Transparent Distributed Processing#

The transparent distributed processing capability is implemented using the QNET protocol.

Source Code Layout#

Amazingly enough, the source code for all of Qnet is available at sys/lsm/qnet (here) . Snowballs in Hell, I know. Thing to keep in mind here is that there are many different variants of Qnet, most of which you probably haven't heard of. The original Qnet variant (classic or "compat") specific source is found under l3_qnx and l4_qnx. This version of Qnet has been deprecated and was replaced some years ago by the "lightweight" or lwl4 qnet which keeps it's variant specific source under the l4_lite and qos directories. Generic Qnet source is found under the nr directory (node resolver), the ndb directory (node database) the kif directory (Kernel InterFace - get it?) directory, and the public directory. The arm, mips, ppc, sh and x86 directories are pretty obviously where you will find the compiled binaries. To avoid confusion, disregard the l4_ixp2800 and l4_rio and l4_tcp directories - think of them as science experiments that never saw the light of day.

The source tree also includes all of the source for io-net (although this source base has only been tested with io-pkt at this time) and will build the associated io-net npm-qnet shared libraries.

How do I use TDP with Core Networking 6.4.0?#

Qnet really hasn't changed a whole lot over the years. Heck, even QNX2 had Qnet, albeit only with ARCnet media. QNX4 had a more generic Qnet, which had support for loadable and unloadable drivers for various media, including ARCNet, ethernet, token ring, FDDI, etc. QNX6 Qnet functions the same under the old io-net and new io-pkt infrastructures - the packet format and protocol is the same, so if you're familiar with using Qnet with io-net, you hopefully won't see much of a change with io-pkt. For both io-net and io-pkt, Qnet is just another protocol (like TCP/IP) which transmits and receives packets.

The QNET module in Core Networking 6.4 is now a loadable shared module (lsm-qnet.so). We only support the "l4_lw_lite" variant. qnet-compat (compatible with neutrino 6.2.1) is no longer supported.

To start the stack with qnet,

- io-pkt-v4 -d*driver* -pqnet

will do the trick (as long as you have your PATH and LD_LIBRARY_PATH environment variables set up properly). You can also mount the protocol after the stack has started

- mount -Tio-pkt *full_path_to_dll*/lsm-qnet.so

Note that the "io-net" option to mount is also supported to provide backward compatibility with existing scripts.

The command line options and general configuration information is the same as it was with io-net (please see the online docs for more information). Our on-line docs are located here (replace "io-net" with "io-pkt" and "npm-qnet.so" with "lsm-qnet.so").

How do I use TDP over IP?#

TDP supports two modes of communications: one directly over ethernet and one over IP. The "straight to ethernet" L4 layer is faster and more dynamic than the IP layer, but it isn't possible to route TDP packets out of a single layer 2 domain. Using TDP over IP is the means whereby you can connect to any remote machine over the Internet. This can be accomplished as follows:

1) TDP must use the dns resolver to get an IP address from a hostname (this is the "resolve=dns" option). Configure the local host name and domain and make sure that that all hostnames that you want to talk to (including the local machine) are resolvable by "gethostbyname".

- Use "hostname" to set the hostname
- Use "setconf _CS_DOMAIN" to set your domain
- If the hosts aren't in a DNS database, create an appropriate name to host resolution file in /etc/hosts which includes the fully qualified node name (including domain) and change the resolver to use the host file instead of using the DNS server.
    - For example "setconf _CS_RESOLVE lookup_file_bind"
    - For more information on name resolution, see the "Name servers" section in TCP/IP Networking.

2) Start (or mount) qnet with the bind=ip,resolve=dns option.

- io-pkt-v4-hc -di82544 -pqnet bind=ip,resolve=dns

or

- mount -Tio-pkt -o bind=ip,resolve=dns "full_path_to_dll"/lsm-qnet.so

Unlike raw Ethernet transport, names won't automatically appear in the /net directory. As you perform TDP operations (e.g. ls /net/host1), the entries will be created as required.

How do I use TDP with wifi?#

Wireless ethernet (aka IEEE 802.11 aka wifi) looks an awful lot like the traditional wired ethernet, but due to how wifi is implemented, it can help to relax the various qnet timeout parameters, to allow the wifi layer to execute it's timeout and retransmission protocol. With that in mind, the following are some command line options that make TDP (qnet) work better with wifi:

io-pkt-v4-hc -di82544 -pqnet
res_ticks=5,res_retries=2,conn_est_timeout=5,conn_est_retries=1,tx_ticks=5,tx_retries=5

Note (1) There are no spaces between the command line options, just commas.

Note (2) We recognize that the above is pretty clumsy - PR68864 has been created for the addition of a "wifi" command line option to qnet, to replace the default timeout values, so in the future you won't have to specify all of the above crud.

Note (3) You might need to relax the timeouts for wifi, even on a node that is connected to traditional wired ethernet, if the remote node is wifi. Or, if you even just have a wifi link between the two, with both the local and remote nodes connected to traditional wired ethernet. Capeesh?