CloudSavvy IT

What Are Unix Sockets and How Do They Work?

ANTHONY HEDDINGS
AUG 27, 2020, 11:00 AM EDT | 3 min read

Shutterstock/asharkyu

Unix sockets are a form of communication between two processes that appears as a file on disk. This file can be used by other programs to establish very fast connections between two or more processes without any network overhead.

What Are Sockets?

Sockets are a direct connection between two processes. Imagine if you wanted to call your friend down the road; you could place a call have it routed up through your telephone company and back down to their house, or you could run a wire directly to their house and cut out the middleman. The latter is obviously impractical in real life, but in the world of Unix it's very common to establish these direct connections between programs.

The proper name for unix sockets is *Unix Domain Sockets*, because they all reside within one computer. In a sense, sockets are a network that is entirely contained within the kernel; rather than using network interfaces to send data, that same data can be sent directly between programs.

Despite creating files on disk, Unix sockets don't actually write the data they send to the disk, as that would be far too slow. Instead, all the data is retained within kernel memory; the only point of the socket file is to maintain a reference to the socket, and to give it filesystem permissions to control access. For example, MySQL's socket is usually at:

```
/var/lib/mysql/mysql.sock
```

This file doesn't contain anything, and you shouldn't modify it directly, except for the permissions where applicable. It's just a name.

How Do Sockets Work?

Sockets simply provide the actual hardware for moving data around. TCP-based sockets are called stream sockets, where all data will arrive in order. UDP-based sockets are datagram sockets, where order (or even delivery) isn't guaranteed. There are also raw sockets, which don't have any restrictions, and are used for implementing different protocols and utilities that need to inspect low-level network traffic, like Wireshark.

ADVERTISEMENT

Sockets usually still use TCP or UDP, as they aren't anything special other than a fancy pipe within the kernel. TCP and UDP are transport protocols that define how data gets from place to place but don't really care about what the data is. TCP and UDP provide the platform for most other protocols like FTP, SMTP, and RDP, which operate at higher levels.

It is possible for an application to use a slightly different implementation of TCP; stream sockets use the `SOCK_STREAM` protocol, which is what TCP also uses for transport almost all the time, and while they're basically interchangeable, they're technically slightly different. Though this is low-level stuff and isn't really something you'll have to worry about, just know that *most* traffic sent through UNIX domain sockets is TCP- or UDP-based, or at least quite similar to it, and TCP sent over UNIX domain sockets is faster than TCP over network interfaces like ports.

Socket Use In Practice

Unix sockets are usually used as an alternative to network-based TCP connections when processes are running on the same machine. Data is usually still sent over the same protocols; it just stays within the same machine and knows it's running in the same domain (hence, the name UNIX domain sockets), so it never has to bother a loopback network interface to connect to itself.

The biggest example of this is Redis, an extremely fast key-value store that operates entirely within memory. Redis is frequently used on the same server that's accessing it, so you'll usually be able to use sockets. At such low levels and with how fast Redis is, sockets provide a 25%-performance boost in some synthetic benchmarks.

If you're connecting to a MySQL database, you can also use a socket. Usually you'd connect to `host:port` from a remote system, but if you're connecting to a database on the same server (for example, an REST API accessing a database), you can use sockets for a speedup. This won't affect normal use, but is very noticable when under load, over 20% on a high end 24 core with 128 concurrent users and a million queries per second. Whether or not you'll see a benefit from sockets is a different story, but at that point you'll likely want to be looking into replication and load balancing anyway.

ADVERTISEMENT

If you want to work with sockets manually, you can use the `socat` utility to expose them over network ports:

```
socat TCP-LISTEN:12345 UNIX-CONNECT:/var/lib/socket.sock
```

This does technically defeat the purpose of Unix domain sockets but can be used for debugging at the transport layer.

ANTHONY HEDDINGS

Anthony Heddings is the resident cloud engineer for LifeSavvy Media, a technical writer, programmer, and an expert at Amazon's AWS platform. He's written hundreds of articles for How-To Geek and CloudSavvy IT that have been read millions of times. **READ FULL BIO »**