



How the Internet Works

What is the Internet?

When you start a short section of the manual with a big question like “What is the Internet?”, you’re asking for problems. So, we’ll warn you upfront that this is (necessarily) a very brief, entry-level explanation that tries to cram a lot into a short space, and leaves a great deal of technical info out. However, this is only intended to be a basic overview of the Internet, so it should suffice.

So, to start things off on the right foot, we’ll answer the question of “What is the Internet?” straight away. The answer is that there is no “Internet.” To confuse you even more, let’s pose another question-and-answer pair: have you ever wondered about the question “Who’s in charge of the Internet?” Well, the answer is: Nobody. Or, rather, *everybody*.

The reason for this rather clearly confusing set of explanations is that the Internet is merely a loose confederation of all of the smaller networks that make it up. The only requirement for membership is a computer that speaks the TCP/IP protocols, and a way to connect to the other networks.

The correct way to envision the Internet is not a hierarchical model like a company with divisions, or even a softball team with a coach and players. Instead, it is a loose confederation like the United Nations, where each member has full authority over their own “territory,” although they may work together or take recommendations from the body as a whole. Remember that the Internet is essentially run by thousands of engineers and computer wizards; these people don’t even like to wear *ties*, let alone bow to any sort of central authority. Each network connected to the Internet rules itself.

However, there have to be “ruling bodies” of a kind, to settle technical matters, or else computers wouldn’t be able to communicate with each other. International committees of engineers (like the International Standards Organization [ISO] or International Telecommunications Union [ITU]) will develop standards and then recommend them to Internet network administrators. And once influential networks (the really big or important ones) start to adopt them, soon all of the others will. Engineers always like cool new technical toys to play with, and they can’t stand falling behind.

Not to belabor the point (too late), but you must understand this: the Internet is only the sum of its parts, which are all of the smaller, privately owned and operated computer networks. There is no Internet, apart from its component networks.

Now that we’ve told you what the Internet isn’t, let’s take a look at what it *is*. The short answer is that it’s a mishmash of different protocols, different networks, different services and different computers.

If you want to irritate an Internet techie, the fastest way to do so (short of setting their computer on fire) is to say something like you’re “surfing the Net” or “logging on to E-mail.” What you are doing is logging on to the Internet, then “surfing” the World Wide Web or accessing your E-mail server to read your mail. This distinction may sound small (not to mention boring), but it’s important.

The Web and E-mail are only *facets* of the Internet, a tiny fraction of the number of different uses for the Internet that are out there. Computers attached to the Internet can “talk to” each other through a large number of protocols, or particular languages and rules for communicating.

The computers that make up the Internet are all united by the ability to “speak” a common set of languages, referred to collectively as TCP/IP (which stand for Transmission Control Protocol and Internet Protocol). This group of protocols (computer-to-computer communications languages) includes all of the other protocols used on the ‘Net.

The Web is nothing more than a bunch of computers (including the one you use to browse or “surf” it) that can send out or understand information in a protocol called HTTP, or Hyper-Text Transfer Protocol. Incidentally, that’s what the `http://` at the beginning of a World Wide Web address means: that you want to talk to a computer through the HTTP

(Web) protocol. When you download a file from the Internet, you're invoking another protocol, even though you may not know it: FTP, or File Transfer Protocol. Similarly, E-mail is delivered through another protocol/language called SMTP, or Simple Mail Transfer Protocol.

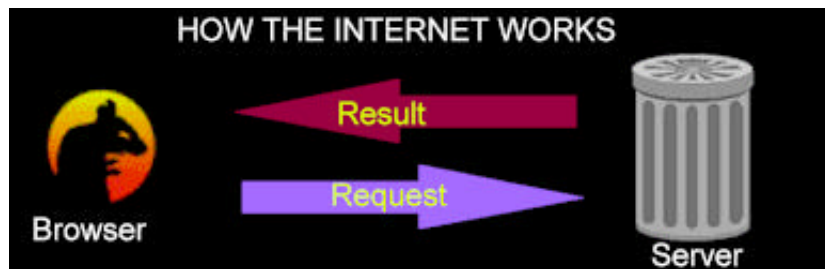
So, what does this mean to you? And why are we already using nerd words to describe it? It means that even though the World Wide Web or E-mail may be the parts of the Internet that you use most often – even if you never knew you were using them – that there is plenty of important stuff going on “under the surface” that controls everything else. Internet-connected computers are constantly communicating with each other through these various protocols, not just the ones that you may be used to “seeing” when you use the Internet. And a problem with any of these protocols may affect the other ones.

Now that you're a little more familiar with protocols, let's walk through how web transfers and CGI scripts work, with the help of some text and illustrations, courtesy of **BigNoseBird.Com**.

How Internet Transfers Work

How Do HTTP Requests Work?

The illustration here is a simple, but accurate representation of the Internet. We have rats out looking for food, which are our browsers. We have our garbage cans, better known as servers providing storage and a way of doling out the food when asked for it. So in a nutshell, the rat asks for food and the garbage can gives it to the rat.



Now of course our garbage can in addition to holding food, also has a bunch of little beasties living in it. You say you never heard of a smart garbage can, have you? Now you have. Let's flip the lid and see what's going on in there.

Well, actually there are other little beasties in the Unix world that are referred to as daemons. Hmm. To make a file readable and writable to everyone, we issue the command `chmod 666`

`filename` . Coincidence? I don't think so... Now here is how all of this is going to make sense to you very quickly:

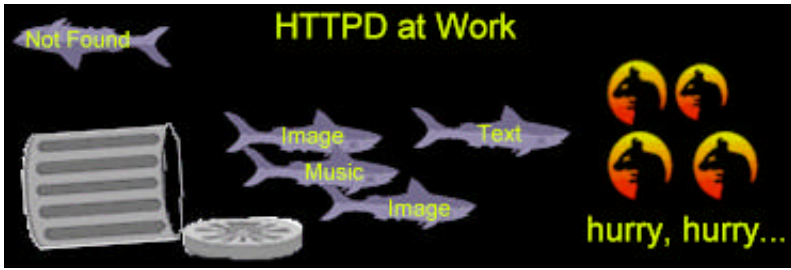
Telnet: This daemon is usually called **telnetd**. Its job in life is to wait for you to try and log on to the system. If at your location window in your browser, you enter `telnet://server.com` , you will get a login prompt. This is assuming your browser has a helper application, and that the server accepts telnet requests.

FTP: This is the `ftpd` daemon. It handles requests that start with `ftp.`, i.e., `ftp://bignosebird.com/bnbform.tar`. FTP stands for File Transfer Protocol.

HTTP: Ah, this is the daemon `httpd`, the one that we must concentrate on. The `httpd` handles all requests that start with `http`. For the purposes of doing HTML and CGI-BIN, this is going to be our favorite beastie!

When you issue a command from your browser, such as `http://bignosebird.com/index.shtml`, the request is sent to my server and the `httpd` daemon finds a document named `index.shtml` , which is then devoured by a school of very nasty sharks. They rip the page to shreds to find out what they have to do. Each shark takes one instruc-





tion. For instance:

If the page has text, three images, and a MIDI music file- in all, five sharks will be called to duty. It is the job of each shark to go and handle retrieving the requested information and packaging it to go back to the requesting browser.

Let's see how our sharks are doing. It looks like the text shark, two of the image sharks, and the music

shark are doing just fine. Uh oh, something is wrong. One of our sharks has gone belly up on us. Did the author name that image file properly? Did somebody delete the file? Well, it's taps for that shark. The browser will display its icon for a missing graphic.

How Do CGI Scripts Work?

Unlike a simple HTML document request, a CGI (Common Gateway Interface; see *CGI Scripts*, page 184 for more information) request requires something a whole lot smarter than a school of hungry, stupid sharks. When you click on a link that looks like:

<http://bignosebird.com/cgi-bin/Guestbk/guestbook.cgi>

You are really instructing the server to run an actual program that handles only this one type of request. In this case it runs the program that calls up the form that allows you to make an entry in the guestbook. Here is what happens:



Your browser sends the request shown above to my server. My server runs the `guestbook.cgi` program. The `guestbook.cgi` program realizes that you are not adding anything yet, so it makes the decision to create and return to you, a blank data entry form. Your browser then displays this form as it would any other HTML document.



like this:

[/cgi-bin/Guestbk/guestbook.cgi?name=bruce&mail=bnb@xyz.com](http://cgi-bin/Guestbk/guestbook.cgi?name=bruce&mail=bnb@xyz.com)

When KJ receives the request, it executes a program called `guestbook.cgi` that is located in the `Guestbk` directory under `cgi-bin`. What is different now is that in addition to a program request, *information* is being fed to KJ. Think of the items listed after each question mark as a fish. Here is what KJ has to do:

1. Load the `guestbook.cgi` program into memory for running.
2. It notices that it has a school of fish to munch on and proceeds to split them up using the `&` as a way of telling where one fish starts and another one ends.
3. KJ then writes a name on each fish so she can remember what is in that fish.
4. Seeing that the E-mail fish has a name in it, sends a thank you note to the person that signed the book.

Please note that the item(s) sent back to the browser do *not* have to be just text; it can be other types of information such as a graphical counter image.

So you take a moment and fill in all the fields and click on the SUBMIT button. Now a more complex series of actions take place. The command in the browser location window may look something

5. Sends a note to the webmaster that a new entry is in the guestbook.
6. Takes all of the fish, and uses their information to create a new entry that is tagged onto the `guestbook.html` document.
7. Creates an HTML document saying “Thank you for signing my guestbook!” that is returned to the browser.
8. KJ’s job is done. Everything went great and let’s have a round of applause for KJ.

The important concept is that in order to use CGIs, you must create a real, however big or small computer program. It is not embedded in your HTML document, but it is installed on your server.

How Internet Backbones Work

The Internet is not really run by companies so much as it is by people. There really aren’t all that many people out there who can run large networks (there is a worldwide shortage of super-nerds), and these people tend to know each other (or have at least heard of each other), and they talk (via E-mail, of course) and trade ideas or work out problems. If some networks “break the rules” (allowing their users to flood the other systems with junk E-mail, etc.), then the other administrators may break off connections to the offending network, thereby enforcing a sort of “peer pressure” to keep things in line.

The upshot of all this is that there is no single central authority that rules the Internet – it’s just the companies and the people that run their own individual parts of it. And this is generally a very good thing – because it means that control of the Internet is left to the technical people who understand it. And so each network is controlled by its own engineers.

Most people envision the Internet (maybe through an overdose of bad “Information Super-Duper-Highway” metaphors) as being like the Interstate highway system – a lot of local roads that lead to the “main highway,” where the “real” Internet is. And like the physical road structure, they somehow imagine this Internet superhighway as being paid for by tax dollars, and perhaps even being worked on by people in yellow hard hats.

There is no public “highway” on the Internet. It’s all toll roads, built by individual companies for paying drivers. Let us reiterate that point: Every single inch of the Internet is *owned* by *someone* (or at least leased by someone from a telephone company).

This myth of a government-controlled “Internet Backbone” comes out of the history of the Internet (much of the popular version of which is myths, anyway, but more on that later). For a long time, the U.S. government did pay for the ARPANet backbone. Eventually, the government decided that the free ride was over – the government would fund the NSFNet backbone, which could only be used for academic purposes. If anyone else wanted to connect to the Internet, private industry would need to step in and provide access.

As the Internet grew, and groups like businesses and home users wanted access, Internet Service Providers paid money to create their own access to it. As more people (and large corporate customers, too) signed up with an ISP, for example, the ISP would make its network larger, to accommodate the needs of all its new customers. Telephone companies – that already possessed large cross-country networks of phone lines in place – jumped on the bandwagon, upping the ante. Tens of millions of dollars were spent creating data pathways for Internet traffic – each pathway owned by the service provider that created it.

Soon, the privately owned connections between networks dwarfed the original government network. In the late 1980s, the government stopped funding NSFNet altogether, preferring largely to make itself a customer of commercial ISPs. The parts of the Internet that are still owned by the government aren’t for public use (MILNET, the network between military installations, for example). This is also why there isn’t a central governing body for the Internet (if the government still owned the Internet, you can be certain they would have created one by now).

There is a rough hierarchy of ISPs, based on the size (but not necessary the quality) of their networks. They are classified as being “tier one,” “tier two” or “tier three,” based on their relationship to all of the other ISP networks.

Remember that point that we were belaboring a while ago? That there is no “Internet;” there’s just all of the networks together, which we collectively call “the Internet?” Well, we’re belaboring it again.

In essence, when you're connecting to the "Internet," what you're really doing is connecting to all of the other networks out there. Networks gather together at places called NAPs (Network Access Points), connect their routers and other equipment together, and then exchange data from one ISP network to another. There are several major NAPs in the United States alone (MAE-East, MAE-West, Pac Bell NAP, Ameritech Chicago NAP Sprint Pennsauken NAP, etc.), and most large ISP networks are connected to at least one of them. ISPs may also choose to have private interconnects with each other, away from these high-usage access points. It is through these NAPs and interconnects that networks exchange traffic.

Why Internet Service isn't Like Phone Service

No matter how much we grumble about our telephone service (some of us more than others), we're pretty used to the way it works: You pick up the telephone, and there's a dial tone, waiting for you to call. Over the past hundred years, tens of thousands of men and women have labored to create one of the greatest engineering feats in history: a phone system, stretching to almost every home in the country, that (almost) always works.

But Internet service is different: sometimes you get a busy signal and can't log on. Sometimes you can access parts of the World Wide Web, but not others. Sometimes your mail can't go through. Sometimes it's just plain slow. Why can't Internet access be like telephone access?

There are plenty of reasons for this, but many of them can be boiled down to this: The Internet is still in its infancy, but it is already (collectively) the most complex machine in the world. Control over the Internet is divided among hundreds of networks, all dependent upon each other for everything to work correctly.

In May 1997, on what Internet technicians still call "Black Friday," a single Internet provider passed incorrect data on routing (sort of a road map for Internet traffic) to a large "backbone" ISP. That ISP's routers (specialized computers that control traffic between networks) then passed the incorrect routing information out to other ISPs, and so on. Within an hour, routers at ISPs all over the planet were malfunctioning and shutting down. Routers from other ISPs simply refused to listen to the malfunctioning routers from the affected networks (a kind of computer quarantine). All of the networks that make up the Internet were effectively cut off from each other. The entire Internet essentially "crashed" for a day, while network engineers fixed the problem.

The hardware problem that caused "Black Friday" was fixed; but who knows what other problems may lurk out there? Because the Internet is young, even the people who understand it best are constantly learning new things – and learning from their mistakes.

The good side is that – for the most part – the Internet is growing more reliable, as more ISPs invest in redundancy (or having multiple connections so that even if one of their data lines fails, their computers are still connected through another line). And as network administrators grow more experienced, and the architecture of the Internet becomes more mature and robust (it sounds like we're describing wine here, doesn't it?), its reliability should increase markedly.

Until then, conditions on the Internet vary from day to day. There are numerous places that provide "weather forecasts" for the Internet (we'll talk more about those later), or show how essential parts of the Internet are performing. But, alas, all ISP networks on the Internet are not equal in size or quality.