

Domain Name Server Access

Obtain IP address given node#

Fri, Feb 23, 1994

Each station has a node# that is used in data requests to indicate where a device resides. Historically, this node# has been part of the physical network address. But with the use of Internet Protocols, this natural node# is less directly useful. Given a node#, how can one determine the IP address needed to support IP communications?

Acnet developed a scheme that uses an alternate node# that is an index in a table of IP addresses that is downloaded to each computer that supports Acnet. As a result, each local station recognizes two node#s. One is the historically natural one, such as 0576; the other is the IP table index, such as 096F.

Until now, specifying a natural node# in a local station page application results in non-IP communications. Use of the 09xx node# forces IP communications. In order to obtain global access to local stations that are on different networks, especially common on ethernet, it may be desirable to force IP communications, if possible. If this is done, how can one derive the IP address given a natural node#? A convention has been used for this purpose in the host support of Macintosh and Sun computers. Rather than use the 09xx values, the natural node#s are used. The Domain Name Server is consulted to translate from a node name such as "node0576.fna1.gov" into an IP address.

Normal domain name resolver code operates synchronously; *i.e.*, it returns only after a reply is received in response to the query to the domain name server. In the context of a local station's real-time operation, this is inconvenient. To use DNS access, it will be necessary to operate asynchronously and to maintain a cache of node# *vs* IP addresses. If a name lookup query is sent to the name server, the frame due to be transmitted may be discarded. When a response from the name server arrives, the IP address can be placed into the cache, so that it will be available the next time access to that node# is needed.

Imagine a table with entries in the following format:

node#	count	IP address
-------	-------	------------

A nonzero node# means the entry is in use. A nonzero IP address means that an entry in use is complete. The counter is used to time out stale entries and also time out responses from the name server. Negative values of the counter mean that a request has been sent to the name server, but no reply has yet been received. Positive values mean that a countdown is in progress until the name server query is retried. If the counters "tick" at once per second, then the maximum delay (32K seconds) is about 9 hours. An IP address of zero (with nonzero node#) means that no response has ever been received from the name server for that node#. If a time out occurs while awaiting a response from the domain name server, and the IP address is nonzero, then leave it the same, as the name server may be temporarily unavailable due to network problems. If a response from the server indicates that the name is undefined, then clear the entry, as an IP address cannot be found for that node#. The table can also be maintained manually, in the absence of an domain name server, if desired.

When a node# is encountered that does not have an entry in the table, use a roving pointer to determine where to start a search for a vacant entry for the new node#. This insures that an entry will not be re-used too soon. In this way, one can use the entry#, perhaps plus an offset, for the request ID in the name server query message. The response ID will then indicate the

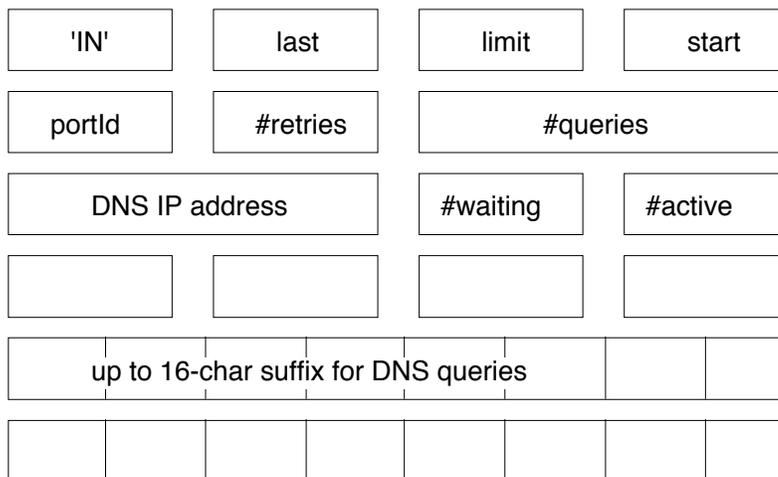
entry whose IP address is to be updated.

This logic is implemented in a local application. Each LA is given a chance to execute each 15 Hz cycle, which provides the opportunity to perform timeout functions. The LA will also run, invoked by the SNAP Task, when a reply is received for the client UDP port, which provides for updating the appropriate table entry. The memory used for the table should be in non-volatile memory, so that the information is not lost upon system reset. As the system code needs access to the table also, the new IP Node Address Table (IPNAT) is system table #27. If the table is undefined, then this feature is supported by the system. If the LA, called DNSQ, is not enabled, then the feature is supported, but only for node#s already in the table. In this way, one can implement support for access to a restricted set of nodes.

How does the system code that queues a message for a network via OUTPQX get the attention of the LA in order to carry out the job of issuing the name server query for an unfamiliar node#? When the LA is enabled, it writes its client *portId*, the entry# in the NETCT table of network and UDP port connections, into the IPNAT table header. The system code can check this *portId* value to lookup the message queue ID used for advising the LA of UDP network messages destined for its UDP port. Writing a special message to this queue advises the LA of the new node# for which it should send a new query. The system clears this *portId* word in the IPNAT header at reset time.

How can a system be configured to use only IP communications? In the third word of the PAGEM table is stored the "broadcast node#" used for name lookups and data requests that are addressed to multiple nodes using multicasting. If this value is in the UDP range of 09Fx, then we can assume the preferred use of IP communications.

IPNAT table header format



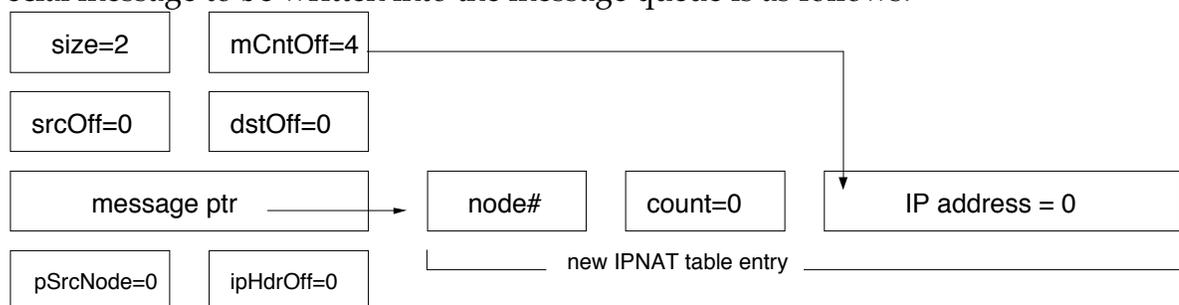
The 'IN' key serves to identify a valid table. (The IP default feature can be disabled by modifying this key.) The *last* word is the offset to the last entry used after searching for an empty one to install a new one. The *limit* word is the size of the table. The *start* word is the offset to the first entry. The *portId* is the index in the NETCT table, placed here by the DNSQ local application when it is enabled. The *#retries* is the count of the number of times that the LA had to repeat sending the name request to the DNS because of no response within the time out period. The *#queries* is the total number of DNS queries since reset. The IP address of the DNS is next, followed by the *#waiting* word, the #entries with waiting queries, and the *#active* word, the #entries actively awaiting a DNS response. A default *suffix* of up to 16 characters,

blank fill, is appended to each node# name of the form "nodexxxx" with xxxx as the node# in hexadecimal. At Fermilab, the suffix ".fnal.gov" is used.

In the OUTPQX routine, when the destination node# is in the appropriate range to be a natural node#, currently 05xx, 06xx, and 07xx, and the system is configured to default to IP communications, *except for reply messages*, consult the entries in the IPNAT table to get an IP address, and call PSNIPARP to obtain a pseudo node# 6xxx. If no IP address is present, make a new entry for this node#, change the destination node# to zero so that it will be discarded, and write a special message into the message queue that is associated with the portID NETCT entry. This will invoke the DNSQ local application, which will see the special message and send a query to the name server for the given node#. (But it will return without waiting for a reply!) During subsequent 15 Hz LA processing, if no response is forthcoming, it will repeat the query. After perhaps 3 times, it will give up, setting the *count* word in the IPNAT entry to retry much later. In this way, responses from the DNS will be cached in the table.

Use of the *count* word for two purposes is as follows: It is divided into two bytes, cntHi and cntLo. Three cases depend upon the cntHi value. If cntHi > 0, cntLo is decremented each second. When it reaches zero, cntHi is decremented. If cntHi reaches zero, the time out has occurred that repeats the query to the DNS to catch up on any IP address changes for that node. In the case that cntHi is negative, cntLo is also decremented each second. When it reaches zero, the query is retried, and the cntHi incremented. If cntHi reaches zero, it means that no response has been received from the DNS at all for this node#, and if the IP address is zero, the entry should be cleared, as the node# may be bogus. If cntHi = 0, no incrementing or decrementing is done. This allows for manual installation of table entries with static IP addresses. This is also useful for foreign nodes where the *suffix* does not apply.

The special message to be written into the message queue is as follows:



This special message is sent via the message queue to the DNSQ local application only when a new entry has just been placed in the IPNAT, the node# to be looked up has been placed into the entry, and the IP address field has been cleared. The *message ptr* is the address of the node# word in the new entry. The *msgOff* value of 4 means that the message count word is the hi word of the IP address field that has been set to zero. The logic in UDPRecv, called by UDPRead, that decrements the message count word to signal that a network message has been processed, first checks to see whether the message count word value is zero; if so, it does not change it. The other four words in this special message are not interpreted by UDPRead. Use of this special message format allows a UDPRead call in the LA to get the network replies from the DNS as well as the special message from the system notifying that a new node# needs to be looked up via the DNS. The LA identifies the special message format by the size=2, just enough for the new node#.