Listen on a TCP port with connections in the TIME-WAIT state

Content

Tested on

Debian (Lenny, Precise)

Objective

To begin listening on a TCP port whilst there are one or more connections to that port in the TIME-WAIT state, without waiting for the TIME-WAIT state to expire.

Background

When a TCP connection is closed then the socket from which the closure was initiated is not destroyed immediately. Instead it is placed in the TIME-WAIT state, where it is required to remain for at least twice the maximum segment lifetime (MSL) to allow any stray network packets to dissipate. During this period it is not permissible for another TCP connection to be established between the same pair of IP addresses and port numbers.

By itself this would be no great burden, but most implementations go further and (by default) do not allow a local address to be bound to a socket if there are any existing sockets using the same IP address and port number (including sockets in the TIME-WAIT state).

The practical effect of this behaviour is that when a network service terminates leaving connections in the TIME-WAIT state, it may not be possible to restart that service until the TIME-WAIT states have expired. The error reported when this happens is `EADDRINUSE`, which glibc renders as “Address already in use”.

Note that TIME-WAIT is not the only issue that could result in an `EADDRINUSE` error. For example, there could be orphaned child processes that were spawned by the network service but are still handling connections. Alternatively there could be another process listening to the port, perhaps because the previously running instance of the network service failed to die. You can check for these conditions by running the `netstat` command, without the `-l` option for connected sockets:

```
netstat -tn
```

and with the `-l` option for listening sockets:

```
netstat -tln
```

The maximum segment lifetime is implementation-dependent, but is typically in the range 30 seconds to 2 minutes. The minimum lifetime of the TIME-WAIT state is therefore typically in the range 1 to 4 minutes.

Scenario

Suppose you are writing a daemon that provides a TCP-based network service. Currently the following sequence of operations is used to open a server socket and listen on the required port:

```
int fd=socket(AF_INET,SOCK_STREAM,0);
if (fd==-1) {
    die("%s",strerror(errno));
}

if (bind(fd,(struct sockaddr*)&addr,sizeof(addr))==-1) {
    die("%s",strerror(errno));
}

if (listen(fd,SOMAXCONN)==-1) {
    die("%s",strerror(errno));
}
```

When the network service is restarted it sometimes fails with the error “Address already in use”. You wish to prevent this from happening.

Method

The error can be avoided by setting the `SO_REUSEADDR` socket option after the socket has been created but before calling `bind`:

```
int fd=socket(AF_INET,SOCK_STREAM,0);
if (fd==-1) {
    die("%s",strerror(errno));
}

int reuseaddr=1;
if (setsockopt(fd,SOL_SOCKET,SO_REUSEADDR,&reuseaddr,sizeof(reuseaddr))==-1) {
    die("%s",strerror(errno));
}

if (bind(fd,(struct sockaddr*)&addr,sizeof(addr))==-1) {
    die("%s",strerror(errno));
}

if (listen(fd,SOMAXCONN)==-1) {
    die("%s",strerror(errno));
}
```

`SO_REUSEADDR` allows a local address to be bound to a socket even if that address is already being used by a connection. This is helpful not only for dealing with connections in the TIME-WAIT state, but also any `ESTABLISHED` connections that are being handled by orphaned child processes.

It is considered safe for a TCP server socket to reuse a local address, because such sockets are used only to listen for connections and do not themselves act as endpoints. When new connections arrive they will need to be checked to ensure that they do not clash with existing ones, but this is something the network stack should be doing anyway: it makes no difference that the server process has been restarted.

In the absence of any good reason for leaving `SO_REUSEADDR` unset, it is considered good practice to set it as a matter of routine when creating TCP server sockets.

Notes

Depending on the implementation, it may be necessary for `SO_REUSEADDR` to be set both before and after the service is restarted.

`SO_REUSEADDR` does not allow two TCP sockets to listen to the same IP address and port number at the same time.

Methods to avoid

Using SO_LINGER

It is possible to prevent the TIME-WAIT state from being entered in the first place by setting the `SO_LINGER` option with a timeout of zero. This changes the behaviour of the close function: instead of performing a graceful shutdown, it aborts the connection by sending an immediate RST. Any unsent data is discarded and the socket immediately reverts to the CLOSED state.

Whilst this would meet the objective as stated, it is not a desirable solution because it circumvents the protection against stray network packets provided by the TIME-WAIT state. Since `SO_REUSEADDR` achieves the desired effect more safely, there is no justification for using `SO_LINGER` to avoid `EADDRINUSE` errors.

Tags: posix