———. "On the Mathematical Powers of the Calculating Engine." In B. Randell, ed., *The Origins of Digital Computers: Selected Papers*. Berlin: Springer-Verlag, 1973; 3rd ed., 1982.

Campbell-Kelly, Martin, ed. *Works of Babbage*. 11 vols. New York: New York University Press, 1989.

———. *Babbage: Passages from the Life of a Philosopher*. Piscataway and New Brunswick, N.J.: Rutgers University Press, 1994.

FURTHER READING

Babbage, Henry P., ed. *Babbage's Calculating Engines: A Collection of Papers*. Los Angeles: Tomash, 1982.

Bromley, Alan G. "Charles Babbage's Analytical Engine." *Annals of the History of Computing*, Vol. 4, No. 3, 1982.

Hyman, Anthony H. *Charles Babbage: Pioneer of the Computer*. Cambridge and New York: Cambridge University Press, 1989.

—*Martin Campbell-Kelly*

Backbone

In networks of computer networks, such as the **Internet**, there are usually numerous possible paths from sender to receiver. Some communication channels are faster than others. If deployed properly, the set of fastest channels can serve as a communication superhighway to move information from one geographic area to another, where it is further relayed to its final destination using slower links. The fastest channels constitute the *backbone* of the network; they transmit data gathered from the slower branches.

The Internet backbone originated from the **ARPANET**. Originally, ARPANET linked only a handful of research institutions on the west and east coasts of the United States. As more universities with heterogeneous computer systems joined ARPANET, the range of communication velocities differed. In 1986, the National Science Foundation (NSF) started NSFnet, which provided communication services between major sites. The NSF funded the establishment of five **supercomputer** centers, which were linked with high-speed channels. The links between the most important network nodes were upgraded periodically. Although NSFnet consisted originally of slow 56-kilobit data links, the backbone was upgraded to a 13-node T-1 (1.25-megabit per second [Mbps]) network in 1988 (see figure). In 1991 the backbone was upgraded again to a 16-node T-3 (45 Mbps) network.

The acceptable use policy (AUP) of the original NSFnet charter explicitly prevented the network from being used commercially. This began to clash with the reality of the Internet in 1990, since much of the traffic through the backbone had commercial purposes. In 1992, the U.S. Congress passed a law—called the Boucher Bill after its sponsor, Representative Rick Boucher (D-Virginia)—revising the AUP to permit commercial traffic on the Internet. As a result, in 1993, NSF announced its decision to put the operation of the backbone into the hands of private companies. The NSFnet backbone was decommissioned at midnight on 30 April 1995.

Since the retirement of the old backbone, the architecture of the network has changed. A vBNS (very high speed backbone network service), operated by the telecommunications company MCI, is now in place, but its use is restricted to organizations that require high speeds. Network access points (NAPs) interconnect the vBNS to other private backbone networks, domestic and foreign. The NAPs are operated by several telephone companies.

Internet backbones have been also installed in other regions, most notably in Europe, following the model of NSFnet. In Germany, for example, the German Association for a Research Network manages the backbone, which was upgraded to gigabit links early in 2000. Universities affiliated with the association pay a subsidized fee for an access point to the backbone.

FURTHER READING

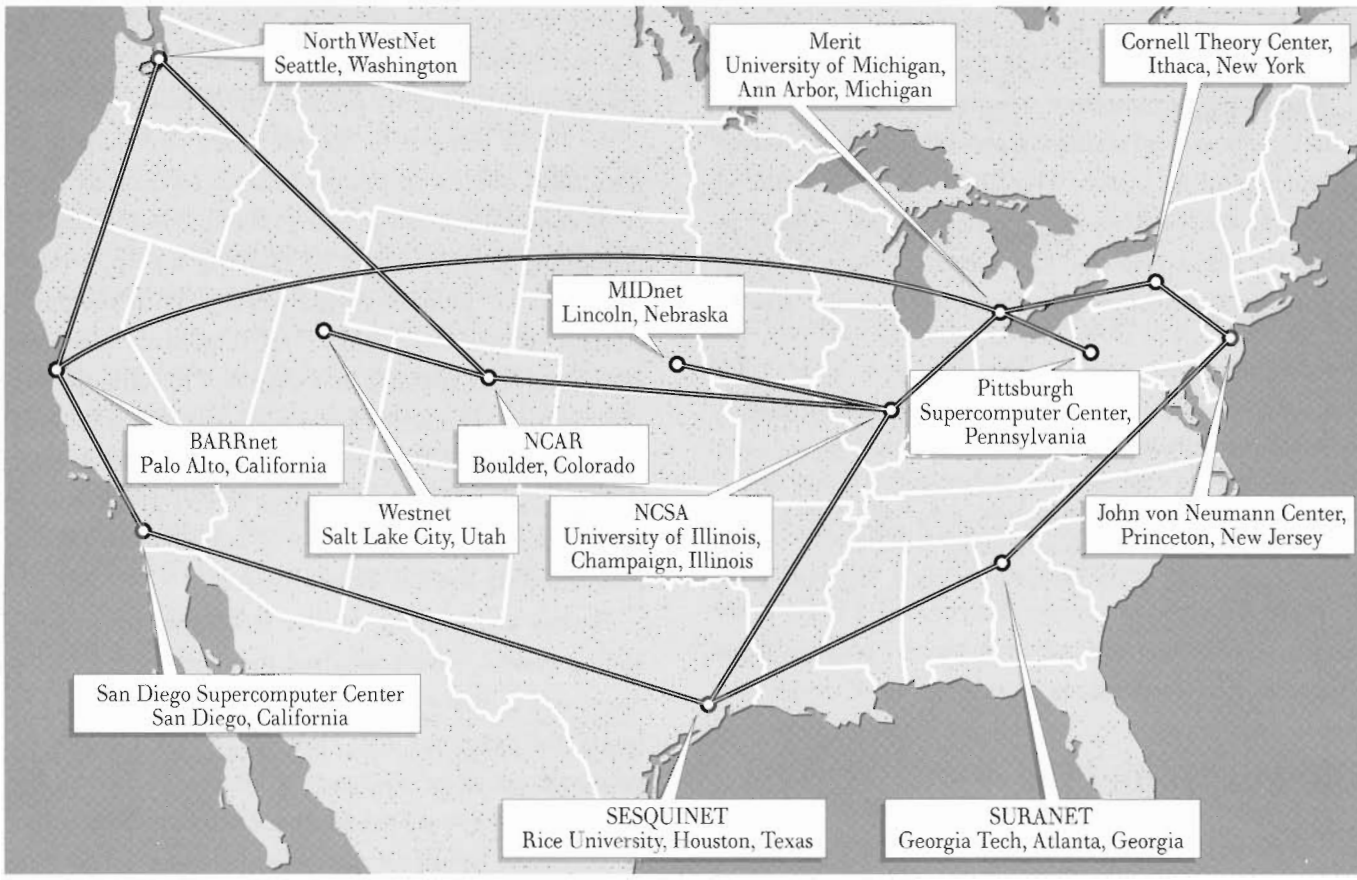
Abbate, Janet. *Inventing the Internet*. Cambridge, Mass.: MIT Press, 1999.

Moschovitis, Christos J. P., Hilary Poole, Tami Schuyler, and Theresa M. Senft. *History of the Internet: A Chronology, 1843 to the Present*. Santa Barbara, Calif.: ABC-CLIO, 1999.

—*Frank Darius*

Back Door

A back door is a secret way of breaking into a system, either left behind by the original programmer or created by someone who gained access illegally. A back



NSFnet backbone as of July 1988.

door is installed by crackers in order to regain access to someone's computer at a later date. In many cases, a back door is just a user account with a password that is not listed in the standard password file of the system.

Many sites on the **World Wide Web** have been defaced by intruders using a back door. The method used by the crackers is to start a program in a remote machine using CGI (common gateway interface): CGI is a way of passing parameters to a program running locally in a remote host computer. Normally, this is done legitimately, to obtain a service from this host. However, if the program script is poorly written or if the cracker has access to the source code of the script, it can be manipulated so that the intruder can start a remote command in the server. The cracker can then send the password file of the remote machine by e-mail to himself or herself, or even remove files from the hard disk. This can be done because in many cases, the Web administrator lets the CGI script interpreter run in privileged mode, which provides access to all files. Once the intruder has the password file, he or she tries to decrypt passwords in order to get a back door into the system. In 1999, both the CIA and FBI Web sites were defaced by crackers using back doors.

In his 1984 **Turing Award** lecture, **Ken Thompson** (1943–) explained how an invisible back door can be installed in a system like **Unix**. He wrote a version of the C compiler (a standard component in the Unix distribution files) that would compile the code for gaining access into the system in such a way that a special password, known only to him, would be recognized. This already constitutes a back door into the system, but that trick could be detected by any programmer who inspects the source code of the C compiler. Thompson went further, rewriting the compiler so that it would include the offending code every time the source code for the compiler was recompiled. He then removed the back door code from the source, recompiled it with the "infected" compiler, and obtained a binary that would perpetuate the back door, even with a clean source code. The moral of all this is that you cannot trust any program that you have not written yourself. You cannot even trust the

compiler used to translate your program, and in general, you cannot trust any executable program sent to you.

Computer emergency response teams (CERTs) exist now in several countries, providing 24-hour advice in the case of an attack. The first CERT was installed by the Advanced Research Projects Agency (**ARPA**) in 1988, the same year that the Internet Worm infected thousands of machines, mainly in the United States.

FURTHER READING

Denning, Peter J. *Computers Under Attack: Intruders, Worms, and Viruses.* Reading, Mass.: Addison-Wesley, 1990.

Stoll, Clifford. *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage.* New York: Pocket Books, 2000.

Thompson, Ken. "Reflections on Trusting Trust." *Communications of the ACM,* Vol. 27, No. 8, Aug. 1984, pp. 761–763.

—Raúl Rojas

Backus, John

1924–

U.S. Computer Scientist

John Backus led the team of programmers at **IBM** who developed the **Fortran** language in 1954. Fortran (Formula Translator) was the first popular high-level programming language and compiler and is still one of the most widely used languages for scientific, engineering, and mathematical problems.

Backus joined IBM in September 1950, soon after completing his master's degree in mathematics at Columbia University. During his first three years at IBM, he worked as a programmer for the IBM **Selective Sequence Electronic Calculator** (SSEC). This early computer contained 21,400 relays and 12,500 vacuum tubes. It used punched paper tape for input and output, including storing the intermediate results of calculations by punching them into paper tape, which was then read back into the program for later reference.

Programming this device was extremely laborious and required many hours of coding in machine language. An early assignment had Backus teamed up with **Edgar Codd** (1923–) later known as the originator of the relational model for databases, to devise an automatic method for locating the source of machine errors during the running of programs on the SSEC. The experience that Backus gained in automating SSEC programming would prove helpful for his later work.

As IBM's computers became more and more complex, IBM and its customers became aware that much more computer time was being spent on developing and debugging programs than on running the applications for which the programs were written. With the development of the 701, IBM's first large-scale electronic computer, IBM realized that they had to solve the programming bottleneck in order to make the machine justifiable economically to its customers. After Backus left the SSEC group to join the group of 701 programmers, he undertook the development of a method for making programming the 701 more efficient.

Backus and a group of five colleagues developed a coding system, which acquired the descriptive name of *Speedcoding*, that substantially decreased the time necessary for programming. However, as IBM began the development of its next-generation computer, the 704, it became clear that Speedcoding would not be adequate to meet the needs of this more powerful machine. Backus decided that programming the 704 required the use of a **compiler**, a program that would process another program written in a user-oriented language producing machine-executable code. He submitted a proposal to his boss at IBM, Cuthbert Hurd (1911–), recommending the development of a compiler and, in 1954, Hurd authorized him to assemble a team to develop the language that came to be called Fortran.

Most computers of the time were used to perform complex mathematical calculations for scientific or engineering applications, so the logical choice for a user-oriented language was one based on algebraic expressions. The basic features of the language proposed were established during 1954 by Backus and his colleagues Harlan Herrick and Irving Ziller. By 1955, the long job of writing the compiler, the program that would read the algebralike instructions of Fortran and produce an efficient machine-executable program, began. Backus had expected the entire project to take about six months, but like most programming efforts, it was more difficult than anticipated. The 704 Fortran compiler was eventually released in April 1957, the result of the work and ingenuity of eight very creative people.