The Simple Times™

THE BI-MONTHLY NEWSLETTER OF SNMP TECHNOLOGY, COMMENT, AND EVENTS℠
VOLUME 1, NUMBER 2 MAY/JUNE, 1992

The Simple Times is an openly-available publication devoted to the promotion of the Simple Network Management Protocol (SNMP). In each issue, ***The Simple Times*** presents: a refereed technical article, an industry comment, and several featured columns. In addition, some issues include brief announcements, summaries of recent publications, and an activities calendar. For information on submissions, see page 16.

In this Issue:

Technical Article

Tracy Cox, Deirdre Kostick, and Kaj Tesink, Bellcore

In this issue: *Sets Are Fun: Introducing the SMDS Subscription MIB Module*

The Switched Multi-megabit Data Service (SMDS) is a high-performance, public, packet-switched data service, and is defined in Bellcore Technical Reference (TR-TSV-000772). Offered as a public network equivalent of a MAC-level service, SMDS is easily integrated into existing networking protocol architectures using Local Area Network (LAN) or Metropolitan Area Network (MAN) technologies such as Ethernet and Fiber Distributed Data Interface (FDDI).

Another Bellcore Technical Advisory (TA-TSV-001062) defines a Customer Network Management (CNM) service to be used in conjunction with SMDS. This service provides SMDS subscribers with SNMP-based access to SMDS subscription parameters and performance monitoring information for their Subscriber Network Interfaces (SNIs). Use of SMDS CNM allows subscribers to directly access information relevant to their use of SMDS. Planned support of subscribers' write-access to SMDS address screening and group address lists will help subscribers reconfigure their use of SMDS in less time than the service order process would take.

For SNMP-based management, the network providing SMDS is modeled as a single, managed resource. An SNMP Agent in the network acts a proxy agent on behalf of the SMDS switches. Using this single, managed resource model, subscribers can view their SMDS SNIs as interfaces to a "box", similar to interfaces into network equipment, such as routers or bridges. Of course, since SMDS is a service, there is some information unique to the service aspects of SMDS. This article briefly describes SMDS and all the MIB modules used to manage SMDS via SNMP. The SMDS Subscription MIB module is described in more detail including a description of the use of the SNMP set-request for reconfiguring certain service features of SMDS. Additional information on SNMP support for SMDS CNM can be found in a forthcoming issue of the *Journal of High-Speed Networking*.

SMDS: The Service

SMDS is a public, datagram service that provides LAN interconnection across a wide area. Users will typically access SMDS using a router attached at the Subscriber Network Interface (SNI). The SMDS Interface Protocol (SIP), which is implemented in the router and in the SMDS network, is a MAC-level protocol which can run over either DS1 or DS3 transmission facilities.

Modeled as a LAN-like service for the wide-area, SMDS is designed to provide low delay and provides other LAN-like services such as multicasting. SMDS offers several service features that make it a unique public data service:

- Group Addressing;
- Address Screening;
- Source Address Validation;
- Access Classes; and,
- Levels of multiple concurrent data units.

The *Group Addressing* feature is similar to multi-casting in LANs. Subscribers identify an SMDS address, called a group address, which can represent at least 128 individual addresses. An SMDS packet that is sent to a group address will be copied by the SMDS network and sent to all members of that group address.

The *Address Screening* feature allows subscribers to establish screening lists which allow or disallow sources and destinations of SMDS traffic. Like address filters common to bridges and routers, the address screening feature lets subscribers establish virtual private data networks with SMDS. There are two types of address screens: individual and group address screens.

Individual address screens contain a list of allowed or disallowed individual addresses and are used for screening destination addresses of SMDS packets sent into the network and source addresses of SMDS packets to be delivered from the network. Group address screens identify allowed or disallowed group addresses and are used to screen destination addresses of packets sent into the network. (Since only individual addresses are allowed for source addresses, group address screening is not used to screen the source address on incoming SMDS traffic.) A minimum of one and a maximum of four group address and individual address screens are supported for each SNI, giving a maximum of eight address screens per SNI. However, each SMDS address that is assigned to an SNI can only be associated with one individual address screen and one group address screen.

The *Source Address Validation* feature provides validation of each SMDS packet to insure that it is originating from an individual SMDS address assigned to that SNI.

The *Access Classes* feature identifies different rates of sustained information transfer for DS3-based access paths. There are five access classes: 4 Mbps, 10 Mbps, 16 Mbps, 25 Mbps, and 34 Mbps. Access Classes apply only to the traffic flow from the subscriber into the network, and will initially be determined at subscription time. (There is no access class enforcement for DS1-based access.)

Use of the *Multiple Concurrent Data Units* feature determines the number of SMDS data units that may be in transit simultaneously between the SMDS network and the router. The choice of 1 or 16 is made at subscription time.

SMDS CNM allows SMDS subscribers to manage their use of SMDS. It is expected that SMDS CNM subscribers will have an SNMP-based Network Management Station (NMS) to manage their LANs. The SMDS CNM service extends this capability to the wide area by providing SNMP read and write access to their SMDS information. The MIBs used to manage the SMDS Interface Protocol (SIP) and the SMDS service features are discussed in the next section.

MIB Mania

SMDS CNM uses the following six MIB modules for SNMP-based management:

- MIB-II - SMDS CNM uses only the system and interfaces groups. The system group provides information on the SNMP Agent that proxies for the SMDS switches in the serving network. The interfaces group provides information on a per-SNI basis. Some SMDS-specific use of the interfaces information is described below.

  `ifIndex` is used to uniquely identify each SNI. Most of the SMDS CNM information is defined on a per-SNI basis.

  `ifType` (type of the managed interface) is sip(31) which refers to the SMDS Interface Protocol and is used to point to the SIP MIB.

  `ifSpeed` is either 1500000 (1.5Mbps) or 45000000 (4.5Mbps) and is used to identify whether the access line is a DS1 or DS3-based transmission facility. This means that the NMS is expected to inspect the value of `ifSpeed` in the Interfaces Group to determine the transmission facility in use, and the corresponding MIB module.

`ifMtu` is 9188 which is the maximum size in octets of the user data part of an SMDS packet.

`ifAdminStatus` (desired status of the managed interface) in MIB-II is defined as read-write. Currently, this is a read-only object for SMDS CNM.

- The SIP MIB module contains packet and error counts for the SMDS Interface Protocol (SIP) for each SNI.
- The DS1/DS3 MIB modules contain physical-level performance counts and configuration information of the DS1 and DS3 transmission facilities.
- The SMDS Subscription MIB module provides the SMDS service feature-specific information. Unlike the previous MIB modules which are implemented in subscriber equipment (such as routers and DSUs), the SMDS Subscription MIB module is only supported by the SMDS network.
- The SNMP Party MIB is supported in the SMDS SNMP CNM agent. Secure SNMP is supported to provide desired security features like authentication, privacy, and message integrity. These features are of particular importance for the write capability, which allows CNM subscribers to modify their service feature information contained in the SMDS Subscription MIB.

The SMDS Subscription MIB Module

An SMDS CNM subscriber uses SNMP to manage service feature-specific information for their SNIs. This information is found in the SMDS Subscription MIB module, which is posted on `venera.isi.edu` as `mib/bellcore.txt`, and is divided into six MIB groups:

- SMDS Subscription Parameters and Violations;
- SMDS Address Table;
- Address Screening;
- Group Addressing Information;
- Service Disagreements; and
- Exchange Access Component of Inter-exchange SMDS.

The *SMDS Subscription Parameters and Violations* group, consists of a table indexed by SNI, which contains SNI contact information, SNI location, access class, MCDUs In, and MCDUs Out. This table also includes counts by type of violation (e.g., unassigned source address, access class exceeded, address screening failure) of SMDS packets that were discarded.

The *SMDS Address Table* group, indexed by SMDS address, associates the SMDS addresses with an `ifIndex` value.

The *Address Screening Group* is a collection of six tables used to identify up to four Individual Address Screens and up to four Group Address Screens. The address screens are associated with different addresses assigned to the SNI.

The first table, `addressScreeningMasterTable`, is used to identify all the screens (i.e., there may be up to eight screens). New screens can be created and deleted by an SMDS CNM subscriber. The second table, `numberAndDefaultScreeningTable`, is used to identify the default screens for each SNI. Default screens are used when an address (either a group address or an individual address) that is assigned to an SNI is not associated with any individual address screen or any group address screen. The default screens may be changed also.

The next two tables are used to associate the addresses assigned to the SNI to a particular individual address screen and a particular group address screen. This information is contained in two tables: one for the SNI addresses associated with the individual address screens, `associatedAddressesIndScreenTable`, and one for the SNI addresses associated with the group address screen, `associatedAddressesGrpScreenTable`.

The remaining two tables in the Address Screening group identify the addresses to be screened (i.e., whether you want to receive SIP L3 PDUs from or send them to a particular address) within the Individual Address Screens and the Group Address Screens.

The *Group Addressing Information* group contains five tables. The Group Address Group contains the `groupAddressTable` which identifies the group address and the individual addresses that are identified by the group address. This group also contains the `numberMemberAddressesTable` which provides the number of individual addresses that are associated with the group address. Individual addresses can be added or deleted to the group address by using this table. Only the sponsor of the group address has read/write access to this information. Members of the group address have only read access. New group addresses cannot be created by using this table, new group addresses can only be assigned by the service provider. The Member Group Address Group identifies the individual address and all the group addresses for which it is a member. Using this table, `memberGroupAddressTable`, the group addresses can only be disassociated from the individual addresses; meaning that the group member can only delete the row and not create a new row in this table. The tables, `memberGroupAddressTable`

and groupAddressTable, contain the same information that is indexed in two different ways and provides different access control. This group also provides the number of group addresses an individual address belongs to, numberGAsForAddressTable, and the number of group addresses that are assigned to each SNI, numberGAsForSNITable.

The *Service Disagreements* group, indexed by SNI and by type of service disagreement, includes the source address, destination address, and timestamp of the last PDU discarded for source address screen violations, destination screen violations, and invalid source address on PDU.

Finally, the *Exchange Access Component of Inter-exchange SMDS* group is used to support inter-LATA SMDS and is not discussed here.

Sets are Fun!

For the SMDS CNM subscriber utilizing an SNMP-based NMS, using set-requests are fun. The SMDS CNM designers developed a one step set-request operation which allows the CNM subscriber to add an entire row to a table by setting a single object. The SNMP CNM agent will use the set-request's operand to create the entire row. Even though adding the row to the table is easy for the SNMP CNM agent, making the corresponding change in the SMDS switch is hard. It is hard, because the SNMP CNM agent does not have complete "authority" (capability) to make the change; instead, the requested change must be made in the SMDS switch.

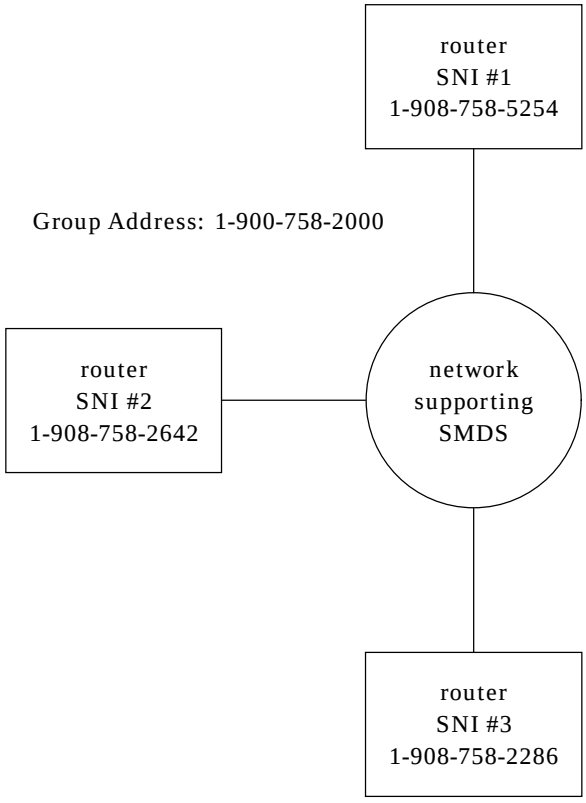
As described earlier, the SNMP CNM agent acts as a proxy agent for the SMDS switch. A set-request relies on a communication between the SNMP CNM agent and the SMDS switch. This "behind-the-scenes" communication may take some time. Therefore, the designers of the SMDS Subscription MIB modeled a set-request of the SMDS Subscription information after the MIB-II ifAdminStatus and ifOperStatus paradigm. For example, to modify the address screening information or the group addressing information, the SMDS CNM subscriber sets a statusChange object and gets an immediate get-response. After the change is processed by the SMDS switch, a trap (identified in the SMDS Subscription MIB) is sent to the CNM subscriber's NMS. The SMDS CNM subscriber can determine whether or not the set-request was successful by comparing the status object (which is read-only) to the statusChange object (which is read-write). If the set-request was successful, the value of the status object will be changed and will be equal to the requested value of the statusChange object. If the set-request was not successful, the value of the statusChange object will revert to its old value, which is equal to the current value of the status object. To better illustrate this, an example is given.

An Example Configuration

Let's look at an example configuration to help explain the use of SNMP's set-request to manage a subscriber's SMDS configuration. Assume that a subscriber has three SNIs served off a single switch. At subscription time, one individual SMDS address is assigned to each SNI as follows:

SNI	SMDS Address
1	1-908-758-5254
2	1-908-758-2642
3	1-908-758-2286

Further, let's assume that the subscriber uses the Address Screening service feature to allow only traffic between its own routers and the Group Address service feature to set up a logical IP subnetwork as defined in RFC 1209. The subscriber sponsors a group address, e.g., 1-900-758-2000, that contains all the individual addresses associated with the three SNIs. So, the group configuration is:



The subscriber uses both group and individual address screens to control incoming and outgoing SMDS traffic. One individual address screen is set up for each SNI to identify allowed individual addresses; each

individual address screen contains all the subscriber's SMDS addresses (except for the individual addresses that are assigned to that particular SNI). Also, each SMDS address, which is assigned to its SNI, is associated with the individual address screen for its SNI. One group address screen is created for each SNI to identify only allowed group addresses and contains the group address 1-900-758-2000. Each SMDS address, which is assigned to its SNI, is associated with the group address screen for its SNI. The subscriber manages a total of three individual address screens and three group address screens.

Now, suppose that the subscriber adds a new SNI, #4. Again, at subscription, a new SMDS address, 1-908-758-2107, is assigned to this SNI. What needs to happen to the group address information and the various address screens?

1. The new SMDS address is added to the existing individual address screens for the first three SNIs to allow communications with the new SNI.
2. The group address is updated to include the new individual address.
3. For the new SNI, an individual address screen containing a list of the subscriber's three existing individual SMDS addresses is created.
4. Also, a group address screen, containing the group address 1-900-758-2000, is created for the new SNI.

Without SMDS CNM, all these configuration changes must be handled through the network provider.

Using SNMP's Set-Request for SMDS CNM Service

With the previous scenario in mind, an SMDS CNM subscriber, who has just added a new, fourth SNI, has to add the new individual address of SNI #4 to the Group Address, set up the individual address screen and group address screen for this SNI, and add the individual address of SNI #4 to the individual address screen of the three other SNIs.

In order to add the individual address to the group address, the SMDS CNM subscriber must be the sponsor of the group address, implying that the sponsor is the person(s) who requested the group address from the SMDS service provider. The sponsor uses the `groupAddressTable` from the Group Address Group to add the individual address to the group address. For this example, the group address is 1-900-758-2000 and the individual address to be added to the group address is 1-908-758-2107.

The operand to the set-request is:

```
SET groupMemberStatusChange.
    E1.90.07.58.20.00.FF.FF.
    C1.90.87.58.21.07.FF.FF = valid(1)
```

Note: An SMDS Address is an octet string of length 8. The first four bits of the octet string identifies whether the address is a group address (1110 or E in hex) or an individual address (1100 or C in hex). The remaining 60-bits are used for the SMDS Address, which is encoded as binary-coded decimals and padded with ones.

The SNMP CNM agent will respond with a get-response containing the same OBJECT IDENTIFIER (OID) and value as received in the set-request. The SNMP CNM agent will forward this request to the SMDS switch. When the SMDS switch makes the change, an enterprise-specific trap will be sent to the CNM subscriber's NMS. This trap will be the `groupAddressChange` TRAP-TYPE as defined in the SMDS Subscription MIB module. Using the OID from the original set-request, the following objects will be added (these are the columns within the row of the `groupAddressTable`):

```
groupAddress.
    E1.90.07.58.20.00.FF.FF.
    C1.90.87.58.21.07.FF.FF = E19007582000
groupMember.
    E1.90.07.58.20.00.FF.FF.
    C1.90.87.58.21.07.FF.FF = C19087582107
groupMemberStatus.
    E1.90.07.58.20.00.FF.FF.
    C1.90.87.58.21.07.FF.FF = valid(1)
```

The other tables within the Group Addressing Information Group will be updated to reflect this change.

To create one individual address screen for SNI #4, the following set-request operand is used (from the `addressScreeningMasterTable`):

```
SET screenStatusChange.4.1.1 = allowed(1)
```

To create one group address screen for SNI #4, the following set-request operand is used (from the `addressScreeningMasterTable`):

```
SET screenStatusChange.4.1.2 = allowed(1)
```

To set the default screens for the associated SNI #4's individual address and group address to equal the one individual address screen and the one group address screen, the following set-request operands are used (from the `numberAndDefaultScreeningTable`):

```
SET defIAScreenForIAsChange.4 = 1
SET defIAScreenForGAsChange.4 = 1
SET defGAScreenForIAsChange.4 = 1
```

(The value 1 indicates screen number 1.) Therefore, the Associated Addresses Group is not needed, because default screens are used.

To add the individual addresses of SNIs #1 thru #3 to SNI #4's one individual address screen, the following set-request operands are used (from the `individualAddressScreenTable`):

```
SET iAScreeningAddressStatusChange.
    4.1.C1.90.87.58.26.42.FF.FF = valid(1)
SET iAScreeningAddressStatusChange.
    4.1.C1.90.87.58.52.54.FF.FF = valid(1)
SET iAScreeningAddressStatusChange.
    4.1.C1.90.87.58.22.86.FF.FF = valid(1)
```

To add the group address to SNI #4's one group address screen, the following set-request operand is used (from the `groupAddressScreenTable`):

```
SET gAScreeningAddressStatusChange.
    4.1.E1.90.07.58.20.00.FF.FF = valid(1)
```

The other objects in each table are created by using the indexing information used in the `statusChange` object. It is left as an exercise for the reader to follow through the addition of SNI #4's individual address to the individual address screen of the three other SNIs.

Conclusions

The SMDS CNM service provides subscribers with a lot of information that is accessed using SNMP. Information ranges from subscription parameters to performance information for each protocol layer and underlying physical transport. The users of this information will need to develop a thorough understanding of the MIB modules. The designers of the SMDS CNM service hope that this information will form the basis for useful network management capabilities for the SMDS CNM subscriber.

In regard to SNMP's set-request, we reach the following conclusions:

- Both reading and writing take some effort.
- Writing is harder than reading.
- Now that we know how to read, let's learn how to write!

Acknowledgments

Much continuing thanks to Dr. SNMP (aka the eminent Professor Jeffrey D. Case), who helped us develop the SMDS Subscription MIB module. Also, special thanks to Ted Brunner, who implemented an SNMP SMDS CNM agent, and to Dave Piscitello, who is a joy to work with and showed us the SNMP light.

Industry Comment

Marshall T. Rose

Welcome to the second issue of ***The Simple Times***.

This issue went a fair bit over the desired page count, so the *Industry Comment* got cut from two pages down to half a column. For now, there are two administrative topics that require brief attention. But, in the next issue, look for a commentary on SNMP evolution.

First, I'm pleased to note that the newsletter appears to be well-received. There are now nearly 600 electronic subscribers (including several re-distribution lists), with slightly more than 10% receiving the MIME edition. In addition, the vendors participating in the hard-copy distribution channel have collectively printed over 6,000 copies of the first issue.

Second, over a fine lunch at the last meeting of the IETF, one subscriber was confused because the approach to bulk retrieval using SNMP described in the first issue's *Technical Article* was greatly at odds with the approach described in RFC 1187, of which the editor is co-author. I pointed out that the Technical Review Board of ***The Simple Times*** considers articles based solely on their merits and applicability. That is, even though the editor prefers a different approach, the approach described in the first issue still represents excellent technical work, as was accepted for publication on that basis.

Applications and Directions

Steven L. Waldbusser

In this issue: *Today's MIB Compilers — Too Much of a Good thing?*

Much attention is focused these days on MIB compilers and the transfer of MIB information from agent manufacturers to management station vendors. In this article, we will investigate why this is so, and why it may be diverting much needed attention from more important issues in network management.

In addition to descriptions of managed objects and their relationships, the MIB contains detailed information concerning each object, for example, whether a parameter is read-only or read-write, what a reasonable default value would be, or a set of enumerated values for the object. This information is invaluable to agent and management station developers as it allows them to agree on characteristics of information that can be retrieved with SNMP. In addition, the MIB may be compiled by a MIB compiler, making some of the information present in the MIB available to management applications such as MIB browsers and graphers. This makes the MIB browser much more useful because it can

automatically format and describe SNMP variables as it presents them to the user. Applications such as MIB browsers and graphers are termed *generic applications*. These are simply applications that gain all of their management capabilities from parsing a MIB, without any intervention from a human — either a programmer, a manager, or an operator. That is, their knowledge of management objects comes solely from a MIB, whether standard, experimental or proprietary, but without the benefit of human insight as to the semantics which the MIB is meant to convey.

Automatic MIB Learning

This ability to mechanically learn MIB descriptions can be very attractive, especially to management station vendors, because it allows them to leverage work on generic applications to cover a wide variety of agents and MIBs. Given the large number of vendor-specific and standard SNMP MIBs available, MIB descriptions reduce the effort required of vendors to keep their applications up to date. (The beneficial effects of this are that developers can concentrate on raising the state of the art of network management and that the price of network management can remain low.)

Vendors have seized on the efficiencies possible and have made generic MIB browsers and MIB compiler technology a major part of their product. On the surface, this seems to have made system integration a matter of loading MIB files and interoperability a matter of fixing syntax errors in a supplied MIB file. In fact, interoperability testing labs spend much of their effort and base much of their characterization on the ease with which MIBs can be compiled into each product.

Desires have been expressed that this trend continue by allowing more types of information in the MIB to be machine readable. This would allow generic applications to be given more detail which should allow them to provide prettier output. In addition, claims have been made that enough information could be learned dynamically in this fashion to allow an application to effectively manage a device never before seen by the management station vendor nor by the user. While MIB compilers have certainly proven useful, it may be a mistake to emphasize this practice too much. The elusive goal of "intelligent" network management cannot be reached with this technology, and further emphasis may divert attention from efforts to achieve this goal.

Limits of the Technology

Unfortunately the most important information in the MIB, the detailed English description of the object, cannot be understood by a MIB compiler (with 20th century technology). For example, a compiler may be able to read a description of an object in MIB-II and learn that the object is an integer with valid values of one and two, the object may be read and written, and the implementation of the object is mandatory. But only a human reader can discern from the natural language description of the object (`ipForwarding`) that if equal to one, the system it describes is acting as a gateway, otherwise it is just a host. Furthermore, there may be information known to an experienced network manager that isn't described at all in the MIB, such as the fact that in the wrong circumstances, it may be dangerous for the system to be a gateway. In order to provide intelligent network management, applications must be written that contain all of that knowledge. As that knowledge cannot be provided in MIB format, it must be placed there by a network management developer.

Without this intelligence, most generic applications are limited to gathering, formatting, and displaying information. This information is then presented to the user, who applies human intelligence to analyze the information. This burden on the network manager may only be eased by embedding intelligence into such applications. Further, it would be unwise to believe that an application could dynamically discover enough about an unknown MIB to effectively manage an unknown device without this intelligence.

In fact, without this intelligence the network manager will be faced with even more data to understand and sort through. Network managers are already dealing with an information overload and need some relief.

Back to the Real Problem

The major problem is that further emphasis on MIB compilers may divert attention from providing intelligent network management. Expectations will be raised amongst users and developers alike that MIB compilers are part of the state of the art. This is obviously an undesirable situation. The advantages of MIB compiler technology should be merely a means to an end. MIB compiler technology should be used to provide leverage to management station vendors so that they can more quickly react to MIB developments and so that more effort may be spent pushing for better network management technology. There are valid improvements to be made in MIB compilers and generic applications, but efforts spent fine-tuning them are often efforts wasted.

Ask Dr. SNMP

Jeffrey D. Case

Dear *Dr. SNMP*,
I have read that SNMP is primarily a request/response protocol augmented by a limited number of event notifications, called traps. However, I find that my management station does not always receive traps sent to it. The traps appear to be lost due to congestion and faults in the network.

It seems that this could have been avoided if the designers had chosen a reliable transport such as TCP instead of chosing UDP, the unreliable datagram protocol. Had it been, SNMP could be more event-oriented, like OSI management, and there would be less need for polling.

Now that the IETF has invited proposals for the evolution of the SNMP management framework, perhaps it is time to correct this design defect. What do you think of this idea?

— *Baffled in Boston*

Dear *Baffled in Boston*,
Dr. SNMP is in doubt about which idea you are asking about: IETF evolution of the SNMP management framework, or your suggested scheme for making the delivery of SNMP traps reliable.

First, it is premature to express an opinion on the IETF process. While the IETF invitation for proposals has been extended, no proposals have been submitted to the process as yet. As a result, Dr. SNMP is necessarily going to defer sharing any general opinions about the process to future issues of ***The Simple Times***. However, your question perhaps warrants specific comments. As stated in the invitation, "There is little community consensus on what the actual deficiencies of the SNMP framework may be" which is to say that it may be difficult to get agreement that your "problem" is a problem. The invitation also states that "there is similarly little consensus that any particular change to the framework warrants the attendant operational or architectural impact." That is, it may be more difficult to get agreement that your "solution" is the right solution.

Second, regarding the "design defect" of using the connectionless UDP, this is not a bug, it is a feature. Part of the success of the SNMP framework is as a result of the correct selection of a connectionless transport service.

So-called "reliable" transport protocols achieve reliability through retransmission. After sending a message, when an acknowledgement is not received in a timely fashion, the sender retransmits the message. That is, it repeats the message. That is, it repeats the message. Note that the protocol does not guarantee delivery, it only guarantees how long and how hard it will retry. If, for example, a backhoe cuts through a cable, the transport protocol will repeatedly attempt to deliver the message in the absence of receipt of an acknowledgement. So, is it possible to deliver these traps reliably when a backhoe cuts through a cable?

Back on the farm, we have a saying:

> "Sure, almost anything is possible. You can even teach a goldfish to play the piano, if you use enough voltage."

Which is to say, when the cable is cut, it doesn't matter what your transport retransmission strategy is, because the signal won't get through until the cable is repaired. What you really need is more voltage, a lot more voltage.

More seriously, our comparative studies of SNMP and the OSI management protocol, CMIP, show that connectionless protocols such as UDP are more robust than connection-oriented protocols such as TCP in networks with elevated packet loss rates. The research shows that in such networks, the use of connection-oriented protocols results in TCP resets, loss of associations, and unacceptably high values for mean time to association establishment. These failures are inconsistent with their application to fault management.

Finally, Dr. SNMP is not particularly in favor of your efforts to rename the "User Datagram Protocol" as the "Unreliable Datagram Protocol".

Dear *Dr. SNMP*,
In the first issue, one of your answers indicated that some variables which have an ACCESS of "read-write" can be set to zero. But you also said that counters should be monotonically increasing, indicating that you don't think that counters should be resettable by a manager. Why is this so?

— *Perplexed in Portland*

Dear *Perplexed in Portland*,
Back on the farm, we have a saying:

> "We could talk about that 'til the cows come home, but ..."

The short answer is that the SMI states that counters are monotonically increasing. Counters were designed that way so that multiple management stations (or multiple independent management applications on a single management station) could use the same counter. If one application resets the counter, warps it to a new value, or otherwise causes a discontinuity, it will probably cause problems for the other application. As a result, management applications should be written such that they do not need to reset counters, rather, displaying

delta values with respect to an initial value maintained internally by the management application.

Of course, management applications should be prepared for asynchronous discontinuities as a result of reboots and counters wrapping when they reach the maximum value.

Dear *Dr. SNMP*,
Why do you so carefully avoid the use of the words "client" and "server" in the SNMP context?
— *Lax in Los Angeles*

Dear *Lax in Los Angeles*,
Dr. SNMP just learned a new expression from a neighbor down here on the farm:

> "Sometimes a dog runs the wrong squirrel up the right tree."

(Actually, Dr. SNMP and the family recently saw the motion picture *Straight Talk*, in which Dolly Parton introduces this expression.)

Dr. SNMP has found that this kind of confusion is exactly what results when we use terms like client and server. For example, suppose you have a network which is structured as a two level hierarchy, consisting of multiple LANs connected via a WAN. You are sitting at your favorite Unix workstation, connected to the LAN at your site. The Unix workstation is running NFS client and X Window server software. The SNMP subsystem has applications which use SNMP to manage the agents on your LAN and to provide summary information about those agents to other management stations elsewhere in the Internet.

And you are wondering why Dr. SNMP finds it difficult to classify this node as a either a client or a server?

Security and Protocols

Keith McCloghrie

In the last issue, we looked at why SNMP Security is needed, and discussed its three primary mechanisms: the MD5 message digest algorithm, the DES encryption algorithm, and loosely synchronized clocks. We saw how these mechanisms are used to provide origin authentication, message integrity, privacy, and replay protection. In this article, we'll look at how the mechanisms are integrated into the protocol. In a future article, we'll discuss issues involved in implementation and deployment.

First, in any form of communication, there is a source and a destination. Authentication and message integrity are dependent upon the source. However, encryption is dependent upon the destination (to see this, consider that if a public-key privacy algorithm were added in the future, then the source would encrypt using the destination's public key and the destination would use its own private key to decrypt). Access control is dependent on which source is trying to access what target, and the target is based on the destination. Thus, in SNMP Security, there is a need to differentiate between source and destination, a need which today's SNMP *community* field does not provide.

SNMP Parties

For this reason, SNMP Security introduces the concept of a SNMP *party*. A SNMP party is defined as an execution context of a SNMP protocol implementation. Whenever a SNMP protocol implementation processes a message, it does so by acting in the role of one of the SNMP parties configured for it. Each SNMP party executes at a specific transport address, and has specific authentication parameters, privacy parameters, proxy information, and a MIB view. The authentication parameters include an algorithm, a secret, and the state information needed to maintain its clock and ensure proper message ordering. The privacy parameters include an algorithm and a secret. The proxy information either indicates no-proxy, or "points" to another SNMP party where the real-agent executes. The MIB view specifies the subset of an agent's management information (i.e. which MIB objects) that the party can access.

Thus, a SNMP Security message is sent from one party to another party. The message is authenticated (or not) according to the authentication parameters of the source party. The message is encrypted (or not) according to the privacy parameters of the destination party. Access control specifies that a specific source party is allowed to originate a particular set of SNMP operations (e.g., get-requests and set-requests, or just get-requests) to a specific destination party. The destination party's MIB view provides the limitation on which MIB objects the message can access.

By including an algorithm in both the authentication and privacy parameters, multiple parties with different capabilities can be defined for execution at a single SNMP protocol entity. One party can use no-authentication and no-privacy; another can use MD5-authentication and no-privacy; and another can use MD5-authentication and DES-privacy.

While this model requires the *wrapper* of a SNMP message to change in order that both the destination and source parties and the appropriate authentication information (e.g., the digest and the time-stamp) can be sent with the message, it does not (fortunately)

entail any change to the basic SNMP PDUs which are encapsulated inside a SNMP message. In fact, an agent implementation which followed the guidelines in the original SNMP protocol specification, should be able to implement SNMP security with additional code but very few changes to the existing code.

The Party Clock

Each SNMP party using MD5-authentication has a (relative) clock. In order to be authentic, a received message must have a time-stamp which, when added to an administratively defined *lifetime*, is greater or equal to the value of the clock at the time of receipt. For example, if the clocks were perfectly synchronized and the lifetime was 1 minute, then the message must be delivered within one minute of its generation.

In practice, each implementation of SNMP needs to keep a local database of party information, both for parties that execute locally and for those remote parties with which local parties communicate. In order to communicate with a remote party using MD5-authentication, a SNMP protocol implementation must retain in its local database a clock value for that party, which is (loosely) synchronized with the remote value. Since clocks tend to drift, it is necessary for the clocks to be inspected periodically, and re-synchronized if necessary. This chore is delegated to the management station. However, a few features are included in the protocol to enhance the synchronization through the regular exchange of messages, and the party clocks are purposely positioned in the MIB to support easy read access to them through both unauthenticated and authenticated get-requests, so that a management station does not need to maintain synchronization with all its agents all of the time, but can let some lapse, and later re-synchronize when necessary.

The lifetime value must be large enough to accommodate variations in communications delay as well as to accommodate a small amount of clock drift. On the other hand, it is the lifetime value which provides the window in time during which a message is valid, and so lifetimes must not be greater than the administrator's desire for protection against replay attacks (e.g., a few minutes).

MIB Views

A party's MIB view is defined as a set of *view subtrees*, where a view subtree is defined by a node in the MIB's naming tree. For each of its each view subtrees, the MIB view either includes or excludes instances of the MIB objects defined within that subtree. In addition, a view mask is defined in order to reduce the amount of configuration required when very granular access control is desired (e.g., access control at the instance level).

As we have seen, SNMP is itself used to maintain and control the parameters of the parties known by an agent. This of course, requires a Party MIB, which contains several MIB tables. Two of the tables contain party information, one for publically-readable information (e.g., the party clocks) about the parties that an agent knows, and a second for the party secrets. Another table is for access control, specifying the types of PDUs which specific remote parties are authorized to send to specific local parties. A fourth table defines the view subtrees for the MIB views of local parties.

Security is never free. The use of MD5 and DES and the access control checking does increase the cpu-time required to process SNMP messages. Experience from prototype implementations indicates that use of MD5 incurs a 15–20% increase, and using a software implementation of DES doubles the processing time. Nevertheless, these increases are considered acceptable. Particularly since if DES is used only to change the SNMP secrets, then its usage is probably at most a few times per week (or per day, for especially security conscious network administrators). Similarly, many network administrators may choose to use MD5 only for network control messages (i.e., SNMP set-requests), which will always be less frequent than messages used for network monitoring.

Standards

David T. Perkins

A characteristic that differentiates the IETF from other standards bodies is the process that is used to create standards. The IETF standards process is one of the reasons for the widespread deployment of IETF-developed protocols. The process encourages new ideas and innovation, yet restricts progress of standards without independent implementation and interoperable deployment.

This contrasts with the process used by other standards bodies which finalize a standard before it has been implemented or deployed. The other differentiating characteristic is the easy access and widespread dissemination of IETF standards, both completed and early drafts, through electronic mail and through network file transfer. Other standards bodies use the sales of standards in paper form as a source of funding for their operations. This increases the costs and slows the dissemination.

Documents generated by the IETF which are meant for distribution are published in the Request for Comments (RFC) series. Not all RFCs are standards. In the past, the standards process was documented in the *IAB*

Official Protocol Standards, a document published as an RFC and updated on an irregular basis. As of March 1992, this process is now described in RFC 1310, entitled *The Internet Standards Process*. This document is over 20 pages long. The remainder of this article will highlight the key points of the standards process. This process continues to be refined over time. The process document itself contains open issues, and additional issues were discussed at the March IETF plenary. Any changes will be reported in future issues of ***The Simple Times***.

According to RFC 1310, the goal of the IETF standards process is to develop specifications that are:

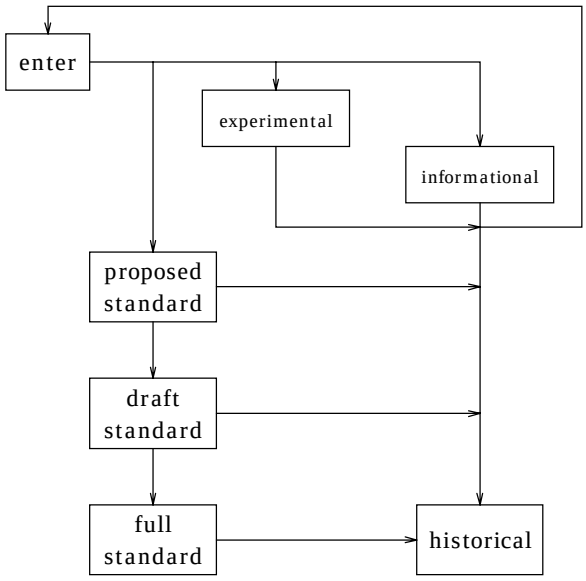
- stable and well-understood;
- are technically competent;
- have multiple, independent, and interoperable implementations with operational experience;
- enjoy significant public support; and,
- are recognizably useful in some or all parts of the Internet.

The Internet Activities Board (IAB) determines the standardization-status of each specification. Changes in status are based on the recommendation of the Internet Engineering Steering Group (IESG). (The working groups of the IETF are split into areas, and each is coordinated by an Area Director. The IESG includes the Area Directors and the IETF chair.) Members of the IETF and its working groups are interested technical contributors and not formal representatives of organizations.

Work in progress is made available in an on-line directory called `internet-drafts`, which is replicated in many locations around the world for review prior to publication as an standard. The standards process requires that a specification be available for at least two weeks in the `internet-drafts` directory before being published.

Progressing through the Standards-Track

Here is how a specification may progress through the standards-track:



To begin, a specification which is either not intended for the standards-track or doesn't fulfill the requirements of an Internet standard is published as either an experimental or an informational document. However, when such a document has been revised or has gathered enough community interest (or implementation experience) it might once again be evaluated for entering the standards track.

To enter onto the standards-track as a proposed standard, a specification must be reasonably stable, internally consistent, well-understood, have received significant community review, and enjoy enough community interest to be considered valuable. Early implementation experience is also helpful in evaluating candidate specifications.

After a minimum of six months have elapsed, and once there are at least two independent and interoperable implementations available to allow for some operational experience, a proposed standard may be considered for advancement to draft standard. Similarly, after a minimum of four months have passed, and there has been significant implementation and operational experience, the draft standard may be considered for advancement to full standard.

Finally, any time a specification has been replaced by a more recent version, the former version should be considered obsolete.

This month's column has highlighted the standards process for Internet standards. The next issue will present the standards process for IEEE network management standards.

Summary of Standards

Full Standards:

- 1155 - Structure of Management Information (SMI);
- 1157 - Simple Network Management Protocol (SNMP); and,
- 1213 - Management Information Base (MIB-II).

Draft Standards:

- 1212 - Concise MIB definitions.

Proposed standards:

- 1229 - Extensions to the generic-interface MIB;
- 1230 - IEEE 802.4 Token Bus Interface Type MIB;
- 1231 - IEEE 802.5 Token Ring Interface Type MIB;
- 1232 - DS1 Interface Type MIB;
- 1233 - DS3 Interface Type MIB;
- 1239 - Reassignment of experimental MIBs to standard MIBs;
- 1243 - AppleTalk MIB;
- 1253 - OSPF version 2 MIB;
- 1269 - BGP version 3 MIB;
- 1271 - Remote LAN Monitoring MIB;
- 1284 - Ether-Like Interface Type MIB;
- 1285 - FDDI Interface Type MIB;
- 1286 - Bridge MIB;
- 1289 - DECnet phase IV MIB;
- 1304 - SMDS Interface Protocol (SIP) Interface Type MIB;
- 1315 - Frame Relay DTE Interface Type MIB;
- 1316 - Character Stream Device MIB;
- 1317 - RS-232 Interface Type MIB; and,
- 1318 - Parallel Printer Interface Type MIB.

Historical:

- 1156 - Management Information Base (MIB-I).

Recently Published Standards

The following MIBs have just recently entered the standards-track:

RFC 1315 - Frame Relay DTE Interface Type MIB
This MIB defines objects for a Frame Relay interface modeled as a single connection to a "multi-access media", not as a group of point-to-point connections. The MIB is composed of three groups. The first is for the Data Link Connection Management Interface, the second for the Circuits, and the third describes errors on the circuits.

RFC 1316 - Character Stream Device MIB
This MIB defines the characteristics of character stream ports. They may be physical or virtual. The MIB consists of two tables. The first table consists of objects that allow character ports to be monitored and controlled. The second table contains objects for monitoring of sessions active on each port.

RFC 1317 - RS-232 Interface Type MIB
This MIB contains objects describing RS-232 and similar types of physical interfaces. These include RS-422, RS-423, V.35, and other asynchronous or synchronous serial physical links with a similar set of control signals. The first table in the MIB contains objects that are common to all types of ports. The second table is specific to asynchronous ports and the third is specific to synchronous ports. The fourth and fifth tables represent input and output signals, respectively, for ports.

1318 - Parallel Printer Interface Type MIB
Physical parallel printer-like interfaces are described using this MIB. The first table contains objects common to all ports. The second and third tables describe input and output signals respectively for ports.

Working Group Synopses

Robert L. Stewart

The working groups supplied plenty of mailing list discussion to report for the last two months, plus the March IETF meeting in San Diego. We'll try to hit the high points, but over 13 pages of notes won't fit into two and one-half pages of newsletter. Ultimately, there's no substitute for subscribing to the mailing lists and attending the meetings.

SNMP in General

Multiplexing SNMP agents was a hot topic, with a BOF meeting in San Diego. Related to this were many

questions about SMUX and DPI. The issue concerns the integration, within general-purpose systems, of independent software components that each have their own MIB modules. A suggestion on the list drew much debate over relaxing SNMP's requirement that all objects in a set request are set as if simultaneously and all succeed or fail as a group. Many consider such a protocol change to be totally unacceptable. The problem may extend not only to multiple software components, but over multiple processors as implementations effectively build a *distributed agent* that appears to its clients as offering a single, integrated MIB. Resolution could involve a new intra-agent protocol, but this could be considered implementation-specific. The meeting attracted a large group, who discussed the topic vigorously. Many working integrated solutions were mentioned, as was simply treating each independent application as having a separate MIB accessed via SNMP proxy.

The relevant components of the ISODE were the topic of several questions. Such questions were referred to `bug-isode@xtel.co.uk` and the ISODE documentation, with increasing bluntness.

A great discussion over the lack of 64–bit counters reached little consensus. Concern over implementation overhead brought a code example which resulted in disagreement over how much overhead is too much. The need is for interface byte counters to handle future ultra-high speed transmission media. The response to a suggestion of lesser resolution was an example of the need for single byte resolution. Results from a straw poll taken at an IEEE meeting were reported, and showed dislike of only 64–bit counters, desiring both 64 and 32–bit.

A plea for a community string MIB as a security feature was severely questioned. Comments included the observation that anyone suggesting use of community strings as a security mechanism should be sued, that a get-next sweep against any community will deny service, and that a community string MIB is of little or no value.

A request for a list of the "most important" MIB variables expanded to include MIB variables that are missing and information on what agents implement what variables. An IP address in the `ifEntry` was suggested, but rejected due to the need for `ifEntry` to support multiple higher level protocols and since the IP address to interface mapping is already available, properly, in the IP group. A plea for a standard for entry creation and deletion brought the declaration that the biggest problem is lack of understanding of how to use MIBs and the suggestion that we need guidance for determining what is necessary to judge a resource's maximum capacity, current use, traffic by time, and reliability. The discussion wandered off into specifics of the X.25 MIB.

Two major topics came up regarding proper behavior of a Network Management System (NMS) in responding to the `tooBig` and column missing conditions. Some NMSs simply quit when they receive a `tooBig` error response. The statement that such behavior is broken, that an NMS should re-issue the request by splitting it into multiple requests, brought the counter-argument that silently splitting the request can cause skew in the relationship between counters read with separate messages. The problem is an NMS needs to know what can be split, and that burden is placed on the user. Including the retrieval of long strings that rarely change (e.g., the value of ifDescr) was suggested as the true culprit. The answer to the problem of missing columns was that NMSs have to deal with it. The response to the statement that mandatory objects MUST be there was the reminder that a row during its creation may be only partially defined and in any case real agents don't necessarily implement everything.

Chassis MIB WG

A new working group to define a MIB for chassis systems met in San Diego for the first time. Before the meeting, two proposals were sent to the mailing list.

At the meeting, the charter was reviewed and accepted. The primary job is to define a MIB for a chassis that can have multiple *virtual network devices* implemented over several physical slots that offer multiple subnetworks in the chassis backplane. A separate MIB for power supplies will be considered, as well as a lower-priority work item on aggregation (e.g., of statistics) over an administratively-defined group of network devices.
Request Address: `chassismib-request@cs.utk.edu`

Ethernet MIB WG

The mailing list received a detailed implementation report that provided relatively clear guidance for the San Diego meeting to improve the MIB. Many objects were removed due to lack of implementation interest, mainly those related to IEEE 802 LLC. A new Internet-Draft is available and will be the basis for the transition from Proposed to Draft Standard status.

Host MIB WG

After much interest at a BOF in San Diego, a new group was formed to define a MIB for host systems, primarily workstations. Relevant vendor MIBs were solicited via the mailing list.
Request Address: `hostmib-request@andrew.cmu.edu`

Multiport Repeater MIB WG

A discussion on the dynamic addition and removal of ports resulted in maintaining the current agreement that "group"s will normally correspond to the hardware configuration but are not required to do so.

Tracking of IEEE work was discussed and will continue closely, less the items already agreed to be omitted. The document was updated with various counter changes submitted to the list for discussion in San Diego.

A complaint that lack of a "hub ID" was inconsistent with the FDDI MIB and should be fixed was rejected since such action had been well discussed at previous meetings and is now a dead issue. The discussion got into ways to access multiple logical devices and was deferred to the Chassis MIB.

Some misunderstanding over the meaning of the `ifAdminStatus` and `ifOperStatus` objects engendered considerable discussion and use of OSI as an example. MIB-II was pointed out to be the proper example. The argument that the model could provide more information, namely whether an administratively disabled port is usable or broken, was suggested to be unworkable in practice, as such a distinction could not be sensed. This discussion continued in San Diego, and resulted in no change to the MIB.

At San Diego the IEEE-alignment updates were accepted. Much discussion of the `notPresent` value for `rptrGroupOperState` resulted in more explanatory text for the meaning of not present and allowing removal of an entry with that state. Discussion of total counters resulted in a gauge for total partitioned ports, total errors, and group counters for frames, octets, and errors. Totals across a whole repeater were rejected due to problems with the semantic definition as groups of ports come and go. A MIB for Medium Access Units (MAUs) was discussed, tracking IEEE work. It was deferred pending completion of the basic MIB, recognizing the problem of where to place it, as a separate MIB or in both Repeater and Ethernet MIBs, due to instancing issues. Pending satisfactory edits, the Repeater MIB is ready for consideration for elevation to Proposed Standard status.

PPP WG

A new draft of the PPP MIB resulted in objections to its model of layering into many `ifEntrys`. This was deferred to work being done by the Network Management Directorate.

The San Diego meeting resulted in a reduction from about 200 objects to a little over a hundred, with some to be added for AppleTalk and IPX. Many of the objects removed were there for debugging purposes. A new Internet-Draft is available.

Remote Network Monitoring (RMON) MIB WG

An interim meeting was held to lay out extensions for Token Ring and resulted in a first draft.

Concern over violating security by setting up an alarm drew the explanation that the setting of alarms is restricted to those MIB views which have the relevant read access.

The answer to a concern over proper operation when losing hosts during `HostTopNTable` duration resulted in the explanation that a report is based on the best information available and doesn't change once prepared. Such reports may include deleted hosts, and therefore corresponding entries in the `HostTable` can not be deleted if they are referenced by a report. Simply deleting a report when a host is deleted is a problem for reports of long duration.

To a suggestion that the history mechanism be expanded to include any object, the response was for future consideration. The present mechanism is a tradeoff between contents and packaging. Likewise, the alarm group should be a candidate for general operation.

At the San Diego Meeting it was decided to hold non-IETF test sessions after Spring Interop and next IETF. The Token Ring draft was revised and a release of a new Internet-Draft was planned by the end of April. For Token Ring, they decided to include a ring order table and they partitioned groups to be useful in non-promiscuous stations.

New Request Address: `rmonmib-request@lexcel.com`

SNMP over Multi-protocol Internet WG

A new working group was formed to handle issues of SNMP over transport protocol stacks other than UDP/IP.

The meeting agreed to many minor changes to the existing documents for mapping onto OSI, AppleTalk and XNS/IPX, and rejected working on mappings for SNA, Ethernet, or TCP. A how-to RFC is to provide general guidelines for this type of document.

On the mailing list, the concern that SNMP over Ethernet needs similar updates continued to receive little sympathy, the reference to "protocol suites of the multi-protocol internet" in the working-group's charter being interpreted not to include that. Specifying that all implementations on a specific transport stack are required to support a larger maximum message size can create problems when crossing network boundaries. Robustness suggests that implementations support the maximum for their transport stack, but not assume that other implementations support more than the normal 484 octets.

Request Address:
`snmp-foo-request@thumper.bellcore.com`

SNMP Security WG

The biggest question was

> "When will the documents become RFCs?"

Currently they are being reviewed by the IAB. Editorial changes are in process, with NO technical changes. The documents should move in perhaps a month.

The next biggest question was

> "How real is this?"

The RFCs are the gating factor. Two openly-available implementations are waiting to be released when the RFCs are official. In addition, interoperable implementations from SNMP Research and Hughes LAN Systems were demonstrated earlier this year.

A suggestion that dynamically adding MIB views is preferable to adding parties was met with the fact that a party maps to a single view and carries all the necessary security and proxy baggage, and views are not less trouble because of the need for additional configuration information. New party names can be assigned from any convenient OID subtree. Overall, security will be a big change for NMSs.

The simple answer to the question

> "Can an OCTET STRING hold a key or secret that is not a multiple of 8 bits long?"

is yes. While it is not currently relevant, since DES and MD5 use multiples of 8 bits, the protocol supports any future algorithm which might use such keys. It would be a simple matter of defining how the key is stored in the OCTET STRING for those algorithms.

X.25 MIB WG

A request for call parameters resulted in adding them to existing call information.

A suggestion that this MIB should wait for finalization of the ISO/CCITT International Standard (IS) was accepted only to the extent of staying interested.

At San Diego, the meeting made many minor changes to the LAPB and X.25 MIBs. As soon as satisfactory editing is accomplished, these will be ready to be considered for elevation to Proposed Standard status, To coordinate with the IETF's "IP Over Large Public Data Networks" WG, "IP over X.25" became "Multiprotocol Interconnect over X.25." The general changes made to the documents include adding DEFVALs and REFERENCEs and a `PositiveInteger` textual convention.

The new LAPB draft is available as an Internet-Draft.

Activities Calendar

- Interop 92 Spring
  May 18–22, Washington, DC
  For information: +1 415–941–3399
- Network & Distributed System Management
  June 15–17, Washington, DC
  For information: +1 310–394–8305
- 24th Meeting of the IETF
  July 13–17, Boston, MA
  For information: +1 703–620–8990

Publication Information

The Simple Times is published with a lot of help from the SNMP community.

Publication Staff

Coordinating Editor:
Dr. Marshall T. Rose — Dover Beach Consulting, Inc.

Technical Review Board:
Dr. Jeffrey D. Case — SNMP Research, Inc.
University of Tennessee
Keith McCloghrie — Hughes LAN Systems, Inc.
Steven L. Waldbusser — Carnegie Mellon University

Featured Columnists:
David T. Perkins — SynOptics Communications, Inc.
Robert L. Stewart — Xyplex, Inc.

Contact Information

Postal: ***The Simple Times***
c/o Dover Beach Consulting, Inc.
420 Whisman Court
Mountain View, CA 94043–2112

Tel: +1 415–968–1052
Fax: +1 415–968–2510
E-mail: st-editorial@simple-times.org
ISSN: 1060–6068

Submissions

The Simple Times solicits high-quality articles of technology and comment. Technical articles are refereed to ensure that the content is marketing-free. By definition, commentaries reflect opinion and, as such, are reviewed only to the extent required to ensure commonly-accepted publication norms.

The Simple Times also solicits terse announcements of products and services, publications, and events. These contributions are reviewed to only the extent required to ensure commonly-accepted publication norms.

Submissions are accepted only in electronic form. A submission consists of ASCII text. (Technical articles are also allowed to reference encapsulated PostScript figures.) Submissions may be sent to the contact address above, either via electronic-mail or via magnetic media (using either 8mm `tar` tape, 1/4in `tar` cartridge-tape, or 3-1/2in MS-DOS floppy-diskette).

Each submission must include the author's full name, title, affiliation, postal address, telephone, and fax numbers. Note that by initiating this process, the submitting party agrees to place the contribution into the public domain.

Subscriptions

The Simple Times is available via electronic-mail in two forms: PostScript and MIME (the multi-media 822 mail format). For more information, send a message to

```
st-subscriptions@simple-times.org
```

with a `Subject` line of

```
help
```

In addition, ***The Simple Times*** has numerous hard-copy distribution outlets. Contact your favorite SNMP vendor and see if they carry it. If not, contact the publisher and ask for a list. (Communications via e-mail or fax are preferred).