

## Host Resources MIB

### Status of this Memo

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

### Abstract

This memo defines a MIB for use with managing host systems. The term "host" is construed to mean any computer that communicates with other similar computers attached to the internet and that is directly used by one or more human beings. Although this MIB does not necessarily apply to devices whose primary function is communications services (e.g., terminal servers, routers, bridges, monitoring equipment), such relevance is not explicitly precluded. This MIB instruments attributes common to all internet hosts including, for example, both personal computers and systems that run variants of Unix.

### Table of Contents

1. The Network Management Framework .....	2
2. Host Resources MIB .....	3
3. Definitions .....	3
4.1 Textual Conventions .....	3
4.2 The Host Resources System Group .....	5
4.3 The Host Resources Storage Group .....	6
4.4 The Host Resources Device Group .....	10
4.5 The Host Resources Running Software Group .....	25
4.6 The Host Resources Running Software Performance Group .....	27
4.7 The Host Resources Installed Software Group .....	29
5. References .....	31
6. Acknowledgments .....	32
7. Security Considerations .....	32
8. Authors' Addresses .....	33

## 1. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. They are:

STD 16, RFC 1155 [1] which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management. STD 16, RFC 1212 [2] defines a more concise description mechanism, which is wholly consistent with the SMI.

STD 17, RFC 1213 [3] which defines MIB-II, the core set of managed objects for the Internet suite of protocols.

STD 15, RFC 1157 [4] which defines the SNMP, the protocol used for network access to managed objects.

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

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Within a given MIB module, objects are defined using STD 16, RFC 1212's OBJECT-TYPE macro. At a minimum, each object has a name, a syntax, an access-level, and an implementation-status.

The name is an object identifier, an administratively assigned name, which specifies an object type. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the object descriptor, to also refer to the object type.

The syntax of an object type defines the abstract data structure corresponding to that object type. The ASN.1[5] language is used for this purpose. However, RFC 1155 purposely restricts the ASN.1 constructs which may be used. These restrictions are explicitly made for simplicity.

The access-level of an object type defines whether it makes "protocol sense" to read and/or write the value of an instance of the object type. (This access-level is independent of any administrative authorization policy.)

The implementation-status of an object type indicates whether the object is mandatory, optional, obsolete, or deprecated.

## 2. Host Resources MIB

The Host Resources MIB defines a uniform set of objects useful for the management of host computers. Host computers are independent of the operating system, network services, or any software application.

The Host Resources MIB defines objects which are common across many computer system architectures.

In addition, there are objects in MIB-II [3] which also provide host management functionality. Implementation of the System and Interfaces groups is mandatory for implementors of the Host Resources MIB.

## 3. Definitions

```
HOST-RESOURCES-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    OBJECT-TYPE                FROM RFC-1212
    DisplayString                FROM RFC1213-MIB
    TimeTicks,
    Counter, Gauge              FROM RFC1155-SMI;
```

```
host      OBJECT IDENTIFIER ::= { mib-2 25 }
```

```
hrSystem      OBJECT IDENTIFIER ::= { host 1 }
hrStorage     OBJECT IDENTIFIER ::= { host 2 }
hrDevice      OBJECT IDENTIFIER ::= { host 3 }
hrSWRun       OBJECT IDENTIFIER ::= { host 4 }
hrSWRunPerf   OBJECT IDENTIFIER ::= { host 5 }
hrSWInstalled OBJECT IDENTIFIER ::= { host 6 }
```

```
-- textual conventions
```

```
-- a truth value
```

```
Boolean ::= INTEGER { true(1), false(2) }
```

```
-- memory size, expressed in units of 1024bytes
```

```
KBytes ::= INTEGER (0..2147483647)
```

```
-- This textual convention is intended to identify the manufacturer,
-- model, and version of a specific hardware or software product.
-- It is suggested that these OBJECT IDENTIFIERS are allocated such
-- that all products from a particular manufacturer are registered
-- under a subtree distinct to that manufacturer. In addition, all
```

```
-- versions of a product should be registered under a subtree
-- distinct to that product.  With this strategy, a management
-- station may uniquely determine the manufacturer and/or model of a
-- product whose productID is unknown to the management station.
-- Objects of this type may be useful for inventory purposes or for
-- automatically detecting incompatibilities or version mismatches
-- between various hardware and software components on a system.
```

```
ProductID ::= OBJECT IDENTIFIER
```

```
-- unknownProduct will be used for any unknown ProductID
```

```
-- unknownProduct OBJECT IDENTIFIER ::= { 0 0 }
```

```
-- For example, the product ID for the ACME 4860 66MHz clock doubled
-- processor might be:
```

```
-- enterprises.acme.acmeProcessors.a4860DX2.MHz66
```

```
-- A software product might be registered as:
```

```
-- enterprises.acme.acmeOperatingSystems.acmeDOS.six(6).one(1)
```

```
DateAndTime ::= OCTET STRING (SIZE (8 | 11))
```

```
--      A date-time specification for the local time of day.
```

```
--      This data type is intended to provide a consistent
--      method of reporting date information.
```

field	octets	contents	range
1	1-2	year (in network byte order)	0..65536
2	3	month	1..12
3	4	day	1..31
4	5	hour	0..23
5	6	minutes	0..59
6	7	seconds (use 60 for leap-second)	0..60
7	8	deci-seconds	0..9
8	9	direction from UTC (in ascii notation)	"+" / "-"
9	10	hours from UTC	0..11
10	11	minutes from UTC	0..59

```
--      Note that if only local time is known, then
--      timezone information (fields 8-10) is not present.
```

```
InternationalDisplayString ::= OCTET STRING
```

```
-- This data type is used to model textual information in some
-- character set.  A network management station should use a local
-- algorithm to determine which character set is in use and how it
-- should be displayed.  Note that this character set may be encoded
-- with more than one octet per symbol, but will most often be NVT
```

```
-- ASCII.

-- The Host Resources System Group
--
-- Implementation of this group is mandatory for all host systems.
hrSystemUptime OBJECT-TYPE
    SYNTAX TimeTicks
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The amount of time since this host was last
        initialized. Note that this is different from
        sysUpTime in MIB-II [3] because sysUpTime is the
        uptime of the network management portion of the
        system."
    ::= { hrSystem 1 }

hrSystemDate OBJECT-TYPE
    SYNTAX DateAndTime
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The host's notion of the local date and time of
        day."
    ::= { hrSystem 2 }

hrSystemInitialLoadDevice OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The index of the hrDeviceEntry for the device from
        which this host is configured to load its initial
        operating system configuration."
    ::= { hrSystem 3 }

hrSystemInitialLoadParameters OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE (0..128))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object contains the parameters (e.g. a
        pathname and parameter) supplied to the load device
        when requesting the initial operating system
        configuration from that device."
    ::= { hrSystem 4 }
```

**hrSystemNumUsers OBJECT-TYPE**

SYNTAX Gauge

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of user sessions for which this host is storing state information. A session is a collection of processes requiring a single act of user authentication and possibly subject to collective job control."

::= { hrSystem 5 }

**hrSystemProcesses OBJECT-TYPE**

SYNTAX Gauge

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of process contexts currently loaded or running on this system."

::= { hrSystem 6 }

**hrSystemMaxProcesses OBJECT-TYPE**

SYNTAX INTEGER (0..2147483647)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The maximum number of process contexts this system can support. If there is no fixed maximum, the value should be zero. On systems that have a fixed maximum, this object can help diagnose failures that occur when this maximum is reached."

::= { hrSystem 7 }

-- The Host Resources Storage Group

--

-- Implementation of this group is mandatory for all host systems.

-- Registration for some storage types, for use with hrStorageType

hrStorageTypes OBJECT IDENTIFIER ::= { hrStorage 1 }

hrStorageOther OBJECT IDENTIFIER ::= { hrStorageTypes 1 }

hrStorageRam OBJECT IDENTIFIER ::= { hrStorageTypes 2 }

-- hrStorageVirtualMemory is temporary storage of swapped

-- or paged memory

hrStorageVirtualMemory OBJECT IDENTIFIER ::= { hrStorageTypes 3 }

hrStorageFixedDisk OBJECT IDENTIFIER ::= { hrStorageTypes 4 }

hrStorageRemovableDisk OBJECT IDENTIFIER ::= { hrStorageTypes 5 }

hrStorageFloppyDisk OBJECT IDENTIFIER ::= { hrStorageTypes 6 }

```
hrStorageCompactDisc    OBJECT IDENTIFIER ::= { hrStorageTypes 7 }
hrStorageRamDisk        OBJECT IDENTIFIER ::= { hrStorageTypes 8 }
```

hrMemorySize OBJECT-TYPE

SYNTAX KBytes

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The amount of physical main memory contained by  
the host."

::= { hrStorage 2 }

hrStorageTable OBJECT-TYPE

SYNTAX SEQUENCE OF HrStorageEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The (conceptual) table of logical storage areas on  
the host."

An entry shall be placed in the storage table for each logical area of storage that is allocated and has fixed resource limits. The amount of storage represented in an entry is the amount actually usable by the requesting entity, and excludes loss due to formatting or file system reference information.

These entries are associated with logical storage areas, as might be seen by an application, rather than physical storage entities which are typically seen by an operating system. Storage such as tapes and floppies without file systems on them are typically not allocated in chunks by the operating system to requesting applications, and therefore shouldn't appear in this table. Examples of valid storage for this table include disk partitions, file systems, ram (for some architectures this is further segmented into regular memory, extended memory, and so on), backing store for virtual memory ('swap space').

This table is intended to be a useful diagnostic for 'out of memory' and 'out of buffers' types of failures. In addition, it can be a useful performance monitoring tool for tracking memory, disk, or buffer usage."

```

 ::= { hrStorage 3 }

hrStorageEntry OBJECT-TYPE
    SYNTAX HrStorageEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A (conceptual) entry for one logical storage area
        on the host.  As an example, an instance of the
        hrStorageType object might be named
        hrStorageType.3"
    INDEX { hrStorageIndex }
    ::= { hrStorageTable 1 }

HrStorageEntry ::= SEQUENCE {
    hrStorageIndex          INTEGER,
    hrStorageType           OBJECT IDENTIFIER,
    hrStorageDescr          DisplayString,
    hrStorageAllocationUnits INTEGER,
    hrStorageSize           INTEGER,
    hrStorageUsed           INTEGER,
    hrStorageAllocationFailures Counter
}

hrStorageIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each logical storage area
        contained by the host."
    ::= { hrStorageEntry 1 }

hrStorageType OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The type of storage represented by this entry."
    ::= { hrStorageEntry 2 }

hrStorageDescr OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A description of the type and instance of the
        storage described by this entry."

```



```
::= { hrStorageEntry 3 }
```

hrStorageAllocationUnits OBJECT-TYPE

SYNTAX INTEGER (1..2147483647)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The size, in bytes, of the data objects allocated from this pool. If this entry is monitoring sectors, blocks, buffers, or packets, for example, this number will commonly be greater than one. Otherwise this number will typically be one."

```
::= { hrStorageEntry 4 }
```

hrStorageSize OBJECT-TYPE

SYNTAX INTEGER (0..2147483647)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The size of the storage represented by this entry, in units of hrStorageAllocationUnits."

```
::= { hrStorageEntry 5 }
```

hrStorageUsed OBJECT-TYPE

SYNTAX INTEGER (0..2147483647)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The amount of the storage represented by this entry that is allocated, in units of hrStorageAllocationUnits."

```
::= { hrStorageEntry 6 }
```

hrStorageAllocationFailures OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of requests for storage represented by this entry that could not be honored due to not enough storage. It should be noted that as this object has a SYNTAX of Counter, that it does not have a defined initial value. However, it is recommended that this object be initialized to zero."

```
::= { hrStorageEntry 7 }
```

```
-- The Host Resources Device Group
--
-- Implementation of this group is mandatory for all host systems.
--
-- The device group is useful for identifying and diagnosing the
-- devices on a system. The hrDeviceTable contains common
-- information for any type of device. In addition, some devices
-- have device-specific tables for more detailed information. More
-- such tables may be defined in the future for other device types.
```

```
-- Registration for some device types, for use with hrDeviceType
hrDeviceTypes          OBJECT IDENTIFIER ::= { hrDevice 1 }
```

```
hrDeviceOther          OBJECT IDENTIFIER ::= { hrDeviceTypes 1 }
hrDeviceUnknown        OBJECT IDENTIFIER ::= { hrDeviceTypes 2 }
hrDeviceProcessor      OBJECT IDENTIFIER ::= { hrDeviceTypes 3 }
hrDeviceNetwork        OBJECT IDENTIFIER ::= { hrDeviceTypes 4 }
hrDevicePrinter        OBJECT IDENTIFIER ::= { hrDeviceTypes 5 }
hrDeviceDiskStorage    OBJECT IDENTIFIER ::= { hrDeviceTypes 6 }
hrDeviceVideo          OBJECT IDENTIFIER ::= { hrDeviceTypes 10 }
hrDeviceAudio          OBJECT IDENTIFIER ::= { hrDeviceTypes 11 }
hrDeviceCoprocessor    OBJECT IDENTIFIER ::= { hrDeviceTypes 12 }
hrDeviceKeyboard       OBJECT IDENTIFIER ::= { hrDeviceTypes 13 }
hrDeviceModem          OBJECT IDENTIFIER ::= { hrDeviceTypes 14 }
hrDeviceParallelPort   OBJECT IDENTIFIER ::= { hrDeviceTypes 15 }
hrDevicePointing       OBJECT IDENTIFIER ::= { hrDeviceTypes 16 }
hrDeviceSerialPort     OBJECT IDENTIFIER ::= { hrDeviceTypes 17 }
hrDeviceTape           OBJECT IDENTIFIER ::= { hrDeviceTypes 18 }
hrDeviceClock          OBJECT IDENTIFIER ::= { hrDeviceTypes 19 }
hrDeviceVolatileMemory OBJECT IDENTIFIER ::= { hrDeviceTypes 20 }
hrDeviceNonVolatileMemory OBJECT IDENTIFIER ::= { hrDeviceTypes 21 }
```

```
hrDeviceTable OBJECT-TYPE
```

```
    SYNTAX SEQUENCE OF HrDeviceEntry
```

```
    ACCESS not-accessible
```

```
    STATUS mandatory
```

```
    DESCRIPTION
```

```
        "The (conceptual) table of devices contained by the
        host."
```

```
    ::= { hrDevice 2 }
```

```
hrDeviceEntry OBJECT-TYPE
```

```
    SYNTAX HrDeviceEntry
```

```
    ACCESS not-accessible
```

```
    STATUS mandatory
```

```
    DESCRIPTION
```

```
        "A (conceptual) entry for one device contained by
```

the host. As an example, an instance of the  
hrDeviceType object might be named hrDeviceType.3"

```
INDEX { hrDeviceIndex }
::= { hrDeviceTable 1 }
```

```
HrDeviceEntry ::= SEQUENCE {
    hrDeviceIndex          INTEGER,
    hrDeviceType           OBJECT IDENTIFIER,
    hrDeviceDescr          DisplayString,
    hrDeviceID             ProductID,
    hrDeviceStatus         INTEGER,
    hrDeviceErrors         Counter
}
```

```
hrDeviceIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each device contained by the
        host. The value for each device must remain
        constant at least from one re-initialization of the
        agent to the next re-initialization."
    ::= { hrDeviceEntry 1 }
```

```
hrDeviceType OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "An indication of the type of device.

        If this value is 'hrDeviceProcessor { hrDeviceTypes
        3 }' then an entry exists in the hrProcessorTable
        which corresponds to this device.

        If this value is 'hrDeviceNetwork { hrDeviceTypes 4
        }', then an entry exists in the hrNetworkTable
        which corresponds to this device.

        If this value is 'hrDevicePrinter { hrDeviceTypes 5
        }', then an entry exists in the hrPrinterTable
        which corresponds to this device.

        If this value is 'hrDeviceDiskStorage {
        hrDeviceTypes 6 }', then an entry exists in the
        hrDiskStorageTable which corresponds to this
        device."
```

```
 ::= { hrDeviceEntry 2 }

hrDeviceDescr OBJECT-TYPE
    SYNTAX DisplayString (SIZE (0..64))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A textual description of this device, including
         the device's manufacturer and revision, and
         optionally, its serial number."
    ::= { hrDeviceEntry 3 }

hrDeviceID OBJECT-TYPE
    SYNTAX ProductID
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The product ID for this device."
    ::= { hrDeviceEntry 4 }

hrDeviceStatus OBJECT-TYPE
    SYNTAX INTEGER {
        unknown(1),
        running(2),
        warning(3),
        testing(4),
        down(5)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The current operational state of the device
         described by this row of the table. A value
         unknown(1) indicates that the current state of the
         device is unknown. running(2) indicates that the
         device is up and running and that no unusual error
         conditions are known. The warning(3) state
         indicates that agent has been informed of an
         unusual error condition by the operational software
         (e.g., a disk device driver) but that the device is
         still 'operational'. An example would be high
         number of soft errors on a disk. A value of
         testing(4), indicates that the device is not
         available for use because it is in the testing
         state. The state of down(5) is used only when the
         agent has been informed that the device is not
         available for any use."
    ::= { hrDeviceEntry 5 }
```

**hrDeviceErrors OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of errors detected on this device. It should be noted that as this object has a SYNTAX of Counter, that it does not have a defined initial value. However, it is recommended that this object be initialized to zero."

::= { hrDeviceEntry 6 }

**hrProcessorTable OBJECT-TYPE**

SYNTAX SEQUENCE OF HrProcessorEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The (conceptual) table of processors contained by the host.

Note that this table is potentially sparse: a (conceptual) entry exists only if the correspondent value of the hrDeviceType object is 'hrDeviceProcessor'."

::= { hrDevice 3 }

**hrProcessorEntry OBJECT-TYPE**

SYNTAX HrProcessorEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A (conceptual) entry for one processor contained by the host. The hrDeviceIndex in the index represents the entry in the hrDeviceTable that corresponds to the hrProcessorEntry.

As an example of how objects in this table are named, an instance of the hrProcessorFrwID object might be named hrProcessorFrwID.3"

INDEX { hrDeviceIndex }

::= { hrProcessorTable 1 }

**HrProcessorEntry ::= SEQUENCE {**

hrProcessorFrwID

ProductID,

hrProcessorLoad

INTEGER

**}****hrProcessorFrwID OBJECT-TYPE**

```
SYNTAX ProductID
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "The product ID of the firmware associated with the
    processor."
::= { hrProcessorEntry 1 }

hrProcessorLoad OBJECT-TYPE
SYNTAX INTEGER (0..100)
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "The average, over the last minute, of the
    percentage of time that this processor was not
    idle."
::= { hrProcessorEntry 2 }

hrNetworkTable OBJECT-TYPE
SYNTAX SEQUENCE OF HrNetworkEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
    "The (conceptual) table of network devices
    contained by the host.

    Note that this table is potentially sparse: a
    (conceptual) entry exists only if the correspondent
    value of the hrDeviceType object is
    'hrDeviceNetwork'."
::= { hrDevice 4 }

hrNetworkEntry OBJECT-TYPE
SYNTAX HrNetworkEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
    "A (conceptual) entry for one network device
    contained by the host. The hrDeviceIndex in the
    index represents the entry in the hrDeviceTable
    that corresponds to the hrNetworkEntry.

    As an example of how objects in this table are
    named, an instance of the hrNetworkIfIndex object
    might be named hrNetworkIfIndex.3"
INDEX { hrDeviceIndex }
::= { hrNetworkTable 1 }
```

```
HrNetworkEntry ::= SEQUENCE {
    hrNetworkIfIndex    INTEGER
}

hrNetworkIfIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The value of ifIndex which corresponds to this
        network device."
    ::= { hrNetworkEntry 1 }

hrPrinterTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HrPrinterEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The (conceptual) table of printers local to the
        host.

        Note that this table is potentially sparse: a
        (conceptual) entry exists only if the correspondent
        value of the hrDeviceType object is
        'hrDevicePrinter'."
    ::= { hrDevice 5 }

hrPrinterEntry OBJECT-TYPE
    SYNTAX HrPrinterEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A (conceptual) entry for one printer local to the
        host. The hrDeviceIndex in the index represents
        the entry in the hrDeviceTable that corresponds to
        the hrPrinterEntry.

        As an example of how objects in this table are
        named, an instance of the hrPrinterStatus object
        might be named hrPrinterStatus.3"
    INDEX { hrDeviceIndex }
    ::= { hrPrinterTable 1 }

HrPrinterEntry ::= SEQUENCE {
    hrPrinterStatus          INTEGER,
    hrPrinterDetectedErrorState OCTET STRING
}
```

**hrPrinterStatus OBJECT-TYPE**

```
SYNTAX INTEGER {
    other(1),
    unknown(2),
    idle(3),
    printing(4),
    warmup(5)
}
```

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The current status of this printer device. When in the idle(1), printing(2), or warmup(3) state, the corresponding hrDeviceStatus should be running(2) or warning(3). When in the unknown state, the corresponding hrDeviceStatus should be unknown(1)."

```
::= { hrPrinterEntry 1 }
```

**hrPrinterDetectedErrorState OBJECT-TYPE**

```
SYNTAX OCTET STRING
```

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This object represents any error conditions detected by the printer. The error conditions are encoded as bits in an octet string, with the following definitions:

Condition	Bit #	hrDeviceStatus
lowPaper	0	warning(3)
noPaper	1	down(5)
lowToner	2	warning(3)
noToner	3	down(5)
doorOpen	4	down(5)
jammed	5	down(5)
offline	6	down(5)
serviceRequested	7	warning(3)

If multiple conditions are currently detected and the hrDeviceStatus would not otherwise be unknown(1) or testing(4), the hrDeviceStatus shall correspond to the worst state of those indicated, where down(5) is worse than warning(3) which is worse than running(2).

Bits are numbered starting with the most



significant bit of the first byte being bit 0, the least significant bit of the first byte being bit 7, the most significant bit of the second byte being bit 8, and so on. A one bit encodes that the condition was detected, while a zero bit encodes that the condition was not detected.

This object is useful for alerting an operator to specific warning or error conditions that may occur, especially those requiring human intervention."

```
::= { hrPrinterEntry 2 }
```

hrDiskStorageTable OBJECT-TYPE

SYNTAX SEQUENCE OF HrDiskStorageEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The (conceptual) table of long-term storage devices contained by the host. In particular, disk devices accessed remotely over a network are not included here.

Note that this table is potentially sparse: a (conceptual) entry exists only if the correspondent value of the hrDeviceType object is 'hrDeviceDiskStorage'."

```
::= { hrDevice 6 }
```

hrDiskStorageEntry OBJECT-TYPE

SYNTAX HrDiskStorageEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A (conceptual) entry for one long-term storage device contained by the host. The hrDeviceIndex in the index represents the entry in the hrDeviceTable that corresponds to the hrDiskStorageEntry. As an example, an instance of the hrDiskStorageCapacity object might be named hrDiskStorageCapacity.3"

INDEX { hrDeviceIndex }

```
::= { hrDiskStorageTable 1 }
```

HrDiskStorageEntry ::= SEQUENCE {

hrDiskStorageAccess	INTEGER,
hrDiskStorageMedia	INTEGER,
hrDiskStorageRemoveble	Boolean,
hrDiskStorageCapacity	KBytes

```
    }

hrDiskStorageAccess OBJECT-TYPE
    SYNTAX INTEGER {
        readWrite(1),
        readOnly(2)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "An indication if this long-term storage device is
        readable and writable or only readable. This
        should reflect the media type, any write-protect
        mechanism, and any device configuration that
        affects the entire device."
    ::= { hrDiskStorageEntry 1 }

hrDiskStorageMedia OBJECT-TYPE
    SYNTAX INTEGER {
        other(1),
        unknown(2),
        hardDisk(3),
        floppyDisk(4),
        opticalDiskROM(5),
        opticalDiskWORM(6),      -- Write Once Read Many
        opticalDiskRW(7),
        ramDisk(8)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "An indication of the type of media used in this
        long-term storage device."
    ::= { hrDiskStorageEntry 2 }

hrDiskStorageRemoveble OBJECT-TYPE
    SYNTAX Boolean
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Denotes whether or not the disk media may be
        removed from the drive."
    ::= { hrDiskStorageEntry 3 }

hrDiskStorageCapacity OBJECT-TYPE
    SYNTAX KBytes
    ACCESS read-only
    STATUS mandatory
```

## DESCRIPTION

"The total size for this long-term storage device."  
 ::= { hrDiskStorageEntry 4 }

## hrPartitionTable OBJECT-TYPE

SYNTAX SEQUENCE OF HrPartitionEntry

ACCESS not-accessible

STATUS mandatory

## DESCRIPTION

"The (conceptual) table of partitions for long-term storage devices contained by the host. In particular, partitions accessed remotely over a network are not included here."

::= { hrDevice 7 }

## hrPartitionEntry OBJECT-TYPE

SYNTAX HrPartitionEntry

ACCESS not-accessible

STATUS mandatory

## DESCRIPTION

"A (conceptual) entry for one partition. The hrDeviceIndex in the index represents the entry in the hrDeviceTable that corresponds to the hrPartitionEntry."

As an example of how objects in this table are named, an instance of the hrPartitionSize object might be named hrPartitionSize.3.1"

INDEX { hrDeviceIndex, hrPartitionIndex }

::= { hrPartitionTable 1 }

HrPartitionEntry ::= SEQUENCE {

hrPartitionIndex

INTEGER,

hrPartitionLabel

InternationalDisplayString,

hrPartitionID

OCTET STRING,

hrPartitionSize

KBytes,

hrPartitionFSIndex

INTEGER

}

## hrPartitionIndex OBJECT-TYPE

SYNTAX INTEGER (1..2147483647)

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"A unique value for each partition on this long-term storage device. The value for each long-term storage device must remain constant at least from one re-initialization of the agent to the next re-

```

        initialization."
    ::= { hrPartitionEntry 1 }

hrPartitionLabel OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE (0..128))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A textual description of this partition."
    ::= { hrPartitionEntry 2 }

hrPartitionID OBJECT-TYPE
    SYNTAX OCTET STRING
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A descriptor which uniquely represents this
        partition to the responsible operating system.  On
        some systems, this might take on a binary
        representation."
    ::= { hrPartitionEntry 3 }

hrPartitionSize OBJECT-TYPE
    SYNTAX KBytes
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The size of this partition."
    ::= { hrPartitionEntry 4 }

hrPartitionFSIndex OBJECT-TYPE
    SYNTAX INTEGER (0..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The index of the file system mounted on this
        partition.  If no file system is mounted on this
        partition, then this value shall be zero.  Note
        that multiple partitions may point to one file
        system, denoting that that file system resides on
        those partitions.  Multiple file systems may not
        reside on one partition."
    ::= { hrPartitionEntry 5 }

-- The File System Table
hrFSTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HrFSEntry

```

ACCESS not-accessible  
 STATUS mandatory  
 DESCRIPTION

"The (conceptual) table of file systems local to this host or remotely mounted from a file server. File systems that are in only one user's environment on a multi-user system will not be included in this table."

::= { hrDevice 8 }

hrFSEntry OBJECT-TYPE

SYNTAX HrFSEntry  
 ACCESS not-accessible  
 STATUS mandatory  
 DESCRIPTION

"A (conceptual) entry for one file system local to this host or remotely mounted from a file server. File systems that are in only one user's environment on a multi-user system will not be included in this table."

As an example of how objects in this table are named, an instance of the hrFSMountPoint object might be named hrFSMountPoint.3"

INDEX { hrFSIndex }  
 ::= { hrFSIndex 1 }

-- Registration for some popular File System types,  
 -- for use with hrFSIndex.

hrFSTypes OBJECT IDENTIFIER ::= { hrDevice 9 }

hrFSOther OBJECT IDENTIFIER ::= { hrFSTypes 1 }

hrFSUnknown OBJECT IDENTIFIER ::= { hrFSTypes 2 }

hrFSBerkeleyFFS OBJECT IDENTIFIER ::= { hrFSTypes 3 }

hrFSSys5FS OBJECT IDENTIFIER ::= { hrFSTypes 4 }

-- DOS

hrFSFat OBJECT IDENTIFIER ::= { hrFSTypes 5 }

-- OS/2 High Performance File System

hrFSHPFS OBJECT IDENTIFIER ::= { hrFSTypes 6 }

-- Macintosh Hierarchical File System

hrFSHFS OBJECT IDENTIFIER ::= { hrFSTypes 7 }

-- Macintosh File System

hrFSMFS OBJECT IDENTIFIER ::= { hrFSTypes 8 }

-- Windows NT

hrFSNTFS OBJECT IDENTIFIER ::= { hrFSTypes 9 }

```

hrFSVNode          OBJECT IDENTIFIER ::= { hrFSTypes 10 }
hrFSJournalled     OBJECT IDENTIFIER ::= { hrFSTypes 11 }
-- CD File systems
hrFSiso9660        OBJECT IDENTIFIER ::= { hrFSTypes 12 }
hrFSRockRidge      OBJECT IDENTIFIER ::= { hrFSTypes 13 }

hrFSNFS            OBJECT IDENTIFIER ::= { hrFSTypes 14 }
hrFSNetware        OBJECT IDENTIFIER ::= { hrFSTypes 15 }
-- Andrew File System
hrFSAFS           OBJECT IDENTIFIER ::= { hrFSTypes 16 }
-- OSF DCE Distributed File System
hrFSDFS           OBJECT IDENTIFIER ::= { hrFSTypes 17 }
hrFSAppleshare    OBJECT IDENTIFIER ::= { hrFSTypes 18 }
hrFSRFS           OBJECT IDENTIFIER ::= { hrFSTypes 19 }
-- Data General
hrFSDGCFs         OBJECT IDENTIFIER ::= { hrFSTypes 20 }
-- SVR4 Boot File System
hrFSBFS           OBJECT IDENTIFIER ::= { hrFSTypes 21 }

```

```

HrFSEntry ::= SEQUENCE {
    hrFSIndex          INTEGER,
    hrFSMountPoint     InternationalDisplayString,
    hrFSRemoteMountPoint InternationalDisplayString,
    hrFSType           OBJECT IDENTIFIER,
    hrFSAccess         INTEGER,
    hrFSBootable       Boolean,
    hrFSStorageIndex   INTEGER,
    hrFSLastFullBackupDate DateAndTime,
    hrFSLastPartialBackupDate DateAndTime
}

```

```

hrFSIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each file system local to this
        host. The value for each file system must remain
        constant at least from one re-initialization of
        the agent to the next re-initialization."
    ::= { hrFSEntry 1 }

```

```

hrFSMountPoint OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE(0..128))
    ACCESS read-only
    STATUS mandatory

```

## DESCRIPTION

"The path name of the root of this file system."  
 ::= { hrFSEntry 2 }

## hrFSRemoteMountPoint OBJECT-TYPE

SYNTAX InternationalDisplayString (SIZE(0..128))

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"A description of the name and/or address of the server that this file system is mounted from. This may also include parameters such as the mount point on the remote file system. If this is not a remote file system, this string should have a length of zero."  
 ::= { hrFSEntry 3 }

## hrFSType OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"The value of this object identifies the type of this file system."  
 ::= { hrFSEntry 4 }

## hrFSAccess OBJECT-TYPE

SYNTAX INTEGER {  
 readWrite(1),  
 readOnly(2)  
}

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"An indication if this file system is logically configured by the operating system to be readable and writable or only readable. This does not represent any local access-control policy, except one that is applied to the file system as a whole."  
 ::= { hrFSEntry 5 }

## hrFSBootable OBJECT-TYPE

SYNTAX Boolean

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"A flag indicating whether this file system is bootable."

```
::= { hrFSEntry 6 }
```

hrFSStorageIndex OBJECT-TYPE

SYNTAX INTEGER (0..2147483647)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The index of the hrStorageEntry that represents information about this file system. If there is no such information available, then this value shall be zero. The relevant storage entry will be useful in tracking the percent usage of this file system and diagnosing errors that may occur when it runs out of space."

```
::= { hrFSEntry 7 }
```

hrFSLastFullBackupDate OBJECT-TYPE

SYNTAX DateAndTime

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The last date at which this complete file system was copied to another storage device for backup. This information is useful for ensuring that backups are being performed regularly.

If this information is not known, then this variable shall have the value corresponding to January 1, year 0000, 00:00:00.0, which is encoded as (hex)'00 00 01 01 00 00 00 00'."

```
::= { hrFSEntry 8 }
```

hrFSLastPartialBackupDate OBJECT-TYPE

SYNTAX DateAndTime

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The last date at which a portion of this file system was copied to another storage device for backup. This information is useful for ensuring that backups are being performed regularly.

If this information is not known, then this variable shall have the value corresponding to January 1, year 0000, 00:00:00.0, which is encoded as (hex)'00 00 01 01 00 00 00 00'."

```
::= { hrFSEntry 9 }
```



```
-- The Host Resources Running Software Group
--
-- Implementation of this group is optional.
--
-- The hrSWRunTable contains an entry for each distinct piece of
-- software that is running or loaded into physical or virtual
-- memory in preparation for running. This includes the host's
-- operating system, device drivers, and applications.
```

hrSWOSIndex OBJECT-TYPE

SYNTAX INTEGER (1..2147483647)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The value of the hrSWRunIndex for the  
hrSWRunEntry that represents the primary operating  
system running on this host. This object is  
useful for quickly and uniquely identifying that  
primary operating system."

::= { hrSWRun 1 }

hrSWRunTable OBJECT-TYPE

SYNTAX SEQUENCE OF HrSWRunEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The (conceptual) table of software running on the  
host."

::= { hrSWRun 2 }

hrSWRunEntry OBJECT-TYPE

SYNTAX HrSWRunEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A (conceptual) entry for one piece of software  
running on the host Note that because the installed  
software table only contains information for  
software stored locally on this host, not every  
piece of running software will be found in the  
installed software table. This is true of software  
that was loaded and run from a non-local source,  
such as a network-mounted file system.

As an example of how objects in this table are  
named, an instance of the hrSWRunName object might  
be named hrSWRunName.1287"

INDEX { hrSWRunIndex }

```
 ::= { hrSWRunTable 1 }

HrSWRunEntry ::= SEQUENCE {
    hrSWRunIndex      INTEGER,
    hrSWRunName       InternationalDisplayString,
    hrSWRunID         ProductID,
    hrSWRunPath       InternationalDisplayString,
    hrSWRunParameters InternationalDisplayString,
    hrSWRunType       INTEGER,
    hrSWRunStatus     INTEGER
}

hrSWRunIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each piece of software running
         on the host. Wherever possible, this should be the
         system's native, unique identification number."
    ::= { hrSWRunEntry 1 }

hrSWRunName OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE (0..64))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A textual description of this running piece of
         software, including the manufacturer, revision,
         and the name by which it is commonly known. If
         this software was installed locally, this should be
         the same string as used in the corresponding
         hrSWInstalledName."
    ::= { hrSWRunEntry 2 }

hrSWRunID OBJECT-TYPE
    SYNTAX ProductID
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The product ID of this running piece of software."
    ::= { hrSWRunEntry 3 }

hrSWRunPath OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE(0..128))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
```

```

        "A description of the location on long-term storage
        (e.g. a disk drive) from which this software was
        loaded."
 ::= { hrSWRunEntry 4 }

hrSWRunParameters OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE(0..128))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A description of the parameters supplied to this
        software when it was initially loaded."
 ::= { hrSWRunEntry 5 }

hrSWRunType OBJECT-TYPE
    SYNTAX INTEGER {
        unknown(1),
        operatingSystem(2),
        deviceDriver(3),
        application(4)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The type of this software."
 ::= { hrSWRunEntry 6 }

hrSWRunStatus OBJECT-TYPE
    SYNTAX INTEGER {
        running(1),
        runnable(2),      -- waiting for resource (CPU, memory, IO)
        notRunnable(3),   -- loaded but waiting for event
        invalid(4)        -- not loaded
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The status of this running piece of software.
        Setting this value to invalid(4) shall cause this
        software to stop running and to be unloaded."
 ::= { hrSWRunEntry 7 }

-- The Host Resources Running Software Performance Group
-- Implementation of this group is optional.
--
-- The hrSWRunPerfTable contains an entry corresponding to
-- each entry in the hrSWRunTable.

```

```

hrSWRunPerfTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HrSWRunPerfEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The (conceptual) table of running software
        performance metrics."
    ::= { hrSWRunPerf 1 }

hrSWRunPerfEntry OBJECT-TYPE
    SYNTAX HrSWRunPerfEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A (conceptual) entry containing software
        performance metrics. As an example, an instance
        of the hrSWRunPerfCPU object might be named
        hrSWRunPerfCPU.1287"
    INDEX { hrSWRunIndex } -- This table augments information in
                           -- the hrSWRunTable.
    ::= { hrSWRunPerfTable 1 }

HrSWRunPerfEntry ::= SEQUENCE {
    hrSWRunPerfCPU      INTEGER,
    hrSWRunPerfMem      KBytes
}

hrSWRunPerfCPU OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of centi-seconds of the total system's
        CPU resources consumed by this process. Note that
        on a multi-processor system, this value may
        increment by more than one centi-second in one
        centi-second of real (wall clock) time."
    ::= { hrSWRunPerfEntry 1 }

hrSWRunPerfMem OBJECT-TYPE
    SYNTAX KBytes
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total amount of real system memory allocated
        to this process."
    ::= { hrSWRunPerfEntry 2 }

```

```
-- The Host Resources Installed Software Group
--
-- Implementation of this group is optional.
--
-- The hrSWInstalledTable contains an entry for each piece
-- of software installed in long-term storage (e.g. a disk
-- drive) locally on this host. Note that this does not
-- include software loadable remotely from a network
-- server.
--
-- This table is useful for identifying and inventorying
-- software on a host and for diagnosing incompatibility
-- and version mismatch problems between various pieces
-- of hardware and software.

hrSWInstalledLastChange OBJECT-TYPE
    SYNTAX TimeTicks
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The value of sysUpTime when an entry in the
        hrSWInstalledTable was last added, renamed, or
        deleted. Because this table is likely to contain
        many entries, polling of this object allows a
        management station to determine when re-downloading
        of the table might be useful."
    ::= { hrSWInstalled 1 }

hrSWInstalledLastUpdateTime OBJECT-TYPE
    SYNTAX TimeTicks
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The value of sysUpTime when the hrSWInstalledTable
        was last completely updated. Because caching of
        this data will be a popular implementation
        strategy, retrieval of this object allows a
        management station to obtain a guarantee that no
        data in this table is older than the indicated
        time."
    ::= { hrSWInstalled 2 }

hrSWInstalledTable OBJECT-TYPE
    SYNTAX SEQUENCE OF HrSWInstalledEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The (conceptual) table of software installed on
```

```

        this host."
    ::= { hrSWInstalled 3 }

hrSWInstalledEntry OBJECT-TYPE
    SYNTAX HrSWInstalledEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A (conceptual) entry for a piece of software
        installed on this host.

        As an example of how objects in this table are
        named, an instance of the hrSWInstalledName object
        might be named hrSWInstalledName.96"
    INDEX { hrSWInstalledIndex }
    ::= { hrSWInstalledTable 1 }

HrSWInstalledEntry ::= SEQUENCE {
    hrSWInstalledIndex      INTEGER,
    hrSWInstalledName       InternationalDisplayString,
    hrSWInstalledID         ProductID,
    hrSWInstalledType       INTEGER,
    hrSWInstalledDate       DateAndTime
}

hrSWInstalledIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each piece of software
        installed on the host. This value shall be in the
        range from 1 to the number of pieces of software
        installed on the host."
    ::= { hrSWInstalledEntry 1 }

hrSWInstalledName OBJECT-TYPE
    SYNTAX InternationalDisplayString (SIZE (0..64))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A textual description of this installed piece of
        software, including the manufacturer, revision, the
        name by which it is commonly known, and optionally,
        its serial number."
    ::= { hrSWInstalledEntry 2 }

hrSWInstalledID OBJECT-TYPE

```

```
SYNTAX ProductID
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "The product ID of this installed piece of
    software."
::= { hrSWInstalledEntry 3 }

hrSWInstalledType OBJECT-TYPE
    SYNTAX INTEGER {
        unknown(1),
        operatingSystem(2),
        deviceDriver(3),
        application(4)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The type of this software."
    ::= { hrSWInstalledEntry 4 }

hrSWInstalledDate OBJECT-TYPE
    SYNTAX DateAndTime
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The last-modification date of this application as
        it would appear in a directory listing."
    ::= { hrSWInstalledEntry 5 }

END
```

## 5. References

- [1] 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.
- [2] Rose, M., and K. McCloghrie, Editors, "Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.
- [3] McCloghrie K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets", STD 17, RFC 1213, Performance Systems International, March 1991.

- [4] 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.
- [5] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization. International Standard 8824, (December, 1987).

## 6. Acknowledgments

This document was produced by the Host Resources MIB working group.

In addition, the authors gratefully acknowledge the comments of the following individuals:

Amatzia Ben-Artzi	NetManage
Steve Bostock	Novell
Stephen Bush	GE Information Systems
Jeff Case	SNMP Research
Chuck Davin	Bellcore
Ray Edgerton	Bell Atlantic
Mike Erlinger	Aerospace Corporation
Tim Farley	Magee Enterprises
Mark Kepke	Hewlett-Packard
Bobby Krupczak	Georgia Tech
Cheryl Krupczak	Georgia Tech
Keith McCloghrie	Hughes Lan Systems
Greg Minshall	Novell
Dave Perkins	Synoptics
Ed Reeder	Objective Systems Integrators
Mike Ritter	Apple Computer
Marshall Rose	Dover Beach Consulting
Jon Saperia	DEC
Rodney Thayer	Sable Technology
Kaj Tesink	Bellcore
Dean Throop	Data General

## 7. Security Considerations

Security issues are not discussed in this memo.



## 8. Authors' Addresses

Pete Grillo  
10915 NW Lost Park Drive  
Portland OR 97229

Phone: +1 503 526 9766  
EMail: pl0143@mail.psi.net

Steven Waldbusser  
Carnegie Mellon University  
4910 Forbes Ave.  
Pittsburgh, PA 15213

Phone: +1 412 268 6628  
Fax: +1 412 268 4987  
EMail: waldbusser@cmu.edu