IEN 61
Section 2.5.6.1

Internet Name Server

Jon Postel

27 October 1978

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291

(213) 822-1511

INTERNET NAME SERVER

NAME SYNTAX

It is strongly recommended that the use of host names in programs be consistent for both input and output across all hosts. To promote such consistency of the internet level, the following syntax is specified:

The SYNTAX of names as presented to the user and as entered by the user is:

```
  ! NET ! REST
```

where:

NET is a network name or number as defined in [1]

and

REST is a host name within that network expressed as a character string or as a number. When a number is used, it is expressed in decimals and is prefixed with a sharpsign (e.g., #1234).

Note that this syntax has minimal impact on the allowable character strings for host names within a network. The only restriction is that a REST string cannot begin with an exclaimation point (!).

The !NET! may be omitted when specifying a host in the local network. That is "!" indicates the network portion of a name string.

NAME SERVER

To aid in the translation of names to internet addresses, several name server processes will be provided. The name server process will accept a name in the above form and will return a name, address pair.

The name server processes will have well-known addresses; addresses that are constant over long periods of time and published in documents such as [1].

A request sent to a name server is sent as an internet datagram [2] with the following content:

```
+--------+--------+--------+--------+--------+--------+---\\---+
!        !        !                                            !
!  NAME  ! LENGTH !                NAME STRING                 !
!        !        !                                            !
+--------+--------+--------+--------+--------+--------+---\\---+
```

where:

NAME is a one octet code indicating that the following is a name,

LENGTH is a one octet count of the number of octets in the name string, and

NAME STRING is an ASCII character string of the form ! NET ! REST.

A reply to a successful translation is sent as an internet datagram with the following content:

```
+--------+--------+--------+--------+--------+--------+---\\---+
!        !        !                                             !
!  NAME  ! LENGTH !                 NAME STRING                 !
!        !        !                                             !
+--------+--------+--------+--------+--------+--------+---\\---+
!        !        !                                    !
! ADDRESS! LENGTH !         INTERNET ADDRESS           !
!        !        !                                    !
+--------+--------+--------+--------+--------+--------+
```

where:

ADDRESS is a one octet code indicating that the following is an internet address,

LENGTH' is a one octet count (=4) of the length of the internet address, and

INTERNET ADDRESS is the internet address.

Actually a particular name might map to several internet addresses, in this case the response would include a list of internet addresses.

When a name is not found, an error is reported via an internet datagram as follows:

```
+--------+--------+--------+--------+--------+--------+---\\---+
!        !        !                                            !
!  NAME  ! LENGTH !                 NAME STRING                !
!        !        !                                            !
+--------+--------+--------+--------+--------+--------+---\\---+
!        !        ! ERROR  !                                   !
! ERROR  ! LENGTH ! CODE   !        ERROR STRING               !
!        !        !        !                                   !
+--------+--------+--------+--------+--------+--------+---\\---+
```

where:

ERROR CODE specifies the error.

ERROR STRING explains the error.

Error Codes

The following error codes are defined:

CODE	MEANING
0	Undetermined or undefined error
1	Name not found
2	Improper name syntax
3	Name not found, but the following similar names exist

Note Error 3 is followed by a list of name strings.

Communication with a Name Server Process

Communication with a name server process is via internet datagrams. Internet datagrams do not guarantee reliable communication. Thus, some requests or replies may be lost.

The name server process is a transaction oriented process; furthermore, the nature of the transactions allows them to be processed in any order and even to be duplicated. This allows the use of a very simple communication protocol.

If a request is made to the name server process and no response is received within a reasonable time, then the requestor should make the request again. This recovers from communication errors which cause the loss of either the request or the reply.

In order to use this simple strategy, care must be taken to allow replies to be properly matched with requests. The name server process does this by including in each reply a copy of the entire request.

The internet datagram does not even guarantee the correctness of the data. The name server protocol must provide measures to prevent incorrect data from being used. For example, an interent address field could be affected by transmission errors.

For this reason, the name server protocol includes a checksum over the information of the requests and replies. The checksum algorithm is the same as is used in the internet protocol (which covers only the internet header).

Format

The requests and replies to and from a name server process are encoded as "items". An item consists of an item-code an item-length and the item-data.

```
  Item  :=  Item-Code  Item-Length  Item-Data

  +--------+--------+--------+--------+--------+--------+---\\---+
  !        !        !                                            !
  !  Item  !  Item  !              Item                          !
  !  Code  ! Length !                     Data                   !
  !        !        !                                            !
  +--------+--------+--------+--------+--------+--------+---\\---+
```

A request is typically one item, and a reply is typically two items. In any case the items communicated at one time are grouped into a "block". A block consists of an octet of zero, an octet identifying this as an name server block, a two octet block length, a two octet checksum field, and some items.

```
  Block  :=  BHead  Items
  BHead  :=  Zero  NS-id  Block-Length  Checksum
  Items  :=  Item  !  Items  Item
```

```
+--------+--------+-----------------+
!        !  Name  !      Block      !
!  Zero  ! Server !      Length     !
+-----------------+--------+--------+
!     Checksum    !ItemCode!Item Len!
+--------+--------+--------+--------+
! .......... Item Data .......... !
+--------+--------+--------+--------+
! ........ Item Data cont ........ !
+--------+--------+--------+--------+
! Item Data cont. !ItemCode!Item Len!
+--------+--------+--------+--------+
! .......... Item Data .......... !
+--------+--------+--------+--------+
```

This format is chosen to be compatible with the proposed multiplexing format for internet protocol. Also the name server is one application using raw internet datagrams as the transmission media, so the internet header protocol field should carry the value indicating raw datagram. An additional field is needed somewhere to multiplex the various applications using raw datagrams.

Item Code Value Assignments:

NAME = 1

ADDRESS = 2

ERROR = 3

Example

a typical request:

```
+--------+--------+-----------------+
!        !  NAME  !                 !
!   0    ! SERVER !       14        !
+-----------------+--------+--------+
!     CHECKSUM    !   1    !   10   !
+--------+--------+--------+--------+
!   !    !   A    !   R    !   P    !
+--------+--------+--------+--------+
!   A    !   !    !   I    !   S    !
+--------+--------+--------+--------+
!   I    !   B    !
+--------+--------+
```

and the reply:

```
+--------+--------+-----------------+
!        ! NAME   !                 !
!   0    ! SERVER !       20        !
+-----------------+--------+--------+
!    CHECKSUM     !   1    !   10   !
+--------+--------+--------+--------+
!   !    !   A    !   R    !   P    !
+--------+--------+--------+--------+
!   A    !   !    !   I    !   S    !
+--------+--------+--------+--------+
!   I    !   B    !   2    !   4    !
+--------+--------+--------+--------+
!   10   !   3    !   0    !   52   !
+--------+--------+--------+--------+
```

References

[1] J. Postel. "Assigned Numbers," RFC 750, NIC 45500, 26 September 1978.

[2] J. Postel. "Internetwork Protocol Specification -- Version 4," IEN 54, USC-Information Sciences Institute, September 1978.