Working with SNMP OIDs for Patton RAS Products

Overview: SNMP, MIBs, and OIDs

SNMP is used to configure and monitor the remote access server. There are numerous third party software applications available that are capable of using SNMP to control the access server.

To interact with the access server, these network management applications need:

- A community string which defines the level of access to the RAS MIB: *Read-only*, or *Read-and-write* access
- An object identifier for the specific parameter you want to view or modify.

Object Identifiers (OIDs) comprise of a series of integers separated by dots that identify a specific parameter (for example, `1.3.6.1.4.1.1768.5.25`)

The series of integers are built by traversing down a tree structure. As a decision is made at each branch of the tree structure, a new integer is added to the object identifier. When the last branch is selected – taking you to the desired parameter - the OID is completed.

The MIB tree for the remote access server is comprised of two sections: The Internet standards section, which is MIBs that deal with Internet standards such as IP, ICMP, Frame-Relay and Ethernet. It contains parameters that could potentially be on any machine that implements these features. The private Patton MIB, which contains MIB variables specific to Patton products.

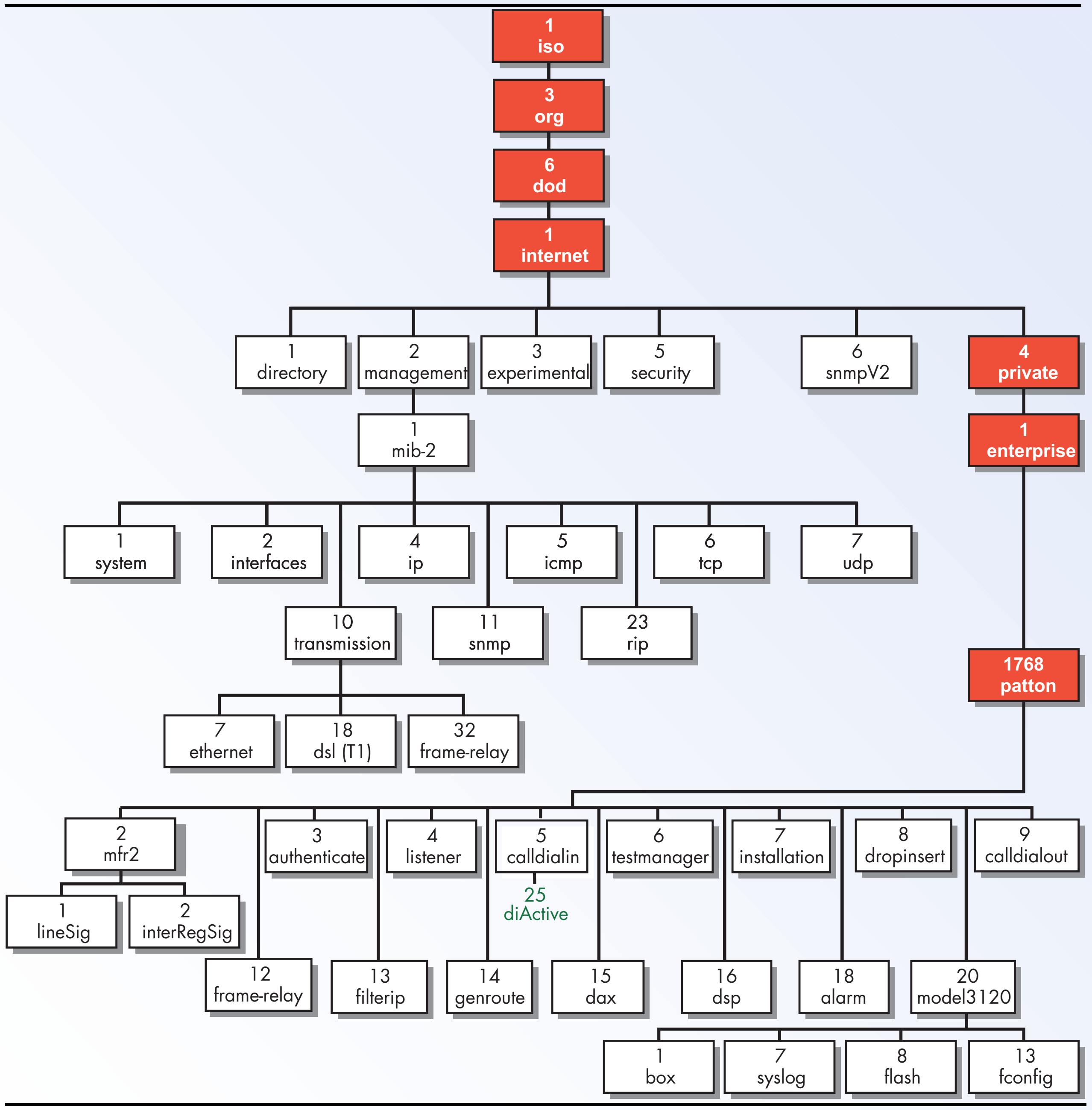
Gathering Information

Patton vendor-specific MIB information

A diagram of the MIB tree is located on the following page, and also at http://www.patton.com/images/tech_support/mib2960.gif.

The MIB files defining the Patton vendor-specific MIB tree are located in the links on the SNMP web page. Corporate MIB defines the overall structure of the RAS MIB. Enterprise MIB defines the MIB variables applicable to a group of Patton products Product MIB defines the MIB variables specific to the particular product

Patton RAS MIB Diagram



Industry-defined MIB information

We also support MIB variables defined in the following RFCs:

1155	- Structure and Identification of Management Information for TCP/IP-based Internets
1213	- Management Information Base for Network Management of TCP/IP-based Internets: MIB-II
1315	- Management Information Base for Frame Relay DTEs
1389	- RIP Version 2 MIB Extension
1406	- Definitions of Managed Objects for the DS1 and E1 Interface Types
1643	- Definitions of Managed Objects for the Ethernet-like Interface types

RFCs can be found at: http://www.faqs.org/rfcs/

Your community strings:

- Read-only community string is the user password.
- Read/Write community string is the superuser password.

Building OIDs

Example 1: Building the OID for the number of active calls

Step 1: **Finding the SNMP Name**

Every piece of information that can be viewed or modified by SNMP can be found on one of the remote access server's web pages. The first step in finding the SNMP name is locating the information you want on one of its web pages. Once you find its location on the web page, go to the Administrator's Reference Guide and find the description for that web page and corresponding parameter you are interested in monitoring.

In this instance the number of active calls is found on the HOME page. The description of the parameters for the HOME page is located in Chapter 2: Section *Operating Status Variables*. The description for the parameter is as follows:

`Active Calls (diActive)`	This number, ranging from 0 to 120 displays the total number of calls being processed (connecting, online, authenticating and so on) in the access server at the time the HOME page was displayed.

The SNMP name is always in parenthesis next to the Screen Identifier.

Step 2: **Finding the section of the MIB tree in which the SNMP parameter resides**:

If this is an Internet standards MIB variable then you will need to consult the RFCs to identify the OID. Some variables are obviously related to RFCs such as Frame-Relay statistics but others are not. One way of determining if it is private or not is to search the MIB files for the SNMP name. If the name does not exist, the parameter is an Internet standards MIB variable.

Step 3: Finding the branch where the SNMP parameter resides

Most of the MIBs are common to all Patton access server products; therefore the parameter is likely to be found in the Enterprise MIB. Go to the SNMP web page and click on Enterprise MIB and open the file `common.mib`. Search for the SNMP name `diActive`, which maps to Active Calls on the HOME page.

Search for the SNMP name `diActive`, which maps to Active Calls on the HOME page. The following entry is listed:

```
diActive   OBJECT-TYPE
SYNTAX     INTEGER
ACCESS     read-write
STATUS     mandatory
DESCRIPTION "The total number of active calls."
::= { calldialin 25 }
```

Step 3 *(continued)*:

The entry includes the name, type, access available, and description of the parameter. The last line tells us that `diActive` is parameter # 25 under the `calldialin` branch. We now know the last digit in the OID and can start building towards the first digit. Now our OID looks like this:

```
calldialin.25
```

Step 4: Climbing the tree

The next step is to determine what the conversion for `calldialin` is and what branch `calldialin` is located. At the top of `common.mib` is a list of imports as seen below:

```
IMPORTS
   mfr2              FROM CORPORAT-MIB
   authenticate      FROM CORPORAT-MIB
   listener          FROM CORPORAT-MIB
   calldialin        FROM CORPORAT-MIB
```

This means that `calldialin` is defined in Corporate MIB and we need to open that file (`corporat.mib`).

Following is the definition of `calldialin`:

```
calldialin   OBJECT IDENTIFIER ::= { patton  5 }
```

`calldialin` is node #5 in the patton branch.

Now our OID looks like this:

```
patton.5.25
```

`patton` is also identified in `corporat.mib` as:

```
patton OBJECT IDENTIFIER ::= { enterprises 1768 }
```

Now we know `patton` is node #1768 in branch enterprises and the OID looks like this:

```
enterprises.1768.5.25
```

Step 4 *(continued)*:

`enterprises` is defined as:

```
   enterprises FROM RFC1155-SMI
```

This means that `enterprises` is defined in RFC 1155. RFC 1155 is laid out in a manner similar to the MIB files you were just looking at so you could perform the same procedure as above using RFC 1155.

All private PATTON MIB variables will be under the `patton` branch which is under the enterprises branch and so on. This means that all private PATTON MIB variables will always start out the same because you always traverse the same branches in the tree to get to our private MIB. The OID will start with `1.3.6.1.4.1.1768`.

The OID for Active Calls is

```
   1.3.6.1.4.1.1768.5.25.
```

Example 2: Building the OID for the state of call ID #1001

Step 1: Finding the SNMP Name

This parameter is located under the Dial-In link. In this instance there is a state for each individual call. The parameter description is located in Chapter 5: Dial-in Section: Dial In main window. The SNMP name is `diactState`.

Step 2: Finding the section of the MIB tree in which the SNMP parameter resides

This is also a PATTON defined variable and not an Internet standards defined variable. This means that the OID will start with `1.3.6.1.4.1.1768`.

Step 3: Finding the branch where the SNMP parameter resides

After searching common.mib, we find the following entry:

```
  diactState OBJECT-TYPE
    SYNTAX      DialinState
    ACCESS      read-write
    STATUS      mandatory
    DESCRIPTION "Indicates current progress and reason for
                 termination."     ::= { diactEntry 3 }
```

This tells us that diactState is the third parameter under diactEntry. The OID ends in

```
  diactEntry.3
```

Step 4: Climbing the tree

Searching for `diactEntry` gives me:

```
diactEntry OBJECT-TYPE
  SYNTAX       DiactEntry
  ACCESS       not-accessible
  STATUS       mandatory
  DESCRIPTION  "An entry of an active call."
  INDEX        { diactIndex }     ::= { diactTable 1 }
```

`diactEntry` is the first branch in `diactTable`. Now we can add to the OID:

```
  diactTable.1.3
```

Searching for `diactTable` gives us the following entry:

```
  diactTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF DiactEntry
    ACCESS      not-accessible
    STATUS      mandatory
    DESCRIPTION "Table of individual active calls."
     ::= { calldialin 100 }
```

`diactTable` is the 100th branch under `calldialin` which gives usthe following for the OID: `calldialin.100.1.3`

From our previous example we know that `calldialin` is the fifth branch of the private PATTON MIB tree so we can table on the OID beginning for all private PATTON MIBs and convert `calldialin` to `5`. Our OID is complete:

```
1.3.6.1.4.1.1768.5.100.1.3
```

. . . or is it???

You may have noticed the following entry when you were looking up `diactState`:

```
DiactEntry ::=
   SEQUENCE {
  diactIndex                    INTEGER,
  diactMultiIndex               INTEGER,
  diactState                    DialinState
```

This indicates that the OID is located in a table and there could be multiple instances of the OID for each entry in the table. In this case, there will be one entry for each call on the dial-in screen. To get to a specific entry in a table, you need to access the table via its INDEX, which in this case is `diactIndex` (see `diactEntry` definition above). `diactIndex` is actually the call ID (see Reference Guide).

Therefore, if we want the state for a particular call then we need to add the call ID to the OID.

The following OID will give us the state of call #1001:

```
1.3.6.1.4.1.1768.5.100.1.3.1001
```

PATTON Electronics Co.
7622 Rickenbacker Drive, Gaithersburg, MD USA 20879
Phone 1.301.975.1000 Fax 1.301.869.9293
http://www.patton.com