             Remote Network Monitoring MIB Protocol Identifier Macros

Status of this Memo

Copyright Notice

Abstract

   This memo contains various protocol identifier examples, which can be
   used to produce valid protocolDirTable INDEX encodings, as defined by
   the Remote Network Monitoring MIB (Management Information Base)
   Version 2 [RFC2021] and the RMON Protocol Identifier Reference
   [RFC2895].

   This document contains protocol identifier macros for well-known
   protocols. A conformant implementation of the RMON-2 MIB [RFC2021]
   can be accomplished without the use of these protocol identifiers,
   and accordingly, this document does not specify any IETF standard.
   It is published to encourage better interoperability between RMON-2
   agent implementations, by providing a great deal of RMON related
   protocol information in one document.

   The first version of the RMON Protocol Identifiers Document [RFC2074]
   has been split into a standards-track Reference portion [RFC2895],
   and an "RMON Protocol Identifier Macros", document (this document)
   which contains the non-normative portion of that specification.

Table of Contents

1.   The SNMP Network Management Framework

   The SNMP Management Framework presently consists of five major
   components:

   o   An overall architecture, described in RFC 2571 [RFC2571].

   o   Mechanisms for describing and naming objects and events for the
        purpose of management. The first version of this Structure of
        Management Information (SMI) is called SMIv1 and described in
        STD 16, RFC 1155 [RFC1155], STD 16, RFC 1212 [RFC1212] and RFC
        1215 [RFC1215]. The second version, called SMIv2, is described
        in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and
        STD 58, RFC 2580 [RFC2580].

   o   Message protocols for transferring management information. The
        first version of the SNMP message protocol is called SNMPv1 and
        described in STD 15, RFC 1157 [RFC1157]. A second version of the
        SNMP message protocol, which is not an Internet standards track
        protocol, is called SNMPv2c and described in RFC 1901 [RFC1901]
        and RFC 1906 [RFC1906]. The third version of the message
        protocol is called SNMPv3 and described in RFC 1906 [RFC1906],
        RFC 2572 [RFC2572] and RFC 2574 [RFC2574].

   o   Protocol operations for accessing management information. The
        first set of protocol operations and associated PDU formats is
        described in STD 15, RFC 1157 [RFC1157]. A second set o
        protocol operations and associated PDU formats is described in
        RFC 1905 [RFC1905].

      o   A set of fundamental applications described in RFC 2573
           [RFC2573] and the view-based access control mechanism described
           in RFC 2575 [RFC2575].

   A more detailed introduction to the current SNMP Management Framework
   can be found in RFC 2570 [RFC2570].

   Managed objects are accessed via a virtual information store, termed
   the Management Information Base or MIB. Objects in the MIB are
   defined using the mechanisms defined in the SMI.

   This memo does not specify a MIB module.

2. Overview

   The RMON-2 MIB [RFC2021] uses hierarchically formatted OCTET STRINGs
   to globally identify individual protocol encapsulations in the
   protocolDirTable.

   This guide contains examples of protocol identifier encapsulations,
   which can be used to describe valid protocolDirTable entries. The
   syntax of the protocol identifier descriptor is defined in the RMON
   Protocol Identifier Reference [RFC2895].

   This document is not intended to be an authoritative reference on the
   protocols described herein. Refer to the Official Internet Standards
   document [RFC2600], the Assigned Numbers document [RFC1700], or other
   appropriate RFCs, IEEE documents, etc. for complete and authoritative
   protocol information.

   This is the the second revision of this document, and is intended to
   replace Section 5 of the first RMON-2 Protocol Identifiers document
   [RFC2074].

   The RMONMIB working group has decided to discontinue maintenance of
   this Protocol Identifier Macro repository document, due to a lack of
   contributions from the RMON vendor community. This document is
   published as an aid in implementation of the protocolDirTable.

2.1. Terms

   Refer to the RMON Protocol Identifier Reference [RFC2895] for
   definitions of terms used to describe the Protocol Identifier Macro
   and aspects of protocolDirTable INDEX encoding.

2.2. Relationship to the Remote Network Monitoring MIB

   This document is intended to describe some protocol identifier
   macros, which can be converted to valid protocolDirTable INDEX
   values, using the mapping rules defined in the RMON Protocol
   Identifier Reference [RFC2895].

   This document is not intended to limit the protocols that may be
   identified for counting in the RMON-2 MIB. Many protocol
   encapsulations, not explicitly identified in this document, may be
   present in an actual implementation of the protocolDirTable. Also,
   implementations of the protocolDirTable may not include all the
   protocols identified in the example section below.

2.3. Relationship to the RMON Protocol Identifier Reference

   This document is intentionally separated from the normative reference
   document defining protocolDirTable INDEX encoding rules and the
   protocol identifier macro syntax [RFC2895]. This allows frequent
   updates to this document without any republication of MIB objects or
   protocolDirTable INDEX encoding rules. Note that the base layer and
   IANA assigned protocol identifier macros are located in Reference
   document, since these encoding values are defined by the RMONMIB WG.

   Protocol Identifier macros submitted from the RMON working group and
   community at large (to the RMONMIB WG mailing list at '
   rmonmib@cisco.com') will be collected and added to this document.

   Macros submissions will be collected in the IANA's MIB files under
   the directory "ftp://ftp.isi.edu/mib/rmonmib/rmon2_pi_macros/" and in
   the RMONMIB working group mailing list message archive file
   "ftp://ftpeng.cisco.com/ftp/rmonmib/rmonmib".

2.4. Relationship to Other MIBs

   The RMON Protocol Identifier Macros document is intended for use with
   the RMON Protocol Identifier Reference [RFC2895] and the RMON-2 MIB
   protocolDirTable [RFC2021]. It is not relevant to any other MIB, or
   intended for use with any other MIB.

3. Protocol Identifier Macros

   This section contains protocol identifier macros for some well-known
   protocols, although some of them may no longer be in use. These
   macros reference the base layer identifiers found in section 4 of the
   RMON Protocol Identifier Reference [RFC2895]. These identifiers are
   listed below:

```
ether2
llc
snap
vsnap
ianaAssigned
802-1Q
```

Refer to the RMON Protocol Identifier Reference [RFC2895] for the
protocol identifier macro definitions for these protocols.

3.1. Protocol Stacks And Single-Vendor Applications

Network layer protocol identifier macros contain additional
information about the network layer, and is found immediately
following a base layer-identifier in a protocol identifier.

The ProtocolDirParameters supported at the network layer are '
countsFragments(0)', and 'tracksSessions(1). An agent may choose to
implement a subset of these parameters.

The protocol-name should be used for the ProtocolDirDescr field. The
ProtocolDirType ATTRIBUTES used at the network layer are '
hasChildren(0)' and 'addressRecognitionCapable(1)'. Agents may choose
to implement a subset of these attributes for each protocol, and
therefore limit which tables the indicated protocol can be present
(e.g. protocol distribution, host, and matrix tables).

The following protocol-identifier macro declarations are given for
example purposes only. They are not intended to constitute an
exhaustive list or an authoritative source for any of the protocol
information given. However, any protocol that can encapsulate other
protocols must be documented here in order to encode the children
identifiers into protocolDirID strings. Leaf protocols should be
documented as well, but an implementation can identify a leaf
protocol even if it isn't listed here (as long as the parent is
documented).

3.1.1. The TCP/IP protocol stack

```
arp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "An Address Resolution Protocol message (request or response).
       This protocol does not include Reverse ARP (RARP) packets, which
       are counted separately."
    REFERENCE
       "RFC 826 [RFC826] defines the Address Resolution Protocol."
```

```
    ::= {
     ether2 0x806,    -- [ 0.0.8.6 ]
     snap   0x806,
     802-1Q 0x806     -- [ 0.0.8.6 ]
    }

ip PROTOCOL-IDENTIFIER
    PARAMETERS {
        countsFragments(0)  -- This parameter applies to all child
                            -- protocols.
    }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
    }
    DESCRIPTION
        "The protocol identifiers for the Internet Protocol (IP). Note
        that IP may be encapsulated within itself, so more than one of
        the following identifiers may be present in a particular
        protocolDirID string."
    CHILDREN
        "Children of 'ip' are selected by the value in the Protocol field
        (one octet), as defined in the PROTOCOL NUMBERS table within the
        Assigned Numbers Document.

        The value of the Protocol field is encoded in an octet string as
        [ 0.0.0.a ], where 'a' is the protocol field .

        Children of 'ip' are encoded as [ 0.0.0.a ], and named as 'ip a'
        where 'a' is the protocol field value.  For example, a
        protocolDirID-fragment value of:
           0.0.0.1.0.0.8.0.0.0.0.1

        defines an encapsulation of ICMP (ether2.ip.icmp)"
    ADDRESS-FORMAT
        "4 octets of the IP address, in network byte order.  Each ip
        packet contains two addresses, the source address and the
        destination address."
    DECODING
        "Note: ether2.ip.ipip4.udp is a different protocolDirID than
        ether2.ip.udp, as identified in the protocolDirTable.  As such,
        two different local protocol index values will be assigned by the
        agent. E.g. (full INDEX values shown):
         ether2.ip.ipip4.udp =
             16.0.0.0.1.0.0.8.0.0.0.0.4.0.0.0.17.4.0.0.0.0
         ether2.ip.udp =
             12.0.0.0.1.0.0.8.0.0.0.0.17.3.0.0.0 "
    REFERENCE
```

```
         "RFC 791 [RFC791] defines the Internet Protocol; The following
         URL defines the authoritative repository for the PROTOCOL NUMBERS
         Table:

            ftp://ftp.isi.edu/in-notes/iana/assignments/protocol-numbers"
     ::= {
        ether2     0x0800,
        llc        0x06,
        snap       0x0800,
           -- ip        4,           ** represented by the ipip4 macro
           -- ip        94,          ** represented by the ipip macro
        802-1Q     0x0800,           -- [0.0.8.0]
        802-1Q     0x02000006        -- 1Q-LLC [2.0.0.6]
     }

  -- ************************************************************
  --
  --                      Children of IP
  --
  -- ************************************************************

 icmp PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Internet Message Control Protocol"
     REFERENCE
        "RFC 792 [RFC792] defines the Internet Control Message Protocol."
     ::= {
      ip 1,
      ipip4 1,
      ipip 1
     }

 igmp PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Internet Group Management Protocol; IGMP is used by IP hosts to
        report their host group memberships to any immediately-
        neighboring multicast routers."
     REFERENCE
        "Appendix A of Host Extensions for IP Multicasting [RFC1112]
        defines the Internet Group Management Protocol."
     ::= {
      ip 2,
      ipip4 2,
      ipip 2
```

```
    }

ggp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Gateway-to-Gateway Protocol; DARPA Internet Gateway
       (historical)"
    REFERENCE
       "RFC 823 [RFC823] defines the Gateway-to-Gateway Protocol."
    ::= {
     ip 3,
     ipip4 3,
     ipip 3
    }

ipip4 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
    }
    DESCRIPTION
       "IP in IP Tunneling"
    CHILDREN
       "Children of 'ipip4' are selected and encoded in the same manner
       as children of IP."
    ADDRESS-FORMAT
       "The 'ipip4' address format is the same as the IP address
       format."
    DECODING
       "Note: ether2.ip.ipip4.udp is a different protocolDirID than
       ether2.ip.udp, as identified in the protocolDirTable.  As such,
       two different local protocol index values will be assigned by the
       agent. E.g. (full INDEX values shown):
        ether2.ip.ipip4.udp =
            16.0.0.0.1.0.0.8.0.0.0.0.4.0.0.0.17.4.0.0.0.0
        ether2.ip.udp =
            12.0.0.0.1.0.0.8.0.0.0.0.17.3.0.0.0 "
    REFERENCE
       "RFC 1853 [RFC1853] defines IP in IP over Protocol 4."
    ::= {
     ip 4,
     ipip4 4,
     ipip 4
    }

st PROTOCOL-IDENTIFIER
```

```
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Internet Stream Protocol Version 2 (ST2); (historical) ST2 is an
       experimental resource reservation protocol intended to provide
       end-to-end real-time guarantees over an internet."
    REFERENCE
       "RFC 1819 [RFC1819] defines version 2 of the Internet Stream
       Protocol."
    ::= {
     ip 5,
     ipip4 5,
     ipip 5
    }

tcp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
      hasChildren(0)
    }
    DESCRIPTION
       "Transmission Control Protocol"
    CHILDREN
       "Children of TCP are identified by the 16 bit Source or
       Destination Port value as specified in RFC 793. They are encoded
       as [ 0.0.a.b], where 'a' is the MSB and 'b' is the LSB of the
       port value. Both bytes are encoded in network byte order.  For
       example, a protocolDirId-fragment of:
           0.0.0.1.0.0.8.0.0.0.0.6.0.0.0.23

       identifies an encapsulation of the telnet protocol
       (ether2.ip.tcp.telnet)"
    REFERENCE
       "RFC 793 [RFC793] defines the Transmission Control Protocol.

       The following URL defines the authoritative repository for
       reserved and registered TCP port values:

          ftp://ftp.isi.edu/in-notes/iana/assignments/port-numbers"
    ::=  {
     ip 6,
     ipip4 6,
     ipip 6
    }

egp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
```

```
    DESCRIPTION
       "Exterior Gateway Protocol (historical)"
    REFERENCE
       "RFC 904 [RFC904] defines the Exterior Gateway Protocol."
    ::= {
     ip  8,
     ipip4  8,
     ipip  8
    }

igp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Any private interior gateway."
    REFERENCE
       "[RFC1700]"
    ::= {
     ip  9,
     ipip4  9,
     ipip  9
    }

nvp2 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "NVP-II; Network Voice Protocol"
    REFERENCE
       "RFC 741 [RFC741] defines the Network Voice Protocol"
    ::= {
     ip 11,
     ipip4 11,
     ipip 11
    }

pup PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "PUP Protocol"
    REFERENCE
       "Xerox"
    ::= {
     ip 12,
     ipip4 12,
     ipip 12
    }
```

```
xnet PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Cross Net Debugger (historical)"
    REFERENCE
       "[IEN158]"
    ::= {
     ip  15,
     ipip4  15,
     ipip  15
    }

chaos PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "CHAOS Protocol; historical"
    REFERENCE
       "J. Noel Chiappa  <JNC@XX.LCS.MIT.EDU>"
    ::= {
     ip 16,
     ipip4 16,
     ipip 16
    }

udp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
      hasChildren(0)
    }
    DESCRIPTION
       "User Datagram Protocol"
    CHILDREN
       "Children of UDP are identified by the 16 bit Source or
       Destination Port value as specified in RFC 768. They are encoded
       as [ 0.0.a.b ], where 'a' is the MSB and 'b' is the LSB of the
       port value. Both bytes are encoded in network byte order.  For
       example, a protocolDirId-fragment of:
           0.0.0.1.0.0.8.0.0.0.0.17.0.0.0.161

       identifies an encapsulation of SNMP (ether2.ip.udp.snmp)"
    REFERENCE
       "RFC 768 [RFC768] defines the User Datagram Protocol.

       The following URL defines the authoritative repository for
       reserved and registered UDP port values:
```

```
           ftp://ftp.isi.edu/in-notes/iana/assignments/port-numbers"
    ::= {
     ip 17,
     ipip4 17,
     ipip 17
     }

mux PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Multiplexing Protocol (historical)"
    REFERENCE
       "IEN-90 [IEN-90] defines the Multiplexing Protocol"
    ::= {
     ip 18,
     ipip4 18,
     ipip 18
     }

hmp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Host Monitoring Protocol; historical"
    REFERENCE
       "RFC 869 [RFC869] defines the Host Monitoring Protocol"
    ::= {
     ip  20,
     ipip4  20,
     ipip  20
     }

xns-idp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "XEROX NS IDP"
    REFERENCE
       "Xerox Corporation"
    ::= {
     ip  22,
     ipip4  22,
     ipip  22
     }

rdp PROTOCOL-IDENTIFIER
    PARAMETERS { }
```

```
     ATTRIBUTES { }
     DESCRIPTION
        "Reliable Data Protocol"
     REFERENCE
        "RFC 908 [RFC908] defines the original protocol; RFC 1151
        [RFC1151] defines version 2 of the Reliable Data Protocol."
     ::= {
      ip 27,
      ipip4 27,
      ipip 27
     }

irtp PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Internet Reliable Transaction Protocol"
     REFERENCE
        "RFC 938 [RFC938] defines the Internet Reliable Transaction
        Protocol functional and  interface specification."
     ::= {
      ip 28,
      ipip4 28,
      ipip 28
     }

iso-tp4  PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "ISO Transport Protocol Specification"
     REFERENCE
        "RFC 905 [RFC905] defines the ISO Transport Protocol
        Specification; ISO DP 8073"
     ::= {
      ip  29,
      ipip4  29,
      ipip  29
     }

netblt PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Bulk Data Transfer Protocol; historical"
     REFERENCE
        "RFC 998 [RFC998] defines NETBLT: A Bulk Data Transfer Protocol."
     ::= {
```

```
     ip 30,
     ipip4 30,
     ipip 30
    }

mfe-nsp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "MFE Network Services Protocol; historical"
    REFERENCE
        "Shuttleworth, B., 'A Documentary of MFENet, a National Computer
        Network', UCRL-52317, Lawrence Livermore Labs, Livermore,
        California, June 1977."
    ::= {
     ip 31,
     ipip4 31,
     ipip 31
    }

idpr PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Inter-Domain Policy Routing Protocol"
    REFERENCE
        "RFC 1479 [RFC1479] defines Version 1 of the Inter-Domain Policy
        Routing Protocol."
    ::= {
     ip 35,
     ipip4 35,
     ipip 35
    }

idpr-cmtp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "IDPR Control Message Transport Protocol"
    REFERENCE
        "RFC 1479 [RFC1479] defines Version 1 of the Inter-Domain Policy
        Routing Protocol."
    ::= {
     ip 38,
     ipip4 38,
     ipip 38
    }
```

```
sdrp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Source Demand Routing Protocol"
    REFERENCE
       "RFC 1940 [RFC1940] defines version 1 of the Source Demand
        Routing: Packet Format and Forwarding Specification"
    ::= {
     ip 42,
     ipip4 42,
     ipip 42
    }

idrp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Inter-Domain Routing Protocol"
    REFERENCE
       "RFC 1745 [RFC1745] defines BGP4/IDRP for IP."
    ::= {
     ip 45,
     ipip4 45,
     ipip 45
    }

rsvp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Resource Reservation Setup Protocol"
    REFERENCE
       "Resource ReSerVation Protocol (RSVP); Version 1 Functional
        Specification [RFC2205]."
    ::= {
     ip 46,
     ipip4 46,
     ipip 46
    }

gre PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "General Routing Encapsulation"
    REFERENCE
        "RFC 1701 [RFC1701] defines Generic Routing Encapsulation (GRE);
```

```
         RFC 1702 [RFC1702] defines Generic Routing Encapsulation over
         IPv4 networks"
      ::= {
       ip 47,
       ipip4 47,
       ipip 47
      }

nhrp PROTOCOL-IDENTIFIER
      PARAMETERS { }
      ATTRIBUTES { }
      DESCRIPTION
         "NBMA Next Hop Resolution Protocol (NHRP)"
      REFERENCE
         "RFC 2332 [RFC2332] defines the Next Hop Resolution Protocol."
      ::= {
       ip 54,
       ipip4 54,
       ipip 54
      }

priv-host PROTOCOL-IDENTIFIER
      PARAMETERS { }
      ATTRIBUTES { }
      DESCRIPTION
         "Pseudo-protocol reserved for any internal host protocol."
      REFERENCE
         "[RFC1700]"
      ::= {
       ip 61,
       ipip4 61,
       ipip 61
      }

priv-net PROTOCOL-IDENTIFIER
      PARAMETERS { }
      ATTRIBUTES { }
      DESCRIPTION
         "Pseudo-protocol reserved for any local network protocol."
      REFERENCE
         "[RFC1700]"
      ::= {
       ip 63,
       ipip4 63,
       ipip 63
      }

priv-distfile PROTOCOL-IDENTIFIER
```

```
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Pseudo-protocol reserved for any distributed file system."
    REFERENCE
        "[RFC1700]"
    ::= {
     ip 68,
     ipip4 68,
     ipip 68
    }

dgp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Dissimilar Gateway Protocol"
    REFERENCE
        "M/A-COM Government Systems, 'Dissimilar Gateway Protocol
        Specification, Draft Version', Contract no. CS901145, November
        16, 1987."
    ::= {
     ip 86,
     ipip4 86,
     ipip 86
    }

igrp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "IGRP; Cisco routing protocol"
    REFERENCE
        "Cisco Systems, Inc."
    ::= {
     ip 88,
     ipip4 88,
     ipip 88
    }

ospf PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Open Shortest Path First Interior GW Protocol (OSPFIGP)."
    REFERENCE
        "RFC 1583 [RFC1583] defines version 2 of the OSPF protocol."
    ::= {
```

```
       ip 89,
       ipip4 89,
       ipip 89
      }

mtp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Multicast Transport Protocol"
    REFERENCE
       "RFC 1301 [RFC1301] defines the Multicast Transport Protocol."
    ::= {
     ip 92,
     ipip4 92,
     ipip 92
    }

ax-25 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "AX.25 Frame Encapsulation"
    REFERENCE
       "RFC 1226 [RFC1226] defines Internet Protocol Encapsulation of
        AX.25 Frames."
    ::= {
     ip 93,
     ipip4 93,
     ipip 93
    }

ipip PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
    }
    DESCRIPTION
       "IP-within-IP Encapsulation Protocol"
    CHILDREN
       "Children of 'ipip' are selected and encoded in the same manner
        as children of IP."
    ADDRESS-FORMAT
       "The 'ipip' address format is the same as the IP address format."
    DECODING
       "Note: ether2.ip.ipip.udp is a different protocolDirID than
        ether2.ip.udp, as identified in the protocolDirTable.  As such,
```

```
     two different local protocol index values will be assigned by the
     agent. E.g. (full INDEX values shown):
      ether2.ip.ipip.udp =
          16.0.0.0.1.0.0.8.0.0.0.0.94.0.0.0.17.4.0.0.0.0
      ether2.ip.udp =
          12.0.0.0.1.0.0.8.0.0.0.0.17.3.0.0.0 "
    REFERENCE
       "RFC 2003 [RFC2003] defines IP Encapsulation within IP."
    ::= {
     ip 94,
     ipip4 94,
     ipip 94
     }

encap PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Encapsulation Header; A Scheme for an Internet Encapsulation
       Protocol: Version 1"
    REFERENCE
       "RFC 1241 [RFC1241] defines version 1 of the ENCAP Protocol."
    ::= {
     ip 98,
     ipip4 98,
     ipip 98
     }

priv-encript PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Pseudo-protocol reserved for any private encryption scheme."
    REFERENCE
       "[RFC1700]"
    ::= {
     ip 99,
     ipip4 99,
     ipip 99
     }

 -- ************************************************************
 --
 --                  Children of UDP and TCP
 --
 -- ************************************************************

tcpmux  PROTOCOL-IDENTIFIER
```

```
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "TCP Port Service Multiplexer Port."
    REFERENCE
        "RFC 1078 [RFC1078] defines the TCP Port Service Multiplexer
        Protocol."
    ::= { tcp 1 }

rje  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote Job Entry Protocol; RJE Logger Port; (historical)."
    REFERENCE
        "RFC 407 [RFC407] defines the Remote Job Entry Protocol."
    ::= { tcp 5 }

echo  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Echo Protocol for debugging  TCP and UDP transports."
    REFERENCE
        "RFC 862 [RFC862] defines the Echo Protocol."
    ::= {
       tcp 7,
       udp 7  }

discard  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Discard Protocol for debugging TCP and UDP transports."
    REFERENCE
        "RFC 863 [RFC863] defines the Discard Protocol."
    ::= {
       tcp 9,
       udp 9  }

systat  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Retrieve the Active Users list; a debugging tool for TCP and UDP
        transports."
    REFERENCE
        "RFC 866 [RFC866] defines the Active Users Protocol."
```

```
   ::= {
      tcp 11,
      udp 11  }

daytime  PROTOCOL-IDENTIFIER
   PARAMETERS { }
   ATTRIBUTES { }
   DESCRIPTION
      "Retrieve the current time of day; a debugging tool for TCP and
      UDP transports."
   REFERENCE
      "RFC 867 [RFC867] defines the Daytime Protocol."
   ::= {
      tcp 13,
      udp 13  }

qotd  PROTOCOL-IDENTIFIER
   PARAMETERS { }
   ATTRIBUTES { }
   DESCRIPTION
      "Quote of the Day Protocol; retrieve a short message (up to 512
      bytes); a debugging tool for TCP and UDP transports."
   REFERENCE
      "RFC 865 [RFC865] defines the Quote of the Day Protocol."
   ::= {
      tcp 17,
      udp 17  }

msp  PROTOCOL-IDENTIFIER
   PARAMETERS { }
   ATTRIBUTES { }
   DESCRIPTION
      "Message Send Protocol"
   REFERENCE
      "RFC 1312 [RFC1312] defines the Message Send Protocol."

   ::= {
      tcp 18,
      udp 18  }

chargen  PROTOCOL-IDENTIFIER
   PARAMETERS { }
   ATTRIBUTES { }
   DESCRIPTION
      "Character Generator Protocol; a debugging tool for TCP and UDP
      transports."
   REFERENCE
      "RFC 864 [RFC864] defines the Character Generator Protocol."
```

```
    ::= {
       tcp 19,
       udp 19   }

ftp-data PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "The File Transfer Protocol Data Port; the FTP Server process
       default data-connection port. "
    REFERENCE
       "RFC 959 [RFC959] defines the File Transfer Protocol.  Refer to
       section 3.2 of [RFC959] for details on FTP data connections."
    ::= { tcp 20 }

ftp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "The File Transfer Protocol Control Port; An FTP client initiates
       an FTP control connection by sending FTP commands from user port
       (U) to this port."
    REFERENCE
       "RFC 959 [RFC959] defines the File Transfer Protocol."
    ::= { tcp 21 }

telnet PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "The Telnet Protocol; The purpose of the TELNET Protocol is to
       provide a fairly general, bi-directional, eight-bit byte oriented
       communications facility.  Its primary goal is to allow a standard
       method of interfacing terminal devices and terminal-oriented
       processes to each other. "
    REFERENCE
       "RFC 854 [RFC854] defines the basic Telnet Protocol."
    ::= { tcp 23 }

priv-mail PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Pseudo-protocol reserved for any private mail system."
    REFERENCE
       "[RFC1700]"
    ::= { tcp 24,
       udp 24 }
```

smtp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "The Simple Mail Transfer Protocol; SMTP control and data
       messages are sent on this port."
    REFERENCE
       "RFC 821 [RFC821] defines the basic Simple Mail Transfer
       Protocol."
    ::= { tcp 25 }

priv-print PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Pseudo-protocol reserved for any private printer server."
    REFERENCE
       "[RFC1700]"
    ::= { tcp 35,
       udp 35 }

time PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Time Protocol"
    REFERENCE
       "RFC 868 [RFC868] defines the Time Protocol."
    ::= { tcp 37,
       udp 37 }

rap PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Route Access Protocol"
    REFERENCE
       "RFC 1476 [RFC1476] defines the Internet Route Access Protocol."
    ::= { tcp 38 }

rlp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Resource Location Protocol"
    REFERENCE
       "RFC 887 [RFC887] defines the Resource Location Protocol."
    ::= { udp 39 }

```
graphics PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Graphics Protocol"
    REFERENCE
       "RFC 493 [RFC493] defines the Graphics Protocol."
    ::= { tcp 41,
      udp 41  }

nameserver  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Host Name Server Protocol"
    REFERENCE
       "IEN 116 [IEN116] defines the Internet Name Server."
    ::= { udp 42 }

nicname  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "NICNAME/WHOIS Protocol"
    REFERENCE
       "RFC 954 [RFC954] defines the NICNAME/Who Is Protocol."
    ::= { tcp 43 }

mpm-flags  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "MPM FLAGS Protocol; (historical)."
    REFERENCE
       "RFC 759 [RFC759] defines the Message Processing Module."
    ::= { tcp 44 }

mpm  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Message Processing Module -- Receiver; (historical)."
    REFERENCE
       "RFC 759 [RFC759] defines the Message Processing Module."
    ::= { tcp 45 }

mpm-snd  PROTOCOL-IDENTIFIER
    PARAMETERS { }
```

```
    ATTRIBUTES { }
    DESCRIPTION
       "Message Processing Module -- Default Send; (historical)."
    REFERENCE
       "RFC 759 [RFC759] defines the Message Processing Module."
    ::= { tcp 46 }

tacacs  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Login Host Protocol (TACACS)"
    REFERENCE
       "An Access Control Protocol, Sometimes Called TACACS [RFC1492]."
    ::= { tcp 49 }

re-mail-ck  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Remote Mail Checking Protocol"
    REFERENCE
       "RFC 1339 [RFC1339] defines the Remote Mail Checking Protocol."
    ::= { udp 50 }

xns-time  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "XNS Time Protocol"
    REFERENCE
       "Xerox Corporation"
    ::= { tcp 52,
       udp 52 }

domain PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Domain Name Service Protocol; DNS may be transported by either
       UDP [RFC768] or TCP [RFC793].  If the transport is UDP, DNS
       requests restricted to 512 bytes in length may be sent to this
       port."
    REFERENCE
       "RFC 1035 [RFC1035] defines the Bootstrap Protocol."
    ::= { udp 53,
       tcp 53  }
```

```
xns-ch PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "XNS Clearinghouse"
    REFERENCE
        "Xerox Corporation"
    ::= { tcp 54,
        udp 54 }

xns-auth PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "XNS Authentication Protocol"
    REFERENCE
        "Xerox Corporation"
    ::= { tcp 56,
        udp 56 }

priv-term PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Pseudo-protocol reserved for any private terminal access
        protocol."
    REFERENCE
        "[RFC1700]"
    ::= { tcp 57,
        udp 57 }

xns-mail PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "XNS Mil Protocol"
    REFERENCE
        "Xerox Corporation"
    ::= { tcp 58,
        udp 58 }

priv-file PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Pseudo-protocol reserved for any private file service."
    REFERENCE
        "[RFC1700]"
```

```
    ::= { tcp 59,
       udp 59 }

tacacs-ds PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Default Server Port; TACACS Access Control Protocol Database
       Service."
    REFERENCE
       "RFC 1492 [RFC1492] defines the TACACS Protocol."
    ::= { tcp 65 }

sqlnet PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Oracle SQL*NET"
    REFERENCE
       "Oracle Corporation"
    ::= { tcp 66 }

bootps PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Bootstrap Protocol Server Protocol; BOOTP Clients send requests
       (usually broadcast) to the bootps port."
    REFERENCE
       "RFC 951 [RFC951] defines the Bootstrap Protocol."
    ::= { udp 67 }

bootpc PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Bootstrap Protocol Client Protocol; BOOTP Server replies are
       sent to the BOOTP Client using this destination port."
    REFERENCE
       "RFC 951 [RFC951] defines the Bootstrap Protocol."
    ::= { udp 68 }

tftp PROTOCOL-IDENTIFIER
    PARAMETERS {
     tracksSessions(1)
    }
    ATTRIBUTES { }
    DESCRIPTION
```

```
        "Trivial File Transfer Protocol; Only the first packet of each
        TFTP transaction will be sent to port 69. If the tracksSessions
        attribute is set, then packets for each TFTP transaction will be
        attributed to tftp, instead of the unregistered port numbers that
        will be encoded in subsequent packets."
    REFERENCE
        "RFC 1350 [RFC1350] defines the TFTP Protocol (revision 2);
         RFC 1782 [RFC1782] defines TFTP Option Extensions;
         RFC 1783 [RFC1783] defines the TFTP Blocksize Option;
         RFC 1784 [RFC1784] defines TFTP Timeout Interval and Transfer
         Size  Options."
    ::= { udp 69 }

gopher PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Internet Gopher Protocol"
    REFERENCE
        "RFC 1436 [RFC1436] defines the Gopher Protocol."
    ::= { tcp 70 }

netrjs-1 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote Job Service Protocol; (historical)."
    REFERENCE
        "RFC 740 [RFC740] defines the NETRJS Protocol."
    ::= { tcp 71 }

netrjs-2 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote Job Service Protocol; (historical)."
    REFERENCE
        "RFC 740 [RFC740] defines the NETRJS Protocol."
    ::= { tcp 72 }

netrjs-3 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote Job Service Protocol; (historical)."
    REFERENCE
        "RFC 740 [RFC740] defines the NETRJS Protocol."
    ::= { tcp 73 }
```

```
netrjs-4 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Remote Job Service Protocol; (historical)."
    REFERENCE
       "RFC 740 [RFC740] defines the NETRJS Protocol."
    ::= { tcp 74 }

priv-dialout PROTOCOL-IDENTIFIER

    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Pseudo-protocol reserved for any private dial out service."
    REFERENCE
       "[RFC1700]"
    ::= { tcp 75,
       udp 75 }

priv-rje PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Pseudo-protocol reserved for any private remote job entry
       service."
    REFERENCE
       "[RFC1700]"
    ::= { tcp 77,
       udp 77 }

finger PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Finger User Information Protocol"
    REFERENCE
       "RFC 1288 [RFC1288] defines the finger protocol."
    ::= { tcp 79 }

www-http PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Hypertext Transfer Protocol"
    REFERENCE
       "RFC 1945 [RFC1945] defines the Hypertext Transfer Protocol
(HTTP/1.0).
```

          RFC 2068 [RFC2068] defines the Hypertext Transfer Protocol
(HTTP/1.1).
          RFC 2069 [RFC2069] defines an Extension to HTTP: Digest Access
             Authentication.
          RFC 2109 [RFC2109] defines the HTTP State Management Mechanism.
          RFC 2145 [RFC2145] defines the use and interpretation of HTTP
             version numbers."
     ::= { tcp 80 }

priv-termlink PROTOCOL-IDENTIFIER

     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Pseudo-protocol reserved for any private terminal link
        protocol."
     REFERENCE
        "[RFC1700]"
     ::= { tcp 87,
        udp 87 }

kerberos PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "The Kerberos Network Authentication Service (V5)"
     REFERENCE
        "RFC 1510 [RFC1510] defines the Kerberos protocol."
     ::= { udp 88 }

supdup PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "SUPDUP Display; (historical)"
     REFERENCE
        "RFC 734 [RFC734] defines the SUPDUP Protocol."
     ::= { tcp 95 }

dixie PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DIXIE Directory Service"
     REFERENCE
        "RFC 1249 [RFC1249] defines the DIXIE Protocol."
     ::= { tcp 96,
        udp 96 }

```
hostname  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "NIC Internet Hostname Server Protocol; (historical)"
    REFERENCE
        "RFC 953 [RFC953] defines the Hostname Server Protocol."
    ::= { tcp 101 }

3com-tsmux PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "3COM-TSMUX"
    REFERENCE
        "3Com, Inc."
    ::= { tcp 106,
       udp 106 }

rtelnet  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote User Telnet Protocol; (historical)."
    REFERENCE
        "RFC 818 [RFC818] defines the Remote User Telnet Service."
    ::= { tcp 107 }

pop2 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Post Office Protocol -- Version 2. Clients establish connections
        with POP2 servers by using this destination port number.
        Historical."
    REFERENCE
        "RFC 937 [RFC937] defines Version 2 of the Post Office Protocol."
    ::= { tcp 109 }

pop3 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Post Office Protocol -- Version 3. Clients establish connections
        with POP3 servers by using this destination port number."
    REFERENCE
        "RFC 1725 [RFC1725] defines Version 3 of the Post Office
        Protocol."
```

```
     ::= { tcp 110,
        udp 110 }       -- RFC defines tcp use


sunrpc PROTOCOL-IDENTIFIER

    PARAMETERS {
        tracksSessions(1) -- learn port mapping of programs
    }
    ATTRIBUTES {
        hasChildren(0)   -- port mapper function numbers
    }
    DESCRIPTION
        "SUN Remote Procedure Call Protocol. Port mapper function
        requests are sent to this destination port."
    CHILDREN
        "Specific RPC functions are represented as children of the sunrpc
        protocol.  Each 'RPC function protocol' is identified by its
        function number assignment. RPC function number assignments are
        defined by different naming authorities, depending on the
        function identifier value.
        From [RFC1831]:

        Program numbers are given out in groups of hexadecimal 20000000
        (decimal 536870912) according to the following chart:

                   0 - 1fffffff   defined by rpc@sun.com
            20000000 - 3fffffff   defined by user
            40000000 - 5fffffff   transient
            60000000 - 7fffffff   reserved
            80000000 - 9fffffff   reserved
            a0000000 - bfffffff   reserved
            c0000000 - dfffffff   reserved
            e0000000 - ffffffff   reserved

        Children of 'sunrpc' are encoded as [ 0.0.0.111], the protocol
        identifier component for 'sunrpc', followed by [ a.b.c.d ], where
        a.b.c.d is the 32 bit binary RPC program number encoded in
        network byte order.  For example, a protocolDirID-fragment value
        of:
            0.0.0.111.0.1.134.163

        defines the NFS function (and protocol).

        Children are named as 'sunrpc' followed by the RPC function
        number in base 10 format. For example, NFS would be named:
            'sunrpc 100003'."
    DECODING
        "The first packet of many SUNRPC transactions is sent to the
```

        port- mapper program, and therefore decoded statically by
        monitoring RFC portmap requests [RFC1831]. Any subsequent packets
        must be decoded and correctly identified by 'remembering' the
        port assignments used in each RPC function call (as identified
        according to the procedures in the RPC Specification Version 2
        [RFC1831]).

        In some cases the port mapping for a particular protocol is well
        known and hard coded into the requesting client. In these cases
        the client will not send portmap requests; instead it will send
        the SUNRPC request directly to the well known port. These cases
        are rare and are being eliminated over time. NFS is the most
        significant SUNRPC program of this class. Such programs should
        still be declared as children of SUNRPC as described under
        CHILDREN above. How an implementation detects this behaviour and
        handles it is beyond the scope of this document.

        The 'tracksSessions(1)' PARAMETER bit is used to indicate whether
        the probe can (and should) monitor portmapper activity to
        correctly track SUNRPC connections."
    REFERENCE
        "RFC 1831 [RFC1831] defines the Remote Procedure Call Protocol
        Version 2. The authoritative list of RPC Functions is identified
        by the URL:
            ftp://ftp.isi.edu/in-notes/iana/assignments/sun-rpc-numbers"
    ::= { tcp 111,
        udp 111 }

auth PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Authentication Service; Identification Protocol."
    REFERENCE
        "RFC 1413 [RFC1413] defines the Identification Protocol."
    ::= { tcp 113 }

sftp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Simple File Transfer Protocol; (historical)."
    REFERENCE
        "RFC 913 [RFC913] defines the Simple File Transfer Protocol."
    ::= { tcp 115 }

uucp-path PROTOCOL-IDENTIFIER

```
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "UUCP Path Service"
    REFERENCE
       "RFC 915 [RFC915] defines the Network Mail Path Service."
    ::= { tcp 117 }

nntp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Network News Transfer Protocol"
    REFERENCE
       "RFC 977 [RFC977] defines the Network News Transfer Protocol."
    ::= { tcp 119 }

cfdptkt PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "CFDPTKT; Coherent File Distribution Protocol"
    REFERENCE
       "RFC 1235 [RFC1235] defines the Coherent File Distribution
       Protocol."
    ::= { udp 120 }

ntp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Network Time Protocol"
    REFERENCE
       "RFC 1305 [RFC1305] defines version 3 of the Network Time
       Protocol."
    ::= { udp 123 }

pwdgen  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Password Generator Protocol"
    REFERENCE
       "RFC 972 [RFC972] defines the Password Generator Protocol."
    ::= { tcp 129,
       udp 129  }

cisco-fna  PROTOCOL-IDENTIFIER
```

```
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "cisco FNATIVE"
    REFERENCE
       "Cisco Systems, Inc."
    ::= { tcp 130,
       udp 130 }

cisco-tna  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "cisco TNATIVE"
    REFERENCE
       "Cisco Systems, Inc."
    ::= { tcp 131,
       udp 131 }

cisco-sys  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "cisco SYSMAINT"
    REFERENCE
       "Cisco Systems, Inc."
    ::= { tcp 132,
       udp 132 }

statsrv  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Statistics Server; (historical)."
    REFERENCE
       "RFC 996 [RFC996] defines the Statistics Server Protocol."
    ::= { tcp 133,
       udp 133 }

 -- defined as nbt-name in IPX section
 -- netbios-ns      137/tcp    NETBIOS Name Service
 -- netbios-ns      137/udp    NETBIOS Name Service
 -- defined as nbt-data in IPX section
 -- netbios-dgm     138/tcp    NETBIOS Datagram Service
 -- netbios-dgm     138/udp    NETBIOS Datagram Service

 -- defined as nbt-session in IPX section
 -- netbios-ssn     139/tcp    NETBIOS Session Service
```

   -- netbios-ssn   139/udp   NETBIOS Session Service

imap2 PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Interactive Mail Access Protocol v2;
         Internet Message Access Protocol v4 (IMAP4) also uses this
        server port."
     REFERENCE
        "RFC 1064 [RFC1064] defines Version 2 of the Interactive Mail
        Access
         Protocol.
         RFC 1730 [RFC1730] defines Version 4 of the Internet Message
        Access
         Protocol."
     ::= { tcp 143 }

iso-tp0 PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "ISO-IP0; ISO-TP0 bridge between TCP and X.25"
     REFERENCE
        "RFC 1086 [RFC1086] defines the ISO-TP0 protocol."
     ::= { tcp 146,
        udp 146 }

iso-ip PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "ISO-IP; Use of the Internet as a Subnetwork for Experimentation
        with the OSI Network Layer"
     REFERENCE
        "RFC 1070 [RFC1070] defines the ISO-IP Protocol."
     ::= { tcp 147,
        udp 147 }

hems PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "HEMS; High Level Entity Management System; (historical)."
     REFERENCE
        "RFC 1021 [RFC1021] defines HEMS."
     ::= { tcp 151 }

```
bftp   PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Background File Transfer Program"
    REFERENCE
        "RFC 1068 [RFC1068] defines the Background File Transfer
        Program."
    ::= { tcp 152 }

sgmp   PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Simple Gateway Monitoring Protocol; (historical)."
    REFERENCE
        "RFC 1028 [RFC1028] defines the Simple Gateway Monitoring
        Protocol."
    ::= { udp 153 }

pcmail-srv   PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "PCMail Server; Distributed Mail System Protocol (DMSP)"
    REFERENCE
        "RFC 1056 [RFC1056] defines the PCMAIL Protocol."
    ::= { tcp 158 }

sgmp-traps   PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Simple Gateway Monitoring Protocol Traps; (historical)."
    REFERENCE
        "RFC 1028 [RFC1028] defines the Simple Gateway Monitoring

        Protocol."
    ::= { udp 160 }

 -- snmp and snmptrap found in the Protocol-Independent section
 -- snmp            161/udp    SNMP
 -- snmptrap        162/udp    SNMPTRAP

cmip-man   PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
```

```
        "CMIP/TCP (CMOT) Manager; (historical)."
    REFERENCE
        "RFC 1095 [RFC1095] defines the Common Management Information
        Services and Protocol over TCP/IP."
    ::= { tcp 163,
        udp 163 }

cmip-agent  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "CMIP/TCP (CMOT) Agent; (historical)."
    REFERENCE
        "RFC 1095 [RFC1095] defines the Common Management Information
        Services and Protocol over TCP/IP."
    ::= { tcp 164,
        udp 164 }

xdmcp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "X Display Manager Control Protocol"
    REFERENCE
        "X11 Consortium"
    ::= { udp 177 }

bgp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Border Gateway Protocol"
    REFERENCE
        "RFC 1267 [RFC1267] defines version 3 of the Border Gateway

        Protocol."
    ::= { tcp 179 }

remote-kis  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote-Knowbot Information Service (KIS)"
    REFERENCE
        "RFC 1739 [RFC1739] describes the KNOWBOT Protocol."
    ::= { tcp 185,
        udp 185 }
```

```
kis  PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Knowbot Information Service (KIS)"
     REFERENCE
        "RFC 1739 [RFC1739] describes the KNOWBOT Protocol."
     ::= { tcp 186,
        udp 186 }

irc  PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Internet Relay Chat Protocol"
     REFERENCE
        "RFC 1459 [RFC1459] defines the Internet Relay Chat Protocol."
     ::= { tcp 194,
        udp 194 }

smux PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "SMUX; SNMP MUX Protocol and MIB; (historical)."
     REFERENCE
        "RFC 1227 [RFC1227] defines the SMUX Protocol."
     ::= { tcp 199 }

 --
 -- AppleTalk applications are defined in the AppleTalk Stack section
 --
 -- at-rtmp          201/tcp    AppleTalk Routing Maintenance
 -- at-rtmp          201/udp    AppleTalk Routing Maintenance
 -- at-nbp           202/tcp    AppleTalk Name Binding
 -- at-nbp           202/udp    AppleTalk Name Binding
 -- at-3             203/tcp    AppleTalk Unused
 -- at-3             203/udp    AppleTalk Unused
 -- at-echo          204/tcp    AppleTalk Echo
 -- at-echo          204/udp    AppleTalk Echo
 -- at-5             205/tcp    AppleTalk Unused
 -- at-5             205/udp    AppleTalk Unused
 -- at-zis           206/tcp    AppleTalk Zone Information
 -- at-zis           206/udp    AppleTalk Zone Information
 -- at-7             207/tcp    AppleTalk Unused
 -- at-7             207/udp    AppleTalk Unused
 -- at-8             208/tcp    AppleTalk Unused
 -- at-8             208/udp    AppleTalk Unused
```

```
z39-50 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "ANSI Z39.50"
    REFERENCE
        "RFC 1729 [RFC1729] describes the Z39.50 Protocol."
    ::= { tcp 210 }

ipx-tunnel  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Tunneling IPX Traffic through IP Networks"
    REFERENCE
        "RFC 1234 [RFC1234] defines the IPX Tunnel Protocol."
    ::= { udp 213 }

mpp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Netix Message Posting Protocol"
    REFERENCE
        "RFC 1204 [RFC1204] defines the Message Posting Protocol."
    ::= { tcp 218 }

imap3  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Interactive Mail Access Protocol v3; (historical)."
    REFERENCE
        "RFC 1203 [RFC1203] defines version 3 of the Interactive Mail
        Access Protocol."
    ::= { tcp 220 }

ldap  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Lightweight Directory Access Protocol"
    REFERENCE
        "RFC 1777 [RFC1777] defines Lightweight Directory Access
        Protocol; RFC 1798 [RFC1798] defines Connection-less Lightweight
        X.500 Directory Access Protocol"
    ::= { tcp 389,      -- RFC 1777
        udp 389  }      -- RFC 1798
```

```
mobileip-agent  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "IP Mobility Support"
    REFERENCE
       "RFC 2002 [RFC2002] defines the IP Mobility Support protocol."
    ::= { udp 434 }

https  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Secure HTTP; HTTP over TLS/SSL"
    REFERENCE
       "Netscape; http://home.netscape.com/eng/ssl3/"
    ::= { tcp 443 }

smtps  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "SMTP protocol over TLS/SSL"
    REFERENCE
       "Netscape; http://home.netscape.com/eng/ssl3/"
    ::= { tcp 465 }

isakmp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Internet Security Association and Key Management Protocol
       (ISAKMP)"
    REFERENCE
       "RFC 2408 [RFC2408]"
    ::= { udp 500 }

login  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "BSD Rlogin; remote login a la telnet"
    REFERENCE
       "RFC 1282 [RFC1282] defines the BSD Rlogin Protocol."
    ::= { tcp 513 }

syslog  PROTOCOL-IDENTIFIER
    PARAMETERS { }
```

```
    ATTRIBUTES { }
    DESCRIPTION
        "syslog"
    REFERENCE
        "[RFC1700]"
    ::= { udp 514 }

uucp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Unix-to-Unix copy protocol"
    REFERENCE
        "[RFC1700]"
    ::= { tcp 540 }

doom  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "DOOM Game;"
    REFERENCE
        " Id Software"
    ::= { tcp 666 }

radius  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Remote Authentication Dial In User Service (RADIUS)"
    REFERENCE
        "RFC 2138 [RFC2138] defines the Radius protocol."
    ::= { udp 1812 }

radiusacct  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "RADIUS Accounting Protocol"
    REFERENCE
        "RFC 2139 [RFC2139] defines the Radius Accounting protocol."
    ::= { udp 1813 }

 --
 -- Portmapper Functions; Children of sunrpc
 --

portmapper PROTOCOL-IDENTIFIER
```

```
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "SUNRPC PORTMAPPER program.  This is the SUNRPC program which is
        used to locate the UDP/TCP ports on which other SUNRPC programs
        can be found."
     REFERENCE
        "Appendix A of RFC 1057 [RFC1057] describes the portmapper
        operation."
     ::= { sunrpc 100000 }

nfs  PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "Sun Network File System (NFS);"
     DECODING
        "NFS is a SUNRPC program which may or may not use the port mapper
        SUNRPC program to connect clients and servers.  In many cases the
        NFS server program runs over UDP/TCP port 2049, but an
        implementation is encouraged to perform further analysis before
        assuming that a packet to/from this port is a SUNRPC/NFS packet.
        Likewise an implementation is encouraged to track port mapper
        activity to spot cases where it is used to locate the SUNRPC/NFS
        program as this is more robust."
     REFERENCE
        "The NFS Version 3 Protocol Specification is defined in RFC 1813
        [RFC1813]."
     ::= {
      sunrpc 100003        --  [0.1.134.163]
     }

xwin PROTOCOL-IDENTIFIER
     PARAMETERS {
         tracksSessions(1)
     }
     ATTRIBUTES { }
     DESCRIPTION
        "X Windows Protocol"
     DECODING
        "The X Windows Protocol when run over UDP/TCP normally runs over
        the well known port 6000.  It can run over any port in the range
        6000 to 6063, however.  If the tracksSessions(1) parameter bit is
        set the agent can and should detect such X Window sessions and
        report them as the X protocol."
     REFERENCE
         "The X Windows Protocol is defined by TBD"
     ::= {
```

```
      tcp 6000,
      udp 6000
      -- lat ?
   }
```

3.1.2. Novell IPX Stack

```
ipx PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
      hasChildren(0),
      addressRecognitionCapable(1)
    }

    DESCRIPTION
       "Novell IPX"
    CHILDREN
       "Children of IPX are defined by the 8 bit packet type field.  The
       value is encoded into an octet string as [ 0.0.0.a ], where 'a'
       is the single octet of the packet type field.

       Notice that in many implementations of IPX usage of the packet
       type field is inconsistent with the specification and
       implementations are encouraged to use other techniques to map
       inconsistent values to the correct value (which in these cases is
       typically the Packet Exchange Protocol).  It is beyond the scope
       of this document to describe these techniques in more detail.

       Children of IPX are encoded as [ 0.0.0.a ], and named as 'ipx a'
       where a is the packet type value.  The novell echo protocol is
       referred to as 'ipx nov-echo' OR 'ipx 2'."
    ADDRESS-FORMAT
       "4 bytes of Network number followed by the 6 bytes Host address
       each in network byte order."
    REFERENCE
       "The IPX protocol is defined by the Novell Corporation

       A complete description of IPX may be secured at the following
       address:
             Novell, Inc.
             122 East 1700 South
             P. O. Box 5900
             Provo, Utah 84601 USA
             800 526 5463
             Novell Part # 883-000780-001"
    ::= {
       ether2       0x8137,       -- [0.0.129.55]
          snap       0x8137,       -- [0.0.129.55]
```

```
        ianaAssigned 1,              -- [0.0.0.1]   (ipxOverRaw8023)
      llc           224,        -- [0.0.0.224]
         802-1Q       0x8137,        -- [0.0.129.55]
       802-1Q        0x020000e0,   -- 1Q-LLC [2.0.0.224]
         802-1Q        0x05000001   -- 1Q-IANA [5.0.0.1]
                                    -- (ipxOverRaw8023)
    }

nov-rip PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell Routing Information Protocol"
    REFERENCE
       "Novell Corporation"
    ::= {
       ipx 0x01,        -- when reached by IPX packet type
       nov-pep 0x0453  -- when reached by IPX socket number
    }

nov-echo PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell Echo Protocol"
    REFERENCE
       "Novell Corporation"
    ::= { ipx 0x02 }

nov-error PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell Error-handler Protocol"
    REFERENCE
       "Novell Corporation"
    ::= { ipx 0x03 }

nov-pep PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
       "Novell Packet Exchange Protocol.  This is really a null protocol
       layer as all IPX packets contain the relevant fields for this
       protocol.  This protocol is defined so that socket-based decoding
       has a point of attachment in the decode tree while still allowing
```

```
           packet type based decoding also."
        CHILDREN
           "Children of PEP are defined by the 16 bit socket values.  The
           value is encoded into an octet string as [ 0.0.a.b ], where 'a'
           and 'b' are the network byte order encodings of the MSB and LSB
           of the socket value.

           Each IPX/PEP packet contains two sockets, source and destination.
           How these are mapped onto the single well-known socket value used
           to identify its children is beyond the scope of this document."
        REFERENCE
           "Novell Corporation"
        ::= {
         -- ipx 0x00      ** Many third party IPX's use this value always
         ipx 0x04         -- Xerox assigned for PEP
         -- ipx 0x11      ** Novell use this for PEP packets, often
        }

nov-spx PROTOCOL-IDENTIFIER
        PARAMETERS { }
        ATTRIBUTES {
        hasChildren(0)
  }
        DESCRIPTION
           "Novell Sequenced Packet Exchange Protocol.  This protocol is an
           extension of IPX/PEP as it shares a common header."
        CHILDREN
           "Children of SPX are defined by the 16 bit socket values.  The
           value is encoded into an octet string as [ 0.0.a.b ], where 'a'
           and 'b' are the network byte order encodings of the MSB and LSB
           of the socket value.

           Each IPX/SPX packet contains two sockets, source and destination.
           How these are mapped onto the single well-known socket value used
           to identify its children is beyond the scope of this document."
        REFERENCE
           "Novell Corporation"
        ::= {
         ipx 0x05 -- Xerox assigned for SPX
        }

nov-sap PROTOCOL-IDENTIFIER
        PARAMETERS {
         tracksSessions(1)
        }
        ATTRIBUTES {
         hasChildren(0)
        }
```

         DESCRIPTION
             "Novell Service Advertising Protocol. This protocol binds
             applications on a particular host to an IPX/PEP or IPX/SPX socket
             number. Although it never truly acts as a transport protocol
             itself it is used to establish sessions between clients and
             servers and barring well-known sockets is the only reliable way
             to determine the protocol running over a given socket on a given
             machine."
         CHILDREN
             "Children of SAP are identified by a 16 bit service type. They
             are encoded as [0.0.a.b], where 'a' is the MSB and 'b' is the
             LSB of the service type.

             Children of SAP are named as 'nov-sap a' where 'a' is the service
             type in hexadecimal notation. The novell NCP protocol is
             referred to as 'nov-sap ncp' OR 'nov-sap 0x0004'."
         DECODING
             "The first packet of any session for a SAP based application
             (almost all IPX/PEP and IPX/SPX based applications utilize SAP)
             is sent to the SAP server(s) to map the service type into a port
             number for the host(s) on which the SAP server(s) is(are)
             running. These initial packets are SAP packets and not
             application packets and must be decoded accordingly.

             Having established the mapping, clients will then send
             application packets to the newly discovered socket number. These
             must be decoded by 'remembering' the socket assignments
             transmitted in the SAP packets.

             In some cases the port mapping for a particular protocol is well
             known and SAP will always return the same socket number for that
             application.

             Such programs should still be declared as children of nov-sap as
             described under CHILDREN above. How an implementation detects a
             client which is bypassing the SAP server to contact a well-known
             application is beyond the scope of this document.

             The 'tracksSessions(1)' PARAMETER bit is used to indicate whether
             the probe can (and should) monitor nov-sap activity to correctly
             track SAP-based connections."
         REFERENCE
             "A list of SAP service types can be found at
                 ftp://ftp.isi.edu/in-notes/iana/assignments/novell-sap-
             numbers"
         ::= { nov-pep 0x0452 }

ncp PROTOCOL-IDENTIFIER

```
    PARAMETERS {
     tracksSessions(1)
    }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
        "Netware Core Protocol"
    CHILDREN
        "Children of NCP are identified by the 8 bit command type field.
        They are encoded as [ 0.0.0.a ] where 'a' is the command type
        value.

        Children of NCP are named as 'ncp a' where 'a' is the command
        type in decimal notation.  The NDS sub-protocol is referred to as
        'ncp nds' OR 'ncp 104'."
    DECODING
        "Only the NCP request frames carry the command type field.  How
        the implementation infers the command type of a response frame is
        an implementation specific matter and beyond the scope of this
        document.

        The tracksSessions(1) PARAMETERS bit indicates whether the probe
        can (and should) perform command type inference."
    REFERENCE
        "Novell Corporation"
    ::= { nov-sap 0x0004,
        nov-pep 0x0451 }

nds PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "The Netware Directory Services sub-protocol."
    REFERENCE
        "Novell Corporation"
    ::= { ncp 104 }

nov-diag PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Novell's diagnostic Protocol"
    REFERENCE
        "Novell Corporation"
    ::= {
     nov-sap 0x0017,    -- [ed., this is the right one]
     nov-pep 0x0456
```

```
     }

nov-sec PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell security - serialization - copy protection protocol."
    REFERENCE
       "Novell Corporation"
    ::= { nov-pep 0x0457 }

nov-watchdog PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell watchdog protocol."
    REFERENCE
       "Novell Corporation"
    ::= { nov-pep 0x4004 }

nov-bcast PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Novell broadcast protocol."
    REFERENCE
       "Novell Corporation"
    ::= { nov-pep 0x4005 }

3.1.3.  The XEROX Protocol Stack

idp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
      hasChildren(0),
      addressRecognitionCapable(1)
    }
    DESCRIPTION
       "Xerox IDP"
    CHILDREN
       "Children of IDP are defined by the 8 bit value of the Packet
       type field.  The value is encoded into an octet string as [
       0.0.0.a ], where 'a' is the value of the packet type field in
       network byte order.

       Children of IDP are encoded as [ 0.0.0.a ], and named as 'idp a'
       where a is the packet type value.  The XNS SPP protocol is
       referred to as 'idp xns-spp' OR 'idp 2'."
```

```
    ADDRESS-FORMAT
        "4 bytes of Network number followed by the 6 bytes Host address
        each in network byte order."
    REFERENCE
        "Xerox Corporation, Document XNSS 028112, 1981"
    ::=  {
        ether2  0x600,      -- [ 0.0.6.0 ]
        snap    0x600,
        802-1Q  0x600       -- [ 0.0.6.0 ]
    }

xns-rip PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Routing Information Protocol."
    REFERENCE
        "Xerox Corporation"
    ::= { idp 1 }

xns-echo PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "XNS echo protocol."
    REFERENCE
        "Xerox Corporation"
    ::= { idp 2 }

xns-error PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "XNS error-handler protocol."
    REFERENCE
        "Xerox Corporation"
    ::= { idp 3 }

xns-pep PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)

    }
    DESCRIPTION
        "XNS Packet Exchange Protocol."
    CHILDREN
        "Children of PEP are defined by the 16 bit socket values.  The
```

         value is encoded into an octet string as [0.0.a.b], where 'a'
         and 'b' are the network byte order encodings of the MSB and LSB
         of the socket value.

         Each XNS/PEP packet contains two sockets, source and destination.
         How these are mapped onto the single well-known socket value used
         to identify its children is beyond the scope of this document."
    REFERENCE
         "Xerox Corporation"
    ::= { idp 4 }

xns-spp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
         "Sequenced Packet Protocol."
    CHILDREN
         "Children of SPP are defined by the 16 bit socket values. The
         value is encoded into an octet string as [0.0.a.b], where 'a'
         and 'b' are the network byte order encodings of the MSB and LSB
         of the socket value.

         Each XNS/SPP packet contains two sockets, source and destination.
         How these are mapped onto the single well-known socket value used
         to identify its children is beyond the scope of this document."
    REFERENCE
         "Xerox Corporation"
    ::= { idp 5 }

3.1.4. AppleTalk Protocol Stack

apple-oui PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
         "Pseudo-protocol which binds Apple's protocols to vsnap."
    CHILDREN
         "Children of apple-oui are identified by the ether2 type field
         value that the child uses when encapsulated in ether2. The value
         is encoded into an octet string as [0.0.a.b], where 'a' and 'b'
         are the MSB and LSB of the 16-bit ether type value in network
         byte order."
    REFERENCE
         "AppleTalk Phase 2 Protocol Specification, document ADPA
          #C0144LL/A."
    ::=   {

```
     vsnap     0x080007,        -- [ 0.8.0.7 ]
     802-1Q    0x04080007       -- 1Q-VSNAP [ 4.8.0.7 ]
     }

aarp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "AppleTalk Address Resolution Protocol."
    REFERENCE
       "AppleTalk Phase 2 Protocol Specification, document ADPA
        #C0144LL/A."
    ::=    {
     ether2    0x80f3,                -- [ 0.0.128.243 ]
     snap      0x80f3,
     apple-oui 0x80f3,
     802-1Q    0x80f3                 -- [ 0.0.128.243 ]
     }

atalk PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
     }
    DESCRIPTION
       "AppleTalk Protocol."
    CHILDREN
       "Children of ATALK are defined by the 8 bit value of the DDP type
        field.  The value is encoded into an octet string as [ 0.0.0.a ],
        where 'a' is the value of the DDP type field in network byte
        order."
    ADDRESS-FORMAT
       "2 bytes of Network number followed by 1 byte of node id each in
        network byte order."
    REFERENCE
       "AppleTalk Phase 2 Protocol Specification, document ADPA
        #C0144LL/A."
    ::=    {
     ether2     0x809b,   -- [ 0.0.128.155 ]
        apple-oui  0x809b,
     802-1Q     0x809b    -- [ 0.0.128.155 ]
     }

rtmp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
```

```
        "AppleTalk Routing Table Maintenance Protocol."
    REFERENCE
        "Apple Computer"
    ::= {
     atalk   0x01,        -- responses
     atalk   0x05         -- requests
    }

aep PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "AppleTalk Echo Protocol."
    REFERENCE
        "Apple Computer"
    ::= { atalk 0x04 }

nbp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "AppleTalk Name Binding Protocol."
    DECODING
        "In order to correctly identify the application protocol running
        over atp NBP packets must be analyzed.  The mechanism by which
        this is achieved is beyond the scope of this document."
    REFERENCE
        "Apple Computer"
    ::= { atalk 0x02 }

zip PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "AppleTalk Zone Information Protocol."
    REFERENCE
        "Apple Computer"
    ::= {
     atalk   0x06,
     atp     3
    }

atp PROTOCOL-IDENTIFIER
    PARAMETERS {
     tracksSessions(1)
    }
    ATTRIBUTES {
     hasChildren(0)
```

```
        }
     DESCRIPTION
        "AppleTalk Transaction Protocol."
     CHILDREN
        "Children of atp are identified by the following (32 bit)
        enumeration:
           1    asp (AppleTalk Session Protocol)
           2    pap (Printer Access Protocol)
           3    zip (Zone Information Protocol)
        Children of atp are encoded as [ a.b.c.d ] where 'a', 'b', 'c'
        and 'd' are the four octets of the enumerated value in network
        order (i.e. 'a' is the MSB and 'd' is the LSB).

        The ZIP protocol is referred to as 'atp zip' OR 'atp 3'."
     DECODING
        "An implementation is encouraged to examine both the socket
        fields in the associated DDP header as well as the contents of
        prior NBP packets in order to determine which (if any) child is
        present.  A full description of this algorithm is beyond the
        scope of this document.  The tracksSessions(1) PARAMETER
        indicates whether the probe can (and should) perform this
        analysis."
     REFERENCE
        "Apple Computer"
     ::= { atalk 0x03 }

adsp PROTOCOL-IDENTIFIER
     PARAMETERS {
      tracksSessions(1)
     }
     ATTRIBUTES {
      hasChildren(0)
     }
     DESCRIPTION
        "AppleTalk Data Stream Protocol."
     CHILDREN
        "Children of adsp are identified by enumeration.  At this time
        none are known."
     DECODING
        "An implementation is encouraged to examine the socket numbers in
        the associated DDP header as well as the contents of prior NBP
        packets in order to determine which (if any) child of ADSP is
        present.

        The mechanism by which this is achieved is beyond the scope of
        this document.

        The tracksSessions(1) PARAMETER indicates whether the probe can
```

          (and should) perform this analysis."
     REFERENCE
        "Apple Computer"
     ::= { atalk 0x07 }

asp PROTOCOL-IDENTIFIER
 PARAMETERS { }
     ATTRIBUTES {
  hasChildren(0)
 }
     DESCRIPTION
        "AppleTalk Session Protocol."
     CHILDREN
        "Children of asp are identified by the following (32 bit)
        enumeration:
           1   afp (AppleTalk Filing Protocol)
        Children of asp are encoded as [a.b.c.d] where 'a', 'b', 'c'
        and 'd' are the four octets of the enumerated value in network
        order (i.e. 'a' is the MSB and 'd' is the LSB).

        The AFP protocol is referred to as 'asp afp' OR 'asp 1'."
     DECODING
        "ASP is a helper layer to assist in building client/server
        protocols. It cooperates with ATP to achieve this; the
        mechanisms used when decoding ATP apply equally here (i.e.
        checking DDP socket numbers and tracking NBP packets).

        Hence the tracksSessions(1) PARAMETER of atp applies to this
        protocol also."
     REFERENCE
        "Apple Computer"
     ::= { atp 1 }

afp PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
         "AppleTalk Filing Protocol."
     REFERENCE
        "Apple Computer"
     ::= { asp 1 }

pap PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "AppleTalk Printer Access Protocol."
     REFERENCE

```
        "Apple Computer"
    ::= { atp 2 }


3.1.5.  Banyon Vines Protocol Stack

vtr PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
        "Banyan Vines Token Ring Protocol Header."
    CHILDREN
        "Children of vines-tr are identified by the 8 bit packet type
        field.  Children are encoded as [ 0.0.0.a ] where 'a' is the
        packet type value.

        The vines-ip protocol is referred to as 'vines-tr vip' OR 'vines-
        tr 0xba'."
    REFERENCE
        "See vip."
    ::= {
     llc           0xBC,       -- declared as any LLC, but really TR only.
     802-1Q        0x020000BC  -- 1Q-LLC [2.0.0.188]
    }

vecho PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Banyan Vines data link level echo protocol."
    REFERENCE
        "See vip."
    ::= {
     ether2        0x0BAF,       -- [0.0.11.175]
     snap          0x0BAF,
     -- vfrp        0x0BAF,
     vtr           0xBB,         -- [ed. yuck!]
     802-1Q        0x0BAF        -- [0.0.11.175]
     }

vip PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
    }
    DESCRIPTION
```

```
        "Banyan Vines Internet Protocol."
    CHILDREN
        "Children of vip are selected by the one-byte 'protocol type'
        field located at offset 5 in the vip header.  The value is
        encoded as [ 0.0.0.a ], where a is the 'protocol type.'  For
        example, a protocolDirId fragment of:

            0.0.0.1.0.0.11.173.0.0.0.1

          identifies an encapsulation of vipc (ether2.vip.vipc)."
    ADDRESS-FORMAT
        "vip packets have 6-byte source and destination addresses.  The
        destination address is located at offset 6 in the vip header, and
        the source address at offset 12.  These are encoded in network
        byte order."
    REFERENCE
        "Vines Protocol Definition - part# 092093-001, order# 003673

          BANYAN,
          120 Flanders Road,
          Westboro, MA 01581 USA"
    ::= {
     ether2  0x0BAD,
     snap    0x0BAD,
     -- vfrp 0x0BAD,
     vtr     0xBA,          -- [ed. yuck!]
     802-1Q  0x0BAD         -- [0.0.11.173]
    }

varp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Banyan Vines Address Resolution Protocol."
    REFERENCE
        "BANYAN"
    ::= { vip 0x04 }

vipc PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
        "Banyan Vines Interprocess Communications Protocol."
    CHILDREN
        "Children of Vines IPC are identified by the packet type field at
        offset 4 in the vipc header.
```

        These are encoded as [0.0.0.a] where 'a' is the packet type
        value. Children of vipc are defined as 'vipc a' where 'a' is the
        packet type value in hexadecimal notation.

        The Vines Reliable Data Transport protocol is referred to as
        'vipc vipc-rdp' OR 'vipc 0x01'."
    DECODING
        "Children of vipc are deemed to start at the first byte after the
        packet type field (i.e. at offset 5 in the vipc header)."
    REFERENCE
        "BANYAN"
    ::= { vip 0x01 }

 -- Banyan treats vipc, vipc-dgp and vipc-rdp as one protocol, IPC.
 -- Vines IPC really comes in two flavours. The first is used to
 -- send unreliable datagrams (vipc packet type 0x00). The second
 -- used to send reliable datagrams (vipc packet type 0x01),
 -- consisting of up to four actual packets.
 -- In order to distinguish between these we need two 'virtual'
 -- protocols to identify which is which.

vipc-dgp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
     }
    DESCRIPTION
        "Vines Unreliable Datagram Protocol."
    CHILDREN
        "Children of vipc-dgp are identified by the 16 bit port numbers
        contained in the vipc (this protocol's parent protocol) header.

        These are encoded as [0.0.a.b] where 'a' is the MSB and 'b' is
        the MSB of the port number in network byte order.

        Children of vipc-dgp are defined as 'vipc-dgp a' where 'a' is the
        port number in hexadecimal notation.

        The StreetTalk protocol running over vipc-dgp would be referred
        to as 'vipc-dgp streettalk' OR 'vipc-dgp 0x000F'.

        The mechanism by which an implementation selects which of the
        source and destination ports to use in determining which child
        protocol is present is implementation specific and beyond the
        scope of this document."
    DECODING
        "Children of vipc-dgp are deemed to start after the single
        padding byte found in the vipc header. In the case of vipc-dgp

          the vipc header is a so called 'short' header, total length 6
          bytes (including the final padding byte)."
     REFERENCE
          "BANYAN"
     ::= { vipc 0x00 }

vipc-rdp PROTOCOL-IDENTIFIER
     PARAMETERS {
      countsFragments(0)
     }
     ATTRIBUTES {
      hasChildren(0)
     }
     DESCRIPTION
          "Vines Reliable Datagram Protocol."
     CHILDREN
          "Children of vipc-rdp are identified by the 16 bit port numbers
          contained in the vipc (this protocol's parent protocol) header.

          These are encoded as [0.0.a.b] where 'a' is the MSB and 'b' is
          the MSB of the port number in network byte order.

          Children of vipc-dgp are defined as 'vipc-rdp a' where 'a' is the
          port number in hexadecimal notation.

          The StreetTalk protocol running over vipc-rdp would be referred
          to as 'vipc-rdp streettalk' OR 'vipc-rdp 0x000F'.

          The mechanism by which an implementation selects which of the
          source and destination ports to use in determining which child
          protocol is present is implementation specific and beyond the
          scope of this document."
     DECODING
          "Children of vipc-rdp are deemed to start after the error/length
          field at the end of the vipc header. For vipc-rdp the vipc
          header is a so called 'long' header, total 16 bytes (including
          the final error/length field).

          vipc-rdp includes a high level fragmentation scheme which allows
          up to four vipc packets to be sent as a single atomic PDU. The
          countsFragments(0) PARAMETERS bit indicates whether the probe can
          (and should) identify the child protocol in all fragments or only
          the leading one."
     REFERENCE
          "BANYAN"
     ::= { vipc 0x01 }

vspp PROTOCOL-IDENTIFIER

```
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
        "Banyan Vines Sequenced Packet Protocol."
    CHILDREN
        "Children of vspp are identified by the 16 bit port numbers
        contained in the vspp header.

        These are encoded as [ 0.0.a.b ] where 'a' is the MSB and 'b' is
        the MSB of the port number in network byte order.

        Children of vspp are defined as 'vspp a' where 'a' is the port
        number in hexadecimal notation.

        The StreetTalk protocol running over vspp would be referred to as
        'vspp streettalk' OR 'vspp 0x000F'.

        The mechanism by which an implementation selects which of the
        source and destination ports to use in determining which child
        protocol is present is implementation specific and beyond the
        scope of this document."
    DECODING
        "The implementation must ensure only those vspp packets which
        contain application data are decoded and passed on to children.
        Although it is suggested that the packet type and control fields
        should be used to determine this fact it is beyond the scope of
        this document to fully define the algorithm used."
    REFERENCE
        "BANYAN"
    ::= { vip 0x02 }

vrtp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Banyan Vines Routing Update Protocol."
    REFERENCE
        "BANYAN"
    ::= { vip 0x05 }

vicp PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Banyan Vines Internet Control Protocol."
    REFERENCE
```

```
            "BANYAN"
     ::= { vip 0x06 }


3.1.6.  The DECNet Protocol Stack

dec PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC"
     REFERENCE
        "Digital Corporation"
     ::= {
      ether2 0x6000,
      802-1Q 0x6000      -- [0.0.96.0]
     }

lat PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Local Area Transport Protocol."
     REFERENCE
        "Digital Corporation"
     ::= {
      ether2 0x6004,
      802-1Q 0x6004      -- [0.0.96.4]
     }

mop PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Maintenance Operations Protocol."
     REFERENCE
        "Digital Corporation"
     ::= {
      ether2 0x6001,    -- mop dump/load
      ether2 0x6002,    -- mop remote console
      802-1Q 0x6001,    -- [0.0.96.1]  VLAN + mop dump/load
      802-1Q 0x6002     -- [0.0.96.2]  VLAN + mop remote console
     }

dec-diag PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Diagnostic Protocol."
```

```
    REFERENCE
        "Digital Corporation"
    ::= {
     ether2 0x6005,
     802-1Q 0x6005      -- [0.0.96.5]
    }

lavc PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "DEC Local Area VAX Cluster Protocol."
    REFERENCE
        "Digital Corporation"
    ::= {
     ether2 0x6007,
     802-1Q 0x6007          -- [0.0.96.7]
    }

drp PROTOCOL-IDENTIFIER
    PARAMETERS {
     countsFragments(1)
    }
    ATTRIBUTES {
     hasChildren(0),
     addressRecognitionCapable(1)
    }
    DESCRIPTION
        "DEC Routing Protocol."

    CHILDREN
        "There is only one child of DRP, NSP.  This is encoded as [
        0.0.0.1 ]."
    ADDRESS-FORMAT
        "There are three address formats used in DRP packets, 2-byte
        (short data packet and all control except ethernet endnode &
        router hello messages), 6-byte (ethernet router & endnode hello
        messages) and 8-byte (long data packet).  All of these contain
        the 2-byte format address in the last 2 bytes with the remaining
        bytes being unimportant for the purposes of system
        identification.  It is beyond the scope of this document to
        define the algorithms used to identify packet types and hence
        address formats.

        The 2-byte address format is the concatenation of a 6-bit area
        and a 10-bit node number.  In all cases this is placed in little
        endian format (i.e. LSB, MSB).  The probe, however, will return
        them in network order (MSB, LSB).  Regardless of the address
```

        format in the packet, the probe will always use the 2-byte
        format.

        For example area=13 (001101) and node=311 (0100110111) gives:
          0011 0101 0011 0111 = 0x3537 in network order (the order the
          probe should return the address in).

          In packets this same value would appear as (hex):

            2-byte 37 35
            6-byte AA 00 04 00 37 35
            8-byte 00 00 AA 00 04 00 37 35

        Notice that the AA 00 04 00 prefix is defined in the
        specification but is unimportant and should not be parsed.

        Notice that control messages only have a source address in the
        header and so they can never be added into the conversation based
        tables."
      DECODING
        "NSP runs over DRP data packets; all other packet types are DRP
        control packets of one sort or another and do not carry any
        higher layer protocol.

        NSP packets are deemed to start at the beginning of the DRP data
        area.

        Data packets may be fragmented over multiple DRP data packets.
        The countsFragments(1) parameter indicates whether a probe can
        (and should) attribute non-leading fragments to the child
        protocol (above NSP in this case) or not.

        Recognition of DRP data packets and fragments is beyond the scope
        of this document."
      REFERENCE
        "DECnet Digital Network Architecture
          Phase IV
          Routing Layer Functional Specification
          Order# AA-X435A-TK
          Digital Equipment Corporation, Maynard, Massachusetts, USA"
      ::= {
       ether2 0x6003,
       snap   0x6003,
       802-1Q 0x6003   -- [0.0.96.3]
      }

nsp PROTOCOL-IDENTIFIER
      PARAMETERS {

```
     tracksSessions(1)
    }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
       "DEC Network Services Protocol."
    CHILDREN
       "Children of NSP are identified by the SCP 8-bit object type.
       Notice that the object type is included only in the session
       establishment messages (connect initiate, retransmitted connect
       initiate).

       Children of NSP are encoded [ 0.0.0.a ] where 'a' is the SCP
       object type.  Children of NSP are named as 'nsp' followed by the
       SCP object type in decimal.  CTERM is referred to as 'nsp cterm'
       OR 'nsp 42'."
    DECODING
       "An implementation is encouraged to examine SCP headers included
       in NSP control messages in order to determine which child
       protocol is present over a given session.  It is beyond the scope
       of this document to define the algorithm used to do this.

       The tracksSessions(1) flag indicates whether the probe can (and
       should) perform this analysis."
    REFERENCE
       "DECnet Digital Network Architecture
         Phase IV
         NSP Functional Specification
         Order# AA-X439A-TK
         Digital Equipment Corporation, Maynard, Massachusetts, USA"
    ::= { drp 1 }

dap-v1 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "DEC Data Access Protocol version 1."
    REFERENCE
       "Digital Corporation"
    ::= { nsp 1 }

dap-v4 PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "DEC Data Access Protocol versions 4 and above."
    REFERENCE
```

```
          "Digital Corporation"
     ::= { nsp 17 }

nice PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Network Information and Control Exchange protocol."
     REFERENCE
        "Digital Corporation"
     ::= { nsp 19 }

dec-loop PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Loopback Protocol."
     REFERENCE
        "Digital Corporation"
     ::= { nsp 25 }

dec-event PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC Event Protocol."
     REFERENCE
        "Digital Corporation"
     ::= { nsp 26 }

cterm PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "DEC CTERM Protocol."
     REFERENCE
        "Digital Corporation"
     ::= { nsp 42 }
```

3.1.7. The IBM SNA Protocol Stack.

```
sna-th PROTOCOL-IDENTIFIER
     PARAMETERS { }
     ATTRIBUTES { }
     DESCRIPTION
        "IBM's SNA TH protocol."
     REFERENCE
        "IBM Systems Network Architecture
```

```
          Format and Protocol
          Reference Manual: Architectural Logic

          SC30-3112-2

          IBM System Communications Division,
          Publications Development,
          Department E02,
          PO Box 12195,
          Research Triangle Park,
          North Carolina 27709."
   ::= {
    llc        0x04,                  -- [0.0.0.4]
    llc        0x08,                  -- [0.0.0.8]
    llc        0x0c,                  -- [0.0.0.12]
       ether2      0x80d5,               -- [0.0.128.213]
    802-1Q     0x02000004,           -- 1Q-LLC [2.0.0.4]
    802-1Q     0x02000008,           -- 1Q-LLC [2.0.0.8]
    802-1Q     0x0200000c,           -- 1Q-LLC [2.0.0.12]
       802-1Q      0x80d5                -- [0.0.128.213]
   }
```

3.1.8.   The NetBEUI/NetBIOS Family

```
-- CHILDREN OF NETBIOS
-- The NetBIOS/NetBEUI functions are implemented over a wide variety of
-- transports.  Despite varying implementations they all share two
-- features.  First, all sessions are established by connecting to
-- locally named services.  Second, all sessions transport application
-- data between the client and the named service.  In all cases the
-- identification of the application protocol carried within the data
-- packets is beyond the scope of this document.]
--
-- Children of NetBIOS/NetBEUI are identified by the following (32 bit)
-- enumeration
--
--      1    smb (Microsoft's Server Message Block Protocol)
--      2    notes (Lotus' Notes Protocol)
--      3    cc-mail (Lotus' CC Mail Protocol)
--
-- Children of NetBIOS/NetBEUI are encoded as [ a.b.c.d ] where 'a', 'b',
-- 'c' and 'd' are the four octets of the enumerated value in network
-- order (i.e.  'a' is the MSB and 'd' is the LSB).
--
-- For example notes over NetBEUI is declared as
--      'notes ::= { netbeui 2 }'
-- but is referred to as
--      'netbeui notes' OR 'netbeui 2'.
```

```
netbeui PROTOCOL-IDENTIFIER
    PARAMETERS {
     tracksSessions(1)
    }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
        "Lan Manager NetBEUI protocol."

    CHILDREN
        "See 'CHILDREN OF NETBIOS'"
    DECODING
        "NETBEUI provides a named service lookup function.  This function
        allows clients to locate a service by (locally assigned) name.
        An implementation is encouraged to follow lookups and session
        establishments and having determined the child protocol, track
        them.

        How the child protocol is determined and how the sessions are
        tracked is an implementation specific matter and is beyond the
        scope of this document."
    REFERENCE
        "IBM"
    ::= {
     llc        0xF0,              --  [0.0.0.240]
     802-1Q    0x020000F0         --  1Q-LLC [2.0.0.240]
    }

nbt-name PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "NetBIOS-over-TCP name protocol."
    REFERENCE
        "RFC 1001 [RFC1001] defines the 'PROTOCOL STANDARD FOR A NetBIOS
        SERVICE ON A TCP/UDP TRANSPORT: CONCEPTS AND METHODS.'  RFC 1002
        [RFC1002] defines the 'PROTOCOL STANDARD FOR A NetBIOS SERVICE ON
        A TCP/UDP TRANSPORT: DETAILED SPECIFICATIONS'."
    ::= {
     udp     137,
     tcp     137
    }

nbt-session PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
```

```
       "NetBIOS-over-TCP session protocol."
    REFERENCE
       "RFC 1001 [RFC1001] defines the 'PROTOCOL STANDARD FOR A NetBIOS
       SERVICE ON A TCP/UDP TRANSPORT: CONCEPTS AND METHODS.'  RFC 1002
       [RFC1002] defines the 'PROTOCOL STANDARD FOR A NetBIOS SERVICE ON
       A TCP/UDP TRANSPORT: DETAILED SPECIFICATIONS'."
    ::= {

     udp      139,
     tcp      139
    }

nbt-data PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
       "NetBIOS-over-TCP datagram protocol."
    CHILDREN
       "See 'CHILDREN OF NETBIOS'"
    REFERENCE
       "RFC 1001 [RFC1001] defines the 'PROTOCOL STANDARD FOR A NetBIOS
       SERVICE ON A TCP/UDP TRANSPORT: CONCEPTS AND METHODS.'  RFC 1002
       [RFC1002] defines the 'PROTOCOL STANDARD FOR A NetBIOS SERVICE ON
       A TCP/UDP TRANSPORT: DETAILED SPECIFICATIONS'."
    ::= {
     udp      138,
     tcp      138
    }

netbios-3com PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES {
     hasChildren(0)
    }
    DESCRIPTION
       "3COM NetBIOS protocol."
    CHILDREN
       "See 'CHILDREN OF NETBIOS'"
    REFERENCE
       "3Com Corporation"
    ::= {
     ether2  0x3C00,
     ether2  0x3C01,
     ether2  0x3C02,
     ether2  0x3C03,
     ether2  0x3C04,
```

```
           ether2  0x3C05,
           ether2  0x3C06,
           ether2  0x3C07,
           ether2  0x3C08,
           ether2  0x3C09,
           ether2  0x3C0A,
           ether2  0x3C0B,
           ether2  0x3C0C,
           ether2  0x3C0D,
           802-1Q  0x3C00,
           802-1Q  0x3C01,
           802-1Q  0x3C02,
           802-1Q  0x3C03,
           802-1Q  0x3C04,
           802-1Q  0x3C05,
           802-1Q  0x3C06,
           802-1Q  0x3C07,
           802-1Q  0x3C08,
           802-1Q  0x3C09,
           802-1Q  0x3C0A,
           802-1Q  0x3C0B,
           802-1Q  0x3C0C,
           802-1Q  0x3C0D
           }

   nov-netbios PROTOCOL-IDENTIFIER
       PARAMETERS { }
       ATTRIBUTES {
        hasChildren(0)
       }
       DESCRIPTION
          "Novell's version of the NetBIOS protocol."
       CHILDREN
          "See 'CHILDREN OF NETBIOS'"
       REFERENCE
          "Novell Corporation"
       ::= {
        nov-sap 0x0020,  -- preferred encapsulation to use, even though
                         -- the following are typically used also
        -- ipx  0x14,       -- when reached by IPX packet type
        -- nov-pep 0x0455   -- when reached by socket number
       }

   burst PROTOCOL-IDENTIFIER
       PARAMETERS { }
       ATTRIBUTES { }
       DESCRIPTION
          "Novell burst-mode transfer"
```

```
    REFERENCE

       "Novell Corporation"
    ::= { nov-pep 0x0d05 }

3.2.  Multi-stack protocols

smb PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Microsoft Server Message Block Protocol."
    REFERENCE
       "Microsoft Corporation"
    ::= {
        netbeui          1,
        netbios-3com     1,
        nov-netbios      1,
        nbt-data         1,
        nbt-session      1,
        nov-pep          0x550,
        nov-pep          0x552
    }

notes PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
       "Lotus Notes Protocol."
    REFERENCE
       "Lotus Development"
    ::= {
     netbeui          2,
     netbios-3com     2,
     nov-netbios      2,
     nbt-data         2,
     tcp              1352,
     udp              1352,
        nov-sap          0x039b
    }

ccmail PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }

    DESCRIPTION
       "Lotus CC-mail Protocol."
    REFERENCE
```

```
        "Lotus Development"
    ::= {
     netbeui          3,
     netbios-3com     3,
     nov-netbios      3,
     nbt-data         3,
     tcp              3264,
     udp              3264
    }

snmp  PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Simple Network Management Protocol. Includes SNMPv1 and SNMPv2
        protocol versions. Does not include SNMP trap packets."
    REFERENCE
        "The SNMP SMI is defined in RFC 1902 [RFC1902].  Version 1 of the
        SNMP  protocol is defined in RFC 1905 [RFC1905].  Transport
        mappings are defined in RFC 1906 [RFC1906]; RFC 1420 (SNMP over
        IPX) [RFC1420]; RFC 1419 (SNMP over AppleTalk) [RFC1419]."
    ::= {
     udp 161,
        nov-pep 0x900f,   -- [ 0.0.144.15 ]
     atalk 8,
     tcp 161
    }

snmptrap PROTOCOL-IDENTIFIER
    PARAMETERS { }
    ATTRIBUTES { }
    DESCRIPTION
        "Simple Network Management Protocol Trap Port."
    REFERENCE
        "The SNMP SMI is defined in RFC 1902 [RFC1902].  The SNMP
        protocol is defined in RFC 1905 [RFC1905].  Transport mappings
        are defined in RFC 1906 [RFC1906]; RFC 1420 (SNMP over IPX)
        [RFC1420]; RFC 1419 (SNMP over AppleTalk) [RFC1419]."
    ::= {
     udp 162,
        nov-pep 0x9010,
     atalk 9,
     tcp 162
    }

-- END
```

4. Intellectual Property

   The IETF takes no position regarding the validity or scope of any
   intellectual property or other rights that might be claimed to
   pertain to the implementation or use of the technology described in
   this document or the extent to which any license under such rights
   might or might not be available; neither does it represent that it
   has made any effort to identify any such rights. Information on the
   IETF's procedures with respect to rights in standards-track and
   standards-related documentation can be found in BCP-11. Copies of
   claims of rights made available for publication and any assurances of
   licenses to be made available, or the result of an attempt made to
   obtain a general license or permission for the use of such
   proprietary rights by implementors or users of this specification can
   be obtained from the IETF Secretariat."

   The IETF invites any interested party to bring to its attention any
   copyrights, patents or patent applications, or other proprietary
   rights which may cover technology that may be required to practice
   this standard. Please address the information to the IETF Executive
   Director.

5. Acknowledgements

   This document was produced by the IETF RMONMIB Working Group.

   The authors wish to thank the following people for their
   contributions to this document:

        Anil Singhal
        Frontier Software Development, Inc.

        Jeanne Haney
        Bay Networks

        Dan Hansen
        Network General Corp.

   Special thanks are in order to the following people for writing RMON
   PI macro compilers, and improving the specification of the PI macro
   language:

        David Perkins
        DeskTalk Systems, Inc.

        Skip Koppenhaver
        Technically Elite, Inc.

6. References

    [IEN158]   J. Haverty, "XNET Formats for Internet Protocol Version
                4", IEN 158, October 1980.

    [RFC407]   Bressler, R., Guida. R. and A. McKenzie, "Remote Job Entry
                Protocol", RFC 407, October 1972.

    [RFC493]   Michener, J., Cotton, I., Kelley, K., Liddle, D. and E.
                Meyer, "E.W., Jr Graphics Protocol", RFC 493, April 1973.

    [RFC734]   Crispin, M., "SUPDUP Protocol", RFC 734, October 1977.

    [RFC740]   Braden, R., "NETRJS Protocol", RFC 740, November 1977.

    [RFC741]   Cohen, D., "Specifications for the Network Voice
                Protocol", RFC 741, ISI/RR 7539, March 1976.

    [RFC759]   Postel, J., "Internet Message Protocol", RFC 759, August
                1980.

    [RFC768]   Postel, J., "User Datagram Protocol", STD 6, RFC 768,
                August 1980.

    [RFC791]   Postel, J., "Internet Protocol - DARPA Internet Program
                Protocol Specification", STD 5, RFC 791, September 1981.

    [RFC792]   Postel, J., "Internet Control Message Protocol - DARPA
                Internet Program Protocol Specification", STD 5, RFC 792,
                September 1981.

    [RFC793]   Postel, J., "Transmission Control Protocol - DARPA
                Internet Program Protocol Specification", STD 5, RFC 793,
                September 1981.

    [RFC818]   Postel, J., "Remote User Telnet service", RFC 818,
                November 1982.

    [RFC821]   Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC
                821, August 1982.

    [RFC823]   Hinden, R. and A. Sheltzer, "The DARPA Internet Gateway",
                RFC 823, September 1982.

    [RFC826]   Plummer, D., "An Ethernet Address Resolution Protocol or
                Converting Network Protocol Addresses to 48-bit Ethernet
                Addresses for Transmission on Ethernet Hardware", STD 37,
                RFC 826, November 1982.

   [RFC854]   Postel, J. and J. Reynolds, "Telnet Protocol
               Specification", STD 8, RFC 854, May 1983.

   [RFC862]   Postel, J., "Echo Protocol", STD 20, RFC 862, May 1983.

   [RFC863]   Postel, J., "Discard Protocol", STD 21, RFC 863, May 1983.

   [RFC864]   Postel, J., "Character Generator Protocol", STD 22, RFC
               864, May 1983.

   [RFC865]   Postel, J., "Quote of the Day Protocol", STD 23, RFC 865,
               May 1983.

   [RFC866]   Postel, J., "Active Users", STD 26, RFC 866, May 1983.

   [RFC867]   Postel, J., "Daytime Protocol", STD 25, RFC 867, May 1983.

   [RFC868]   Postel, J., "Time Protocol", STD 26, RFC 868, May 1983.

   [RFC869]   Hinden, R., "A Host Monitoring Protocol", RFC 869,
               December 1983.

   [RFC887]   Accetta, M., "Resource Location Protocol", RFC 887,
               December 1983.

   [RFC904]   International Telegraph and Telephone Co., D. Mills,
               "Exterior Gateway Protocol Formal Specification", STD 18,
               RFC 904, April 1984.

   [RFC905]   McKenzie, A., "ISO Transport Protocol Specification - ISO
               DP 8073", RFC 905, April 1984.

   [RFC908]   Velten, D., Hinden, R., and J. Sax, "Reliable Data
               Protocol", RFC 908, July 1984.

   [RFC913]   Lottor, M., "Simple File Transfer Protocol", RFC 913,
               September 1984.

   [RFC915]   Elvy, M. and R. Nedved, "Network mail path service", RFC
               915, December 1984.

   [RFC937]   Butler, M., Chase, D., Goldberger, J., Postel, J., and J.
               Reynolds, "Post Office Protocol - version 2", RFC 937,
               February 1985.

   [RFC938]   Miller, T., "Internet Reliable Transaction Protocol", RFC
               938, February 1985.

   [RFC951]   Croft, W. and J. Gilmore, "BOOTSTRAP Protocol (BOOTP)",
               RFC 951, September 1985.

   [RFC953]   Feinler, E., Harrenstien, K. and M. Stahl, "Hostname
               Server", RFC 953, October 1985.

   [RFC954]   Feinler, E., Harrenstien, K. and M. Stahl,
               "NICNAME/WHOIS", RFC 954, October 1985.

   [RFC959]   Postel, J., and J. Reynolds, "File Transfer Protocol", STD
               9, RFC 959, October 1985.

   [RFC972]   Wancho, F., "Password Generator Protocol", RFC 972,
               January 1986.

   [RFC977]   Kantor, B. and P. Lapsley, "Network News Transfer
               Protocol: A Proposed Standard for the Stream-Based
               Transmission of News", RFC 977, February 1986.

   [RFC996]   Mills, D., "Statistics server", RFC 996, February 1987.

   [RFC998]   Clark, D., Lambert, M. and L. Zhang, "NETBLT: A Bulk Data
               Transfer Protocol", RFC 998, March 1987.

   [RFC1001]   NetBIOS Working Group in the Defense Advanced Research
               Projects Agency, Internet Activities Board, End-to-End
               Services Task Force. "Protocol standard for a NetBIOS
               service on a TCP/UDP transport: Concepts and methods",
               STD 19, RFC 1001, March 1987.

   [RFC1002]   NetBIOS Working Group in the Defense Advanced Research
               Projects Agency, Internet Activities Board, End-to-End
               Services Task Force. "Protocol standard for a NetBIOS
               service on a TCP/UDP transport: Detailed
               specifications.", STD 19, RFC 1002, March 1987.

   [RFC1021]   Partridge, C. and G. Trewitt, "High-level Entity
               Management System HEMS", RFC 1021, October 1987.

   [RFC1028]   Case, J., Davin, J., Fedor, M. and M. Schoffstall, "Simple
               Gateway Monitoring Protocol", RFC 1028, November 1987.

   [RFC1035]   Mockapetris, P., "Domain Names - Implementation and
               Specification", STD 13, RFC 1035, November 1987.

   [RFC1056]   Lambert, M., "PCMAIL: A distributed mail system for
               personal computers", RFC 1056, June 1988.

   [RFC1057]   Sun Microsystems, Inc, "RPC: Remote Procedure Call
               Protocol Specification version 2", RFC 1057, June 1988.

   [RFC1064]   Crispin, M., "Interactive Mail Access Protocol: Version
               2", RFC 1064, July 1988.

   [RFC1068]   DeSchon, A. and R. Braden, "Background File Transfer
               Program BFTP", RFC 1068, August 1988.

   [RFC1070]   Hagens, R., Hall, N. and M. Rose, "Use of the Internet as
               a subnetwork for experimentation with the OSI network
               layer", RFC 1070, February 1989.

   [RFC1078]   Lottor, M., "TCP port service Multiplexer TCPMUX", RFC
               1078, November, 1988.

   [RFC1086]   Onions, J. and M. Rose, "ISO-TP0 bridge between TCP and
               X.25", RFC 1086, December 1988.

   [RFC1095]   Warrier, U. and L. Besaw, "Common Management Information
               Services and Protocol over TCP/IP (CMOT)", RFC 1095, April
               1989.

   [RFC1112]   Deering, S., "Host Extensions for IP Multicasting", STD 5,
               RFC 1112, August 1989.

   [RFC1155]   Rose, M. and K. McCloghrie, "Structure and Identification
               of Management Information for TCP/IP-based Internets", STD
               16, RFC 1155, May 1990.

   [RFC1157]   Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple
               Network Management Protocol", STD 15, RFC 1157, May 1990.

   [RFC1203]   Rice, J., "Interactive Mail Access Protocol - Version 3",
               RFC 1203, February 1991.

   [RFC1204]   Lee, D. and S. Yeh, "Message Posting Protocol (MPP)", RFC
               1204, February 1991.

   [RFC1212]   Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD
               16, RFC 1212, March 1991.

   [RFC1213]   McCloghrie, K. and M. Rose, "Management Information Base
               for Network Management of TCP/IP-based internets: MIB-II",
               STD 17, RFC 1213, March 1991.

   [RFC1215]   Rose, M., "A Convention for Defining Traps for use with
               the SNMP", RFC 1215, March 1991.

   [RFC1226]   Kantor, B., "Internet Protocol Encapsulation of AX.25
               Frames", RFC 1226, May 1991.

   [RFC1227]   Rose, M., "SNMP MUX Protocol and MIB", RFC 1227, May 1991.

   [RFC1234]   Provan, D., "Tunneling IPX Traffic through IP Networks",
               RFC 1234, June 1991.

   [RFC1235]   Ioannidis, J. and G. Maguire, Jr., "The Coherent File
               Distribution Protocol", RFC 1235, June 1991.

   [RFC1241]   Mills, D. and R. Woodburn, "A Scheme for an Internet
               Encapsulation Protocol: Version 1", RFC 1241, July 1991.

   [RFC1249]   Howes, T., Smith, M. and B. Beecher, "DIXIE Protocol
               Specification", RFC 1249, August 1991.

   [RFC1267]   Lougheed, K. and Y. Rekhter, "A Border Gateway Protocol 3
               (BGP-3)", RFC 1267, October 1991.

   [RFC1282]   Kantor, B., "BSD Rlogin", RFC 1282, December 1991.

   [RFC1288]   Zimmerman, D., "The Finger User Information Protocol", RFC
               1288, December 1991.

   [RFC1301]   Amstrong, S., Freier, A. and K. Marzullo, "Multicast
               Transport Protocol", RFC 1301, February 1992.

   [RFC1305]   Mills, D., "Network Time Protocol (v3)", RFC 1305, April
               1992.

   [RFC1312]   Nelson, R. and G. Arnold, "Message Send Protocol", RFC
               1312, April 1992.

   [RFC1339]   Dorner, S. and P. Resnick, "Remote Mail Checking
               Protocol", RFC 1339, June 1992.

   [RFC1350]   Sollins, K., "TFTP Protocol (revision 2)", RFC 1350, July
               1992.

   [RFC1413]   St. Johns, M., "Identification Protocol", RFC 1413,
               February 1993.

   [RFC1419]   Minshall, G. and M. Ritter, "SNMP over AppleTalk", RFC
               1419, March 1993.

   [RFC1420]   Bostock, S., "SNMP over IPX", RFC 1420, March 1993.

   [RFC1436]   Anklesaria, F., McCahill, M., Lindner, P., Johnson, D.,
               John, D., Torrey, D. and B. Alberti, "The Internet Gopher
               Protocol (a distributed document search and retrieval
               protocol)", RFC 1436, March 1993.

   [RFC1459]   Oikarinen, J. and D. Reed, "Internet Relay Chat Protocol",
               RFC 1459, May 1993.

   [RFC1476]   Ullmann, R., "RAP: Internet Route Access Protocol", RFC
               1476, June 1993.

   [RFC1479]   Steenstrup, M., "Inter-Domain Policy Routing Protocol
               Specification: Version 1", RFC 1479, July 1993.

   [RFC1483]   Heinanen, J., "Multiprotocol Encapsulation over ATM
               Adaptation Layer 5", RFC 1483, July 1993.

   [RFC1492]   Finseth, C., "An Access Control Protocol, Sometimes Called
               TACACS", RFC 1492, July 1993.

   [RFC1510]   Kohl, J. and B. Neuman, "The Kerberos Network
               Authentication Service (V5)", RFC 1510, September 1993.

   [RFC1583]   Moy, J., "OSPF Version 2", RFC 1583, March 1994.

   [RFC1700]   Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC
               1700, October 1994.

   [RFC1701]   Hanks, S., Li, T., Farinacci, D. and P. Traina, "Generic
               Routing Encapsulation (GRE)", RFC 1701, October 1994.

   [RFC1702]   Hanks, S., Li, T., Farinacci, D. and P. Traina, "Generic
               Routing Encapsulation over IPv4 networks", RFC 1702,
               October 1994.

   [RFC1725]   Myers, J. and M. Rose, "Post Office Protocol - Version 3",
               RFC 1725, November 1994.

   [RFC1729]   Lynch, C., "Using the Z39.50 Information Retrieval
               Protocol in the Internet Environment", RFC 1729, December
               1994.

   [RFC1730]   Crispin, M., "Internet Message Access Protocol - Version
               4", RFC 1730, December 1994.

   [RFC1739]   Kessler, G. and S. Shepard, "A Primer On Internet and
               TCP/IP Tools", RFC 1739, December 1994.

   [RFC1745]   Varadhan, K., Hares, S. and Y. Rekhter, "BGP4/IDRP for
               IP---OSPF Interaction", RFC 1745, December 1994.

   [RFC1757]   Waldbusser, S., "Remote Network Monitoring MIB", RFC 1757,
               February 1995.

   [RFC1777]   Yeong, W., Howes, T. and S. Kille, "Lightweight Directory
               Access Protocol", RFC 1777, March 1995.

   [RFC1782]   Malkin, G. and A. Harkin, "TFTP Option Extension", RFC
               1782, March 1995.

   [RFC1783]   Malkin, G. and A. Harkin, "TFTP BlockOption Option", RFC
               1783, March 1995.

   [RFC1784]   Malkin, G. and A. Harkin, "TFTP Timeout Interval and
               Transfer Size Options", RFC 1784, March 1995.

   [RFC1798]   Young, A., "Connection-less Lightweight Directory Access
               Protocol", RFC 1798, June 1995.

   [RFC1813]   Callaghan, B., Pawlowski, B. and P. Staubach, "NFS Version
               3 Protocol Specification", RFC 1813, June 1995.

   [RFC1819]   Delgrossi, L. and L. Berger, "Internet Stream Protocol
               Version 2 (ST2)", RFC 1819, August 1995.

   [RFC1831]   Srinivasan, R., "Remote Procedure Call Protocol Version
               2", RFC 1831, August 1995.

   [RFC1853]   Simpson, W., "IP in IP Tunneling", RFC 1853, October 1995.

   [RFC1901]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Introduction to Community-based SNMPv2", RFC 1901,
               January 1996.

   [RFC1902]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Structure of Management Information for version 2 of the
               Simple Network Management Protocol (SNMPv2)", RFC 1902,
               January 1996.

   [RFC1903]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Textual Conventions for version 2 of the Simple Network
               Management Protocol (SNMPv2)", RFC 1903, January 1996.

   [RFC1904]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Conformance Statements for version 2 of the Simple
               Network Management Protocol (SNMPv2)", RFC 1904, January
               1996.

   [RFC1905]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Protocol Operations for Version 2 of the Simple Network
               Management Protocol (SNMPv2)", RFC 1905, January 1996.

   [RFC1906]   Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
               "Transport Mappings for Version 2 of the Simple Network
               Management Protocol (SNMPv2)", RFC 1906, January 1996.

   [RFC1940]   Estrin, D., Li, T., Rekhter, Y., Varadhan, K. and D.
               Zappala, "Source Demand Routing: Packet Format and
               Forwarding Specification (Version 1)", RFC 1940, May 1996.

   [RFC1945]   Berners-Lee, T. and R. Fielding, "Hypertext Transfer
               Protocol -- HTTP/1.0", RFC 1945, November 1995.

   [RFC2002]   Perkins, C., "IP Mobility Support", RFC 2002, October
               1996.

   [RFC2003]   Perkins, C., "IP Encapsulation within IP", RFC 2003,
               October 1996.

   [RFC2021]   Waldbusser, S., "Remote Network Monitoring MIB (RMON-2)",
               RFC 2021, January 1997.

   [RFC2037]   McCloghrie, K. and A. Bierman, "Entity MIB using SMIv2",
               RFC 2037, October 1996.

   [RFC2068]   Fielding, R., Gettys, J., Mogul, J., Frystyk, H. and T.
               Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1",
               RFC 2068, January 1997.

   [RFC2069]   Franks, J., Hallam-Baker, P., Hostetler, J., Luotonen, P.
               A. and E. L. Stewart, "An Extension to HTTP: Digest Access
               Authentication", RFC 2069, January 1997.

   [RFC2074]   Bierman, A. and R. Iddon, "Remote Network Monitoring MIB
               Protocol Identifiers", RFC 2074, January 1997.

   [RFC2109]   Kristol, D. and L. Montulli, "HTTP State Management
               Mechanism", RFC 2109, February 1997.

   [RFC2138]   Rigney, C., Rubens, A., Simpson, W. and W. Willens,
               "Remote Authentication Dial In User Service (RADIUS)", RFC
               2138, April 1997.

   [RFC2139]   Rigney, C., "RADIUS Accounting", RFC 2139, April 1997.

   [RFC2145]   Mogul, J., Fielding, R., Gettys, J. and H. Frystyk, "Use
               and interpretation of HTTP version numbers", RFC 2145, May
               1997.

   [RFC2205]   Braden, R., Zhang, L., Berson, S., Herzog, S. and S.
               Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1
               Functional Specification", RFC 2205, September, 1997.

   [RFC2233]   McCloghrie, K. and F. Kastenholz, "The Interfaces Group
               MIB Using SMIv2", RFC 2233, November, 1997.

   [RFC2271]   Harrington, D., Presuhn, R. and B. Wijnen, "An
               Architecture for Describing SNMP Management Frameworks",
               RFC 2271, January 1998.

   [RFC2272]   Case, J., Harrington D., Presuhn R. and B. Wijnen,
               "Message Processing and Dispatching for the Simple Network
               Management Protocol (SNMP)", RFC 2272, January 1998.

   [RFC2273]   Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications",
               RFC 2273, January 1998.

   [RFC2274]   Blumenthal, U. and B. Wijnen, "User-based Security Model
               (USM) for version 3 of the Simple Network Management
               Protocol (SNMPv3)", RFC 2274, January 1998.

   [RFC2275]   Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based
               Access Control Model (VACM) for the Simple Network
               Management Protocol (SNMP)", RFC 2275, January 1998.

   [RFC2332]   Luciani, J., Katz, D., Piscitello, D., Cole, B. and N.
               Doraswamy, "NBMA Next Hop Resolution Protocol (NHRP)", RFC
               2332, April 1998.

   [RFC2408]   Maughan, D., Schertler, M., Schneider, M. and J. Turner,
               RFC 2408, November 1998.

   [RFC2570]   Case, J., Mundy, R., Partain, D. and B. Stewart,
               "Introduction to Version 3 of the Internet-standard
               Network Management Framework", RFC 2570, April 1999.

   [RFC2571]   Harrington, D., Presuhn, R. and B. Wijnen, "An
               Architecture for Describing SNMP Management Frameworks",
               RFC 2571, April 1999.

   [RFC2572]   Case, J., Harrington D., Presuhn R. and B. Wijnen,
               "Message Processing and Dispatching for the Simple Network
               Management Protocol (SNMP)", RFC 2572, April 1999.

   [RFC2573]   Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications",
               RFC 2573, April 1999.

   [RFC2574]   Blumenthal, U. and B. Wijnen, "User-based Security Model
               (USM) for version 3 of the Simple Network Management
               Protocol (SNMPv3)", RFC 2574, April 1999.

   [RFC2575]   Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based
               Access Control Model (VACM) for the Simple Network
               Management Protocol (SNMP)", RFC 2575, April 1999.

   [RFC2578]   McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J.,
               Rose, M. and S. Waldbusser, "Structure of Management
               Information Version 2 (SMIv2)", STD 58, RFC 2578, April
               1999.

   [RFC2579]   McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J.,
               Rose, M. and S. Waldbusser, "Textual Conventions for
               SMIv2", STD 58, RFC 2579, April 1999.

   [RFC2580]   McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J.,
               Rose, M. and S. Waldbusser, "Conformance Statements for
               SMIv2", STD 58, RFC 2580, April 1999.

   [RFC2600]   Reynolds, J. and R. Braden, "Internet Official Protocol
                Standards", STD 1, RFC 2600, March 2000.

   [RFC2895]   Bierman, A., Bucci, C. and R. Iddon, "RMON Protocol
                Identifier Reference", RFC 2895, August 2000.

7. Security Considerations

   This document contains textual descriptions of well-known networking
   protocols, not the definition of any networking behavior. As such,
   no security considerations are raised by its publication.

8. Authors' Addresses

   Andy Bierman
   Cisco Systems, Inc.
   170 West Tasman Drive
   San Jose, CA USA 95134

   Phone: +1 408-527-3711
   EMail: abierman@cisco.com

   Chris Bucci
   Cisco Systems, Inc.
   170 West Tasman Drive
   San Jose, CA USA 95134

   Phone: +1 408-527-5337
   EMail: cbucci@cisco.com

   Robin Iddon
   c/o 3Com Inc.
   Blackfriars House
   40/50 Blackfrias Street
   Edinburgh, EH1 1NE, UK

   Phone: +44 131.558.3888
   EMail: None

9. Full Copyright Statement

Acknowledgement