Network Working Group
Request for Comments: 1611
Category: Standards Track

R. Austein
Epilogue Technology Corporation
J. Saperia
Digital Equipment Corporation

May 1994

DNS Server MIB Extensions

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes a set of extensions which instrument DNS name server functions. This memo was produced by the DNS working group.

With the adoption of the Internet-standard Network Management Framework [4,5,6,7], and with a large number of vendor implementations of these standards in commercially available products, it became possible to provide a higher level of effective network management in TCP/IP-based internets than was previously available. With the growth in the use of these standards, it has become possible to consider the management of other elements of the infrastructure beyond the basic TCP/IP protocols. A key element of

the TCP/IP infrastructure is the DNS.

Up to this point there has been no mechanism to integrate the management of the DNS with SNMP-based managers. This memo provides the mechanisms by which IP-based management stations can effectively manage DNS name server software in an integrated fashion.

We have defined DNS MIB objects to be used in conjunction with the Internet MIB to allow access to and control of DNS name server software via SNMP by the Internet community.

2. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework consists of four major components. They are:

- RFC 1442 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
- STD 17, RFC 1213 defines MIB-II, the core set of managed objects for the Internet suite of protocols.
- RFC 1445 which defines the administrative and other architectural aspects of the framework.
- RFC 1448 which defines the protocol used for network access to managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

2.1. Object Definitions

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) defined in the SMI. In particular, each object object type is named by an OBJECT IDENTIFIER, an administratively assigned name. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the descriptor, to refer to the object type.

3. Overview

In theory, the DNS world is pretty simple. There are two kinds of entities: resolvers and name servers. Resolvers ask questions. Name servers answer them. The real world, however, is not so simple.

Implementors have made widely differing choices about how to divide DNS functions between resolvers and servers. They have also constructed various sorts of exotic hybrids. The most difficult task in defining this MIB was to accommodate this wide range of entities without having to come up with a separate MIB for each.

We divided up the various DNS functions into two, non-overlapping classes, called "resolver functions" and "name server functions." A DNS entity that performs what we define as resolver functions contains a resolver, and therefore must implement the MIB groups required of all resolvers which are defined in a separate MIB Module. A DNS entity which implements name server functions is considered to be a name server, and must implement the MIB groups required for name servers in this module. If the same piece of software performs both resolver and server functions, we imagine that it contains both a resolver and a server and would thus implement both the DNS Server and DNS Resolver MIBs.

3.1. Resolvers

In our model, a resolver is a program (or piece thereof) which obtains resource records from servers. Normally it does so at the behest of an application, but may also do so as part of its own operation. A resolver sends DNS protocol queries and receives DNS protocol replies. A resolver neither receives queries nor sends replies. A full service resolver is one that knows how to resolve queries: it obtains the needed resource records by contacting a server authoritative for the records desired. A stub resolver does not know how to resolve queries: it sends all queries to a local name server, setting the "recursion desired" flag to indicate that it hopes that the name server will be willing to resolve the query. A resolver may (optionally) have a cache for remembering previously acquired resource records. It may also have a negative cache for remembering names or data that have been determined not to exist.

3.2. Name Servers

A name server is a program (or piece thereof) that provides resource records to resolvers. All references in this document to "a name server" imply "the name server's role"; in some cases the name server's role and the resolver's role might be combined into a single program. A name server receives DNS protocol queries and sends DNS protocol replies. A name server neither sends queries nor receives replies. As a consequence, name servers do not have caches. Normally, a name server would expect to receive only those queries to which it could respond with authoritative information. However, if a name server receives a query that it cannot respond to with purely authoritative information, it may choose to try to obtain the

necessary additional information from a resolver which may or may not be a separate process.

3.3. Selected Objects

Many of the objects included in this memo have been created from information contained in the DNS specifications [1,2], as amended and clarified by subsequent host requirements documents [3]. Other objects have been created based on experience with existing DNS management tools, expected operational needs, the statistics generated by existing DNS implementations, and the configuration files used by existing DNS implementations. These objects have been ordered into groups as follows:

- Server Configuration Group
- Server Counter Group
- Server Optional Counter Group
- Server Zone Group

This information has been converted into a standard form using the SNMPv2 SMI defined in [9]. For the most part, the descriptions are influenced by the DNS related RFCs noted above. For example, the descriptions for counters used for the various types of queries of DNS records are influenced by the definitions used for the various record types found in [2].

3.4. Textual Conventions

Several conceptual data types have been introduced as a textual conventions in this DNS MIB document. These additions will facilitate the common understanding of information used by the DNS. No changes to the SMI or the SNMP are necessary to support these conventions.

Readers familiar with MIBs designed to manage entities in the lower layers of the Internet protocol suite may be surprised at the number of non-enumerated integers used in this MIB to represent values such as DNS RR class and type numbers. The reason for this choice is simple: the DNS itself is designed as an extensible protocol, allowing new classes and types of resource records to be added to the protocol without recoding the core DNS software. Using non-enumerated integers to represent these data types in this MIB allows the MIB to accommodate these changes as well.

4. Definitions

```
DNS-SERVER-MIB DEFINITIONS ::= BEGIN

IMPORTS
    mib-2
        FROM RFC-1213
    MODULE-IDENTITY, OBJECT-TYPE, OBJECT-IDENTITY,
    IpAddress, Counter32, Gauge32
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION, RowStatus, DisplayString, TruthValue
        FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP
        FROM SNMPv2-CONF;

dns OBJECT-IDENTITY
    STATUS  current
    DESCRIPTION
            "The OID assigned to DNS MIB work by the IANA."
    ::= { mib-2 32 }

dnsServMIB MODULE-IDENTITY
    LAST-UPDATED "9401282251Z"
    ORGANIZATION "IETF DNS Working Group"
    CONTACT-INFO
            "       Rob Austein
            Postal: Epilogue Technology Corporation
                    268 Main Street, Suite 283
                    North Reading, MA 10864
                    US
               Tel: +1 617 245 0804
               Fax: +1 617 245 8122
            E-Mail: sra@epilogue.com

                    Jon Saperia
            Postal: Digital Equipment Corporation
                    110 Spit Brook Road
                    ZKO1-3/H18
                    Nashua, NH 03062-2698
                    US
               Tel: +1 603 881 0480
               Fax: +1 603 881 0120
             Email: saperia@zko.dec.com"
    DESCRIPTION
            "The MIB module for entities implementing the server side
            of the Domain Name System (DNS) protocol."
    ::= { dns 1 }
```

```
   dnsServMIBObjects       OBJECT IDENTIFIER ::= { dnsServMIB 1 }

   -- (Old-style) groups in the DNS server MIB.

   dnsServConfig           OBJECT IDENTIFIER ::= { dnsServMIBObjects 1 }
   dnsServCounter          OBJECT IDENTIFIER ::= { dnsServMIBObjects 2 }
   dnsServOptCounter       OBJECT IDENTIFIER ::= { dnsServMIBObjects 3 }
   dnsServZone             OBJECT IDENTIFIER ::= { dnsServMIBObjects 4 }


   -- Textual conventions

   DnsName ::= TEXTUAL-CONVENTION
       -- A DISPLAY-HINT would be nice, but difficult to express.
       STATUS  current
       DESCRIPTION
               "A DNS name is a sequence of labels.  When DNS names are
               displayed, the boundaries between labels are typically
               indicated by dots (e.g. 'Acme' and 'COM' are labels in
               the name 'Acme.COM').  In the DNS protocol, however, no
               such separators are needed because each label is encoded
               as a length octet followed by the indicated number of
               octets of label.  For example, 'Acme.COM' is encoded as
               the octet sequence { 4, 'A', 'c', 'm', 'e', 3, 'C', 'O',
               'M', 0 } (the final 0 is the length of the name of the
               root domain, which appears implicitly at the end of any
               DNS name).  This MIB uses the same encoding as the DNS
               protocol.

               A DnsName must always be a fully qualified name.  It is
               an error to encode a relative domain name as a DnsName
               without first making it a fully qualified name."
       REFERENCE
               "RFC-1034 section 3.1."
       SYNTAX  OCTET STRING (SIZE (0..255))

   DnsNameAsIndex ::= TEXTUAL-CONVENTION
       STATUS  current
       DESCRIPTION
               "This textual convention is like a DnsName, but is used
               as an index componant in tables.  Alphabetic characters
               in names of this type are restricted to uppercase: the
               characters 'a' through 'z' are mapped to the characters
               'A' through 'Z'.  This restriction is intended to make
               the lexical ordering imposed by SNMP useful when applied
               to DNS names.

               Note that it is theoretically possible for a valid DNS
```

```
                name to exceed the allowed length of an SNMP object
                identifer, and thus be impossible to represent in tables
                in this MIB that are indexed by DNS name.  Sampling of
                DNS names in current use on the Internet suggests that
                this limit does not pose a serious problem in practice."
        REFERENCE
                "RFC-1034 section 3.1, RFC-1448 section 4.1."
        SYNTAX  DnsName

    DnsClass ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "2d"
        STATUS  current
        DESCRIPTION
                "This data type is used to represent the class values
                which appear in Resource Records in the DNS.  A 16-bit
                unsigned integer is used to allow room for new classes
                of records to be defined.  Existing standard classes are
                listed in the DNS specifications."
        REFERENCE
                "RFC-1035 section 3.2.4."
        SYNTAX  INTEGER (0..65535)

    DnsType ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "2d"
        STATUS  current
        DESCRIPTION
                "This data type is used to represent the type values
                which appear in Resource Records in the DNS.  A 16-bit
                unsigned integer is used to allow room for new record
                types to be defined.  Existing standard types are listed
                in the DNS specifications."
        REFERENCE
                "RFC-1035 section 3.2.2."
        SYNTAX  INTEGER (0..65535)

    DnsQClass ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "2d"
        STATUS  current
        DESCRIPTION
                "This data type is used to represent the QClass values
                which appear in Resource Records in the DNS.  A 16-bit
                unsigned integer is used to allow room for new QClass
                records to be defined.  Existing standard QClasses are
                listed in the DNS specification."
        REFERENCE
                "RFC-1035 section 3.2.5."
        SYNTAX  INTEGER (0..65535)
```

```
   DnsQType ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "2d"
       STATUS  current
       DESCRIPTION
               "This data type is used to represent the QType values
               which appear in Resource Records in the DNS.  A 16-bit
               unsigned integer is used to allow room for new QType
               records to be defined.  Existing standard QTypes are
               listed in the DNS specification."
       REFERENCE
               "RFC-1035 section 3.2.3."
       SYNTAX  INTEGER (0..65535)

   DnsTime ::= TEXTUAL-CONVENTION
       DISPLAY-HINT "4d"
       STATUS  current
       DESCRIPTION
               "DnsTime values are 32-bit unsigned integers which
               measure time in seconds."
       REFERENCE
               "RFC-1035."
       SYNTAX  Gauge32


   DnsOpCode ::= TEXTUAL-CONVENTION
       STATUS  current
       DESCRIPTION
               "This textual convention is used to represent the DNS
               OPCODE values used in the header section of DNS
               messages.  Existing standard OPCODE values are listed in
               the DNS specifications."
       REFERENCE
               "RFC-1035 section 4.1.1."
       SYNTAX  INTEGER (0..15)

   DnsRespCode ::= TEXTUAL-CONVENTION
       STATUS  current
       DESCRIPTION
               "This data type is used to represent the DNS RCODE value
               in DNS response messages.  Existing standard RCODE
               values are listed in the DNS specifications."
       REFERENCE
               "RFC-1035 section 4.1.1."
       SYNTAX  INTEGER (0..15)
```

```
   -- Server Configuration Group

   dnsServConfigImplementIdent OBJECT-TYPE
       SYNTAX      DisplayString
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "The implementation identification string for the DNS
               server software in use on the system, for example;
               'FNS-2.1'"
       ::= { dnsServConfig 1 }

   dnsServConfigRecurs  OBJECT-TYPE
       SYNTAX      INTEGER { available(1),
                             restricted(2),
                             unavailable(3) }
       MAX-ACCESS  read-write
       STATUS      current
       DESCRIPTION
               "This represents the recursion services offered by this
               name server.  The values that can be read or written
               are:

               available(1) - performs recursion on requests from
               clients.

               restricted(2) - recursion is performed on requests only
               from certain clients, for example; clients on an access
               control list.

               unavailable(3) - recursion is not available."
        ::= { dnsServConfig 2 }

   dnsServConfigUpTime OBJECT-TYPE
       SYNTAX      DnsTime
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "If the server has a persistent state (e.g., a process),
               this value will be the time elapsed since it started.
               For software without persistant state, this value will
               be zero."
       ::= { dnsServConfig 3 }

   dnsServConfigResetTime OBJECT-TYPE
       SYNTAX      DnsTime
       MAX-ACCESS  read-only
       STATUS      current
```

```
        DESCRIPTION
                "If the server has a persistent state (e.g., a process)
                and supports a 'reset' operation (e.g., can be told to
                re-read configuration files), this value will be the
                time elapsed since the last time the name server was
                'reset.'  For software that does not have persistence or
                does not support a 'reset' operation, this value will be
                zero."
        ::= { dnsServConfig 4 }

    dnsServConfigReset OBJECT-TYPE
        SYNTAX      INTEGER { other(1),
                              reset(2),
                              initializing(3),
                              running(4) }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "Status/action object to reinitialize any persistant name
                server state.  When set to reset(2), any persistant
                name server state (such as a process) is reinitialized as
                if the name server had just been started.  This value
                will never be returned by a read operation.  When read,
                one of the following values will be returned:
                    other(1) - server in some unknown state;
                    initializing(3) - server (re)initializing;
                    running(4) - server currently running."
        ::= { dnsServConfig 5 }


    -- Server Counter Group

    dnsServCounterAuthAns OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of queries which were authoritatively answered."
        ::= { dnsServCounter 2 }

    dnsServCounterAuthNoNames OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of queries for which 'authoritative no such name'
                responses were made."
        ::= { dnsServCounter 3 }
```

```
   dnsServCounterAuthNoDataResps OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries for which 'authoritative no such data'
               (empty answer) responses were made."
       ::= { dnsServCounter 4 }

   dnsServCounterNonAuthDatas OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries which were non-authoritatively
               answered (cached data)."
       ::= { dnsServCounter 5 }

   dnsServCounterNonAuthNoDatas OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries which were non-authoritatively
               answered with no data (empty answer)."
       ::= { dnsServCounter 6 }

   dnsServCounterReferrals OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of requests that were referred to other servers."
       ::= { dnsServCounter 7 }

   dnsServCounterErrors OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of requests the server has processed that were
               answered with errors (RCODE values other than 0 and 3)."
       REFERENCE
               "RFC-1035 section 4.1.1."
       ::= { dnsServCounter 8 }

   dnsServCounterRelNames OBJECT-TYPE
       SYNTAX      Counter32
```

```
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received by the server for names that
                are only 1 label long (text form - no internal dots)."
        ::= { dnsServCounter 9 }

    dnsServCounterReqRefusals OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of DNS requests refused by the server."
        ::= { dnsServCounter 10 }

    dnsServCounterReqUnparses OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received which were unparseable."
        ::= { dnsServCounter 11 }

    dnsServCounterOtherErrors OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests which were aborted for other (local)
                server errors."
        ::= { dnsServCounter 12 }

    -- DNS Server Counter Table

    dnsServCounterTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DnsServCounterEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "Counter information broken down by DNS class and type."
        ::= { dnsServCounter 13 }

    dnsServCounterEntry OBJECT-TYPE
        SYNTAX      DnsServCounterEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "This table contains count information for each DNS class
```

```
                and type value known to the server.  The index allows
                management software to to create indices to the table to
                get the specific information desired, e.g., number of
                queries over UDP for records with type value 'A' which
                came to this server.  In order to prevent an
                uncontrolled expansion of rows in the table; if
                dnsServCounterRequests is 0 and dnsServCounterResponses
                is 0, then the row does not exist and 'no such' is
                returned when the agent is queried for such instances."
        INDEX     { dnsServCounterOpCode,
                    dnsServCounterQClass,
                    dnsServCounterQType,
                    dnsServCounterTransport }
        ::= { dnsServCounterTable 1 }

    DnsServCounterEntry ::=
        SEQUENCE {
            dnsServCounterOpCode
                DnsOpCode,
            dnsServCounterQClass
                DnsClass,
            dnsServCounterQType
                DnsType,
            dnsServCounterTransport
                INTEGER,
            dnsServCounterRequests
                Counter32,
            dnsServCounterResponses
                Counter32
            }

    dnsServCounterOpCode OBJECT-TYPE
        SYNTAX      DnsOpCode
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "The DNS OPCODE being counted in this row of the table."
        ::= { dnsServCounterEntry 1 }

    dnsServCounterQClass OBJECT-TYPE
        SYNTAX      DnsClass
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "The class of record being counted in this row of the
                table."
        ::= { dnsServCounterEntry 2 }
```

```
   dnsServCounterQType OBJECT-TYPE
       SYNTAX      DnsType
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
               "The type of record which is being counted in this row in
               the table."
       ::= { dnsServCounterEntry 3 }

   dnsServCounterTransport OBJECT-TYPE
       SYNTAX      INTEGER { udp(1), tcp(2), other(3) }
       MAX-ACCESS  not-accessible
       STATUS      current
       DESCRIPTION
               "A value of udp(1) indicates that the queries reported on
               this row were sent using UDP.

               A value of tcp(2) indicates that the queries reported on
               this row were sent using TCP.

               A value of other(3) indicates that the queries reported
               on this row were sent using a transport that was neither
               TCP nor UDP."
       ::= { dnsServCounterEntry 4 }

   dnsServCounterRequests OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of requests (queries) that have been recorded in
               this row of the table."
       ::= { dnsServCounterEntry 5 }

   dnsServCounterResponses OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of responses made by the server since
               initialization for the kind of query identified on this
               row of the table."
       ::= { dnsServCounterEntry 6 }
```

```
   -- Server Optional Counter Group

   -- The Server Optional Counter Group is intended for those systems
   -- which make distinctions between the different sources of the DNS
   -- queries as defined below.
   --
   -- Objects in this group are implemented on servers which distinguish
   -- between queries which originate from the same host as the server,
   -- queries from one of an arbitrary group of hosts that are on an
   -- access list defined by the server, and queries from hosts that do
   -- not fit either of these descriptions.
   --
   -- The objects found in the Server Counter group are totals.  Thus if
   -- one wanted to identify, for example, the number of queries from
   -- 'remote' hosts which have been given authoritative answers, one
   -- would subtract the current values of ServOptCounterFriendsAuthAns
   -- and ServOptCounterSelfAuthAns from servCounterAuthAns.
   --
   -- The purpose of these distinctions is to allow for implementations
   -- to group queries and responses on this basis.  One way in which
   -- servers may make these distinctions is by looking at the source IP
   -- address of the DNS query.  If the source of the query is 'your
   -- own' then the query should be counted as 'yourself' (local host).
   -- If the source of the query matches an 'access list,' the query
   -- came from a friend.  What constitutes an 'access list' is
   -- implementation dependent and could be as simple as a rule that all
   -- hosts on the same IP network as the DNS server are classed
   -- 'friends.'
   --
   -- In order to avoid double counting, the following rules apply:
   --
   -- 1. No host is in more than one of the three groups defined above.
   --
   -- 2. All queries from the local host are always counted in the
   --    'yourself' group regardless of what the access list, if any,
   --    says.
   --
   -- 3. The access list should not define 'your friends' in such a way
   --    that it includes all hosts.   That is, not everybody is your
   --    'friend.'

   dnsServOptCounterSelfAuthAns OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of requests the server has processed which
               originated from a resolver on the same host for which
```

```
                there has been an authoritative answer."
        ::= { dnsServOptCounter 1 }

    dnsServOptCounterSelfAuthNoNames OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from a resolver on the same host for which
                there has been an authoritative no such name answer
                given."
        ::= { dnsServOptCounter 2 }

    dnsServOptCounterSelfAuthNoDataResps OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from a resolver on the same host for which
                there has been an authoritative no such data answer
                (empty answer) made."
        ::= { dnsServOptCounter 3 }

    dnsServOptCounterSelfNonAuthDatas OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from a resolver on the same host for which a
                non-authoritative answer (cached data) was made."
        ::= { dnsServOptCounter 4 }

    dnsServOptCounterSelfNonAuthNoDatas OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from a resolver on the same host for which a
                'non-authoritative, no such data' response was made
                (empty answer)."
        ::= { dnsServOptCounter 5 }

    dnsServOptCounterSelfReferrals OBJECT-TYPE
        SYNTAX      Counter32
```

```
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of queries the server has processed which
                originated from a resolver on the same host and were
                referred to other servers."
        ::= { dnsServOptCounter 6 }

    dnsServOptCounterSelfErrors OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from a resolver on the same host which have
                been answered with errors (RCODEs other than 0 and 3)."
        REFERENCE
                "RFC-1035 section 4.1.1."
        ::= { dnsServOptCounter 7 }

    dnsServOptCounterSelfRelNames OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received for names that are only 1
                label long (text form - no internal dots) the server has
                processed which originated from a resolver on the same
                host."
        ::= { dnsServOptCounter 8 }

    dnsServOptCounterSelfReqRefusals OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of DNS requests refused by the server which
                originated from a resolver on the same host."
        ::= { dnsServOptCounter 9 }

    dnsServOptCounterSelfReqUnparses OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received which were unparseable and
                which originated from a resolver on the same host."
        ::= { dnsServOptCounter 10 }
```

```
   dnsServOptCounterSelfOtherErrors OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of requests which were aborted for other (local)
               server errors and which originated on the same host."
       ::= { dnsServOptCounter 11 }

   dnsServOptCounterFriendsAuthAns OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries originating from friends which were
               authoritatively answered.  The definition of friends is
               a locally defined matter."
       ::= { dnsServOptCounter 12 }

   dnsServOptCounterFriendsAuthNoNames OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries originating from friends, for which
               authoritative 'no such name' responses were made.  The
               definition of friends is a locally defined matter."
       ::= { dnsServOptCounter 13 }

   dnsServOptCounterFriendsAuthNoDataResps OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries originating from friends for which
               authoritative no such data (empty answer) responses were
               made.  The definition of friends is a locally defined
               matter."
       ::= { dnsServOptCounter 14 }

   dnsServOptCounterFriendsNonAuthDatas OBJECT-TYPE
       SYNTAX      Counter32
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
               "Number of queries originating from friends which were
               non-authoritatively answered (cached data). The
               definition of friends is a locally defined matter."
```

```
        ::= { dnsServOptCounter 15 }

    dnsServOptCounterFriendsNonAuthNoDatas OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of queries originating from friends which were
                non-authoritatively answered with no such data (empty
                answer)."
        ::= { dnsServOptCounter 16 }

    dnsServOptCounterFriendsReferrals OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests which originated from friends that
                were referred to other servers.  The definition of
                friends is a locally defined matter."
        ::= { dnsServOptCounter 17 }

    dnsServOptCounterFriendsErrors OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests the server has processed which
                originated from friends and were answered with errors
                (RCODE values other than 0 and 3).  The definition of
                friends is a locally defined matter."
        REFERENCE
                "RFC-1035 section 4.1.1."
        ::= { dnsServOptCounter 18 }

    dnsServOptCounterFriendsRelNames OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received for names from friends that
                are only 1 label long (text form - no internal dots) the
                server has processed."
        ::= { dnsServOptCounter 19 }

    dnsServOptCounterFriendsReqRefusals OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
```

```
        STATUS      current
        DESCRIPTION
                "Number of DNS requests refused by the server which were
                received from 'friends'."
        ::= { dnsServOptCounter 20 }

    dnsServOptCounterFriendsReqUnparses OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests received which were unparseable and
                which originated from 'friends'."
        ::= { dnsServOptCounter 21 }

    dnsServOptCounterFriendsOtherErrors OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Number of requests which were aborted for other (local)
                server errors and which originated from 'friends'."
        ::= { dnsServOptCounter 22 }


    -- Server Zone Group

    -- DNS Management Zone Configuration Table

    -- This table contains zone configuration information.

    dnsServZoneTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DnsServZoneEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "Table of zones for which this name server provides
                information.  Each of the zones may be loaded from stable
                storage via an implementation-specific mechanism or may
                be obtained from another name server via a zone transfer.

                If name server doesn't load any zones, this table is
                empty."
        ::= { dnsServZone 1 }

    dnsServZoneEntry OBJECT-TYPE
        SYNTAX      DnsServZoneEntry
        MAX-ACCESS  not-accessible
```

```
        STATUS      current
        DESCRIPTION
                "An entry in the name server zone table.  New rows may be
                added either via SNMP or by the name server itself."
        INDEX     { dnsServZoneName,
                    dnsServZoneClass }
        ::= { dnsServZoneTable 1 }

    DnsServZoneEntry ::=
        SEQUENCE {
            dnsServZoneName
                DnsNameAsIndex,
            dnsServZoneClass
                DnsClass,
            dnsServZoneLastReloadSuccess
                DnsTime,
            dnsServZoneLastReloadAttempt
                DnsTime,
            dnsServZoneLastSourceAttempt
                IpAddress,
            dnsServZoneStatus
                RowStatus,
            dnsServZoneSerial
                Counter32,
            dnsServZoneCurrent
                TruthValue,
            dnsServZoneLastSourceSuccess
                IpAddress
        }

    dnsServZoneName OBJECT-TYPE
        SYNTAX      DnsNameAsIndex
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "DNS name of the zone described by this row of the table.
                This is the owner name of the SOA RR that defines the
                top of the zone. This is name is in uppercase:
                characters 'a' through 'z' are mapped to 'A' through 'Z'
                in order to make the lexical ordering useful."
        ::= { dnsServZoneEntry 1 }

    dnsServZoneClass OBJECT-TYPE
        SYNTAX      DnsClass
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "DNS class of the RRs in this zone."
```

```
        ::= { dnsServZoneEntry 2 }

    dnsServZoneLastReloadSuccess OBJECT-TYPE
        SYNTAX      DnsTime
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Elapsed time in seconds since last successful reload of
                this zone."
        ::= { dnsServZoneEntry 3 }

    dnsServZoneLastReloadAttempt OBJECT-TYPE
        SYNTAX      DnsTime
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Elapsed time in seconds since last attempted reload of
                this zone."
        ::= { dnsServZoneEntry 4 }

    dnsServZoneLastSourceAttempt OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "IP address of host from which most recent zone transfer
                of this zone was attempted.  This value should match the
                value of dnsServZoneSourceSuccess if the attempt was
                succcessful.  If zone transfer has not been attempted
                within the memory of this name server, this value should
                be 0.0.0.0."
        ::= { dnsServZoneEntry 5 }

    dnsServZoneStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
                "The status of the information represented in this row of
                the table."
        ::= { dnsServZoneEntry 6 }

    dnsServZoneSerial OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Zone serial number (from the SOA RR) of the zone
```

```
                represented by this row of the table.  If the zone has
                not been successfully loaded within the memory of this
                name server, the value of this variable is zero."
        ::= { dnsServZoneEntry 7 }

    dnsServZoneCurrent OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Whether the server's copy of the zone represented by
                this row of the table is currently valid.  If the zone
                has never been successfully loaded or has expired since
                it was last succesfully loaded, this variable will have
                the value false(2), otherwise this variable will have
                the value true(1)."
        ::= { dnsServZoneEntry 8 }

    dnsServZoneLastSourceSuccess OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "IP address of host which was the source of the most
                recent successful zone transfer for this zone.  If
                unknown (e.g., zone has never been successfully
                transfered) or irrelevant (e.g., zone was loaded from
                stable storage), this value should be 0.0.0.0."
        ::= { dnsServZoneEntry 9 }

    -- DNS Zone Source Table

    dnsServZoneSrcTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DnsServZoneSrcEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "This table is a list of IP addresses from which the
                server will attempt to load zone information using DNS
                zone transfer operations.  A reload may occur due to SNMP
                operations that create a row in dnsServZoneTable or a
                SET to object dnsServZoneReload.  This table is only
                used when the zone is loaded via zone transfer."
        ::= { dnsServZone 2 }

    dnsServZoneSrcEntry OBJECT-TYPE
        SYNTAX      DnsServZoneSrcEntry
        MAX-ACCESS  not-accessible
```

```
        STATUS      current
        DESCRIPTION
                "An entry in the name server zone source table."
        INDEX     { dnsServZoneSrcName,
                    dnsServZoneSrcClass,
                    dnsServZoneSrcAddr }
        ::= { dnsServZoneSrcTable 1 }

    DnsServZoneSrcEntry ::=
        SEQUENCE {
            dnsServZoneSrcName
                DnsNameAsIndex,
            dnsServZoneSrcClass
                DnsClass,
            dnsServZoneSrcAddr
                IpAddress,
            dnsServZoneSrcStatus
                RowStatus
        }

    dnsServZoneSrcName OBJECT-TYPE
        SYNTAX      DnsNameAsIndex
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "DNS name of the zone to which this entry applies."
        ::= { dnsServZoneSrcEntry 1 }

    dnsServZoneSrcClass OBJECT-TYPE
        SYNTAX      DnsClass
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "DNS class of zone to which this entry applies."
        ::= { dnsServZoneSrcEntry 2 }

    dnsServZoneSrcAddr OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "IP address of name server host from which this zone
                might be obtainable."
        ::= { dnsServZoneSrcEntry 3 }

    dnsServZoneSrcStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
```

```
        STATUS      current
        DESCRIPTION
                "The status of the information represented in this row of
                the table."
        ::= { dnsServZoneSrcEntry 4 }


    -- SNMPv2 groups.

    dnsServMIBGroups        OBJECT IDENTIFIER ::= { dnsServMIB 2 }

    dnsServConfigGroup OBJECT-GROUP
        OBJECTS   { dnsServConfigImplementIdent,
                    dnsServConfigRecurs,
                    dnsServConfigUpTime,
                    dnsServConfigResetTime,
                    dnsServConfigReset }
        STATUS      current
        DESCRIPTION
                "A collection of objects providing basic configuration
                control of a DNS name server."
        ::= { dnsServMIBGroups 1 }

    dnsServCounterGroup OBJECT-GROUP
        OBJECTS   { dnsServCounterAuthAns,
                    dnsServCounterAuthNoNames,
                    dnsServCounterAuthNoDataResps,
                    dnsServCounterNonAuthDatas,
                    dnsServCounterNonAuthNoDatas,
                    dnsServCounterReferrals,
                    dnsServCounterErrors,
                    dnsServCounterRelNames,
                    dnsServCounterReqRefusals,
                    dnsServCounterReqUnparses,
                    dnsServCounterOtherErrors,
                    dnsServCounterOpCode,
                    dnsServCounterQClass,
                    dnsServCounterQType,
                    dnsServCounterTransport,
                    dnsServCounterRequests,
                    dnsServCounterResponses }
        STATUS      current
        DESCRIPTION
                "A collection of objects providing basic instrumentation
                of a DNS name server."
        ::= { dnsServMIBGroups 2 }
```

```
   dnsServOptCounterGroup OBJECT-GROUP
       OBJECTS   { dnsServOptCounterSelfAuthAns,
                   dnsServOptCounterSelfAuthNoNames,
                   dnsServOptCounterSelfAuthNoDataResps,
                   dnsServOptCounterSelfNonAuthDatas,
                   dnsServOptCounterSelfNonAuthNoDatas,
                   dnsServOptCounterSelfReferrals,
                   dnsServOptCounterSelfErrors,
                   dnsServOptCounterSelfRelNames,
                   dnsServOptCounterSelfReqRefusals,
                   dnsServOptCounterSelfReqUnparses,
                   dnsServOptCounterSelfOtherErrors,
                   dnsServOptCounterFriendsAuthAns,
                   dnsServOptCounterFriendsAuthNoNames,
                   dnsServOptCounterFriendsAuthNoDataResps,
                   dnsServOptCounterFriendsNonAuthDatas,
                   dnsServOptCounterFriendsNonAuthNoDatas,
                   dnsServOptCounterFriendsReferrals,
                   dnsServOptCounterFriendsErrors,
                   dnsServOptCounterFriendsRelNames,
                   dnsServOptCounterFriendsReqRefusals,
                   dnsServOptCounterFriendsReqUnparses,
                   dnsServOptCounterFriendsOtherErrors }
       STATUS      current
       DESCRIPTION
               "A collection of objects providing extended
               instrumentation of a DNS name server."
       ::= { dnsServMIBGroups 3 }

   dnsServZoneGroup OBJECT-GROUP
       OBJECTS   { dnsServZoneName,
                   dnsServZoneClass,
                   dnsServZoneLastReloadSuccess,
                   dnsServZoneLastReloadAttempt,
                   dnsServZoneLastSourceAttempt,
                   dnsServZoneLastSourceSuccess,
                   dnsServZoneStatus,
                   dnsServZoneSerial,
                   dnsServZoneCurrent,
                   dnsServZoneSrcName,
                   dnsServZoneSrcClass,
                   dnsServZoneSrcAddr,
                   dnsServZoneSrcStatus }
       STATUS      current
       DESCRIPTION
               "A collection of objects providing configuration control
               of a DNS name server which loads authoritative zones."
       ::= { dnsServMIBGroups 4 }
```

```
   -- Compliances.

   dnsServMIBCompliances OBJECT IDENTIFIER ::= { dnsServMIB 3 }

   dnsServMIBCompliance MODULE-COMPLIANCE
       STATUS      current
       DESCRIPTION
               "The compliance statement for agents implementing the DNS
               name server MIB extensions."
       MODULE -- This MIB module
           MANDATORY-GROUPS { dnsServConfigGroup, dnsServCounterGroup }
           GROUP   dnsServOptCounterGroup
           DESCRIPTION
               "The server optional counter group is unconditionally
               optional."
           GROUP   dnsServZoneGroup
           DESCRIPTION
               "The server zone group is mandatory for any name server
               that acts as an authoritative server for any DNS zone."
           OBJECT  dnsServConfigRecurs
           MIN-ACCESS      read-only
           DESCRIPTION
               "This object need not be writable."
           OBJECT  dnsServConfigReset
           MIN-ACCESS      read-only
           DESCRIPTION
               "This object need not be writable."
       ::= { dnsServMIBCompliances 1 }

   END
```

5. Acknowledgements

This document is the result of work undertaken the by DNS working group. The authors would particularly like to thank the following people for their contributions to this document: Philip Almquist, Frank Kastenholz (FTP Software), Joe Peck (DEC), Dave Perkins (SynOptics), Win Treese (DEC), and Mimi Zohar (IBM).

6. References

[1] Mockapetris, P., "Domain Names -- Concepts and Facilities", STD 13, RFC 1034, USC/Information Sciences Institute, November 1987.

[2] Mockapetris, P., "Domain Names -- Implementation and Specification", STD 13, RFC 1035, USC/Information Sciences Institute, November 1987.

[3] Braden, R., Editor, "Requirements for Internet Hosts -- Application and Support, STD 3, RFC 1123, USC/Information Sciences Institute, October 1989.

[4] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets", STD 16, RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[5] McCloghrie, K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1156, Hughes LAN Systems, Performance Systems International, May 1990.

[6] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[7] Rose, M., and K. McCloghrie, Editors, "Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.

[8] McCloghrie, K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, Hughes LAN Systems, Performance Systems International, March 1991.

[9] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1442, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon

University, April 1993.

[10] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Textual Conventions for version 2 of the the Simple Network Management Protocol (SNMPv2)", RFC 1443, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

[11] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Conformance Statements for version 2 of the the Simple Network Management Protocol (SNMPv2)", RFC 1444, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

[12] Galvin, J., and K. McCloghrie, "Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1445, Trusted Information Systems, Hughes LAN Systems, April 1993.

[13] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1448, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.

[14] "Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)", International Organization for Standardization, International Standard 8824, December 1987.

7. Security Considerations

Security issues are not discussed in this memo.

8. Authors' Addresses

Rob Austein
Epilogue Technology Corporation
268 Main Street, Suite 283
North Reading, MA 01864
USA

Phone: +1-617-245-0804
Fax: +1-617-245-8122
EMail: sra@epilogue.com

Jon Saperia
Digital Equipment Corporation
110 Spit Brook Road
ZKO1-3/H18
Nashua, NH 03062-2698
USA

Phone: +1-603-881-0480
Fax: +1-603-881-0120
EMail: saperia@zko.dec.com