Port

When working with TCP or UDP protocol, this is part of a connection's information. Target and source are defined by the network address, like 192.168.0.1 communicating with 8.8.8.8. But as you are aware, there can be many programs running on the same computer. Like you are using your web browser while having a chat program open, and maybe at the same time, listening to a web radio or watching a video. In order to make communication with multiple programs on the same computer possible, every connection information does not only consist of target and source addresses but also target and source ports.

A port in the TCP/IP sense is a number between 0 and 65535. So every program that want's to communicate with another (even when it is connecting to its own host) uses a destination port, which it tries to communicate with, and expects replies to it on the port it uses itself. One full communication's participant's information therefore would be 8.8.8.8:53 (UDP). This is for one of Google's dns servers, 8.8.8.8 being the IP address, while 53 is the port number and UDP the protocol that is to be used. One program can use multiple ports simultaneously and multiple programs use different ports than other programs. There are two different situations:

Working as Server

When a program works as a server (e. g. a dedicated server), the own port of that server is pre-defined. This is where the server waits for incoming communication. This is important, because clients must know in advance, at which port they will have to send their request. E. g. when connecting to https://www.google.de, the HTTPS protocol tells your web browser already in advance, that this connection will most likely use port 443 TCP and it will try that. Normal HTTP works on standard port 80. You don't have to guess a random number this way. For a server that is the norm.

Working as Client

On the other hand, that does not go for the other side, the client. Your web browser may have connections to many web servers, even sumultaneously. That implies, that it will require, aside from its address, also multiple ports on its end, one for each connection. Imagine watching a video on Youtube while at the same time reading an article about something on a news web site while downloading the latest update for one of your games. All three cannot be using the same connection, so the browser has to use at least three different communication channels. This is solved by using port numbers. Trying to giving the same ports to different outgoing request would give port collisions (your computer would not know, where a certain package is to be put, firefox? Your downloadmanger? The updating game?), so the ports are usually selected randomly from the available ones.

This works, as the client will try to connect to the server. It knows where to send its request (destination address and port are known, e. g. 142.250.74.3:443 TCP for https://www.google.de). The server on the other hand can simply look into the package he just received from you, where the source address and port are noted. So he also knows, where exactly to send the data back to. The randomness of the port does not cause a lack of required information.

More complicated Ways of Using Ports

This also allowed techniques such as Network Address Translation (NAT), where your router acts as if it were doing all the requests all by itself, while it actually only relays requests of other computers. And in turn makes port forwarding an issue.

[Games Database] [Game Related Terms] [Network Terms]