The NIC Name Server--A Datagram Based Information Utility

by

John R. Pickens, Elizabeth J. Feinler, and James E. Mathis

July 1979

SRI International
Telecommunications Sciences Center
333 Ravenswood
Menlo Park, California 94025

NIC Name Server   July 1979

                              Abstract

   In this paper a new method for distributing and updating host
name/address information in large computer networks is described.
The technique uses datagrams to provide a simple
transaction-based query/response service. A provisional service
is being provided by the Arpanet Network Information Center (NIC)
and is used by mobile packet radio terminals, as well as by
several Arpanet DEC-10 hosts. Extensions to the service are
suggested that would expand the query functionality to allow more
flexible query formats as well as queries for service addresses.
Several architectural approaches with potential for expansion
into a distributed internet environment are proposed. This
technique may be utilized in support of other distributed
applications such as user identification and group distribution
for computer based mail.

1. INTRODUCTION

   In large computer networks, such as the Arpanet [1],
network-wide standard host-addressing information must be
maintained and distributed. To fulfill that need, the Arpanet
Network Information Center (NIC) at SRI International has
maintained, administered, and distributed the host addressing
data base to Arpanet hosts since 1972 [2].

   The most common method for maintaining an up-to-date copy on
each host is to bulk-transfer the entire data base to each host
individually. This technique satisfies most host addressing
needs on the Arpanet today. However, some hosts maintain neither
a complete nor a current copy of the data base because of limited
memory capacity and infrequent processing of updates. In
addition, with the expansion of the Arpanet into the internet
environment [3, 4], a strong need has arisen for new techniques
to augment the distribution of name/address information.

   One method currently being investigated is the dynamic
distribution of host-address information via a transaction-based,
inquiry-response process called the Name Server [5, 6]. To
support this investigation, the NIC has developed a provisional
Name Server that is compatible with a level of service specified
in the Defense Advanced Research Projects Agency (DARPA) Internet
experiment [5]. The basic Name Server is described in this paper
and a set of additional functions that such a service might be
expected to support is proposed.

   The discussion is structured as follows: Section 1 describes
the NIC Name Server and how it is derived from the NIC data base
service. Section 2 describes extensions of the name server which
allow a richer syntax and queries for service addresses. Section

3 discusses architectural issues, and presents some preliminary
thoughts on how to evolve from the current centralized,
hierarchic service to a distributed Name Server service.

2. THE NIC NAME SERVER

   A Name Server service has been installed on SRI-KA, an Arpanet
DEC-10. Inquiry-response access is via the Internet Name Server
protocol [5], which in turn employs the DARPA Internet Protocol,
IP4 [7].

   To demonstrate the service a simple interactive facility is
provided to format user input into name server requests--e.g. a
query of the form !ARPANET!FOO-TENEX returns an address such as
"10 2 0 9" (NET=10, HOST=2, LOGICALHOST=0, IMP=9; for details of
host address formats see [8]).

   User access to the name server has been implemented on several
Arpanet DEC-10 TENEX and Packet Radio Network LSI-11 Terminal
Interface Unit (TIU) hosts [9, 10]. While the TENEX program
serves primarily as a demonstration vehicle, the LSI-11 program
provides a valuable augmentation of the TIU's host table. A
typical scenario is that when the packet radio TIU is initiated
or initialized, it contains only a minimal host table, with the
addresses of a few candidate name servers. The user queries the
name server with a simple manual query process, and then uses the
address obtained to open a TELNET connection to the desired host.

   The information to support the name server originates from the
NIC's Arpanet host address data base. An optimized version of
this data base is presented to the name server upon program
initiation as an input file.

3. NAME SERVER ISSUES

   The basic name server provides a simple address-binding service
[5]. In response to a datagram query [7, 11], the name server
returns either an address, a list of similar names if a unique
match is not found, or an error indication. Several useful
additional functions can be envisioned for the name server such
as service queries and broader access to host-related
information.

3.1. Similar Names

   An important issue to be resolved is that of the interpretation
given to the "similar names" response. A standard definition
should be given and not be left to implementors' whims.

If the "similar names" response is used primarily to provide
helpful information to a human-interface process, then it may be
useful to model the behavior of the name server on the behavior
of other known processes that present host name information on
demand. An example of this is a common implementation of virtual
terminal access on the Arpanet, User TELNET [12], in which three
different functions occur:

1. Upon termination of host name input (e.g. <CR>), the
   user is warned only with an audible alarm if the name
   used is not unique. If the name is unique, the name is
   completed, and the requested operation is initiated.

2. In response to <ESC>, either the name will be completed
   if unique or the user will be warned with an audible
   alarm if the name is not unique.

3. Only in response to "?" will a list of similar names be
   printed. "Similar names", in this case, means all
   names that begin with the same character string. The
   list is alphabetized.

   In support of this style of user interface, it may be
appropriate to return the "similar names" response only when
requested. Two ways to achieve this might be either to set an
option bit or to use "?" to force the similar names response.

3.2. Query Syntax

   A second issue in the provision of name server service is that
of query syntax. The basic level of service previously described
allows only a few query functions. With more intelligent name
servers, complex queries can be supported.

   The current Internet name server requires two fields in the
query string--a network name field and a host name field. If the
network field is non-existent, a local network query is assumed.

   Since network independent queries are desirable, an expanded
query functionality must be specified. One way this might be
done is to add to the notion of "missing field", which means
"local", the notion of a special character like "*", which means
"all".

   The semantic range of queries afforded by adopting this
convention is listed below (Note: ~ is used to mean "null". If
both network and host fields are null the representation is "~
~". "N" means "network" and "H" means "host"):

```
~  ~     local net, local host (validity check?)

~  *     local net, all hosts

~  H     local net, named host

*  ~     all nets, local host (inverse search)

*  *     all nets, all hosts (probably prohibited)

*  H     all nets, named host

N  ~     named net, local host (inverse search)

N  *     named net, all hosts

N  H     named net, named host
```

   By combining the on-demand similar-names function, "all" and
"local", and by allowing "*" to be prefixed or appended to the
query string as a wild card character, one can query as follows:

```
~  SRI*?         All hosts named SRI* on local net

*  SRI*?         All hosts named SRI* on all nets

*  *UNIX*?       All hosts named *UNIX* on all nets
```

3.3. Service Queries

   It has been suggested that the name server be generalized into
a binding function [13, 14]. In this context, allowing service
queries is a very useful extension. One application of this
service, that exists within the Packet Radio Project at SRI, is
the need to find the addresses of Hosts that support the
LoaderServer service--a service that allows packet radio TIUs to
receive executable programs via downline loading.

   Service querying, unlike host names querying, requires a
multiple response capability. The querying process would, upon
receiving multiple service descriptors, attempt to establish
access to each service, one at a time, until successful.

   Service descriptors consist of an official name, a list of
alias names, and a network-dependent address. In the case of
Arpanet Internet services, this address field would consist of
the host address(32 bits), port(32 bits), and protocol number(8
bits). For Arpanet NCP services, the address would consist of a
host address(24 bits) and a socket(32 bits).

```

Syntactically, service queries can be derived from host queries by the addition of a service name field, as below:

NET HOST SERVICE

A network-independent service query, for example, can be represented as:

* * SERVICE


## 3.4. Name Server Options

The concept of options has been introduced in the discussion of the "similar names" function.  Another group of options may be used to specify the format of the reply.  At one extreme is the compact, binary, style such as exists in the basic level of service.  At the other extreme is an expanded, textual, style such as is represented by a NIC host table record with official and alias names included.  Options can be envisioned that specify:

- Binary versus text format

- Inclusion of each field in the reply

- Inclusion of official name, per field, in the reply

- Inclusion of alias names, per field, in the reply

- Inclusion of other miscellaneous information, such as
    operating system, machine type, access restrictions,
    and so on.

Other options can be envisioned that specify the scope of the search, such as "find SERVER hosts only".  An alternate form for specifying formats might be to settle on several standard ones, and allow an option to select among them.

Certainly, not all name servers can support all such options, and not all options are equally useful.  Thus, the proposed list will be expanded or contracted to fit the actual needs of processes using the name server service.


## 3.5. MORE DATA Protocol

It should be noted that some of the proposed name server extensions have the potential for generating more than a single datagram's worth of reply, since the current DARPA Internet Protocol limits the size which all networks must support to 576 octets [15].  In spite of this, the size of such replies need not require a full-blown stream protocol.  Several alternatives

exist:

1. Disallow options that imply large replies.

2. Truncate the packet for large replies.

3. Ignore the recommended maximum datagram size.

4. Utilize an alternate base protocol for such requests.

5. Develop a MORE DATA protocol.

If alternative 1 is chosen, the potential for overflow exists, even with the basic level of service.  Alternative 2 implies unpredictable behavior to the user of the name server service. Alternative 3 reduces the availability of the service. Alternative 4 is certainly possible, but may be overkill.

   Alternative 5 appears to be a reasonable solution and could be implemented very simply.  The name server could return, as part of the reply, a code of the following form:

```
+------+---------+
| MORE | ID_NEXT |
+------+---------+
```

where ID_NEXT is a name-server-chosen quantity that allows the name server to find the next block of reply, the next time it is queried.  This quantity may be an internal pointer, a block number, or whatever the name server chooses.  Follow-on queries may be implemented by recomputing the entire original query and discarding output until the ID_NEXT block is reached, or by efficiently storing the entire reply in a cache, fragmented into blocks (with appropriate decay algorithms).


3.6. Dynamic Updates

   In the previous discussion, the host name data base was assumed to have been a static or slowly changing entity with an administrative and manual update authority.  This model reflects most of the network needs in the Arpanet and Internet communities.  However, dynamic automated updating of the host table may be needed in the future, especially in mobile environments such as the Packet Radio Network.

   In a closed user group community (such as a local network of mutually trusting hosts), dynamic updating becomes simply a technical question concerning packet formats.  In wider communities, a mechanism to authenticate the change request must be developed; however, the authentication issue is outside the scope of this paper.

## 4. ARCHITECTURE

   The Name Server concept is invaluable for allowing hosts with
incomplete knowledge of the network address space to obtain full
access to network services.  Whether for reasons of insufficient
kernel space or of dynamically changing environments, the need
for the service is little questioned.  However, significant
design issues revolve around the methods for providing the
service and for administering and updating the data base.

   In the current NIC Name Server, the service is centralized, and
is supported by a data base administered by a single authority.
In the long range, other architectures are possible that present
more flexible ways to distribute host information within and
between networks and administrative entities.  These present
opportunities for dynamic, automated, approaches to the
maintenance and sharing of data--particularly host name data.

   From an evolutionary point of view, initial Name Server
services will likely exist as a centralized service, possibly
with one large Name Server that has knowledge of multiple
networks.  From this beginning, an expansion in two orthogonal
directions can be envisioned.

   - In the direction of internal distribution, the name
     server can be partitioned into multiple, cooperating
     processes on separate hosts.  The data base can be
     replicated exactly or managed as a distributed data
     base.

   - In the direction of administrative distribution,
     multiple autonomous name servers may exist that
     exchange data in an appropriate fashion, on a per
     network or other administrative basis.

   For hosts with small host tables, caching might be used,
whereby local, temporary copies would be maintained of subsets of
the addressing data base.  Such copies may be obtained either by
remembering previous queries made of name servers, or by
receiving automatic distributions of data from name servers.  For
mobile hosts, in which even the home network is unknown, it would
be possible to maintain a very sparse table with the addresses of
only a few name servers.

   For hosts lacking even the knowledge of name server addresses,
a very primitive name server function can be provided by all
network hosts that would allow real name servers to be located;
e.g., a query of the form "*  *  RealNameServer" addressed to an
arbitrarily selected host could elicit the address of a real Name
Server.

   Finally, the possibility exists for multiple name servers to
communicate dynamically in attempting to resolve queries.  If,

for example, a name server on the Arpanet receives a query for a
host on the Packet Radio Network, then the Arpanet name server
can conceivably query the Packet Radio Network Name Server to
resolve the reply.


## 5. FUTURE WORK

The techniques discussed in this paper for providing dynamic
access to and maintenance of host address information are
generally applicable to other information-providing services.
Two possibilities currently under investigation at SRI include
user identification servers [16] and time servers, which offer
people/address and time-stamp information, respectively, as
datagram driven information utilities.

Further work is needed to refine the implementation of the name
server and its using query processes.  Two items in particular
are direct system calls into the NIC data base management system
for general access to host information and process-level query
interfaces for using processes.  The design issues for efficient
implementation are complex and involve some trade-offs.  The most
obvious trade-off is volume of network traffic versus "freshness"
of information. The local host table handler can either send a
message to the name server for every address request, or it can
use some type of local cache to remember frequently requested
names.  SRI is exploring both the process-level interface for the
LSI-11 TIU and the problems of local host table management in
small machines operating in a dynamic environment.

Information services such as the Name Server are integral
components of distributed systems and applications.  However, the
effective utilization of such services is a relatively unstudied
and unexplored area.  One area in which SRI plans to study their
impact on distributed architectures is in computer based mail
applications.  Information utilities in this environment can
range from the support of simple mailbox address queries to
complex address list management needed for inter-organizational
and group resolution.


## 6. CONCLUSION

A provisional Name Server service, based on the NIC host
address data base was described in this paper.  In addition, a
collection of design ideas for an internet Name Server service
has been presented.

Work is continuing on the refinement of the NIC name server
service.  New requirements and opportunities for functional
distribution are being investigated.  Many questions have been

raised in exploring an expansion of the existing service. Such an expansion is expected to result in more useful support of internet (and intranet) capability.


REFERENCES

1.    L. G. Roberts and B. D. Wessler, "Computer Network Development to Achieve Resource Sharing," in Proceedings of SJCC, pp. 543-549 (AFIP, 1970).

2.    M. D. Kudlick and E. J. Feinler, Host Names On-line, RFC 608, Stanford Research Institute, Menlo Park, California (January 1974).

3.    V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Interconnection," IEEE Transactions on Communication Technology, Vol. COM-22, pp. 637-641 (1974).

4.    V. G. Cerf and P. T. Kirstein, "Issues in Packet-Network Interconnection," Proc. IEEE, Vol. 66, No. 11, pp. 1386-1408 (November 1978).

5.    J. Postel, Internet Name Server, IEN 89, Information Sciences Institute, Univ. of Southern Calif., Marina Del Rey, California, May 1979.

6.    J. R. Pickens, E. J. Feinler and J. E. Mathis, An Experimental Network Information Center Name Server (NICNAME), IEN 103, SRI International, Menlo Park, California (May 1979).

7.    J. Postel, Internet Protocol, IEN 81, Information Sciences Institute, Univ. of Southern Calif., Marina Del Rey, California (February 1979).

8.    J. Postel, Address Mappings, IEN 91, Information Sciences Institute, Univ. of Southern Calif., Marina Del Rey, California (May 1979).

9.    R. E. Kahn, "The Organization of Computer Resources into a Packet Radio Network," IEEE Transactions on Communications, Vol. COM-25, No. 1, pp. 169-178 (January 1977).

10.   R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, "Advances in Packet Radio Technology," Proc. IEEE, Vol. 66, No. 11, pp. 1468-1496 (November 1978).

11.   J. Postel, User Datagram Protocol, IEN 88, Information Sciences Institute, Univ. of Southern Calif., Marina Del Rey, California (May 1979).

12.    E. Leavitt, TENEX USER'S GUIDE, Bolt Beranek and Newman,
       Inc., Cambridge, Massachusetts.

13.    R. Bressler, A Proposed Experiment With a Message Switching
       Protocol (section on Information Operator), pp. 26-31, RFC
       333, Bolt Beranek and Newman Inc, Cambridge, Mass. (May 15,
       1972).

14.    Y. Dalal, Internet Meeting, January 24,25 1979, (Group
       Discussion).

15.    J. Postel, Internet Meeting Notes - 25,26 January 1979, pp.
       12, IEN 76, Information Sciences Institute, Univ. of
       Southern Calif., Marina Del Rey, California (February
       1979).

16.    E. J. Feinler, "The Identification Data Base in a
       Networking Environment," in NTC '77 Conference Record, pp.
       21:3 (IEEE, 1977).

# Table of Contents