

Forum:[Hack Forums](#)

Thread: [Multi-Threaded Server Socket Connection + built in memory Load-Balancer](#)

nucleo_type09 - April 23, 2024, 7:08 am

C++ Memory-Safe Multi-Threaded Server Socket Connection Program, can accept multiple connections at the same time using a detached thread for each connection operation, provides memory safety and concurrency control using locking mechanism, uses an internal built in memory load balancing mechanism, the library is very easy to use, just create an instance and call Connect on it, and the server is up and running, you don't need to handle extra configurations, manageable configurations are: [ip-address, port number, IDLE timeout, auto shutdown mechanism, max n of peers, etc...], memory cleanup is already handled by the destructor.

Compiled using gcc compiler on Linux System using c++20, using additional memory security flags such as -fsanitizer-address for catching memory leaks, the full code and library documentation is on github, you can find it here: https://github.com/nucleoType98/Unix_Server_Socket.git

Simple Connection

```
Code #include "server.cpp"
using namespace ServerSocket;
int main()
{
    Socket *SockChannel = new Socket();
    SockChannel->Connect();

    // server is up and running....

    delete SockChannel;
    return 0;
}
```

With More Configurations

```
Code #include "server.cpp"

using namespace ServerSocket;

int main(int argc, char **argv)
{

    // create new Socket Instance, this can also be non-pointer
    Socket *SockChannel = new Socket();

    // @Optional: set verbosity to true, default is true
    SockChannel->set_verbosity_mode(true);

    // @Optional: set auto shutdown mode, after idle timeout expires it will self-shutdown, you need
```

```

to set an idle timeout value for this to have
effect, by default it does NOT automatically shutdown
SockChannel->use_auto_shutdown();

// @Optional: set server max idle time in seconds before auto shutdown for inactivity
SockChannel->set_max_server_idle_time(10);

// @Optional: set server ip address, defaults to localhost AKA 127.0.0.1
SockChannel->set_ip_address("127.0.0.1");

// @Optional: default port is 3300, most systems use this port range for networking operations
anyway
SockChannel->set_port(3300);

// @Optional: set the maximum number of peer connections before rejecting new ones, defaults
to 100
SockChannel->accept_max_peers(50);

// @Optional: use resource memory balancer to prevent system crash
SockChannel->use_memory_balancer();

// @Optional: reserve x megabytes amount of memory to always be available for other system
processes, is good practice to keep this
value not below 500
SockChannel->reserve_memory(900);

// @Required: connect socket server and wait for connections
SockChannel->Connect();

delete SockChannel;

return 0;
}

```

Stanley - April 23, 2024, 10:21 am

I'm Stanley HF's helper bot. The OP (original poster) of this thread wants everyone to see this page and has elected to bump it to the top of the showthread listing. Since HF doesn't want a ton of LQP (Low Quality Posts) this is the best method of bumping your content. There is a small ?yte fee for each bump. Any OP can bump any of their threads on HF from the linked icon on the top of their first post.

If you haven't already read the first post of this thread we encourage you to do so.

Thanks,
Stanley

Hackers - April 23, 2024, 3:48 pm

Nice, I didn't realize that this had load balancing already handled.

Is this something where you could easily see this being used to have multiple chatrooms with their own groups of users connecting? Or would it be better just to run different instances of this same program on different ports?

I'm assuming there's no reason you couldn't have multiple channels in 1 main server running on different ports, but just curious mostly.

nucleo_type09 - April 23, 2024, 4:10 pm

(Apr 23, 2024 03:48 PM)nodes Wrote: [CITE](#) Nice, I didn't realize that this had load balancing already handled.

Is this something where you could easily see this being used to have multiple chatrooms with their own groups of users connecting? Or would it be better just to run different instances of this same program on different ports?

I'm assuming there's no reason you couldn't have multiple channels in 1 main server running on different ports, but just curious mostly.

well it handles how many connections desired, they're all handled by different threads detached from the main thread, all using the same instance, with the same port yes, no need for more instances, just one instance, and all clients will hook to that instance, but i honestly i wrote this to serve another purpose rather than a chat room, it can even be used as a chat room server end point as well tho,as you said, it has builtin memory load balancer, which can be configured as well, it has an IDLE timeout builtin mechanism as well, so when there are no clients connected, it will automatically shutdown after X amount of seconds, this can also be configured haha ?