wesley. ISBN 0-201-53082-1.
- Pippenger, Nicholas (1997). *Theories of Computability* (1st ed.). Cambridge, England: Cambridge University Press. ISBN 0-521-55380-6 (hc).
- Rodger, Susan; Thomas Finley (2006). *JFLAP: An Interactive Formal Languages and Automata Package* (1st ed.). Sudbury, MA: Jones and Bartlett. ISBN 0-7637-3834-4.
- Sipser, Michael (2006). *Introduction to the Theory of Computation* (2nd ed.). Boston Mass: Thomson Course Technology. ISBN 0-534-95097-3.
- Wood, Derick (1987). *Theory of Computation* (1st ed.). New York: Harper & Row, Publishers, Inc.. ISBN 0-06-047208-1.

### **Abstract state machines in theoretical computer science**

- Yuri Gurevich (2000), *Sequential Abstract State Machines Capture Sequential Algorithms*, ACM Transactions on Computational Logic, vl. 1, no. 1 (July 2000), pages 77–111. <http://research.microsoft.com/~gurevich/Opera/141.pdf>

### **Machine learning using finite-state algorithms**

- Mitchell, Tom M. (1997). *Machine Learning* (1st ed.). New York: WCB/McGraw-Hill Corporation. ISBN 0-07-042807-7.

### **Hardware engineering: state minimization and synthesis of sequential circuits**

- Booth, Taylor L. (1967). *Sequential Machines and Automata Theory* (1st ed.). New York: John Wiley and Sons, Inc.. Library of Congress Card Catalog Number 67-25924.
- Booth, Taylor L. (1971). *Digital Networks and Computer Systems* (1st ed.). New York: John Wiley and Sons, Inc.. ISBN 0-471-08840-4.
- McCluskey, E. J. (1965). *Introduction to the Theory of Switching Circuits* (1st ed.). New York: McGraw-Hill Book Company, Inc.. Library of Congress Card Catalog Number 65-17394.
- Hill, Fredrick J.; Gerald R. Peterson (1965). *Introduction to the Theory of Switching Circuits* (1st ed.). New York: McGraw-Hill Book Company. Library of Congress Card Catalog Number 65-17394.

## Finite Markov chain processes

"We may think of a Markov chain as a process that moves successively through a set of states  $s_1, s_2, \dots, s_r, \dots$  if it is in state  $s_i$  it moves on to the next stop to state  $s_j$  with probability  $p_{ij}$ . These probabilities can be exhibited in the form of a transition matrix" (Kemeny (1959), p. 384)

Finite Markov-chain processes are also known as subshifts of finite type.

- Booth, Taylor L. (1967). *Sequential Machines and Automata Theory* (1st ed.). New York: John Wiley and Sons, Inc.. Library of Congress Card Catalog Number 67-25924.
- Kemeny, John G.; Hazleton Mirkil, J. Laurie Snell, Gerald L. Thompson (1959). *Finite Mathematical Structures* (1st ed.). Englewood Cliffs, N.J.: Prentice-Hall, Inc.. Library of Congress Card Catalog Number 59-12841. Chapter 6 "Finite Markov Chains".

## External links

- Modeling a Simple AI behavior using a Finite State Machine (<http://blog.manuvra.com/modeling-a-simple-ai-behavior-using-a-finite-state-machine/>) Example of usage in Video Games
  - Free On-Line Dictionary of Computing (<http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=finite+state+machine>) description of Finite State Machines
  - NIST Dictionary of Algorithms and Data Structures (<http://www.nist.gov/dads/HTML/finiteStateMachine.html>) description of Finite State Machines
  - Video lecture (<http://video.google.com/videoplay?docid=-5837841629284334824>)
-

---

--END WEEK ONE--

---



# Article Sources and Contributors

**Internet** *Source:* <http://en.wikipedia.org/w/index.php?oldid=516833827> *Contributors:* -Majestic-, 000o, Owaldo, 12.235.7.xxx, 16@r, 199.196.144.xxx, 1exec1, 2dogs, 3idiot, 424242, 75th Trombone, 802geek, 90, A. B., ABCD, AJR, Aaron Schulz, AaronKaplan, Abbos, Abrooke, AbstractClass, Acruzizer, Acdx, Acprisp, Acroterion, Adashiel, Administration, Adw2000, Aervanath, Agather, Agathoclea, Ageekgal, Ageing Geek, AgentPeppermint, Ahmad halawani, Ahoerstemeier, Ahruman, Aitias, Akamad, AlMac, Aladdin Sane, Alan Liefing, AlanBarrett, Alansohn, Aldie, AlefZet, Alem800, AlexPlante, Alexf, AlexiusHoratius, Alexwcovington, Alias Flood, AlistairMcMillan, All Is One, AllrOund, Allyoursanity, Alphachimp, Altzinn, Alvestrand, Alyska, Am088, Ambrose Brightside, AmiDaniel, Amillar, Amire80, Amitesh3011, Anagramology, Andonic, Andre Engels, Andre3k1, Andres, Andrevan, Andrew Gray, AndrewHowse, Andrewkorney, Andrewpmk, Andris, AndyBQ, Andyiou52345, Andypandy.UK, Anetode, AngelOfSadness, Angela, Angr, Animum, Ankiththegreat, Ann Stoutter, Anna Lincoln, Annexia, Anonymasty, Anonymous Dissident, Anonymous anonymous, Anonymous editor, AnotherDeadPerson, Antandrus, Anthony, Anthony Appleyard, Antimatter15, Antonrojo, Antony, joseph, Aphilo, Aranel, ArdClose, Arda Xi, Areeb cool, Arekku'xx, Arfan, ArielGold, Arnoel, Arsenal0328, Art LaPella, Aruton, Arx Fortis, Asdfghty, Ashenai, Asmo86, AstroHurricane001, Astroview120mm, Atgthatsme, Atomic23112, Atropos, Aujlakam, Aurelie Branchereau-Giles, Aurora Glory Paradise, Auroranorth, Autocracy, Avalean, Aviaris, Avochelm, Avraham, Avuton, Awolf002, AxelBoldt, Az1568, Azkar, AznShortBoi8021, B9 hummingbird hovering, BK08, BWD, BWP1234, Babyface123, Bad Bud, Bambuway, Bandan, Bandj, Barberio, Barefootguru, Barneca, Bart133, Bayerischermann, Baylink, Bband11th, Bbao, Bbatsell, Bcanoerrecart, Beao, Beland, Ben D., Bender235, Bennetchipper, Bentonmas214, Berezyne, Berk, Berro9, Bethenco, Bevo, Bfax, Bhuvanneshat, Biederman, Big Brother 1984, Bigbrotha3, Bigtimepeace, Bigwig77, Billion, Bimach, Biriturul, Bisqwit, Bitbit, Bjb, Blahma, Blake-, Bicarson, Blightsoot, Bloodshedder, Blossom the Awesome, Blowdart, BlueAg09, Bluejm2, Bluemask, Bluemoose, Bob A, Bob f it, Bobblewik, Bobet, Bobo192, Bodnotbod, Bogdangiusa, Bomazi, Bonadea, Bongwarrior, Bookh, Bookuser, Bordello, Borislav, BossOfTheGame, Brandmeister (old), Brandon5485, Brandork, Bratsche, Breakmoved, Brendan Moody, Breno, Brezzo, Briaboru, Brian0918, BrianRecchia, Brianga, BrokenSegue, Bry9000, Bryan Derksen, Burner0718, Bushytails, Bwaav, C'est moi, CF90, Cfreland, CJLL Wright, CLW, CMW275, CWY2190, Cahk, Calliopejen1, Callipides, Caltas, Cam809, CambridgeBayWeather, CameronHarris, Cameronmas214, Camc, Can't sleep, clown will eat me, Canadian-Bacon, Canaima, Capouch, Captain Disdain, Captainpancreas, Carbonite, CaribDigitA, Carlsoir, Carmen.banea, Casper2k3, Cbaxter1, Cdc, Cdxnolan, Cees webmaster, Cenarium, Centered1, Central2, Centrx, CesarB, Ceycocke, Cferro, Cgeorge316, Chabby, Chalybs, Chanting Fox, Charles7, Charlesblack, Charlesincharge, Chcknwrm, Chensiyuan, Cherkash, Cheung1303, ChewT, Chime Shinsen, Chmod007, Choaibatton, Chocolateboy, Chodorkovskiy, Chotisormmas214, Chris 73, Chris the speller, Chris21192, ChrisGualtieri, Christian List, Christopherlin, Chroniclev, Chun-hian, Chunky Rice, Cimon Avaro, Cirt, Citicat, Classicfilms, CloudworkSoul, Clothed so hardsm, CloudNine, Clpo13, CobraWiki, Cockip, Codemastercode, Codus, Coelacan, Collin, CommodoreAngry, CommonsDeflinker, Complex, Computageek95, Computerjoe, Computequip, Conversion script, Coolcaesar, Coolhawks88, Coolness107, CopperMurdoch, Corrupt one, Cortalux, Cotoco, Couchpotato99, Cpom, Crazycomputers, Crempuff222, Curps, Cuan, Cybaxter, Cybercobra, Cyberevil, Cyberitis, Cyclopia, Cynical, Cyp01, D. Recorder, DJ Clayworth, DVD R W, DaManWitDaPlan, Daemon8666, Dgvdadorj, Damian Yerrick, Damiens.rf, Dan larsen, Dana60Cummins, Daniel, Daniel5127, Danny, Dancscol, Danskil14, Darius Bacon, Darrendeng, Darth Vader, Daveb, Daveg, David.Monniaux, DavidJGW, Davidiad, Davodd, Dbeilhartz, Dchall1, DeadEyeArrow, Deedub1983, Deflagro, Dekisugi, Delirium, Delldot, Den fjättrade ankan, DerHexer, Derekdb, Dess, Dethem0w, Devonellie, Dforest, Dgwg, Dickguertin, Dicklyon, Diderot, Didimos, DigitalSorcerer, Digresser, Dijkstra, Diletante, Dillard421, Dinardi, Dinestyfaith, Dinosaur puppy, Dirkvdm, Discharger12, Discospinster, Diverock, Djegan, Dijkbox, Djr xi, Dlohciechierk, Dnas, Doc glasgow, Dodecki, Domstabsdogs, Dooky, Dorftrottel, Dorgan65, Dotancohen, DoubleBluu, Dpicquiro, Dr Debug, Dr Zen, Dr. Blofeld, Dr.K., Drangob123, Drappel, Drat, Dreadstar, Dream Focus, DreamGuy, Drew thomas amirault, Drew46, Driftwoodzebulin, DriveMySol, Drivera90, Dnm8, DropDeadGorgias, Drosera, Drphilharmonic, Drunkenmonkey, Dryice2000, Drzoidberg, Dtdcthingy, Dubious Irony, Duoduoduo, Dusso Janladde, Dust Filter, Dvc214, Dwheeler, Dylan Lake, Dylan W. E. Sn0 =31337=, EKN, ESKG, EVula, Eaaiscool, Eagardea, EagleOne, EamonnPKean, Ebaur5, Echuck215, Ederjar, Edgar181, Editmachine, Edonovan, Edwina Storie, Edwy, Eirik (usurped), Ekjon Lok, ElBenevolente, Elcobbola, Eleassar777, Elmidmibbs, Elonka, Eloquence, Eltonhoyantam, Emijrp, Enchantian, Enereccio, Entirety, Entrpy, EoGuy, Epa101, Epidown, Eran of Arcadia, Eric outdoors, Erik9, Erlikringham, Erikotto, EronMain, EryZ, Erzeneg, EthanNeuen, Eu.stefan, Eurosong, Euryalus, EventHorizon, Evercat, Everything, Evil Moon, Evil saltine, EvilZak, Evilphoenix, Excelsior f, Excirial, Exeunt, Expert at this, Expertu, Eybaybay, Eyu100, FF2010, Fab, Fabricationary, Fadookie, FaerieInGrey, Fan-1967, FancyPants, Fantasy, Faseidman, Favonian, FelixtheMagnificent, Fennec, Filelakeshoe, Firetrap9254, Fishal, Flash man999, Flowerpotman, Flubberdubbe, Fluppy, FlushinQwnzNyc, Flyingember, Foboy, Folajimi, Foodmarket, Fortethefourthversion, Fottrey5516, Fountain09, FrancoGG, Frank G Anderson, Frap, Frappyjohn, Frazzydee, Freakofnurture, Fredbauder, Fredrik, Freewayeric, FreplySpang, Freshacconi, Freshbakedpie, Freyr, Friejens, Frigglinn, Fujifisher, Fulldenced, FundieBuster, Funky Monkey, Fuzheado, Fvw, G0dswed, GDonato, GHE, GMR5, GULLman, Gadium, Gahuntly, Galwhaa, Gamahucheur, Gamefreek76, Gamer007, Gannzor, Gantster, Gardar Rurak, Garfield 80, Garfield226, Gary King, Gblxyz, Geape, Geekmax, GeorgeTheCar, Gianfranco, Giftlite, Gimmeurm0ney, Gingaspice, Gioto, Glenn, Gluttenus, Gnoluyr, Gobonobo, Gogaeles4321, Gogo madhur, Gogo Dodo, Golbez, Golfballock, Goobergunch, Good Olfactory, GoodStuff, Goodnewsfortheinsane, Goodnightmush, Goplat, Gppande, Gracenotes, GraemeL, Grafen, Graham87, Grandpafootsoldier, Grassmaker, Graue, GreatLiver, Greatlyfingsock, Green caterpillar, GregAsche, GregNorc, Grenavitar, Grey666, Grich, Grison, Gronky, Grunt, Gsarwa, Gscshoyru, Gsklee, Gsociology, Gtrmp, Guade00, Guaka, Guanaco, Guffydrawers, Gurch, Gurchzilla, Guthrie, Guy M, Guy Peters, Gwernol, Gwyllim a, H3llbringer, HJV, HackerOnSteroids69, Hadal, Hadians, Haemo, Haham hanuka, HairY Dude, Hallenrm, Handface, Hans Dunkelberg, HappyCamper, HappyInGeneral, Hardcoregayslaveman, HardwareBob, HarryMcDeath, Harrystons, Harryzilber, Haukurth, Hawaiian717, Hazhk, Haxr0x, Hdr83, Headbomb, Hegiiman, Heimstern, HeinzdderMann, Hekxcieksdl, Helios Entity 2, Helix84, Henry Flower, Hephaestos, Herdrick, Heron, Hetar, Hhhhgateam, Hhhhhh, HiB2Bornot2B, Hiddenfromview, Hiii98, Hirschtick3, HisSpaceResearch, Hm2k, Hobbit fingers, Hrvatska, HulleGranz, Hunter-Jing, Hydrajr, Hydrox, Hyperthermia, I Am Not Willy On Wheels, I amm Beowulf!, ICTlogist, INVERTED, INaNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, IRelayer, IanYossarian, Iceflamephoenix, Icewindfresnow, Iconoclact, Igoldste, Ihope127, Ilmari Karonen, Imacg3, Imthesmartduke, In4matt, InMooseWeTrust, InShaneue, Incnis Mrsi, Incompetence, Indon, Inkypaws, Insiriusdenial, Interflop, Intrg, Intiendes, Into The Fray, InverseHypercube, Invincible Ninja, Iridescent, Irish Souffle, Irrelevant, Intrav, Irvingbird, Isomega, Isotope23, Istiamar, Italianewy931, Ixfid64, Izwalito, J.delanoy, J.puckett, JEN9841, J.Hunter!, JIMBO WALEs, JLaTondre, JMD, JMJimmy, JRod451, Jacek Kendysz, Jack slack, Jacobko, JNimAtE, I

Reliablesources, Relly Komaruzaman, Remember the dot, Renaissance Man, Resslerdylan, Rettetast, ReXNL, ReyBrujo, Reymysterio01, Rfl, Rflorenc, Rhobite, Riana, Rich Farnbrough, Richard Weil, Richard Ye, RichardF, Richemond, Rick Block, RickK, Rinothan2, RjCan, Rjwilmsi, Rklawton, Robchurch, Robomaeyhem, RockMFR, Rockhall, Rockinduck, Ronz, Ronz91, Roozbezh, Rory096, Roscheray, RossPatterson, Roy Brumbach, RoyBoy, Roybadami, Rp, Rppgprog, Rrburke, Rsabbatini, Rscash22, RucasHost, Rud Hud Hudibras, Rumping, RunOrDie, RunningBon, Rursus, Ruud Koot, RxS, Ryan, Ryan Postlethwaite, Ryt, Ryulong, S M Woodall, SEOXpert, SJP, SNlyer12, SSZ, ST47, STA654, Saga City, Saghar2, Sam Li, Samir, Samj72, Samsara, Samuel Curtis, Samurai107, Sapocast, Sarenne, SasiSasi, Satori Son, Savantpol, Savidan, Sawdon2, Saxmaniac777, Sciepia, Sceptre, Schneelocke, Schoolproject2, SchulteMS214, Schzmo, Sciriurine, Scott981992, Scottbell, Scrolls, Scurra, Sdornan, Seav, Seg381, Seinfreak37, Senator2029, Septenia, Serpent-A, Seth Ilys, Seth662, Severa, Sfmmammia, Shadan, Shade11sayshello, Shadowlynk, Shady Hippo, Shane3x, ShaneCavanaugh, Shanes, Shangrilaista, Shantavira, Shantu123, Sharkticon Guy, Sharprs, ShaunES, Shawry 00, Shebaby102, Shengii, Shenme, Shervink, Shii, Shoeofdeath, Shreyash02, Sibbic, Sidepocket, SignorSimon, Siliconov, Simetrical, SimonP, SimonShlosberg, Sinn, Sinnerwiki, Sir Nicholas de Mimsy-Porpington, Sirkh1, Siroxo, Sjakalle, Sjjupadhay, Skb8721, SkyWalker, Skyhuperut, Sl, SI021, Slakr, Slark, Slayerteez, Sleepyasthesouth, Sloud74, Slowking Man, Sm8900, Smalljim, Sman24, Smartse, Smeira, Smilesfozwood, Smokizzy, Smoothy, Snoom haplub, SnowFire, Snoyes, Softlavender, Solipsist, Solitude, Solphusion, SomeStranger, Someguy1221, Someone35, Sonicsuns, Sonjaaa, Soph, Soptep, Soumyasch, Sparky the Seventh Chaos, Sparsh717, SparsityProblem, Spartan-James, Spazm, Spearhead, Speculatrix, Spedwehavedisabilaty, Speedysam, Spellcast, Spellmaster, Spencer, Spick And Span, SpigotMap, Splarka, Spliffy, SpookyMulder, SpuriousQ, SpyMagician, Spyke411, SqueakBox, Squiquifox, Sri555666, Srice13, Srmelson, Sstrader, St ttb, Stagefrog2, Staile, Stardust8212, Starnestommy, Stateguy, Steel, Steinberger, Stephan Leeds, Stephen, Stephen B Streater, Stephen Shaw, Stephenb, Stephenchou0722, Stereotek, Steverapaport, Stevey7788, Stevietheaman, Sthow, Stickguy, Stirling Newberry, Stormy Waters, Stratocracy, Striver, Strongirley, Student17, Stwalkerster, Subatomicguy, Suicidalhamster, Sukmas214, SummerPhD, SummerWithMorons, Sundar, Sunshine22 858, Sup chily monster, Superfreak37, Supertouch, Supten, SusanBrackman, Susurus, Suttipongkanasaimas214, Swollib, Sycthos, SymlynJ, Syvanen, T.J.V., THEODORMAS214, THEemu, TI85, Taimaster, TakuyaMurata, Tale, Tallungs, TangoFett, Tangotango, Tannin, Tapir Terrific, Tarret, Taskmaster99, Teamcoltra, Tech2blog, Technician2, Technobadger, Tedickey, Tehmechappe, Tehori, TekeeTakShak, Tenpin477, Terence, Tertz, Texture, Tfcollective, Thatdog, The Anome, The Flying Purple Hippo, The JPS, The Thing That Should Not Be, The Transhumanist, The Transhumanist (AWB), The teller of all 666, The undertow, The-secret-asian-man, The1pato, TheDoctor10, TheFloppyOne, TheJosh, TheKMan, TheLeopard, TheObtuseAngleOfDoom, TheRhani, TheStick, Theda, Thedjatclubrock, Theguns, Thehelpfulone, Themightyone, Thewallowmaker, Thingg, Thinker jones, Thiseye, Thparkth, Thyruuldf, Thue, Tiddly Tom, Tigerjag, Tim Chambers, TimMartin, Timir Saxa, Timknell, Tiptoety, Tiramisoo, Titoxd, Tkeller, Tkttkt, Tklynerd, Tmh, Tobias Bergemann, Tobias Wolter, Tocio, Tokyogirl119, Tomatote, Tomstein, Tomsega, Tony1, TonyClarke, TonySt, TonyTheTiger, Toolinguy, Tothwolf, Tovias, Tpbadbury, Traroth, Trashday03, TreasuryTag, Tregoweth, Trekkie711, Trevor MacInnis, Trollinator, Tronno, Trusilver, Tsavage, Tuuuuuuubies, Tverbeek, Twilsonb, Twobells, Tylermillersucks, Tzhoudeka, Tznkai, UBER, URBOSAUR, Uberdude85, UncleBubba, UniQue tree, United88, Untame Zerg, Unyoyega, UpTheBracket, Urhixidur, UrsuaLinguaBWD, Urzadek, Use the force, Usergreatpower, Utcursch, VRoscioli, Vald, Valermos, Valkyryn, Vanished user 39948282, Vary, Vegaswikian, Velvetron, Verdlanco, Versageek, Vespriantiano, Vgy7ujm, Viajero, Vidgmchtr, VigilancePrime, Vignaux, Vilerage, Villafanuk, Vinnie, Violetriga, Viriditas, Vjlenin, Vocation44, Voyager, Vulcanstar6, Vybr8, Vzbs34, W Nowicki, W163, W1k13rh3nry, WikiHorseMan, WJBScribe, WJctChao, WQDW412, Wafulz, Wagers, Waldir, Wangi, Wardron, Warhol13, Wavelength, Wawplok, Waxigloo, Wayne Hardman, WayneRay, Wayward, Wbm1058, We4reLegion, Wenli, Weregiberl, West Brom 4ever, West wikipedia, Weyes, Whale plane, Where, Whiney, WhisperToMe, Whitehorse1413, Whkoh, Wiggleintatter, Wiki Edit 12, Wiki alf, WikiWikiMan, Wikibob, Wikidemon, Wikidogia, Wikifun95, Wikigreatest, WikipedianMarlith, Wikipediaphile, Wikisanchez, Wikiwiki, Wiknerd, Wile E. Heresiarch, Will Beback Auto, WillMak050389, Willemdd, Willtron, Wimt, Wisejoker, Wisq, Wjfox2005, Wmahan, WojBob, Wolfkeeper, Wompa99, Wonderstruck, Woodshed, Wordsmith, Wossi, Wri0013, Wrs1864, Wutasumi, Www06, X1, XJAmRastafire, Xagent86, Xaosflux, Xavier86, Xaxafrad, Xeddy, Xeo-Wizzard, Xeysz, Xezbeth, Xkeeper, Xklsv, Xyzzyplugh, YTMND64, YUL89YYZ, Yahel Guhan, Yamaka122, Yamamoto Ichiro, Yansa, Yangharoth, Yatesie, Yath, Yelgrun, YellowMonkey, Yelyos, Yet-another-user, YixitiTephon, Yoganate79, Yonatan, Yosri, You must have cookies enabled to log in to Wikipedia., YourEyesOnly, Ytmnd6, Zachary, ZakTek, Zavon25, Zellfaze, Zeno McDohl, ZerD, Zerk-Tul, Zerbu, Zeronbarnotation, Zhente, Zhonghuo, Zib redlektab, Zidane tribal, ZimZalaBim, Zizzybaluba, Zmasterodinternetz, Zombi333, Zondor, Zundark, Zungaphile, Zzuuz, Zzyzx11, ^demon, 3030 anonymous edits محبوب عالم

**Internet protocol suite** *Source:* http://en.wikipedia.org/w/index.php?oldid=515653549 *Contributors:* 130.243.79.xxx, 203.109.250.xxx, 213.253.39.xxx, 66.169.238.xxx, A8UDI, Aapo Laitinen, Abdull, Abdullais4u, Acceptus, Acroterion, Aeonx, Ahoerstemeier, Alansohn, Albanaco, Aldie, Ale2006, Alek Baka, AliMaghrebi, Aliasptr, Alireza.usa, AlistairMcMillan, Amungale, Ana Couto, Aneah, Anna Lincoln, Anon lynx, Anorom, Arcenciel, ArchonMagnus, Arteitle, ArticCyant, Avand Guard, Avicennasis, Axxess, AxelBoldt, B4hand, Barberio, Barnacle157, Beland, Bender235, Bentogoa, Bernard François, Betterworld, Bezenek, Bhavin, Biot, Bloodshedder, Bmcomicp, Branko, Breno, Brian.fsm, Brion VIBBER, Camw, Canthusus, CaptainVindaloo, Carnildo, Casey Abell, Cate, Cburnett, Cf. Hay, Chadernook, Cheesycow5, Ckatz, Clark42, Coasting, Conversion script, Coolcaesar, CrinklyCrunk, Ctm314, Cybercobra, Cynthia Rhoads, DARTH SIDIOUS 2, Damian Yerrick, Daniel Staal, DanielCD, Darkhalfact, DavidDW, DavidDouthitt, Denisarona, DerekLaw, Dgtsyb, Dicklyon, Disavian, Dmerranda, Dnas, Dogcow, Donrude, Doradus, Dorgan65, Doug Bell, Drphilharmonic, Duffman, EagleOne, Ed g2s, Edmline, Edward, Edwardando, Eekster, Ekashp, Electron9, Elylwa, Elwood j blues, EnOreg, EncMstr, Enjoia4586, Epbr123, Eptin, Equendil, Ericl234, Erik Sandberg, Ethanthej, Etu, Evil Monkey, Evil saltine, Expensivehat, Falcon9x5, Falcor84, Ferkelparade, Fixman88, Fr34k, Freyr, GaelicWizard, Geneb1955, Gilliam, Glane23, Glenn, GlobalEdge 2010, Globemasterthree, Golbez, GordonMcKinney, Graham87, Gringo.ch, Gsl, Guy Harris, Haakon, Hadal, Haiy Dude, HarisM, Harryzilver, Hasty001, Hcberkowitz, Headbomb, Helix84, Here, Hore, Holylamposts, Hpnguyen83, Hyad, IMSoP, Ilario, Imcdnzl, Imran, Indeterminate, Indinking, Inundanceency, Inomyabcs, Intrgr, Itai, J.delanoy, JTN, Jackqu7, James Mohr, JamesBWatson, Jantangring, Jaktns, JesterXXV, Jimp, Jmdavid1789, Jnc, Joanjoc, John Vandenberg, Johnblade, Johnuniq, JonHarder, Jorunn, Jrogern, Jsoon ue, Jusdafax, JustAGal, KYPark, Kaare, Kasperd, Katieh5584, Kbrose, Kim Bruning, Kim Rubin, KnowledgeOfSelf, Kocio, Konman72, Koyaanis Qatsi, Krauss, Krellis, Kungming2, Kusma, Kving, Kyng, Labongo, Larree, Law, Layer, Leapfrog314, Lee Carre, Locketine, Logitheo, Lova Falk, Luna Santin, Magioladitis, Magister Mathematicae, Maltest, Mandarax, Mange01, Manop, Marcika, Martyman, Martyvis, Master Conjuror, Matt Dunn, Mattbrundage, Matthew Woodcraft, Matusz, Mav, Mckoss, Mechanical digger, Meiskam, Mendel, Merlissimo, Metaclassing, Michael Hardy, Miles, MilesMi, Mintguy, Mothmolevna, Mrzaies, Mukkakukaku, Mwarren us, Mzje, NMChico24, Nasz, Navedahmed123, NawlinWiki, Nealcardwell, Nealmb, NewEnglandYanke, Ngriffeth, Nhorton, Nick C, Niteowineils, Nivix, Nixdorf, Nknight, Nmacu, Nobody Ent, Northamerica1000, Nubiatech, Nr8200p, Obradovic Goran, Oheckmann, Olathe, Otets, OttoTheFish, Oxwil, Oxymoron83, Palfrey, Papadopa, Patilravi1985, Paul, Paul Koning, Paulkramer, Perfectpasta, Periputs, Pfalstad, Pharaoh of the Wizards, Phgao, Piano non troppo, PloM, Plugwash, Pokeywiz, Posflit, Pps, Public Menace, Punjabi101, Putdust, Quinxorin, R'nB, RaNo, Radagast83, Ramnath R Iyer, Raymond, Rbhagat0, Reaper Eternal, RedWolf, Reliableresources, RerVagnarok, Rich Farnbrough, Rich257, Renardwhiuk, Rick Block, RickJwidlmsi, RobEby, RobertG, RobertL30, Roberta F, Robost, Rodeosmurf, Ross Fraser, Rrelf, Rserpool, Runis57, Ruthherrin, SJP, STHayden, SWAdair, Samjoopin, SasiSasi, Seaphoto, Sheldrake, Shii, Shiraun, Shiro jdn, Shizhao, Sietse Snel, Sitush, Sjakalle, Skullketon, Smartse, Smjg, Snaxe920, SnoFox, Spmion, Stahla92, StanQuayle, Staszek Lem, Stefan Milosevski, Stephan Leeds, Stephenb, Suffusion of Yellow, Sully, Sunray, SuperWiki, Suruenna, Sveck, Whiteheadf, Swpb, Ta bu shi da yu, Tagishisimon, Tarquin, Tassedeth, Techmonk, Techpro30, Template namespace initialisation script, Tfi, That Guy, From That Show!, Thatguyflint, The Anome, The Nut, TheOtherJesse, Theresa knott, Thingg, Thumperward, Thunderboltz, Thw1309, Tide rolls, Tim Watson, Timwi, TinaSDCE, Tmaufer, Tmchck, Tobias Hoevekamp, TomPhil, Topbanana, Tr606, Tyler, Typhoon, Ukexpat, Unyoyega, Vanis314, Vegaswikian, Victor Liu, Violetriga, W163, Wadamja, Wagers, Wavelength, WeregEEK, West.andrew.g, Weyplin, Whereizenb, Wikid7, Wikiklrsc, William Avery, Wimt, Winston Chuen-Shih Yang, Wolfkeeper, Wonderstruck, Woohookitty, Wrs1864, XJAmRastafire, Xeehs, Xojx, Xosé, Yakudza, Yas, Ydalal, Yudiweb, Yunshui, ZNott, Zac439, Zeerak88, Zfr, Zigger, Zoicon5, Zondor, Zundark, Zvezda1111, ^demon, شات سوني, 831 anonymous edits

**OSI model** *Source:* http://en.wikipedia.org/w/index.php?oldid=516839240 *Contributors:* 0612, 0x6D667061, 1337 JN, 1966batfan, 24.12.199.xxx, 28bytes, 336, 63.227.96.xxx, 7, 75th Trombone, 802geek, 90 Auto, @modi, A412, A930913, ABF, Abarry, Abune, Adamantios, Addshore, Adibob, Adityagaar 7, Adj08, Adoniscik, Adrianwn, Advancedtelcovt, Ageekgal, Ahoerstemeier, Aitias, Ajo Mama, Ajw901, Alansohn, Albanaco, Aldie, Ale jrb, AlistairMcMillan, Allens, Alplachimp, Alucard 16, Alvestrand, Amillar, Amithbatia76, Amtanoli, Andjohn2000, Andre Engels, Andybryant, Angrysockhop, Animum, Anjola, AnkhhMorpork, Anna Lincoln, Anon lynx, Anonymous anonymous, Another-anomaly, Apocryphie, Apparition11, Arroww, Artur Perwenis, Arunachalammanohar, Ashutosh.mcse, Aslambasha09, Asn1tlv, AtomicDragon, Atreyu42, Audunn, Avitesh, AxelBoldt, Ayengar, B4hand, BACbKA, BDerry, Bakilas, Balajia82, Bariswheel, Bchnip, Bdamokos, Beelaj, BenLyanage, Beno1000, Biblbrosks, Bjelleklang, Bletch, Blueskies238, Bmylez, Bobo192, Bogdangiusca, Boikej, Bojer, BommelDing, Bonobosarenicer, Booyabazooka, Borge, Brambleclawx, Brandon, Brick Thrower, Brougham96, Bryan Derksen, BuickCenturyDriver, Bzimage.it, Bücherwürmlin, CDima, Cleland, CMBJ, Caerwine, Caesura, Calmer Waters, Caltas, CambridgeBayWeather, Camw, Can You Prove That You're Human, Can't sleep, clown will eat me, CanadianLinuxUser, Candc4, Caper13, Carre, Casey Abell, Causa sui, Cburnett, Cbustapeck, Cflm001, Charles Edward, Charm, Che090572, Chester Markel, Chfcalcao, Chimpex, Chirag, Chrislk02, Chupon, Ckicdragan, Citicat, Closedmouth, Cokoli, Cometstyles, Conquest ace, Conversion script, Coriron, Courcelles, Cputdoc, CraSH, CraigBox, Crashearl, Crimsonmargarine, Cs mat3, Ctboldt, Cxxl, Cybercobra, CyborgTosser, Cykthus, CynicalWiki, DARTH SIDIOUS 2, DJPohly, DSParillo, Damian Yerrick, Daniel, Danlev, Dave2, Davetrainer, David Edgar, David Gerard, David0811, DavidBarak, DavidLevinson, Davidjd, Dcooper, Dcovell, Deagle AP, Delfeye, Delldot, DeltaQuad, Demitsu, Denisarona, DennyColt, Dgtsyb, Dicklyon, Dili, Dino.korah, Discospinster, Dispenser, Djib, Djmoa, DmitryKo, Dmohantyatgmail, Doniag, Dpark, DrDOS, DrSpice, Drat, Dreish, Drwarpmind, Duey111, Dumbledad, Dzubint, EJDykzen, EJSawyer, ENeville, EagleOne, Eazy007, Ed g2s, EdH, Edivorce, Edward, ElKevbo, Eldiablo, Eleassar, Elfosardo, Eliezerb, Elipongo, Emperorbma, EnOreg, Enjoia4586, Enoclau, Epbr123, Eric Soyke, Everyking, Evillawngnome, Ewlyahoooom, Excirial, FF2010, Falk.H.G., Farg Aili, Feezo, Fiable.biz, Filemon, Finlay McWalter, Fjpanna, Fleg31789, Flewis, Flowanda, Fraggel81, FrankTobia, Fred Bradstadt, Fredrik, Free Bear, FreshPrinz, Fresheneesz, Friday, Friedo, Frigintor, Fullstop, Fumitot, Fzheado, Fvw, Fæ, GDonato, GGShinobi, Gadium, Gafex, GarethGilson, Gary King, Gasp01, Gazpacho, Geek2003, General Rommel, Ghosttalker, Giftlite, Gilliam, GlassCobra, Glenn, Goodnightmush, Graeme Bartlett, Grafen, Graham.rellinger, GreYfoXGTi, Grendelkhan, Grubber, Gsl, Gurchzilla, Guy Harris, Gwernol, Gökhan, H2g2bob, H34d, HMGb, Haakon, Hadal, HamatoKameko, HarisM, Hatch68, Hcberkowitz, Hdante, Helix84, Hellomarius, Henrikholm, Herbee, Heron, Hes Nikke, Hetar, HexaChord, Hgerstung, Hiddekel, Highpriority, Honeymey, I dream of horses, IMSoP, IReceivedDeathThreats, Iambk, Iambossatghari, Ideoplex, Ifroggie, Ilario, Immunize, Inkhorn, Inking, Insinaterhythm, Intrgr, Inversetime, InvisibleK, Inwind, Iridescent, Irvavash, IronGargoyle, Ishikawa Minoru, Island Monkey, Isofox, Isthishtingworking, Itpastorn, Itusg15q4user, Ivinney, J.delanoy, JMatthews, JV Smithy, Jake Wartenberg, JamesBWatson, Jannetta, Jatinchina, Jauerback, Jchrstn, Jcw69, Jdrmrk, JeTataMe, Jeanjour, Jeff G, Jeffrey Jay, Jessemerridan, Jetekus, Jhilvering, JidGom, Jim1138, Jimw338, Jjenkins5123, Jmorgan, Jcn, JoanneB, JodyB, Joebeone, John Hopley, John Vandenberg, John254, Johnblade, Johnleemk, Johnuniq, JohnHarran, Jonathanwagner, Jonwatson, Joodas, Josef Šábl cz, Josh Parris, Jovianeye, Joy, Jpta, Jrodor, Jschnur, Jschoon4, Jsonehd, Jusdafax, Kaaveh Ahangar, Kallaspriit, Kareklic, Karpouzi, Kaszeta, Katalaveno, Kaz219, Kazrak, Kbrose, Kcordina, KerryVeenstra, Kesla, Kevin Rector, Kgrr, Khat17, Killiondude, Kim Rubin, Kingpin13, Kirill Lokshin, Kkbairi, KnowledgeOfSelf, Kraftlos, Kramerino, Krampo, Krellis, Kuru, Kvng, Kyllis, LOL, LOTRnet, Lachlancooper, Lankivell, Lawrence Cohen, Lazarus666, Leafyplant, Lear's Fool, Lectoran, Lee Carre, Lighthouse, Lights, LittleOldMe, LittleWink, LizardJr8, Lockcole, Logitheo, Logthis, Lomn, Looxix, Lord Chamberlain, the Renowned, Lordeaswar, Lotje, Lulu of the Lotus-Eaters, Luna Santin, Lupin, Lynallendaly, M, MBisanz, MER-C, MIT Trekkie, Maguscrowley, Mahanga, Mahesh Sarmalkar, Majorly, Mange01, Manishar us, Marek9, Mark Sutton, MarkWahl, Markb, Markhard, Markolinsky, MartinHarper, Martinkop, Marvin01, Materialscientist, Mattallan, Mattanga Yeager, Mattgalloway, Mattmill30, Mbc362, Mboverload, McGinnis, Mcnutti, Mdd, Meepster, MelbourneStar, Mendel, Mephistophelian, Merlion444, Metaclassing, Micahcowan, Michael Hardy, Michael miceli, Mike Rosoff, Mikel Ward, Mikeo, Mikeyeh56, Milind m2255, Minimac, Mkweise, Mlewis000, Mmeeran, Mmernex, Mmmeg,

Mobius R, Mohitjoshi999, Mohitsport, Mojalefa247, Monterey Bay, Morten, Moxfyre, Mr Elmo, Mr Stephen, Mr.ghlban, MrOllie, Mrankur, MrsValdry, Mtd2006, MuZemike, Mudasir011, Mulad, Mwtoews, Myanaw, Myheadpainsincircles, N-Man, N5iln, Naishadh, Nanshu, Naohiro19, Naresah jangra, Nasa-verve, Natarajuab, Nate Silva, Nathanashleywid, NawlinWiki, Nbarth, Nbhata, Neevan99, Nejko, Nemesis of Reason, Nethgib, Netsnipe, Niax, Nick, Nickshanks, Nicolas1981, Nisavid, Nitecruzr, Nivix, Nk, Nkansahreford, Noahspurrier, Nolyann, Northamerical000, Nsaa, Nubiatech, NuclearWarfare, Nux, OSUKid7, Octahedron80, Odie5533, Ogress, Oita2001, OIEnglish, Omicronpersei8, Orange Suede Sofa, Ore4444, Originalharry, Ott, Ottosmo, Ouishoebcan, Oxymoron83, PGWG, Palltrast, Pamri, Panser Born, Paparodo, Parakalo, Pastore Italy, Patch1103, Patrikor, Patstuart, Paul August, PaulWIKIJeffery, Payal1234, Pb30, Peni, Penno, Pethr, Petrb, Phatom87, Phil Boswell, Philip Trueman, PhilipMW, Pluyo9899, Pmorkert, Pseudomonas, Psiphiorg, Public Menace, Puchiko, Puckly, PyreneesJIM, Pytom, RainbowOfLight, Raju5134, RandomAct, Ravikiran r, RazorICE, Rcannon100, Rebroad, Recognizance, RedWolf, Reedy, Rejax, Rettetast, Rfc1394, Rgilchrist, Rhobite, Rich Farmbrough, RichardVaryard, Richwales, Rick Sidwell, Rjgodoy, Rjstync, Rlaager, Rnbc, RobEby, Robert K S, RobertL30, RockMFR, Rohwigan03, Ronz, Roo314159, RoscoMck, RossPatterson, Roux, Roux-HG, RoyBoy, Rsiddharth, Runis57, Runtux, Rursus, Ryan au, Ryt, Ryulong, S, S3000, SMC, Saad ziyad, Saddy Dumpington, Safety Cap, Saintfiends, Sakurambo, Savh, SaxicolousOne, Scarian, Schumi555, Scientus, Scohoust, Scolobb, Scottonsocks, Seaphoto, Sesu Prime, Shadow1, Shadowjams, SharePointStacy, Shell Kinney, Shirik, Shoefdeath, ShornAssociates, Shradha deshमुख, Shrofami, Sietse Snel, Simonfl, Simple Bob, SineChristoNon, Sir Nicholas de Mimsy-Porpington, Skier Dude, Sliceofmiami, Slobertson, Smalljim, Smokizzy, SnowFire, Snowwolf, Soosed, Sp33dyphil, SpaceFlight89, Speaker to Lampposts, SpeedyGonsales, Spitfire8520, SpuriousQ, Sridev, StaticGull, Stemonitis, Stephan Leeds, Stephen Gilbert, StephenFalken, Stevage, Steven Zhang, StuartBrady, Subfrowns, Sunilmalik1107, Suruena, Suyashparth, Swapcouch, Syntaxsystem, TAS, THEN WHO WAS PHONE?, Tagishsimon, Tangatango, Tarekadi, Taruntan, Tbsdy lives, Tcnvc, Techtoucin, Tedickej, Tellyaddict, Tempodivalse, The Anome, The Athlon Duster, The Haunted Angel, The Thing That Should Not Be, Theopolismo, Therumakna, Thief12, Thingg, Think4mit, Threede912, ThunderBird, Tide rolls, Tim Q. Wells, TinyTimZamboni, Tom harrison, TomPhil, Tommy2010, Tompsci, Tony1, Tooki, TpbBradbury, Tpvibes, Tranzz, Travelbird, Tree Biting Conspiracy, Trevor MacInnis, Triona, TripleF, Triwbe, Troy 07, Turb0chrg, Tyler.szabo, UU, Umair ahmed123, Uncle Dick, Unknownid123, Vegaswagwag, Venu62, Versus22, VidGa, Vishnava, Visor, Vx anantha, Vmгурurасath, Voidxor, WLU, Waggars, Warrierrakesh, Wayfarer, Wergegibil, Whitejay251, WikiDan61, Wikipelli, William Avery, Willking1979, Wilson.canadian, Willy duck, Wingman417, Winston Chuen-Shih Yang, Wire323, Wireless friend, Wishington, Wizardist, Wknight94, WoiKiCK, Woohookitty, Wrlce, Wrs1864, Wtmitchell, Wtshymanski, Yamamoto Ichiro, YamiKaitou, Yamike, Yms, YolanCh, Yuokool12, ZX81, ZachPruckowski, Zachary, Zoobee79, བུ་མཚན་, ȳuman, 3273 anonymous edits

**Internet Protocol** *Source:* http://en.wikipedia.org/w/index.php?oldid=516149058 *Contributors:* 2w133, A-moll9, A. B., ARUNKUMAR P.R, Abb615, Abdull, Adagio Cantabile, Addihockey10, Aekton, Aggelos.Biboudis, Ahoerstermeier, Aldie, Altesys, Alvin-cs, Andareed, Andre Engels, Andywandy, Angelo.biboudis, Anon lynx, Antiuser, Anttin, Anwar saatad, Ardonik, Arkrishna, Attilios, BAXelrod, Badanedwa, Bdesham, Beefman, Bennor, Bentogoa, Biot, Bjornwiren, Blanchardb, Blehfu, Blue520, Bobguy7, Bobo192, Borislav, Brest, Brianga, Bridgecross, Brim, Brouhaha, Bryan Derksen, Bryanbuang1993, CALR, Caltech, Camilo Sanchez, Capricorn42, Caramdir, CarloJerna, Casey Abell, Cbodorset, Cburnett, Cdyson37, Cco, CesarB, Chealer, Chenzw, Christian List, Closedmouth, Conversion script, Coolcaesar, Coralmizu, Corruptcopper, Courcelles, Cverska, Cwofsheep, Cybercobra, Cyclonenim, Cynthia Rhoads, DARTH SIDIOUS 2, Dan D. Ric, Daniel Staal, Daniel.Cardenas, DanielCD, DeadEyeArrow, Defyant, Denial12358, Denisarona, Dgw, Dmaftei, Dnas, Doria, DrBag, Drugonot, Ducknish, Dwhewler, EagleOne, Echuck215, Eclipsed, Eequor, El C, Elfiggy, Enjo4586, Eprb123, Erodium, EverGreg, Evil saltine, FGont, Felipe1982, Ferkelparade, Fish147, Florentino floro, Formulax, Forton, Fredrik, FrummerThanThou, Fæ, Galoubet, Gamera2, Gareth Griffith-Jones, GaryW, General Wess, Giftlite, Glane23, Glenn, Goatbilly, Graciella, GraemeL, Grafen, GaramB7, Granbarreman, GreenRoot, Hairly Dude, Hardyplants, Harvester, Hayabusa future, Helix84, Hemanshu, Hetar, Hughcharlesparker, I am Me true, IRedRat, Iamregin, Imcdnzl, Imroy, Intgr, Ixf64, J.delanoy, JHolman, JTN, Jack Phoenix, Jadounrahul, JamesWatson, Jarble, Jasper Deng, Jauhienji, JavierMC, Jdforrester, Jedysurya, Jeff Carr, Jheiv, Jiddish, Jimgeorge, Jimys salonika, Jnc, Jno, JohnGrantNineTiles, JonHarris, Justsee, KD5TVI, Karl McClendon, Karol Langner, Katalaveno, Kbrose, Kgfleischmann, Kim Bruning, Kocio, Krellis, Kubigula, Kukini, Kvng, L Kensington, Latitudinarian, Liangent, Limitmagic1, Lion789, Looxix, Lotje, Love manjeet kumar singh, Maerk, Mahabub398, Mailtomeet, Mammad2002, Mange01, Manop, ManuelGR, Markr123, Marr75, Max Naylor, Melter, Mike Rosoff, Mikieiminnow, Mindmatrix, Mion, NGNWiki, Naive cynic, Nakon, NawlinWiki, NeoNorm, Nialldawson, Nightraider0, Nimiew, Nixdorf, Noformation, Noldoaran, Northamerical000, Nubcake, Nubiatech, Nugzthepirate, O, Ogress, Olathe, Otolumur crassicaudatus, Oxymoron83, Paolopal, Patrick, Paul, PaulHanson, PedroPVZ, Phatom87, Philip Trueman, Pinkadelica, Piotrus, Plustgarten, Poelq, Poweroid, Ppchailley, Python eggs, Rabbit67890, Recognizance, Reedy, Reub2000, Rgclegg, Rich Farmbrough, Richard Ye, Rjgodoy, Rkrirkorian, RobertG, Robocoder, Rsduhamel, Ryamigo, Ryanmcdaniel, Sandstein, SasiSasi, Scientizzle, Scientus, Scopecreep, Senator2029, Shiftoften66, Shlomiz, Shoefdeath, Sir Arthur Williams, Sjaak, SkyLined, SmilingBoy, Smyth, Some jerk on the Internet, Sonix0519, Sopus Bie, SpaceRocket, SpeedyGonsales, Spexios, Stephenb, Stevey7788, Supalex, Suruena, Sysiphe, THEN WHO WAS PHONE?, TakuyaMurata, Tarquin, Teles, Template namespace initialisation script, Tezdog, The Anome, The Transhumanist (AWB), Thorpe, Timwi, Tiuks, TkGy, Tobias Bergemann, Tommy2010, Torla42, Trevor MacInnis, Tristantech, TutterMouse, UncleBubba, Urvabara, V-ball, Varnav, Vary, Versageek, Versus22, VillemVillemVillem, Warren, Wayiran, Wayne Slam, Wcourtney, Weroon, Where, Whitepaw, Wik, Wikibob, Wikisierracharie, William Avery, Winston Chuen-Shih Yang, Wknight94, Woohookitty, Wrs1864, Wtmitchell, Yahooollian, Yamamoto Ichiro, Yausman, Yidisheryid, Yyy, Zarcillo, Zundark, Zzuuzz, شات صوني, བུ་མཚན་, 521 anonymous edits

**Transmission Control Protocol** *Source:* http://en.wikipedia.org/w/index.php?oldid=516667414 *Contributors:* 10metreh, 2620:0:102C:9:221:6AFF:FE6F:2758, 5 albert square, A-moll9, Al1vast, A5b, Access Denied, Acxd, AgadaUrbanit, Agent Koopa, Agi896, Ahson7, Akamad, Akill, Akshaymathur156, Alca Isilon, Aldie, Ale2006, Alex, Alexescalona, Alexh19740110, Alexius08, Alt-sysrq, Alvin-cs, Amalthea, Andre Engels, Andy M. Wang, Anichandran.mca, Anna Lincoln, AnotherNtPicker, Anwar saatad, Aprice457, Arnavchaudhary, Armhemcr, Arsenal9b0i, Asenine, Ashwinshadit, Astronomerren, Avono, Awy997, B4hand, BAXelrod, Banje, Beccus, Beland, Betterworld, Bezenek, Bigdumbindiosaur, Bilbo1507, Bill Malloy, BillMcGonigle, Biot, Bkell, Bkbbad, Blacksqr, Blanchardb, Boothy443, Boscobiscotti, Brech, Breno, Brighterorange, Brion VIBBER, Bstrand, Bubba hotep, Butros, C10191, Can't sleep, clown wi eat me, CanadianLinuxUser, Canthusus, Cbrettin, Cburnett, Centrew, Charles.partition, Chealer, Cheburashka, Christopher P, Cincapatrin, Ciprian Dorin Craciun, Cjdanield, Clark-gr, CobbSalad, Coinchon, Colonies Chris, Cometstyles, Conversion script, Cpaausch, CraSH, CrizCraig, D235, DKEdwards, Daev, Daniel Staal, Daniel.Cardenas, Danielbarnabas, Danielgrad, DariuszT, DarkAudit, DaveSymonds, David.bar, Dcoetzee, DeadEyeArrow, Dewet, Dgreen34, Dharmabum420, Dina, Djdawso, Dnas, DnetSvg, Doc Strange, Dori, DrHannibal216, Drake Redcrest, Duckbill, DylanW, Egg, Ego White Tray, Eirik (usurped), El pak, EncMstr, Encognito, Enjo4586, Enviroboy, Eprb123, Equendil, Eric-Wester, Erik Sandberg, Esmond.pitt, Evices, Evil Monkeys, Explicit, FDD, FGont, Fabiob, Fawcett5, Fcp999, Fishal, Fisherisland14, Fleminra, Foobaz, Fred Bradstadt, Frederico1234, Fredrik, Fredrikh, Frenecheigh, Fresheneesz, Fsiler, Fubar Obfusco, Gaius Cornelius, GalaxiaGuy, Gamera2, Garion96, GarryAnderson, Ghetoblaster, Giftlite, GilHamilton, Glenn Willen, Gmaran23, Gmaxwell, GoingBatty, Goplat, Graeme Bartlett, Graham.Fountain, Graham87, Guy Harris, Gwinnaadin, Haggis, Hairly Dude, Hamtechperson, Harput, HarrisonLi, Harryzilber, Harvester, Helix84, Henriktudborg, HiB2Bornt02B, Hilgerdenaar, I already forgot, I-bal'L, IOLJeff, IRP, Idcmp, Ideoplex, Iitywybmad, Imcdnzl, Imroy, Intgr, Inwind, Izn0, J. Nguyen, JCO312, JTN, Jaan513, Jackol, Nabus-causa, Jec, JeffClarkis, Jehorn, Jesant13, Jewbacca, Jfmantis, Jgeer, Jgrahn, Jjino, Jnc, JoanneB, Jogers, John Vandenberg, Johnuniq, JonHarder, Jonathan Hall, Jondel, Jowagner, Jsavage, Jtk, Juliancolton, JuneGloom07, Jvsdafax, Jxw13, Karada, Kartano, Karthick.s5, Kasperl, Katimawean2005, Kbrose, Kenyon, Kevinmon, Kgfleischmann, Kim Bruning, Kinema, Klhuillier, Krellis, Kubanczyk, Kumarat9pm, Kung, Kwamikagami, Kwiki, L Kensington, LachlanA, Lam Kim Keung, Lanilsson, Lark ascending, LeifZhu, LeoNomis, Leolaursen, Leszek Jarczuk, Leyo, LiDaobing, Lightdarkness, Lights, LilHelpa, Lilac Soul, Lime, Logicwax, Lokaci443, Lone boatman, Longuniongirl, Loor39, Loukris, Ltsampro2, Lucasbfr2, Luk, Luna Santin, Lunadesign, Lupo, M.S.K., MER-C, Mac, MacStep, Maclion, Maerk, Maimai009, Makomk, Maksym.Yehorov, Mange01, Manlyjacques, Manop, Marek69, Mark Bergsma, Markrod, Martijn Hoekstra, Marty Pauley, Materialscientist, Mattabab, Mboverload, Mbruck, MeekMark, Meekohi, Mendel, Mhändler, Mxycanthous, Oxymoron83, Pöper, PBSurf, PS3ninja, Padmini Gaur, Pak21, Palica, Panaplex, Pangelos, Paphos, Patar, Papadopa, PaulWay, Pde, PedroPVZ, Pegasus1138, Perfgeek, PeterB, Peytons, Pfalstad, Pg8p, Pgr94, Phandel, Phantomsteve, Phatom87, PhilKnight, Phuyal, Piano non troppo, Plainsong, Plugwash, Pluknet, Pmadrid, Poelq, Prakash mit, Prasannaxd, Prashant.khodade, Prunk, PureRumble, Purplefeltangel, Putdust, PxT, Quantumobserver, Quibik, Qutezuze, RA0808, Raul654, Rdone, RedWolf, Rettetast, RevRagnarok, Revolus, Rick Sidwell, Rjwilmsi, Rodowen, RodrigoCruzatti, Rogper, RoyBoy, Rtclawson, Ru.in.au, Ryan Stone, SHIMONSHA, SWAdair, Saaga, SamSim, Samuel, Sasha Callahan, Savagejumpin, Shnoble, Scientus, Scil100, Scoriundiain, Scrool, Sdomin3, Sepper, Sergiodc, Sethwm, Sh manu, Shaddack, Shadowjain, Shadowjams, Shultzc, Sietse Snel, Silverwindx, Sim, Sleigh, Smappy, Smsarmad, Smyth, Snowolf, Some Wiki Editor, Spearhead, SpeedyGonsales, Splint9, Spoon!, Squidish, StephenHemminger, Stephenb, Stevenwagner, Stonesand, StradivariusTV, StubbyT, Suruena, Svinodh, SwisterTwister, Synchrite, Syp, THEN WHO WAS PHONE?, Talel Atlas, TangentCube, Tariqajbotu, Tasc, Technobadger, TediousFellow, Teles, Template namespace initialisation script, Tfl, The Anome, The Monster, The Thing That Should Not Be, The-tenth-zdog, TheVoid, Thine Antique Pen, Tide rolls, Timotheus Canens, Timwi, Toh, Tomchiuck, Tommy2010, Torla42, Trou, TuukkaH, Umar420e, UncleBubba, Unixguy, Urmajest, Ursushoribilis, Uruaiame, Vedantm, Velella, Via strass, Vinu Padmanabhan, Vrenator, WLU, WikHead, WikiLaurent, Wimt, Wkcheang, Wlgrin, WojBob, Wolfkeeper, Woohookitty, Wrs1864, Xaphnir, Ymiaji, Yonatan, Youpilot, Ysangkok, Yyy, Zachlipton, ZeroOne, Zeroboo, Zundark, Zvar, شات صوني, 1136 anonymous edits

**User Datagram Protocol** *Source:* http://en.wikipedia.org/w/index.php?oldid=514987674 *Contributors:* 1exec1, 28421u2232nfencenc, A5b, Aapo Laitinen, Alfio, Ali@gwc.org.uk, Alison22, Alvin-cs, Andrew Kember, Aram33, Ardonik, Areh adeola, Ashmoo, Ayeroxor, BAXelrod, Beacon11, Beefman, BenAveling, Bender235, Benqmonitor, Bentogoa, Bobby Cadwell, Bobstopper, Borgx, Breno, Bryan Derksen, Burisch, Cassan, Cburnett, Chenzw, Chiefmanzzz, Cincapatrin, Conrad Irwin, Conversion script, CraSH, CrookedAsterisk, Calvda, DSatz, Damian Yerrick, Daniel Staal, DavidSol, DeadEyeArrow, DennyColt, DesTroy, Dfurbeck, Dicentra, Djordjes, Dkogh, E Wing, E090, Eagleal, Eclipsed, Enjo4586, Erik Garrison, Esmond.pitt, Falcon9x5, Fennec, Fightn' Phillie, Flewis, Fredrikh, FrisoB, Frodet, Fromagestciel, Fubar Obfusco, Fudgefr, Gamera2, Giftlite, Ginkgo100, Gniphallir, Golwengaud, Gordoni, Gorryf, Graceofol, Graham87, Gsmgm, Guy Harris, Hdante, Headbangerkenny, Hitherebrian, I dream of horses, IRedRat, Icestorm815, Ideoplex, InverseHypercube, Irrypride, Itai, JNighthawk, JTN, Jengelh, Jjino, Jncraton, Joachim Wuttke, Joanjoc, Johnny O, Johnuniq, Jtk, Jóna Þórunn, Kbrose, Kgfleischmann, Kinema, Koviano, Krellis, Kvng, Kwamikagami, Kyng, Lissajous, LI1324, Locos epraix, MLeb, Mahjongg, Mam711, Mange01, Mani1, Materialsscientist, Matt Darby, Michael Zimmermann, Mike Dill, Mike6271, Misterdom, Modulatum, Mpeg4codec, Mwholt, Neile, Nixdorf, Nroets, Nubiatech, Od Mishehu, Oumitch, Oxymoron83, PatheticCopyEditor, Paulish, PedroPVZ, Pelesl, Petri Krohn, Phatom87, Plindemann, Plugwash, Pocuscualin, Public Menace, Quiddity, R2richar, Razorflame, Recurring dreams, RedWolf, Rememberway, Reub2000, RevRagnarok, Reywas92, Rick Sidwell, Robbe, RobertL30, Romanm, Ronz, Rotlink, Rwww, S-n-ushakov, SWAdair, Samersarhan83, Sammydee, Samuel, Sci13960, Smeggygasm, Spearsall, Ssd293, Suruena, Tdangkhoa, Teapeat, TechyOne, Teemuk, Teh roflmaor, Template namespace initialisation script, Terrycoons, The Anome, The Monster, The Utahraptor, The-tenth-zdog, ThePolymis, Thray, Tjdw, Tobias Bergemann, Towel401, TpbBradbury, Twinxor, UncleBubba, Upadhyaykc, Vanished user 5zariu3jis0j4iirj, Vinu Padmanabhan, Voidxor, WestwoodMatt, Wiarthurhu, Wingman417, Wkcheang, Xezbeth, Yelajakit, Zara1709, ^demon, Александр, Тиверополник, 352 anonymous edits

**Internet Control Message Protocol** *Source:* <http://en.wikipedia.org/w/index.php?oldid=516828139> *Contributors:* Adamianash, Ajk91, Alerante, Alvin-es, Andareed, Anggarda, ArielGold, Athaenara, B20180, Beowulf king, Bgraabek, BobHackett, Bradycardia, Caesura, Celtkin, Chealer, Conversion script, Courcelles, Dark knight, DeadEyeArrow, Dnevil, DrThompson, Drj, Electrocute, Enjoia586, Face, Favonian, Fbriere, Forton, Fresheneesz, Fubar Obfusco, Goreatic, Graham87, Grenavitat, Gwern, Hari36533, Hawaiiaan717, Huwr, Jdelaney, JTN, Jadhav.m, Jasminek, Jay-Sebastos, Jengelh, Inc, Joseanes, Khrose, Kenyon, Kgfleischmann, Kinema, Kku, LAX, Mahboud, Mange01, Matusz, Mayonaise15, Meand, Mintleaf, Monkeymaker, Monkeyman, Mr link, Mremai1964, My76Srat, Naveenpf, Nimiew, Niteowneils, Nixdorf, Nuno Tavares, Omar35880, Oskilian, PeterB, Phatom87, Philleb, Piper8, Plustigarten, Pmsyyz, Pointy haired fellow, Polluks, Poweroid, Promethean, Qubik, Qwertys, RHaworth, Ranjiths, Rigodory, Romanm, SarahEmm, Shellreef, Sietse Snel, Sioux.cz, SixSix, Suruena, Template namespace initialisation script, The Anome, Tjpayne, Tobias Bergemann, Tstojano, TuukkaH, UncleBubba, Unixguy, Verdatum, VictorianMutant, Voidxor, Voomoo, Wayne Hardman, Whbstare, Wikisierracharie, William Avery, Woohookitty, Wrs1864, Xmm0, YordanGeorgiev, 228 anonymous edits

**Hypertext Transfer Protocol** *Source:* <http://en.wikipedia.org/w/index.php?oldid=516023840> *Contributors:* 1234r00t, 4342, A'eX, A-moll9, Aapo Laitinen, Abdull, Abu ali, Acdx, Ace of Spades, Adoniscik, Agawish, Aagpte, Agus puryanto, Ahoerstemeier, Aitias, Alansohn, Aldie, AlefZet, Alerante, Ali@gwc.org.uk, AlistairMcMillan, Amalthea, Andread, Andkim99, Andre Engels, Android Mouse, Aneah, Angus Lepper, Anna Lincoln, AnotherSprocket, Anwar saadat, Apparition11, Arch o median, Ash, Ashley Y, Asj123fds, Asteiner, Astroviev120mm, Atemperman, Audriusa, Avb, Avnjay, Baa, Babbage, Babedacus, Babysabre, Bamkin, BaronLarf, Basil.bourque, BeakerK44, Beefyt, Beinder, Beland, Ben-Zin, BenBildstein, Benandorsqueaks, Benanhalt, Bernfarr, Betacommand, Betterworld, Bevan7, Bevo, Bg, Bgwvlm, Bihoc, Bill, Bjankuloski06en, Blade44, Blanchardb, Blm, Bluemask, Bluerasberry, Bobo192, Bongwarrior, Booyabazooka, Bpantalone, Brian.fsm, Brownsteve, Bsdjan, Bull3t, C. A. Russell, C.Fred, CWii, Calabe1992, Callmederek, Calor, Caltas, Calvin 1998, Cameltrader, Can't sleep, clown will eat me, Canderson7, Capricorn42, CardinalDan, Catamorphosis, Churnett, CeciliWard, Cedy, Chadfiller, Charles Gaudette, Chasingol, Chealer, Checking, Chengqiang.prc, ChongDe, Christian75, Chuck, Coasting, Conversion script, Coroberti, CorpX, Cryptic, Cwolfoosheep, Cybersthe3000X, CyrilB, DKEdwards, DVdm, Daf, Dalitvoice, Damian Yerrick, Damreds, DanBLOO, DanBishop, Danculley, Daniboyd007, Danielfolsom, DarTar, DasRakel, Davidjk, Davilima, Dawn Bard, Doeetee, Ddas, De728631, DeadEyeArrow, Delicates, Delirium, DerBorg, DerHexer, DigitalNinja, Dionyziz, Discospinster, Dmufasa, Dmyersturnbull, Doc Armitage, Don4of4, Doria, Dr bab, Drwright, Dspradua, Ducknish, ENeville, Earthlyreasion, Edward, Ehn, El C, ElKevbo, Electron9, Eli the Bearded, Elimerl, Ellywa, Elving, Englishrose, Enjoia586, Ent, Epr123, EphemeralJung, Equendil, Esanchez7587, Evil Monkey, Ewawer, Extra999, Extransit, Eyreland, Falsestuff, Favonian, Femto, Ferkelparade, Firetrap254, Flockmeal, Frap, Fraxtil, Fred Bradstadt, Fresheneesz, Furrykef, Fuzzygenius, GGSinobi, GJHC1, Gaccolin, Gaius Cornelius, Galund, Gamernoterd, Gavinmorrice, Gdo01, Gen. von Klinkerhoffen, Gfoley4, Giftlite, GimliDotNet, GioeleBarabucci, Godtrog, GordonMcKinney, Gracenotes, Graciella, Graham87, GrahamColm, Grantglendinning, Gregkaye, Grouse, Gschoyru, Gsiegegan, Gurchzilla, Guy Harris, Haakon, Habbabub91, Haiiiiilllsatana, Ham Pastrami, Hans Genten, HappyCamper, Happysailor, HarisM, Haseo9999, Hashar, Hashproduct, Helix84, HexaChord, Himichellet, Horsten, Hrafnkell.palsson, Hrvoje Simic, Htonl, Hu12, Huw Powell, Hydrogen Iodide, Hyperthermia, IMSoP, IRedRat, IShadowed, Icairns, Idyllic press, Ilatejava, Iluvcapra, Imnotminkus, Imroy, Incnis Mersi, InfoRetrieval, Interior, Intgr, Irrypride, Itpastorn, Ivank06, Ivko, Ixfdd64, J.Dong820, J.delanoj, JHolman, JTN, Jaan513, Jacobulus, JamesBWatson, Jarble, Jarkosprat, Jarkspratt, Jasper Deng, Jdowland, Jeffq, Jeltz, Jennica, Jleedev, Jmar777, John Quiggin, John Vandenberg, John of Reading, Joghutten, Jonathan O'Donnell, Jonnabuz, JonoF, Josh3736, Joyous!, Jrobinjapan, Jstrater, Jusdafax, JustAnotherJoe, KAtremer, KGasso, Ka-Ping Yee, Karingo, Karl Dickman, Khrose, Keilana, Kesac, Kesla, Kgfleischmann, Kinema, Kiwi128, Knutux, Koavf, Kocio, Konnirret3, Koralin, KrakatoaKatie, Krellion, Krellis, Ksn, Kundor, Kungfuadam, KurtRaschke, Kvgv, Kwi, Kyla, Kymacpherson, La Parka Your Car, Lady Serena, Lannm, Leafyplant, Lee Carre, Lee J, Kheyland, Lethe, Lezek, Lightdarkness, Lilac Soul, Linusnk, Lloydpick, Lotje, Lotu, Lproven, Luna Santin, Lupinoid, Lysdexia, M0rphzone, M4gnum0n, MLupa, Mabdul, Macademe, MackSalmon, Madhero88, Maebmij, Magaman dude, Magioladitis, Mange01, Marcos canbeiro, Marek69, Mark Arsten, Martin BENOTT, Marx01, Materialscientist, Mathiasctk, Mauro Lanari, Mblumber, Mdd, Mdf, Meekohi, Mehfulz, Meneth, MikeF, Mikerlynn, Mindmatrix, Mjb, Mlibby, Mmj, Mnot, Modster, Mohammedsheikh786, Mormegil, Mortense, Moustafaza, Mpeylo, Mr Stephen, MrExplosive, MsDivaginn, Mschel, Mumia-w-18, Mwarren us, Mwtoews, Nageh, Nanshu, NawlinWiki, Negrulio, Nevyn, NewEnglandYankee, Newone, Nezzadar, Nicolas1981, Nigelj, Nightraider0, Nightstallion, Nikola Smolenski, Ninly, Niqueco, Nixdorf, Nixeagle, Nknight, Nlu, No Guru, Nubiatech, Nurg, Nyelvmark, ObfuscatePenguin, Okyea, Omniplex, Oneiros, Onlynone, Optiguy54, Orbnauticus, OsoLeon, Oxymoron83, Papadakis2007, Part Deux, Pascal666, Patrick, Paul Mackay, Paul-1, Pb30, Perceptes, Persian Poet Gal, Petero9, Pforret, Pgan002, Phatom87, Philip Trueman, Piano non troppo, Pioneerking, Pip2andahalf, Pnm, Praefectorian, Produke, Psychonaut, Pxma, Qff828, R3m0t, RHaworth, RJaguar3, RainbowOfLight, Raimman-sr, Raise exception, RandomHumanoid, Ravik, Reach Out to the Truth, RedWolf, Reschke, Retiono Virginian, RexNL, Rhaskallion, Rich Farmbrough, Rischmueller, Rjwilmsi, Rnhermen, Rmosler2100, RobMattson, Robert Xia, RockMFR, Ronhijones, RossPatterson, Rron, RyanCross, Rylong, SJP, SWAdair, Sae1962, Saejorn, Sakkath, Salvio giuliano, Samuel, Sander Saide, Saoshany, Saturn star, SchnitzelMannGreek, Schwallex, Scientus, Sdrirts, Sean Whitton, Seb-Gibbs, Sekro, Shadowjams, Sharon08tam, SheeEttin, Sheldon Rampton, Shiftoften66, Shuipzv3, Sietse Snel, SiobhanHansa, SixSix, SkE, Smsarmad, Socrates2008, SoftX, Spaceman85, Spandrawn, Sparky132, SqueakBox, Sri Krishna GEOVA Allah, Sses401, Ssteevvee, Stephan Leeds, StephenBuxton, Stephen, SteveLoughran, Stevenup7002, Stolee, Stupid Corn, Suppie, Suruena, Syrriths, T-borg, THEN WHO WAS PHONE?, Tabledthof, TakuyaMurata, Tbhocht, TechTony, Template namespace initialisation script, Tempodivalse, TenOfAllTrades, Th.matt.wiki, The Anome, The Thing That Should Not Be, The Utahraptor, The wub, TheJosh, Theopolisme, Think777, Thumperward, Tianjiao, Tide rolls, Timc1212, Timmywimpy, Timneu22, Tlesher, Tobby72, Tobias Bergemann, Tobias382, Todd Vierling, Tomaxer, Tommy2010, TonyAiuto, Toto Mommam, Tprbadbury, Tpirfan28, Tregoweth, Trusilver, Trygviss, TuukkaH, UU, Uncle Dick, Uncle G, UncleBubba, Universalls, Universimmedia, Urruamme, Utcursch, VMS Mosaic, Valenciano, Versus22, VictorAnyakin, Vinoth.t, Vxxii, Vrenator, WDGraham, Wavelength, Wayne Hardman, Widefox, Wikipelli, Wintonian, Wizzard2k, Wolfomatic5, Woohookitty, Wtmitchell, Yacht, Yaronf, Yomcat, Yonatan, Yuhong, Zachlipton, Zalnass, Zekefast, Zvn, ^demon, Ile flottante, 1316 anonymous edits

**Skype protocol** *Source:* <http://en.wikipedia.org/w/index.php?oldid=510627437> *Contributors:* Anwesender, BD2412, Benjamin, Brandon, Bunnyhop11, Camptown, Chealer, ChrisGualtieri, Cmskog, CosineKitty, CyberSkull, DStoykov, DataWraith, Dzsi, Firsfron, Frap, Gioto, Hairy Dude, Ian Cheese, Intgr, Jasper Deng, Jmorgam, Kastchei, Keenan Pepper, LilHelpa, Loyeyoung, Milan Kerslager, Mmemo, Nurg, Operating, Orenburg1, Ospalh, Ouan, Pgr94, Phatom87, Phil Boswell, PhilippGs, Pnm, Rich Farmbrough, Saimhe, Seaphoto, Sfan00 IMG, Skintigh, Timwi, Tomaxer, Toussaint, TreveX, Tustin2121, Welsh, Ysangkok, 58 anonymous edits

**BitTorrent** *Source:* <http://en.wikipedia.org/w/index.php?oldid=516174920> *Contributors:* 100110100, 1337 JN, 2 black, 4v410n42, 7h0r, =Xotic=, AMsissscorister, Aaron Brenneman, AaronSw, Abby, Acidburn24m, Acp1989, Adam Conover, Adamwankenobi, Adashiel, AgentSmith15, Agente, Agro1986, Aitias, Ajfweb, Akersmc, Alanl, Alerante, AlexR, Alexp73, Alianware377, Alimony, AlistairMcMillan, Alistaircasc, Aljullu, Alpha Quadrant, Aldesigins, Always remember don't forget, Alxndr, Amren, An Sealgar, Analoguedragon, Andersc1, Andkore, Andrew Benton, AndrewMrse, Andrewericoleman, Andrewlp1991, Andrewpwm, Anduin13, Anna Lincoln, Anomaly1, Anonymous Dissident, Anonymous editor, Anopheles, Antaeus Feldspar, Antandrus, Anthony, Antidrug, Antilived, Appraiser, Aquilioson, Arctic-Editor, Arnold Go, Arpingstone, Arru, Arsenic88, Asbestos, Asdf903, Atheros1, Augustz, Avsa, AxelBoldt, Assimon, Azcolvin429, Azimuth1, B&W Anime Fan, B-Con, Babajobu, Bad Byte, Badagnani, Badmonkey0001, Barfly, Barneyboo, Barte, Bbatsell, Bbik, Bcharles, Beauluc, Beebaumax, Beestra, Befryxell, Beland, Benandorsqueaks, Bender235, Benmoreassynt, Beno1000, Betacommand, Bevo, Beyazid, Beyondo, Bhagwad, Bige1977, Billymac00, Bitbit, Bkell, Black Falcon, Blahaccountblah, Blanchardb, Bleu'dove, Blu Aardvark, Bmendoc, Bmkos, Bncok, Boba5fett, Bobblewik, Bobet, Bobo192, Bobz, Bongwarrior, BonzoESC, Bootleg42, BorisFromStockdale, Bovski, Bposert, Brasetvik, Bratch, Bratsche, Breno, Brian Geppert, Brianjd, Bridson, BrokenSegue, Bruce89, Bryan Derksen, Bsroiaadn, Btipling, Buchanan-Hermit, Bulmbriefs144, BurnDownBabylon, Burschik, Buttonius, Bytemaster, CBDunkerson, CableCat, Caesura, Caiaffa, Calton, Cambodianholiday, Can't sleep, clown will eat me, CanisRufus, Canththinkofusername, Capt. James T. Kirk, Caramelman, Carbol2000, Carlosguitar, Cartoonmaster, Catfoo, Catgut, Cbraga, Cdonges, Cebra, Ceplm, CesarB, Ceyockey, Chahael Riens, ChangChienF, CharlesC, Chris Wood, Chrisprall, Christopher Parham, CIPHERgoh, Cippher03, CliranceC63, Clement Cherlin, Clifflinger407, Cmdrjameson, Cmousse, Codewench, Coelacan, Coludacid, Colonel panic, Connelly, Consumed Crustacean, Conti, Contristo, CooPs89, CorbinSimpson, Corevette, Cornilleau, CorpX, Corti, CosineKitty, CountZer0, Courcelles, Covracer, Coyote376, Cprompt, Cradel, Crazy4her, Crazyman, Crazyviolinist, Crf, Crimson30, Crispmmuncher, Crossmr, Crscrs, Curps, Cw.fire, CyberSkull, Cyberevil, Cynical, Cypherior, D Marcescu, D33d, DBaba, DBigXray, DH85868993, D21953, Da Yynpi, Daminfio52, Damian Yerrick, Dan100, Dancingmadness, Danfreak, DanielCardenas, Daniel770, DanielCD, Danielizatz, Darkone, Darksun, Darthonard37, DaveBurstein, Davewho2, David Levy, Davidgothberg, Davidknag, Davidlawrence, Daxx wp, Dck7777, Ddxc, Deamon chum, Deathphoenix, Debresser, Decrease789, Decrypt3, Deewhite, Dekker, Delirium, Delpino, Demonkey36, DerHexer, Derek.cashman, Deskanra, Dexter gper, Dhcmrlchtdj, DiaB\*ZeRo, Diaa abdelmoneim, Diannaa, Didimos, Dina, Discospinster, Dissolve, Dmritis-bat-girl, DocWawdon2, Doctroxenbriery, Dodo bird, Donwilson, Doogooie, Doradus, Dp462090, Dr. F.C. Turner, DrMoc, Dratman, Dreaded Walrus, Dundersc1, Duk, Dupz, Dxc0, Dycedarg, Dysepsion, Dysprosia, E. Sn0 =31337=, ENoggin, Eadric, Easwarno1, Edderso, EdoDodo, Edurant, Edwards, Ehn, Einstein9073, Ejay, Ekashp, ElTYrant, Elfugy, Eliashe, Eloquence, Emkrasn, Emufarmers, Emurphy42, En3r0, EndingPop, Endlessnameless, Englishmen, Epr123, Epsoul, EquusAustralus, Erkan, Ernesto99, Esemono, Espo111, Esrever, Ethifacal, Eugene van der Pijil, Eugman, Euphrosyne, Even248, Everyking, Evrice, Evil Monkey, Evilweevil, EwanMcLean2005, Exile.mind, Eyreland, Ezue, F, FT2, Fab1968, Fabioh, Falcon8765, Fallout boy, Farshad83, Felix Wiemann, Fennec, Feureau, Finlay McWalter, FireBallAX1, Fish and karate, Fishnet37222, Flarn2006, Flockmeal, Folken de Fanel, Fondaprog, Foobaz, Fourchette, Fragglet, FrancoGG, Frap, Frazzyede, FreakQNC, Freakmignity, Freakofnuture, Fredrikh, Fsilr, Ftiercel, Fubar Obfusco, Funchords, Funnybunny, Furrykef, Fuzheado, Fvw, G-RaZoR, Gabbe, Gaius Cornelius, Gambuzino, Gardar Rurak, Gargaj, Garion96, Garo, Gary King, GatesPlusPlus, Gavinatkinson, GenOkide, Gengw2000, Georgewilliamherbert, Gephe, Geschichte, Ggoddard, Ghen, Ghindo, Ghost Freeman, Giftlite, Ginsengbomb, Godcast, GodofLuigi, Gokusandwich, Golbez, Goodboywonder, Goto, Gracefool, GraemeL, Graft, Grawity, Green caterpillar, Greenrd, GregorB, Greyviz, Gronky, GrubLond, Guaka, Guillaume2303, Gujjar123, Gunboat Diplomat, Gunnar Helgason, Guyjohnston, Guzenkov, Gwern, Gzhanstong, H2pymee, Haakon, Hadal, Hadees, Haeleth, Halfmoonkitty, Ham Pastrami, Hao2lian, Haseo9999, Hashar, Hateless, Hbdragon88, Hdkiller, Hdd83, Hede2000, Helios solaris, Helohe, Heron, HFastedge, HiS oWn, Holdenmcrutch1, Honta, HowardStrong, Howdoesthiswo, Hpkomic, Htmilmis, Huji, Hulsie, Hurricane111, Husky, Hydrargyrum, IGod, INO Exodus, IRedRat, IRn, Iamunknown, Ian Dunster, Icedog, ImMAW, Imroy, InShanee, Ind9ie, Inexplicable, InfoMike84, Inter, Intgr, Intimidated, Ioscius, Ipsigin, Idrepesca572, Isilaness, Itai, Itemirus, Ixfdd64, J Di, J3ff, JLaTondre, JYouyang, JaGa, Jack Cox, Jaffa phi, Jahiegeel, Janhuss, JavaTenor, JayKeaton, Jayen466, Jcorby, Jdowland, Jdub7, Jerryobjekt, Jersey eml, Jetbackwards, Jfmantis, Jghaines, Jimmyjackjohn1, Jipcy, Jivecat, Jleedev, Jm1262, Jmabel, JoanneB, JoeSmack, Joel7687, Joelholdsworth, Jogloran, John Fader, John13535, John254, Johnhoo776059, Johnleach, Johnleemk, Johnm115, JohnyDog, Jolivierd, Jonas Viper, Jonne, Josemanimala, Josh Parris, Joshik, Joshua Issac, Jossi, Jossuy!, Jpettit, Jrlighton, Jernest, Jtepper, Jtkiefer, Jusdafax, Justinwiley, Juux, K.K.Nevelsteen, K1Bond007, Kaldosh, Kalebdf, KamasamaK, Karasuhebi, Karl-Henner, Karlo918, Karlwiegand, Karn, Karnesky, Karol Langner, Karthikndr, Kasreyn, Katalaveno, Kate, Kateshortforbob, Kazvorpall, Kelly Martin, Kennethkrabat, Kerotan, Kevin, Kf4bdy, Khalid, Khokkanen, Khukri, Kidburla, Killdevil, Kinema, Kingturtle, Kitsunegami, Kiwifilm, Kizar, Kizzle, KjeXz, Kjm, K14m-AWB, Knidu, Knuemo2, Knutaldrin, Koogunmo, Krakau, Krassotkin, KrazyA1pha, Krogstadst, Kuru, Kwekubo, Kwirky88, Kx1186, Kylene, LACHLANAC, LBarsov, Labenset, LamontCranston, Langelgin, Lavenderleaves, Lee1026, Leedrick, Legolost, LeighsOptimvsMaximvs, Lejarrag, Lemming, Lensi, LeHoSeka, Leonoz, Lheraupt, Leuk he, Leuko, Lexein, Liamzebedee, Liao, LieroMan, Lightmouse, Linguisticgeek, Linkminer, Lino Mastrodomenico, Liantamer, Little Mountain 5, LittleDan, Littlealien182, Logan, LoganTheGeshrat, Loganishappye, Logixoul, LokoC2, Lollerskates, Lotje, Ludling, Lulu of the Lotus-Eaters, Lunkwill, Lysis rationale, M, M.B, M.luke.myers, M.nelson, M4gnum0n, MC



# Image Sources, Licenses and Contributors

**Image:Internet map 1024.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Internet\\_map\\_1024.jpg](http://en.wikipedia.org/w/index.php?title=File:Internet_map_1024.jpg) *License:* Creative Commons Attribution 2.5 *Contributors:* Barrett Lyon The Opte Project

**Image:Crystal\_Clear\_app\_browser.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Crystal\\_Clear\\_app\\_browser.png](http://en.wikipedia.org/w/index.php?title=File:Crystal_Clear_app_browser.png) *License:* GNU Lesser General Public License *Contributors:* Everaldo Coelho and YellowIcon

**File:Leonard-Kleinrock-and-IMP1.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Leonard-Kleinrock-and-IMP1.png> *License:* Public Domain *Contributors:* Leonard Kleinrock

**Image:NSFNET-backbone-T3.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:NSFNET-backbone-T3.png> *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* JuTa

**File:First Web Server.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:First\\_Web\\_Server.jpg](http://en.wikipedia.org/w/index.php?title=File:First_Web_Server.jpg) *License:* GNU Free Documentation License *Contributors:* User:Coolcaesar at en.wikipedia

**Image:UDP encapsulation.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:UDP\\_encapsulation.svg](http://en.wikipedia.org/w/index.php?title=File:UDP_encapsulation.svg) *License:* GNU Free Documentation License *Contributors:* en:User:Cburnett original work, colorization by en:User:Kbrose

**File:Internet Connectivity Distribution & Core.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Internet\\_Connectivity\\_Distribution\\_&\\_Core.svg](http://en.wikipedia.org/w/index.php?title=File:Internet_Connectivity_Distribution_&_Core.svg) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:Ludovic.ferre

**File:Icannheadquarters.jpg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Icannheadquarters.jpg> *License:* GNU Free Documentation License *Contributors:* Original uploader was Coolcaesar at en.wikipedia

**File:Internet users per 100 inhabitants ITU.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Internet\\_users\\_per\\_100\\_inhabitants\\_ITU.svg](http://en.wikipedia.org/w/index.php?title=File:Internet_users_per_100_inhabitants_ITU.svg) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:W163

**File:InternetUsersByLanguagePieChart.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:InternetUsersByLanguagePieChart.svg> *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:W163

**Image:WebsitesByLanguagePieChart.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:WebsitesByLanguagePieChart.svg> *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:W163

**File:SRI First Internetworked Connection diagram.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:SRI\\_First\\_Internetworked\\_Connection\\_diagram.jpg](http://en.wikipedia.org/w/index.php?title=File:SRI_First_Internetworked_Connection_diagram.jpg) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:Russavia

**File:SRI Packet Radio Van.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:SRI\\_Packet\\_Radio\\_Van.jpg](http://en.wikipedia.org/w/index.php?title=File:SRI_Packet_Radio_Van.jpg) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* User:Russavia

**Image:IP stack connections.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:IP\\_stack\\_connections.svg](http://en.wikipedia.org/w/index.php?title=File:IP_stack_connections.svg) *License:* GNU Free Documentation License *Contributors:* en:User:Kbrose

**File:OSI-model-Communication.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:OSI-model-Communication.svg> *License:* Public Domain *Contributors:* Runtux

**File:UDP encapsulation.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:UDP\\_encapsulation.svg](http://en.wikipedia.org/w/index.php?title=File:UDP_encapsulation.svg) *License:* GNU Free Documentation License *Contributors:* en:User:Cburnett original work, colorization by en:User:Kbrose

**File:Tcp state diagram fixed.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Tcp\\_state\\_diagram\\_fixed.svg](http://en.wikipedia.org/w/index.php?title=File:Tcp_state_diagram_fixed.svg) *License:* GNU Free Documentation License *Contributors:* Tcp\_state\_diagram\_new.svg: \*derivative work: Sergiodc2 (talk) Tcp\_state\_diagram.svg: dnet derivative work: Marty Pauley (talk)

**File:TCP CLOSE.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:TCP\\_CLOSE.svg](http://en.wikipedia.org/w/index.php?title=File:TCP_CLOSE.svg) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* Iustitia, 1 anonymous edits

**Image:Tcp.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Tcp.svg> *License:* Creative Commons Attribution-ShareAlike 3.0 Unported *Contributors:* Mike de

**File:Internet1.jpg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Internet1.jpg> *License:* Creative Commons Attribution-Share Alike *Contributors:* Rock1997

**File:Tim Berners-Lee CP 2.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Tim\\_Berners-Lee\\_CP\\_2.jpg](http://en.wikipedia.org/w/index.php?title=File:Tim_Berners-Lee_CP_2.jpg) *License:* Creative Commons Attribution 2.0 *Contributors:* Silvio Tanaka

**Image:Http request telnet ubuntu.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Http\\_request\\_telnet\\_ubuntu.png](http://en.wikipedia.org/w/index.php?title=File:Http_request_telnet_ubuntu.png) *License:* Public Domain *Contributors:* TheJosh

**Image:Torrentcomp small.gif** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Torrentcomp\\_small.gif](http://en.wikipedia.org/w/index.php?title=File:Torrentcomp_small.gif) *License:* GNU Free Documentation License *Contributors:* Frigotoni, Holek, Joolz, Mdd, Senator2029, Sputnik, Wikiadd, Wknight94, Ypacarai, 3 anonymous edits

**Image:Turnstile state machine colored.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Turnstile\\_state\\_machine\\_colored.svg](http://en.wikipedia.org/w/index.php?title=File:Turnstile_state_machine_colored.svg) *License:* Creative Commons Zero *Contributors:* User:Chetvorno

**Image:Torniqueterevolution.jpg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:Torniqueterevolution.jpg> *License:* Creative Commons Attribution-Share Alike *Contributors:* Sebasgui

**File:UML state machine Fig5.png** *Source:* [http://en.wikipedia.org/w/index.php?title=File:UML\\_state\\_machine\\_Fig5.png](http://en.wikipedia.org/w/index.php?title=File:UML_state_machine_Fig5.png) *License:* Creative Commons Attribution-Sharealike 3.0 *Contributors:* Mirosamek (talk)

**Image:SdlStateMachine.png** *Source:* <http://en.wikipedia.org/w/index.php?title=File:SdlStateMachine.png> *License:* Creative Commons Attribution-ShareAlike 3.0 Unported *Contributors:* Manu31415, Theon144

**File:Finite state machine example with comments.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Finite\\_state\\_machine\\_example\\_with\\_comments.svg](http://en.wikipedia.org/w/index.php?title=File:Finite_state_machine_example_with_comments.svg) *License:* Public Domain *Contributors:* Macguy314

**File:Fsm parsing word nice.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Fsm\\_parsing\\_word\\_nice.svg](http://en.wikipedia.org/w/index.php?title=File:Fsm_parsing_word_nice.svg) *License:* Public Domain *Contributors:* en:User:Thowa, redrawn by User:Stannered

**File:DFAexample.svg** *Source:* <http://en.wikipedia.org/w/index.php?title=File:DFAexample.svg> *License:* Public Domain *Contributors:* Cepheus

**File:Fsm mealy model door control.jpg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Fsm\\_mealy\\_model\\_door\\_control.jpg](http://en.wikipedia.org/w/index.php?title=File:Fsm_mealy_model_door_control.jpg) *License:* Public Domain *Contributors:* Original uploader was Thowa at en.wikipedia

**File:Finite State Machine Logic.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:Finite\\_State\\_Machine\\_Logic.svg](http://en.wikipedia.org/w/index.php?title=File:Finite_State_Machine_Logic.svg) *License:* Public Domain *Contributors:* jjbeard

**File:4 bit counter.svg** *Source:* [http://en.wikipedia.org/w/index.php?title=File:4\\_bit\\_counter.svg](http://en.wikipedia.org/w/index.php?title=File:4_bit_counter.svg) *License:* Public Domain *Contributors:* Gargan

# License

---

Creative Commons Attribution-Share Alike 3.0 Unported  
//creativecommons.org/licenses/by-sa/3.0/