

Network Working Group
Internet-Draft
Expires: May 2001

Dave Thaler
Microsoft Corporation
Satyen Chandragiri
Lucent Technologies
November 2000

Subtree Retrieval MIB
<[draft-irtf-nmrg-get-subtree-mib-01.txt](#)>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

Draft

GET-SUBTREE MIB

November 2000

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects used for retrieving subtrees of MIB objects without the overshoot problems of the SNMP GetBulkRequest operation.

2. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- o An overall architecture, described in [RFC 2571](#) [1].
- 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 [RFC 1155](#) [2], [RFC 1212](#) [3] and [RFC 1215](#) [4]. The second version, called SMIV2, is described in [RFC 2578](#) [5], [RFC 2579](#) [6] and [RFC 2580](#) [7].
- o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in [RFC 1157](#) [8]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in [RFC 1901](#) [9] and [RFC 1906](#) [10]. The third version of the message protocol is called SNMPv3 and described in [RFC 1906](#) [10], [RFC 2572](#) [11] and [RFC 2574](#) [12].
- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in [RFC 1157](#) [8]. A second set of protocol operations and associated PDU formats is described in [RFC 1905](#) [13].
- o A set of fundamental applications described in [RFC 2573](#) [14] and the view-based access control mechanism described in [RFC 2575](#) [15].

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.

Expires May 2001

[Page 2]

Draft

GET-SUBTREE MIB

November 2000

This memo specifies a MIB module that is compliant with the SMIV2. A MIB conforming to the SMIV1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine readable information in SMIV2 will be converted into textual descriptions in SMIV1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB.

[3.](#) Overview

A major shortcoming of SNMP is the lack of a mechanism to efficiently retrieve large amounts of MIB data from a device. SNMPv1, though widely deployed, contains no provision for bulk retrieval and the manager must use GetNextRequests to traverse the MIB. This requires a large number of request-response exchanges leading to high latency. To address this problem, the GetBulkRequest operation [[13](#)] was introduced in SNMPv2.

The GetBulkRequest operation aims to minimize the number of protocol exchanges required to retrieve a large amount of management information by returning a series of variable bindings in a single response. The requester is required to specify a "max-repetitions" count, and the agent then fills in as many variable bindings as it can without exceeding either this count, or the maximum message size.

The main problem with retrieving tables using GetBulkRequest is that the manager typically does not know the number of rows in the table, and hence cannot set max-repetitions to the optimal value. As a result, the manager must either set max-repetitions to some huge value, resulting in a potentially large waste of bandwidth when many more variable bindings are returned than are needed (sometimes called "overshoot"), or else must issue multiple GetBulkRequests sequentially to traverse a large table such as the

routing table in a backbone router.

This document describes a MIB that can be used by an application to retrieve MIB information in a more efficient way without any change to the SNMP protocol or SNMP protocol engines. This operation, referred to as a "GetSubtree" request, allows a manager to request the agent to retrieve a subtree of MIB data starting at the root OID (specified by the manager) and including all object

Expires May 2001

[Page 3]

Draft

GET-SUBTREE MIB

November 2000

instances whose OID is prefixed by the root OID. The manager is also provided the facility to specify several subtrees of data to be retrieved in parallel by the agent.

This approach eliminates the guesswork involved in using the GetBulkRequest operation and makes good use of bandwidth as only those MIB objects of interest to the manager are retrieved (albeit with some overhead).

[3.1](#). Structure of the MIB

This MIB consists of two tables:

- o getSubtreeRootTable - this table stores the root OID(s) of the subtree(s) to be retrieved. Rows in this table are indexed by getSubtreeRootOperationID which uniquely identifies the retrieval operation in progress, and a secondary index (getSubtreeRootOIDIndex) which identifies the root OIDs associated with the request.
- o getSubtreeControlTable - this table stores the main getSubtree request information. The manager must create a row which identifies the application (represented by an entry in the snmpTargetAddrTable) as the response destination. Rows in this table are indexed by getSubtreeRootOperationID from the getSubtreeRootTable.

Note: The application must have been pre-configured as a legal notification target in the SNMP-TARGET-MIB [\[14\]](#).

[3.2.](#) Operation

To set up a retrieval, an application issues a SetRequest to perform a row creation in the getSubtreeRootTable using an arbitrarily selected value for the primary index (getSubtreeRootOperationID), a value of "1" for the secondary index (getSubtreeRootOIDIndex), and the root OID of the subtree to be retrieved. If the application wishes to retrieve multiple subtrees simultaneously, it may do so by creating additional rows for each subtree's root OID using the same primary index value and sequentially increasing values for getSubtreeRootOIDIndex. The purpose of starting the secondary index values at "1" for each new GetSubtree operation is to avoid concurrency problems if another

Expires May 2001

[Page 4]

Draft

GET-SUBTREE MIB

November 2000

management application happens to select the same value of the primary index for creating its rows in this table. Note that creating rows in the getSubtreeRootTable in itself does not start the retrieval process.

The manager must then create a row in the getSubtreeControlTable using the same value of getSubtreeRootOperationID as the index and identifying itself via the getSubtreeControlTarget object. Creating a row in this table triggers the agent to begin sending traps containing the requested information to the application, so long as there are one or more entries in the getSubtreeRootTable indexed by the same value of getSubtreeRootOperationID. If there are no such entries, then this newly created row in the getSubtreeControlTable will be deleted automatically by the agent. It may also be that a manager may make one or more rows in the getSubtreeRootTable and fail to make any entries in the getSubtreeControlTable to drive the operation. In this case, the agent may choose what to do with these "orphaned" entries. Options include, but are not limited to: letting them exist indefinitely; deleting them after some timeout period has elapsed; or deleting them if there is a resource shortage. The choice is implementation dependent.

Each trap generated by the agent contains a sequence number object (which can be used to detect losses), a counter object, a flag

object signalling whether the trap is the final one in the sequence, and a series of varbinds containing the next chunk from the requested subtree. The agent will send as many varbinds in the trap as possible without exceeding the maximum message size and without going beyond the subtree. Multiple traps will be sent until the entire subtree has been retrieved. If more than one subtree has been requested, the agent will send repetitions containing varbinds from each subtree until all subtrees have been retrieved. The agent must ensure that each trap contains only an integral number of repetitions.

If a manager detects a lost trap (or a sequence of lost traps), it can request that portion of the subtree to be retransmitted using a GetBulkRequest. The counter in each trap indicates the number of repetitions transmitted thus far (including those sent in the current trap). A management application can use this counter as a hint when selecting a value for the max-repetitions field in the GetBulkRequest when sending the retransmission request.

Once the entire retrieval operation is complete, the request will

be deleted from `getSubtreeControlTable` along with all its corresponding rows in `getSubtreeRootTable`.

Furthermore, if the request was in error (e.g., a human entered a different OID from what was intended, causing the application to receive large amounts of unwanted data), the MIB also provides a way to halt an operation in progress, if the agent is able to support this. Halting an operation in progress is accomplished simply by allowing the application to delete the conceptual row in `getSubtreeControlTable` corresponding to the outstanding operation. If the retrieval operation is successfully aborted, the agent will also automatically delete the rows from `getSubtreeRootTable` corresponding to that operation.

[3.3.](#) Limitations

The limitations of this approach that come with not changing SNMP include:

- o To use this MIB to retrieve subtrees of information, the application must be able to issue SETs (at least to this MIB), not just GETs.
- o The command responder and notification originator need to be tightly coupled, as well as the command generator and notification responder.
- o The subagent implementing this MIB must be able to call back into the SNMP engine to walk other MIBs, without causing a deadlock.

Expires May 2001

[Page 6]

Draft

GET-SUBTREE MIB

November 2000

[4.](#) Definitions

GET-SUBTREE-MIB DEFINITIONS ::= BEGIN

IMPORTS

| | |
|----------------------------------|--------------------------|
| MODULE-IDENTITY, OBJECT-TYPE, | |
| NOTIFICATION-TYPE, Unsigned32, | |
| Counter32 | FROM SNMPv2-SMI |
| RowStatus, TruthValue | FROM SNMPv2-TC |
| MODULE-COMPLIANCE, OBJECT-GROUP, | |
| NOTIFICATION-GROUP | FROM SNMPv2-CONF |
| SnmpAdminString | FROM SNMP-FRAMEWORK-MIB; |

```

getSubtreeMIB MODULE-IDENTITY
    LAST-UPDATED "0010201200Z"
    ORGANIZATION "IRTF Network Management Research Group"
    CONTACT-INFO
        "Dave Thaler
        Microsoft Corporation
        One Microsoft Way
        Redmond, WA 98052-6399
        EMail: dthaler@microsoft.com

        Satyen Chandragiri
        Lucent Technologies, Inc.
        101 Crawfords Corner Road
        Holmdel, NJ 07733-3030
        EMail: satyen@lucent.com"
    DESCRIPTION
        "This MIB module provides the ability to retrieve
        arbitrary subtrees of OIDs (in parallel) by receiving
        traps."
    ::= { XXX }

getSubtreeMIBObjects OBJECT IDENTIFIER ::= { getSubtreeMIB 1 }

getSubtree          OBJECT IDENTIFIER ::= { getSubtreeMIBObjects 1 }
getSubtreeTraps      OBJECT IDENTIFIER ::= { getSubtreeMIBObjects 2 }

--
-- GetSubtree Root Table
--

getSubtreeRootTable OBJECT-TYPE

```

Expires May 2001

[Page 7]

Draft

GET-SUBTREE MIB

November 2000

```

SYNTAX      SEQUENCE OF GetSubtreeRootEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The (conceptual) table containing the root OID(s) of
    the MIB subtree(s) to be retrieved. Multiple rows with
    different values for getSubtreeRootOID may be created

```


if the manager wishes to retrieve several subtrees
simultaneously"
 ::= { getSubtree 1 }

getSubtreeRootEntry OBJECT-TYPE

SYNTAX GetSubtreeRootEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry (conceptual row) specifying the root OID of
the MIB subtree to be retrieved"

INDEX { getSubtreeRootOperationID, getSubtreeRootOIDIndex }

::= { getSubtreeRootTable 1 }

GetSubtreeRootEntry ::= SEQUENCE {

getSubtreeRootOperationID Unsigned32,

getSubtreeRootOIDIndex Unsigned32,

getSubtreeRootOID OBJECT IDENTIFIER,

getSubtreeRootStatus RowStatus

}

getSubtreeRootOperationID OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An integer uniquely identifying the GetSubtree
operation in progress. This value should be randomly
generated by a manager before attempting to create the
row."

::= { getSubtreeRootEntry 1 }

getSubtreeRootOIDIndex OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An index value used to associate various root OIDs

be one for the first entry associated with a particular retrieval operation and should be sequentially incremented for each additional entry."
 ::= { getSubtreeRootEntry 2 }

getSubtreeRootOID OBJECT-TYPE
SYNTAX OBJECT IDENTIFIER
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The root OID of the subtree to be retrieved."
 ::= { getSubtreeRootEntry 3 }

getSubtreeRootStatus OBJECT-TYPE
SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The status of this row. The manager should create one row for each subtree to be retrieved. After the entire subtree retrieval operation is completed (or if the operation is aborted by the user) all rows in this table corresponding to that operation are automatically deleted by the agent.

Objects in this table cannot be modified while a GetSubtree retrieval operation is in progress."
 ::= { getSubtreeRootEntry 4 }

--
-- GetSubtree Control Table
--

getSubtreeControlTable OBJECT-TYPE
SYNTAX SEQUENCE OF GetSubtreeControlEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The (conceptual) table used to control GetSubtree operations in progress. This table is used in conjunction with getSubtreeRootTable.

A manager interested in retrieving a subtree of MIB

objects must first create an entry in the getSubtreeRootTable specifying the root OID (s) of the subtree(s) to be retrieved. It must then make a corresponding entry in this table using the same getSubtreeRootOperationID value and identifying itself (via getSubtreeControlTarget) as the recipient of the traps.

When the agent completes retrieval of all MIB instances within the specified subtree(s) this conceptual row will be automatically deleted."

::= { getSubtree 2 }

getSubtreeControlEntry OBJECT-TYPE

SYNTAX GetSubtreeControlEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry (conceptual row) containing the information on a particular GetSubtree operation in progress."

INDEX { getSubtreeRootOperationID }

::= { getSubtreeControlTable 1 }

GetSubtreeControlEntry ::= SEQUENCE {

getSubtreeControlTarget SnmpAdminString,

getSubtreeControlSeqNumber Counter32,

getSubtreeControlCount Counter32,

getSubtreeControlDone TruthValue,

getSubtreeControlStatus RowStatus

}

getSubtreeControlTarget OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object selects a management target defined in the snmpTargetAddrTable (in the SNMP-TARGET-MIB). The selected target is defined by an entry in the snmpTargetAddrTable whose index value (snmpTargetAddrName) is equal to this object."

::= { getSubtreeControlEntry 2 }

getSubtreeControlSeqNumber OBJECT-TYPE

SYNTAX Counter32

Draft

GET-SUBTREE MIB

November 2000

MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of trap responses previously sent for this
 GetSubtree request."
::= { getSubtreeControlEntry 3 }

getSubtreeControlCount OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of repetitions transmitted thus far (up to
 and including those sent in the current trap)"
::= { getSubtreeControlEntry 4 }

getSubtreeControlDone OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This is set to true in the last trap sent, and is set
 to false otherwise."
::= { getSubtreeControlEntry 5 }

getSubtreeControlStatus OBJECT-TYPE

SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
 "The status of this row, by which new entries may be
 created, or old entries deleted from this table. Once
 created, the row may be deleted, but other objects in
 the row may not be modified. A row (and corresponding
 rows in the getSubtreeRootTable) will be deleted
 automatically by the agent once the operation has
 completed. A row may be created using an instance
 value for getSubtreeRootOperationID even if no entries
 exist in getSubtreeRootTable indexed by that value.

Creating a row will cause the subtree retrieval

operation to commence if there are entries in the
getSubtreeRootTable indexed by the same value of
getSubtreeRootOperationID as this row. If the agent
allows aborting operations in progress, deleting a row

Expires May 2001

[Page 11]

Draft

GET-SUBTREE MIB

November 2000

will cause the operation to halt. If the operation is
successfully aborted, the rows in the
getSubtreeRootTable corresponding to this request will
also be automatically deleted by the agent."
::= { getSubtreeControlEntry 6 }

-- traps

getSubtreeTrapPrefix OBJECT IDENTIFIER ::= { getSubtreeTraps 0 }

getSubtreeResponse NOTIFICATION-TYPE

OBJECTS {
 getSubtreeControlSeqNumber,
 getSubtreeControlCount,
 getSubtreeControlDone
}

STATUS current

DESCRIPTION

"In addition to the three objects above, this trap
also contains a series of varbinds containing the next
chunk of the subtree. The generating entity will
append, in order, as many variables to the variable-
bindings field as it can without exceeding the maximum
message size, and without going beyond the subtree of
OIDs requested. A series of such traps will be
generated until the end of the subtree is reached."

::= { getSubtreeTrapPrefix 1 }

-- conformance information

getSubtreeMIBConformance

OBJECT IDENTIFIER ::= { getSubtreeMIB 2 }

getSubtreeMIBCompliances

OBJECT IDENTIFIER ::= { getSubtreeMIBConformance 1 }

```

getSubtreeMIBGroups OBJECT IDENTIFIER ::= { getSubtreeMIBConformance 2 }

-- compliance statements

getSubtreeMIBCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "The compliance statement for the GetSubtree MIB."
    MODULE -- this module
    MANDATORY-GROUPS {

```

Expires May 2001

[Page 12]

Draft

GET-SUBTREE MIB

November 2000

```

        getSubtreeObjectGroup,
        getSubtreeNotificationGroup
    }
    ::= { getSubtreeMIBCompliances 1 }

-- units of conformance

getSubtreeObjectGroup OBJECT-GROUP
    OBJECTS {
        getSubtreeRootOID,
        getSubtreeRootStatus,
        getSubtreeControlTarget,
        getSubtreeControlSeqNumber,
        getSubtreeControlCount,
        getSubtreeControlDone,
        getSubtreeControlStatus
    }
    STATUS current
    DESCRIPTION
        "A collection of objects to support requests for
        subtree retrieval operations."
    ::= { getSubtreeMIBGroups 1 }

getSubtreeNotificationGroup NOTIFICATION-GROUP
    NOTIFICATIONS { getSubtreeResponse }
    STATUS current
    DESCRIPTION
        "The notification which an entity is required to
        implement."

```

```
::= { getSubtreeMIBGroups 2 }
```

END

Expires May 2001

[Page 13]

Draft

GET-SUBTREE MIB

November 2000

[5.](#) Security Considerations

While unauthorized read access to the objects in this MIB is relatively innocuous, unauthorized write access could trigger sending of a potentially large amount of data to an authorized notification receiver, which could be viewed as a denial-of-service attack.

Hence, the support for SNMP operations in a non-secure environment without proper protection can have a negative effect on network operations.

SNMPv1 by itself is such an insecure environment. Even if the network itself is secure (for example by using IPSec [16]), even then, there is no control as to who on the secure network is allowed to access and SET (change/create/delete) the objects in this MIB.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC 2274](#) [12] and the View-based Access Control Model [RFC 2275](#) [15] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to this MIB, is properly configured to give access to those objects only to those principals (users) that have legitimate rights to access them.

[6.](#) Authors' Addresses

Dave Thaler
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
Phone: +1 425 703 8835
EMail: dthaler@microsoft.com

Satyen Chandragiri
Lucent Technologies, Inc.
101 Crawfords Corner Road
Holmdel, NJ 07733-3030
Phone: +1 732 949 2080
EMail: satyen@lucent.com

Expires May 2001

[Page 14]

Draft

GET-SUBTREE MIB

November 2000

[7.](#) References

- [1] Wijnen, B., Harrington, D., and R. Presuhn, "An Architecture for Describing SNMP Management Frameworks", [RFC 2571](#), Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.
- [2] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, [RFC 1155](#), Performance Systems International, Hughes LAN Systems, May 1990.
- [3] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, [RFC 1212](#), Performance Systems International, Hughes LAN Systems, March 1991.
- [4] M. Rose, "A Convention for Defining Traps for use with the

SNMP", [RFC 1215](#), Performance Systems International, March 1991.

- [5] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIv2)", STD 58, [RFC 2578](#), April 1999.
- [6] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, [RFC 2579](#), April 1999.
- [7] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, [RFC 2580](#), April 1999.
- [8] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, [RFC 1157](#), SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.
- [9] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", [RFC 1901](#), SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.
- [10] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1906](#), SNMP Research, Inc.,

Expires May 2001

[Page 15]

Draft

GET-SUBTREE MIB

November 2000

Cisco Systems, Inc., Dover Beach Consulting, Inc.,
International Network Services, January 1996.

- [11] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", [RFC 2572](#), SNMP Research, Inc., Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, April 1999.
- [12] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol

(SNMPv3)", [RFC 2574](#), IBM T. J. Watson Research, April 1999.

- [13] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1905](#), SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.
- [14] Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", [RFC 2573](#), SNMP Research, Inc., Secure Computing Corporation, Cisco Systems, April 1999.
- [15] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", [RFC 2575](#), IBM T. J. Watson Research, BMC Software, Inc., Cisco Systems, Inc., April 1999.

[8.](#) Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into

Expires May 2001

[Page 16]

Draft

GET-SUBTREE MIB

November 2000

languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.