The Simple Times™

THE BI-MONTHLY NEWSLETTER OF SNMP TECHNOLOGY, COMMENT, AND EVENTS℠
VOLUME 2, NUMBER 4 JULY/AUGUST, 1993

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 19.

In this Issue:

Technical Article

Shawn A. Routhier, Epilogue Technology Corporation

In this issue: *Implementation Experience For SNMPv2*

This article outlines some insights I have gained while extending an implementation of an SNMP version 1 engine to include support for SNMPv2. My intention is to give advice and support to other SNMPv2 implementors and, if possible, to help clarify sections of the specifications that may be confusing to developers. Some of these comments have been voiced earlier, either on the SNMP mailing lists, or at the NM Open Area meeting held at the Amsterdam IETF. In reading this article, it's helpful to be familiar with SNMPv2, specifically the documents which deal with textual conventions (RFC 1443), security (RFCs 1445–7), and the protocol itself (RFC 1448).

The SNMP engine I extended is one of my company's products. We supply it to other companies as a toolkit; our customers then add more code to produce their own network manager or agent. Packaging the code as an engine rather than an agent or manager does allow it to be used in many environments; however, an engine implementation imposes two significant constraints:

- Since the actual end use for an engine is unknown, almost all of the SNMPv2 specification, including any optional capabilities (e.g., instance-level granularity) must be implemented. It also means that the code must be compartmentalized to allow our customers the ability to remove options they do not require for their applications.

- Since the code must be portable to many environments, it must work without being tightly coupled to a particular operating system. In SNMPv1, this coupling was not an issue. In SNMPv2, the addition of a number of objects that should reside in non-volatile memory makes the relationship to the operating system more significant.

Let's look at two key areas: coding experience and resource requirements. For each, there are a number of considerations. We'll start with coding issues.

Wrappers

Writing the encoding and decoding wrappers for privacy and authentication was straightforward, although it did require reading the specification several times to ensure that all the data objects (such as time stamps or party object identifiers) were in the correct order. The only difficulty encountered was encoding the `privData` section of the `SnmpPrivMsg`. Unlike the messages, which are implicit sequences, the `privData` section is an `IMPLICIT OCTET STRING` and so should be encoded as an 81 hexadecimal instead of an A1 (or A2 in the case of the `SnmpMgmtCom` message).

Get-Bulk

Implementing the `get-bulk` operator was relatively simple. The only difficulty I encountered was from misreading part of the specification. The `get-next` operator may return an error code indicating a response packet would be too big; in contrast, the `get-bulk` operator will either return a packet, perhaps with no variable bindings; otherwise, if this "empty" packet is still too large, drop the packet. In my original implementation, both operators were handled by the same routine, but the code didn't account for this difference, so the `get-bulk` operator could return the "too big" error. Adding code to test for an empty response for the `get-bulk` operator is easy; the difficult part is reading the specification closely enough to notice that an invocation of the `get-bulk` operator is not precisely equal to repeated invocations of the `get-next` operator.

Given the ease of implementing the `get-bulk` operator and the expected performance gain, it easily passes the "thrust/weight ratio" test.

Row Creation

The definition of the row status textual convention provides a standard way to manipulate rows. It is complicated and can be difficult to implement correctly, especially if dribble style creation (creating a conceptual row with one object per packet) is allowed. The major difficulty is determining whether the row being created is consistent. (Consistent means being able to go to a state of "not in service" or "active" as defined in the MIB that describes the row.) One method of simplifying the consistency check is to use a temporary storage area to collect all information about the row from both the PDU and the partially created row, if any. This makes it easier to perform consistency tests because it is no longer necessary to hunt through the PDU for each specific item. The difficulty of the consistency checks will, however, depend on the definitions in the MIB. As more linkages are added between items, especially items that are not in the same table, the consistency checks will become larger and more difficult to write. So this becomes a MIB design problem; MIBs should be designed to use as few dependencies as possible to solve their specific problem.

Views

Views, especially views involving instance-level granularity, have been the subject of much discussion, and most of that discussion has been about possible performance problems. With these discussions in mind, I started thinking about how to implement views in a computationally-efficient manner.

I examined several schemes, but discarded all of them as being too complicated and (because I did not know how views will be used in the final application) of uncertain usefulness. Instead, I implemented a simple view check routine which I could optimize if operational experience demonstrated a need for more efficiency. The routine can handle instance-level granularity and seems to deliver acceptable performance in limited testing (using my test-bed and with moderate views). To date, the test criterion has been the "feel" of a manager-agent interaction, and so far the response time has been shorter than the human response time. More performance testing is needed, but more important will be data about operational systems and the views that they use. It may be that most entities will find a simple view check scheme sufficient for their purposes, and only entities with the most complex views will need a more involved scheme to avoid the performance penalties from the larger view families.

It was pointed out on the SNMP mailing list that dynamically mountable subtrees may cause conflicts with views if the entity does not support instance-level granularity. The conflict arises if, before a subtree is mounted, a view is set that would require instance-level granularity of the subtree. Several solutions were proposed: limit the views such that none of them will have instance-level granularities, disallow mounting of any subtree for which views with instance-level granularities exist, or, when mounting a subtree trim any branches that would cause a view to have an instance-level granularity.

Clocks

The use of 32 bits for clocks (4 billion units) presents some difficulties for machines with 32–bit words and clocks. In my original implementation I planned to have one system clock; each party would have a single 32–bit offset which would be added to the system clock to determine the authentication clock. After thinking

about this for some time I was unable to find a method to allow use of the full range and to detect clock wrap using a single 32–bit offset, and concluded that at least 33 bits were required. I elected to use two 32-bit words to keep track of the time stamps. If the clocks were reduced to 31 bits (2 billion units) then a single 32–bit offset would be sufficient, although the code would still be more complicated than simply adding an offset to the clock and testing the result.

Parties

I had some trouble interpreting the section of the specification dealing with creating parties. I interpreted "created" in the statement

> "Once an instance of this object is created, its value cannot be changed."

from the `partyAuthProtocol`, `partyAuthLifetime` and, `partyPrivProtocol` objects to mean when the row attained a status of either "not in service" or "active". Discussions on the mailing list indicated that it should mean when that individual object is written, either directly by a set request for that object or indirectly by a set request for the `partyCloneFrom` object.

Contexts

Another section of the specification I found unclear is the one detailing the interaction between the proxy information and view index in a context entry. RFC 1447 requires a consistent (status of "active" or "not in service") context entry to have a non-zero view index or, if the view index is zero, some proxy information. If the view index is non-zero the RFC states that the value of an instance of `contextProxyDstParty`, `contextProxySrcParty`, and, `contextProxyContext` is 0.0. It does not state whether the objects themselves are modified or if they are changeable while the view index is non-zero. In my implementation I choose to return 0.0 but to maintain the underlying objects. In addition, I allow these objects to be set while they are in this state. This was done to allow a manager to configure the proxy information of an existing context and then change the view index to zero to use the proxy functions.

Proxys

While adding code in the engine to support proxies I noticed a need to maintain some state even for native proxies. Originally, I thought that the information kept in the proxy section of the context entry would be sufficient; unfortunately, a packet may use a source address that does not match the address given in the source party, so the source address of the original request must be saved for use as the destination address of the final response.

Access Control

In implementing the ACL table I was unsure if an ACL entry should be created if the parties and contexts it referenced were either not ready for use or non-existent. In many tables referencing an entry that doesn't exist yet would not be a problem as the reference is unambiguous. However, an ACL entry references parties and contexts by their index which is not user-settable. I concluded that, for the indexes to properly reference a party or context, the party or context must already exist and must be in either the "not in service" or "active" states, and implemented the ACL routines to enforce this conclusion.

Method Routines

I was able to make all of the changes to the main engine without needing to modify any of my method routines. I believe this was one of the goals of the original SMP specification. However, in order to take full advantage of the expanded error semantics of SNMPv2, some method routines will need to be modified slightly to use the new error codes defined in SNMPv2. (With many engine designs only the `set` type of method routines will need modification.)

I am also looking into giving the `get` and `get-next` type of method routines a hint that the `get-bulk` operator is being executed. The method routine for a complicated table might be able to take advantage of this information. For example, it might build a sorted list and cache the result for the benefit of the next call.

Now let's look at the impact SNMPv2 had on the resource requirements of my implementation.

Configuration

As many people expected, configuration is difficult. Hand-configuring a small test-bed (several machines on the same local network) was painful and prone to errors. I believe that end-users will reject any scheme that requires them to do such configuration on a large network, and that we as a community will need to provide ways to do most of the configuration automatically. Presumably we will use several types of tools to solve different pieces of the problem. Perhaps we will need one tool to configure a new system before installation, another to update configurations over the network, and ways to allow both our tools and the systems themselves to do some amount of automatic configuration.

At the recent NM Open Area meeting held at the Amsterdam IETF, some different ideas on automating configuration were presented. However, this work is preliminary.

Non-Volatile Memory

The amount of non-volatile memory required greatly depends on the configuration chosen. A minimal agent (supporting only the no auth/no priv initial information specified in RFC 1447) should need little or no non-volatile memory. A manager for a campus-wide network could use huge amounts of non-volatile memory. Clever coding may reduce the amount of non-volatile memory required, but real reductions will require finding ways to shrink the number of entries required for configuration. Using proxies, generating entries dynamically or algorithmically, and delegating tasks to a lower-level manager are all potential methods for reducing the number of entries required. Trying to maintain a correct and consistent configuration while using these methods is another reason for trying to automate most of the configuration task.

Code Space

My implementation grew a good deal with the addition of SNMPv2. However, almost all of the growth was in the method routines to handle the Party MIB objects and the routines to implement the databases to hold these objects. The growth due to the new wrappers, the `get-bulk` PDU, and the added ASN.1 types was minimal. A minimal agent could be built without most of the party MIB method routines and database. A manager would not be able to modify such an agent's party MIB structures, but the agent would be only slightly larger than an SNMPv1 agent. If an agent already supports a large collection of MIB method routines, then the relative increase in size, including the party MIB, will probably be moderate.

Data Space

The data space requirements, like the non-volatile memory requirements, will depend on the configuration. Again, clever coding can reduce the memory required, but the primary question will be how large and complex a configuration is needed. Some of the methods that may help reduce the non-volatile memory requirements would also reduce the memory requirements in general.

In Summary

In general I found the specifications (RFCs 1441 to 1453) somewhat dry and sometimes difficult to read. I attribute this difficulty to an attempt by the authors to close any loopholes that could allow someone to create a non-interoperable program. (No, I do not have a solution to this problem.) I did find the examples useful and encourage their continued inclusion and perhaps an expansion where reasonable.

I think that the specifications are implementable. There are some sections that are not obvious and may lead to some implementation differences, but they should not affect interoperability at the protocol level. They may present difficulties at the network management system level (e.g., given my earlier comments on contexts, imagine a manager expecting to be able to modify the `contextProxyDstParty` while the view index is non-zero and the agent not allowing this event).

The resources required for an SNMPv2 implementation will be greater than for an SNMPv1 implementation but many of the increases may be avoided by systems if desired. However, the price will be an accompanying reduction in the services offered by the implementation.

We need operational experience to determine the costs and benefits of some features, such as views — especially instance-level granularity. We also need better data about traffic patterns under real loads to better determine the performance effects of security and features like the `get-bulk` operator.

Industry Comment

Marshall T. Rose

In this issue: *SNMPv2 and MIB Modules*

As the IESG Area Director for Network Management, I publish a monthly *State of the Area* report which contains, among other things, information on what's going on in the IETF with respect to network management standardization activities. This report is published on the IETF and SNMP mailing lists.

One topic contained in the report is the AD's policy on how working groups developing MIB modules must be mindful for coexistence between SNMPv1 and SNMPv2. In this issue of ***The Simple Times***, we'll take a look at this policy.

The Need

Both versions of SNMP use a language termed the "Structure of Management Information" (SMI) to describe objects which are managed. SNMPv2 was very

carefully designed to minimize interworking problems with SNMPv1. For example, an implementor can take an SNMPv1 product, add support for SNMPv2, and not have to change *any* of the code which implements MIB support in the product. This means that any MIB module compliant to SNMPv1's SMI will work with an SNMPv2 product.

However, there is the question as to how MIB modules compliant with SNMPv2's SMI will work in an SNMPv1 product. There are two competing pressures here: SNMPv2's SMI has a number of features which make it superior for defining managed objects, so it would be good to start using the newer SMI right away; on the other hand, there is a large number of existing MIB modules written using SNMPv1's SMI, and it really doesn't make any sense to invalidate those.

RFC 1452, the SNMPv1/SNMPv2 coexistence document, describes the differences between two versions of the SMI. Fortunately, there are very few features in SNMPv2's SMI which are incompatible with SNMPv1. So, with some care, we can devise a policy which will allow existing MIB modules to be relatively unaffected by SNMPv2, whilst newly-defined MIB modules can take advantage of the new features.

The Policy

The policy, which applies only to working groups in the IETF which produce MIB modules, consists of three stages.

In the first stage, which was in effect from April (when the SNMPv2 documents were published) until the beginning of August, all MIB modules were required to use SNMPv1's SMI (as defined in RFCs 1155 and 1212). However, each working group was responsible to take special care in minimizing the transformation necessary to use SNMPv2's SMI (RFC 1442).

The goal of the first stage was simply to provide a four months' "heads-up" to the Internet community.

The second stage is in effect until SNMPv2's SMI becomes a Draft standard, which (and I am guessing here) probably won't be until April of next year. In this stage, MIB modules are required to use SNMPv2's SMI, but are not allowed to use those parts of the SMI which are not compatible with SNMPv1. This means that any MIB module standardized during the second stage would work with an SNMPv1 agent. In addition, working groups developing MIB modules must now start writing conformance statements using the mechanisms provided by SNMPv2.

The impact of the second stage on existing MIB modules is quite minimal — a change in syntax, not semantics. Basically, the prefix of each module changes to import definitions from RFC 1442, instead of RFCs 1155 and 1212. In addition, the syntactic skeleton of each managed object slightly changes. (In fact, these changes can be entirely automated by a simple program.) Of course, any new MIB modules defined will also work with either version of SNMP. It is important to note that not a single fielded product will have to be changed as a result of the second stage.

However, if a working group can demonstrate a clear need to use the full SMI for SNMPv2, they can request a waiver. Such a waiver has been granted to the Interfaces MIB working group, which needs to use 64–bit counters for some objects.

The final stage goes into effect when SNMPv2's SMI becomes a Draft standard. In this stage, all of the features of SNMPv2's SMI may be used. Of course, "may" needn't imply "must". Existing MIB modules which meet the requirements of the second stage need not be changed at all. So, these MIB modules will continue to work with both SNMPv1 and SNMPv2. However, new MIB modules, those which require the advanced features of SNMPv2's SMI, will work only with SNMPv2. Of course, only new products will deploy these newly-defined MIB modules, and these will be the products which implement SNMPv2.

The focus of the policy is clear: the existing SNMPv1 infrastructure is largely unaffected — no products need be changed. However, it is important to gain experience using SNMPv2's SMI now, so that SNMPv2 can progress along the standards track. (To achieve Draft standard status, implementation, deployment and interoperability experience must be demonstrated.) Finally, note that existing MIB modules need be upgraded to stage two only when they are considered for advancement.

Experiences

Now that we have entered stage 2, two issues have emerged.

First, although compilers for SNMPv2 have emerged in product, many products remain capable of dealing only with SNMPv1's SMI. To remedy this, a V2–to-V1 SMI translation tool will be made openly-available. This will take a MIB module written under the second stage (using the subset of SNMPv2's SMI) and convert it to a MIB module written using SNMPv1's SMI. So, when a MIB module is published under the second stage (which will be virtually all existing MIB modules), an "auxiliary" copy will be made available for products which understand only SNMPv1's SMI.

Second, SNMPv2 refines SNMPv1's concept of a "textual convention", which is a way of adding semantics to a data type. Of the textual conventions initially

defined by SNMPv2, only one, RowStatus, makes use of an SNMPv2–specific protocol feature. New MIB modules written under the second stage may want to make use of this textual convention. So, an SNMPv1–compatible version of RowStatus is being developed. This will allow newly fielded SNMPv1 products to make use of these MIB modules.

Applications and Directions

Steven L. Waldbusser

In this issue: *Why Plug and Play Networks are Hard to Manage*

Plug and play networks are crucial to the continued growth of networking, especially into non-technical environments. Plug and play promises to eliminate the requirement for assigning addresses and configuring hosts by hand. This may also be attempted for routers and servers. For small networks this means that a network can be set up without hiring or consulting with a network expert. Larger networks will still require network expertise to install and manage the network, but plug and play can simplify some tasks — while making others more complex.

Many sites have had experience with the plug and play nature of AppleTalk. The IETF is nearing completion of the Dynamic Host Configuration Protocol to add plug and play features to TCP/IP, and other protocol suites have these features as well. One should also consider that many routing, naming, and address resolution protocols have similar dynamic configuration characteristics. As the use of plug and play grows and as larger networks are built with it, it is increasingly important to understand how to manage it and what problems it adds to the network management puzzle.

Transient Addressing

A general difficulty in managing networks with dynamic address assignment is that the network address is transient, and therefore not suitable for identifying a host. This complicates many network management tasks. Most network managers (and network management systems) tend to identify a host by its network address and often by a name associated with that address (such as a domain name associated with an IP address). The network manager can assume that when communicating to that address that a particular host is responding. For example, if a file server has the address 128.2.10.2, the network manager will assume that SNMP commands sent to that address are monitoring or configuring that server. In fact, an icon on a network map may be devoted to that server at that address. None of this is possible with dynamic addresses. When network addresses can change every time a host reboots, the network manager must rediscover the network address of the host, potentially before each management operation.

If the address can change dynamically, the manager needs to find the correct network address associated with the system it needs to manage. The first thing the manager needs to do use choose a different "handle" than the network address to identify the system. The most readily available value is the hardware address of the system but sometimes a serial number is more appropriate. If using the hardware address, be aware that it might change if someone changes the network card. With this identifying attribute, the manager needs to discover the correct network address. If the address is allocated by a server, that would be the most direct place to get this information. Otherwise, one could consult other sources of information such as a router's ARP table (or AARP table for AppleTalk) or a remote monitoring device.

Once the manager has discovered the correct network address and is communicating with the managed system, it will continuously need to verify that the addressing has not changed causing requests to go to the wrong system. To achieve this, the manager will need to verify that the system it is communicating with has the expected "handle". If this is the hardware address, for example, every GET operation could include a request for ifPhysAddress. If this ever changes, the response would be discarded and the manager would re-discover the network address.

Dynamic addressing clearly introduces some headaches into the management process that managers (human or software) are not yet ready to handle. However, one can get many of the benefits of dynamic addressing while relieving the management burden by dynamically assigning addresses at installation time and keeping the addresses constant throughout the life of a machine. This is possible with server-based address allocation schemes such as TCP/IP's Dynamic Host Configuration Protocol (unfortunately this isn't the case for AppleTalk's DDP). This allows a user to unpack a new machine, plug it in and have it acquire a network address without configuration. From that point, the manager can configure that address into the management system without fear of it changing.

It should also be pointed out that this is not widely a problem now because most management is performed using SNMP over TCP/IP. Because TCP/IP does not have dynamic addressing, the manager can be fairly confident that the network address is not changing. Most AppleTalk environments don't have this problem because

network management is performed using SNMP over TCP/IP. While the AppleTalk address of a managed system may change, the IP address stays constant, solving the dynamic addressing problem. The problems above can be encountered when using SNMP over AppleTalk, but this new standard includes a simple mechanism for discovering the network addresses when they change.

The Dangers of Auto Configuration

There are many other pieces of configuration information required to get a system running on the network (default routers, file servers, naming information, and so on). In plug and play networks, these are commonly provided by a server that answers network requests for configuration information.

A common problem on large networks is for unauthorized configuration servers to be attached to the network. These servers provide incorrect information to hosts on the network and can misconfigure many hosts before they are discovered and shut down. Often these servers are installed as the innocent byproduct of installing a (misconfigured) file server or router on a network (in the case of AppleTalk, all routers and many file servers provide configuration information about routing, ZIP, and NBP to any client which asks).

The Catch-22 in this situation is that the obvious way to solve this problem is to provide authentication between the hosts and the servers. Unfortunately, this requires that an authentication key be manually configured in the host, and we were trying to avoid manual configuration in the first place! In the end, we might find that configuration of a single authentication key is not burdensome (especially with public key technology) and will solve the problem of unauthorized servers.

In the meantime, it will be helpful for the network manager to keep track of all servers on the net that are providing configuration information. If any unauthorized servers are discovered, they should immediately be shut down before they corrupt the configuration of hundreds or thousands of hosts that can no longer access network services.

As an example, routers that advertise routes are providing configuration information to hosts on the net. If a host incorrectly advertises a route and is incapable of forwarding those requests, traffic to one or many networks may cease. At Carnegie Mellon, RIP broadcasts are monitored. If any unauthorized hosts are sending routing packets they are shut down. It is often the case of a UNIX host running `routed` even though it is not a router. The policy at CMU is that this is an accident waiting to happen and the `routed` process must be stopped.

Finally, some server and router products support auto-configuration. This is ill-advised in many cases, especially for large networks. The implications of auto-configuring hosts are bad enough, but routers and servers should be robust enough to be counted on if all else fails. It is hard for a router to provide effective firewall protection if it becomes part of the problem.

Handle with Care

Plug and play is very important to networking. But it can cause problems with network management, especially in large networks. There are a few things that can be done to make things easier, but in any case, it should be handled with care.

Ask Dr. SNMP

Jeffrey D. Case

Dear *Dr. SNMP*,
I am concerned that the security features of SNMPv2 will render it much slower than SNMPv1. Can you comment on that?
— *Speed-Demon from Spokane*

Dear *Speed-Demon from Spokane*,
Down on the farm, we have a saying:

> "It's faster than a six-legged jack rabbit."

The first major public demonstration of SNMPv2 interoperability occurred in April at the International Federation of Information Processing (IFIP) International Symposium on Integrated Network Management in San Francisco, California. This was the first time that Technology Centers were a part of this bi-annual conference on network management. The SNMPv2 Technology Center demonstration showed many of the new features of SNMPv2. These included demonstration of the speed of SNMPv2 with security compared with SNMPv1 without security. This demonstration utilized a visual tool inspired by Digital Equipment Corporation's "bricks demo," which is famous for illustrating the performance advantages of FDDI compared to Ethernet. It was easy to see that the improved performance of SNMPv2's `get-bulk` operator more than offset the added cost of authentication. When retrieving large tables, SNMPv2 with authentication was more than ten times faster than SNMPv1 without authentication. SNMPv2 with `get-bulk` and without authentication was even faster. Even when the most expensive security features were enabled (authentication plus software-based encryption to afford privacy), SNMPv2 was still approximately twice

as fast as SNMPv1 which provides neither authentication nor privacy.

Your mileage may vary, because exact results are dependent upon the maximum packet size supported, the number of columns in a table, and the sizes of each data element retrieved. However, in any case, SNMPv2 should quell any complaints about the time it takes to retrieve large tables using the Internet-standard Network Management Framework based on the SNMP.

Dear *Dr. SNMP*,
Now that the SNMP version 2 documents are published, what features are you planning for SNMP version 3?
— *Journalist from Jupiter*

Dear *Journalist*,
Down on the farm, we have a saying:

> "Now there's something that's about as welcome as a skunk at a lawn party."

I know that as soon as you finish writing something, you must immediately begin thinking of next week's edition. However, not everybody lives that way. To the best of my knowledge, there are no dreams about doing this again. Those are nightmares.

Dear *Dr. SNMP*,
Why is it that you keep referring to "down on the farm" in your replies? Don't you live in Knoxville? Isn't this a city with a big university (the University of Tennessee, Knoxville) and a prestigious college football team (the VOLS)? This doesn't sound like "farm" to me, it sounds like "town". What's the story? Inquiring minds want to know!
— *Mystified in Manhattan*

Dear *Mystified in Manhattan*,
Down on the farm, we have a saying:

> "We live so far out in the country that some of our neighbors think that a seven course dinner is a six-pack and a possum."

While I do not think barbecued possum is a treat, the neighborhood is full of wonderful, nice folks.

We have a Knoxville postal address, but we do not exactly live "in Knoxville". We are actually about 30 minutes outside of Knoxville, in the foothills of the Smoky Mountains, on a 75–acre farm.

Security and Protocols

Keith McCloghrie

In addition to implementing new functionality, the upgrade of an agent from SNMPv1 to SNMPv2 requires a number of changes to the existing code. Fortunately, only one of these has any impact on the object-specific code (the "method routines") already written for existing MIB objects. This change concerns the use of error codes. In this article, we'll examine the new error codes, and how to use them.

SNMPv1 error codes

First, let's recall SNMPv1's use of error codes. SNMPv1 specifies four error codes for use by agents in the header of a response PDU:

- `tooBig` – the response PDU was too big to transmit;
- `noSuchName` – there was something wrong with the name of a variable in the request;
- `badValue` – there was something wrong with the value of a variable in a set request; and,
- `genErr` – something else was wrong.

(`readOnly` is also defined, but there are no circumstances defined in RFC 1157 under which an SNMPv1-compliant agent will generate `readOnly`).

There are two issues with these codes: one, the use of "something" in most of the above cases; that is, in SNMPv1, an agent can do no better than be rather vague in telling the management station what kind of error occurred. Two, whenever anything is wrong, the whole request fails; this is particularly frustrating on retrieval requests where the ability to retrieve the values of the individual variables is normally independent.

SNMPv2 error and exception codes

SNMPv2 addresses both of these issues by defining thirteen new error codes, and introducing the notion of exception codes.

Exception codes are used in responses to retrieval requests to indicate a condition affecting the retrieval of a specific variable, so that the response can include the values of any/all other variables which can be retrieved. To achieve this, the exception is not indicated in the header of the response PDU, but rather as a specially tagged value. Three such special tags are defined. All three are used in place of the SNMPv1 `noSuchName` error-code. Two of them, `noSuchObject` and `noSuchInstance`, allow an agent responding to

a `get-request` PDU, to provide distinction between a MIB object which is not implemented, and one for which the requested instance does not exist. The third, `endOfMibView`, allows an agent responding to a `get-next-request` or a `get-bulk-request` PDU, to indicate that there is no lexicographical successor. Note that each of these refers to the presence of objects/object instances in the relevant MIB view. No distinction is made between the presence or not of objects/instances outside the MIB view (for the same reason as login sequences typically prompt for a password even when the username is unknown).

Two of the SNMPv1 error codes are retained with their same meanings: `tooBig` and `genErr`. However, their usage varies slightly: in SNMPv2, the use of `tooBig` specifies that the `response` PDU is formatted with an empty variable bindings list, which increases the probability of being able to send the response; and `genErr` now represents a smaller set of conditions since it excludes those identified by the new error codes.

The remaining SNMPv1 error codes, `noSuchName`, `badValue`, and `readOnly`, are also retained, but not for use by SNMPv2 agents. Rather, they are retained so that proxy agents, between an SNMPv2 manager and an SNMPv1 agent, can forward a response PDU without changing the old error codes since no exact translation is possible.

Of the new error codes, one, `authorizationError`, indicates that the particular PDU type is not authorized for the particular combination of source party, destination party, and context; for example, if an agent is configured so that `set-requests` are only authorized for authenticated requests, then a `set-request` sent using noAuth/noPriv parties, would generate an `authorizationError`.

The other twelve new error codes are used only in response for sets. They are either permanent or transient error conditions; The permanent conditions are:

- `noAccess` – the variable is not accessible by this request;
- `noCreation` – the variable does not exist and cannot be created;
- `notWriteable` – the variable is read-only;
- `wrongType` – the value specified has the wrong (ASN.1) type for the variable;
- `wrongLength` – the value specified is too long/short for the variable;
- `wrongEncoding` – the value specified is incorrectly encoded; and,
- `wrongValue` – the value specified cannot be assigned to the variable.

The transient conditions are:

- `inconsistentName` – the variable does not exist and cannot be created at this time;
- `inconsistentValue` – the specified value cannot be assigned at this time;
- `resourceUnavailable` – a resource required in order to assign the value is presently unavailable;
- `commitFailed` – the actual assignment of this variable failed even though the maximum amount of pre-checking of the assignment was successful; however, the agent was able to return all variables to their state prior to starting the assignments; and,
- `undoFailed` – an error occurred in undoing the assignment of this variable after an assignment failed even after the maximum amount of pre-checking of all assignments had been successful.

Note that the specification strongly encourages implementations to take all possible measures to avoid use of either commitFailed or undoFailed. These two error codes are NOT defined to allow lazy/sloppy implementations.

Changes to method-routines

A typical SNMPv1 agent implementation consists of a "protocol engine" which is independent of the specific MIB objects supported, plus a "mib tree" table which contains information about specific MIB objects, including pointers to various "method routines". Such method routines are called in the appropriate processing phase of the various types of request. Examples of method routines are: a routine which returns the lexicographically next instance, a routine which checks that a new value is acceptable, a routine which assigns a new value, and so on.

To upgrade an existing SNMPv1 agent, the definition of the new error codes and exception tags typically requires no changes to the method routines used for retrieval. For the set operation, the only change required is typically the additional ability, on the occurrence of certain types of errors, to indicate which of the new error codes is applicable. These types of errors are likely to be: `noCreation`, `wrongLength`, `wrongValue`, `inconsistentName`, `inconsistentValue`, and, `resourceUnavailable`.

Bi-lingual agents

Bi-lingual agents are agents which implement both SNMPv1 and SNMPv2. The use of bi-lingual agents is not recommended by RFC 1452. Instead, RFC 1452 specifies two strategies for the transition and co-existence between SNMPv1 and SNMPv2: one, the use of proxy agents; and, two, the use of bi-lingual managers. However, for the short-term, when few networks have management stations supporting SNMPv2, agent implementors are likely to provide bi-lingual agents.

Fortunately, due to the minimal changes to method routines required by SNMPv2, as noted above, bi-lingual agents need have only one set of method routines. These method routines are the SNMPv2-upgraded routines which return SNMPv2 error codes. They can be called irrespective of whether an SNMPv1 or SNMPv2 request is being processed. The protocol engine can then, for SNMPv1 requests, translate an SNMPv2 error code into an SNMPv1 error code to be included in the SNMPv1 response PDU. In particular, `noCreation` and `inconsistentName` can be mapped to `noSuchName`; `wrongLength`, `wrongValue`, and, `inconsistentValue` can be mapped to `badValue`; and, `resourceUnavailable` can be mapped to `genErr`.

Standards

David T. Perkins

In the past two months, two previously published documents have been updated and republished. These are the Bridge MIB module and the Network Management Tools Catalog. The authors and chair of the Bridge MIB module should be congratulated for progressing their document to the middle rung on the standards ladder. And all of you who know of network management tools not in the catalog, please send an e-mail message as instructed in the document to add or update entries.

A family of four documents describing point-to-point protocol (PPP) has been published. And as predicted in the previous column, the RFC that defines managed objects for multiprotocol interconnect using X.25 was published.

Many MIB modules are in the final stages of review. These include but are not limited to: DNS, Host Resources, Token Ring RMON, and Bridge source-routing. There are many active working groups (WGs) developing MIB modules. All the development activity and the preparation for transition to the SNMPv2 SMI has caused a strain on the people resources needed to ensure high-quality IETF MIB modules. To help get the already started work through the system, the IESG Area Director for Network Management has imposed a moratorium on starting new MIB working groups until after the beginning of 1994. Too many MIB modules are at the first rung (Proposed) on the standards ladder. Going to the next rung (Draft) is the most difficult, since it requires multiple independent complete implementations, operational experience, and interoperability testing of all aspects of the MIB module. A delay in starting new MIB WGs should allow the existing developments to complete, and allow for the focus needed to move MIB modules along the standards-track!

Recently Published RFCs

RFC 1461 - X.25 MultiProtocol Interconnect MIB (Proposed standard)
This MIB module defines objects used for managing Multiprotocol Interconnect traffic carried over X.25 as defined by RFC 1356. Systems that implement the encapsulation of multiprotocol packets (IP, CLNP, ES-IS, or SNAP) in X.25 frames as defined by RFC 1356 should implement this MIB module. The MIB module consists of three tables. The first table, `mioxPleTable`, defines information for each interface used to carry the traffic. The second table, `mioxPeerTable`, defines information about each possible peer that may exchange traffic with the system. The last table, `mioxPeerEncTable`, defines each encapsulation with each peer.

RFC 1470 - A Network Management Tool Catalog (Informational)
This is a resource book listing openly-available and commercial products spanning all aspects of network management.

RFC 1471 - PPP Link Control Protocol (LCP) MIB (Proposed standard)
This MIB module defines the core objects for managing the Point-to-Point Protocol (PPP). The Link group contains two tables, `pppLinkStatusTable` and `pppLinkConfigTable`. These provide information for each PPP interface. The LQR group contains two tables for the basic status and configuration objects that apply to the PPP LQR protocol. The LQR extensions group contains a table with the most recently received LQR for each PPP interface. Also defined are the identity of PPP tests to be used with `ifExtnsTestTable`.

RFC 1472 - PPP Security MIB (Proposed standard)
This MIB module defines two tables used for security in PPP. The first, `pppSecurityConfigTable`, specifies the preference of security protocol for each possible PPP interface. The second, `pppSecuritySecretsTable`, contains the security information about each active identity for each PPP interface.

RFC 1473 - PPP IP Network Control Protocol (NCP) MIB (Proposed standard)
This MIB module defines the objects for using IP over PPP. It consists of two tables. The first, `pppIpTable`, contains IP parameters and statistics for the local PPP entity. The second, `pppIpConfigTable`, contains administrative and compression information.

RFC 1474 - PPP Bridge Network Control Protocol (NCP) MIB (Proposed standard)
This MIB module defines the objects that support Bridging over PPP. The `pppBridgeTable` has status information about each PPP interface used for bridging. Configuration information for each PPP interface used for bridging is specified in objects contained in `pppBridgeConfigTable`. The status and configuration for each MAC type on each PPP interface used for bridging is specified in objects in `pppBridgeMediaTable` and `pppBridgeMediaConfigTable`.

RFC 1493 - Bridge MIB (Draft standard)
This is an updated version of RFC 1286. The major change was to remove the definitions of the source-route bridging objects. These are to be specified in another MIB document. There were also several other changes which were cleanups to the MIB module.

Standards Process Update

In a previous issue of ***The Simple Times***, this column reported on the IETF standards process. Since then, the POISED effort has resulted in changes to the process of how the IETF is governed. RFC 1310, the current specification for the IETF standards process is being updated to reflect the new procedures and to incorporate improvements to the current process. One such improvement is a new classification for some RFCs, termed "Prototype." This will be a welcome addition if it is not abused. Organizations developing for-profit products are generally not interested in experimental RFCs — usually a technology must be on the standards-track before a company will adopt it. The Prototype label is meant for those specifications that describe a technology destined to become a standard, but due to complexity or lack of rough consensus, needs implementation experience to move forward.

Note that an Internet-Draft is a work in progress. As stated in the successor to RFC 1310:

> "Under no circumstances should an Internet-Draft be referenced by any paper, report, Request-for-Proposal, nor should a vendor claim compliance with an Internet-Draft."

In contrast, an RFC with the Prototype label may be referenced and the technology deployed in a product. This label may be abused if it is used to circumvent the review by a working group. It may also be the "compromise solution" for WGs that deadlock over fundamental issues. To keep from being abused, this label should be used infrequently.

In the next issue, we'll look at the challenges in the standards area as we transition to SNMPv2's SMI.

Summary of Standards

SNMPv1 Framework (Full Standards):

- 1155 - Structure of Management Information (SMI);
- 1157 - Simple Network Management Protocol (SNMP);
- 1212 - Concise MIB definitions; and,
- 1213 - Management Information Base (MIB-II).

SNMPv2 Framework (Proposed Standards):

- 1441 - Introduction to SNMPv2;
- 1442 - SMI for SNMPv2;
- 1443 - Textual Conventions for SNMPv2;
- 1444 - Conformance Statements for SNMPv2;
- 1445 - Administrative Model for SNMPv2;
- 1446 - Security Protocols for SNMPv2;
- 1447 - Party MIB for SNMPv2;
- 1448 - Protocol Operations for SNMPv2;
- 1449 - Transport Mappings for SNMPv2;
- 1450 - MIB for SNMPv2;
- 1451 - Manager-to-Manager MIB; and,
- 1452 - Coexistence between SNMPv1 and SNMPv2.

Full Standards:

- 1213 - Management Information Base (MIB-II).

Draft Standards:

- 1398 - Ether-Like Interface Type MIB; and,
- 1493 - Bridge MIB.

Proposed Standards:

- 1229 - Extensions to the generic-interface MIB;

- 1231 - IEEE 802.5 Token Ring 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;
- 1285 - FDDI Interface Type MIB;
- 1289 - DECnet phase IV MIB;
- 1304 - SMDS Interface Protocol (SIP) Interface Type MIB;
- 1315 - Frame Relay DTE Interface Type MIB;
- 1316 - Character Device MIB;
- 1317 - RS-232 Interface Type MIB;
- 1318 - Parallel Printer Interface Type MIB;
- 1354 - SNMP IP Forwarding Table MIB;
- 1368 - IEEE 802.3 Repeater MIB;
- 1381 - X.25 LAPB MIB;
- 1382 - X.25 PLP MIB;
- 1389 - RIPv2 MIB;
- 1406 - DS1/E1 Interface Type MIB;
- 1407 - DS3/E3 Interface Type MIB;
- 1414 - Identification MIB;
- 1418 - SNMP over OSI;
- 1419 - SNMP over AppleTalk;
- 1420 - SNMP over IPX;
- 1461 - Multiprotocol Interconnect over X.25 MIB;
- 1471 - PPP Link Control Protocol (LCP) MIB;
- 1472 - PPP Security Protocols MIB;
- 1473 - PPP IP Network Control Protocol (NCP) MIB; and,
- 1474 - PPP Bridge Network Control Protocol (NCP) MIB.

Experimental:

- 1187 - Bulk table retrieval with the SNMP;
- 1224 - Techniques for managing asynchronously generated alerts;
- 1227 - SNMP MUX protocol and MIB;
- 1228 - SNMP Distributed Program Interface (SNMP-DPI); and,
- 1238 - CLNS MIB.

Informational:

- 1215 - A convention for defining traps for use with the SNMP;
- 1270 - SNMP communication services;
- 1303 - A convention for describing SNMP-based agents;
- 1321 - MD5 message-digest algorithm; and,
- 1470 - A network management tool catalog.

Historical:

- 1156 - Management Information Base (MIB-I);
- 1230 - IEEE 802.4 Token Bus Interface Type MIB;
- 1232 - DS1 Interface Type MIB;
- 1233 - DS3 Interface Type MIB;
- 1252 - OSPF version 2 MIB;
- 1283 - SNMP over OSI;
- 1284 - Ether-Like Interface Type;
- 1286 - Bridge MIB;
- 1298 - SNMP over IPX;
- 1351 - SNMP Administrative Model;
- 1352 - SNMP Security Protocols; and,
- 1353 - SNMP Party MIB.

Working Group Synopses

Frederick J. Baker, Deirdre C. Kostick, and Kaj Tesink

This column is a summary of activities. There is no substitute for actually participating in a working group. Even if you cannot go to the meetings, you can subscribe to the mailing lists. Included in each working group's summary is the address of the group's mailing list. To subscribe, simply append "`-request`" on to the local-part of the address. For example, the submission address for the SNMP general discussion list is

```
snmp@psi.net
```

so to subscribe, you'd send a message to

```
snmp-request@psi.net
```

If you are interested in a group's activities and do not subscribe to the mailing list, you should!

SNMP General Discussion

Submissions: `snmp@psi.net`

The formation of the Atlanta SNMP Users Group was announced. The first meeting was held on July 12, 1993. The group will focus on how to make use of the "raw" information made available by SNMP. Contact (`cheryl@empiretech.com`) for more information.

There was a question as to whether it is legal to constrain `Counter` and `Gauge`, e.g.,

```
SYNTAX Gauge (5..100)
```

The response was an opinion that it is not legal to constrain a `Counter` as the rollover point is defined. A Gauge *could* be constrained in the sense of limiting the values that one would expect it to ever contain. However, this was viewed as an odd usage. The SNMPv2 SMI clarifies the syntax rules.

There was a question as to whether the data structures produced by MIB compilers are intended for the agent, the manager, or both? Both is the answer.

There was a question as to how can trap severity be specified in SNMP? The answer came in two parts: first, a couple of alternatives were suggested (e.g., identifying enterprise-specific objects); and, secondly, even though trap severity could be identified, there are some problems. There are different viewpoints about what conditions are important. Further, to do something with the severity indication, the software must be able to handle the different severity levels for various traps. It was recommended that specifying trap severity was in the realm of management software and user configuration.

There was a question as to how can SNMP be used to implement a general-purpose logging capability for in-house software? One response gave a possible solution, another suggested using either the `syslogd` protocol or building a simple program to log received UDP packets, either of which may be a more appropriate solution to this problem.

Finally, a list of FTP repositories of SNMP packages was posted. (Please note that this information has not been verified.)

`dnpap.et.tudelft.nl:/pub/btng`
RMON for OS/2, Tricklet (Perl-based SNMP tool for Unix or OS/2)

`nic.nikhef.nl:/pub/monet/monet-0.10.tar.Z`
Xmonet network monitoring tools

`ftp.synoptics.com:/eng/mibcompiler/src.tar.Z`
SMIC (MIB Compiler)

`ftp.synoptics.com:/eng/mibcompiler/mibs.tar.Z`
Public MIB modules

`munnari.OZ.AU:pub/cmu-mu-snmp1.5.tar.Z`
MIB-II enhancements to CMU's SNMPv1 API

`ncgia.ucsb.edu:pub/etc/xsnmp21.tar.Z`
xsnmp (X-windows based snmp program)

`cs.ubc.ca:/pub/local/src/snacc`
SNACC (MIB compiler with MIB-II macros and C, C++ BER routines)

`venera.isi.edu:/ftp/mib various`
Public MIB modules

`ftp.cisco.com:`
MIB modules for Cisco routers

`nic.near.net:/pub/cmu-snmp1.2u.tar.Z`
version 1.2(unofficial) CMU SNMP code with MIB-II support

`zippy.telcom.arizona.edu:/pub/snm/agents/*`
Schema and OIDs for SunNet Manager

`ptt.lcs.mit.edu:/pub/snmp`
MIT SNMP code with MIB-II support

Appletalk/IP Working Group

Submissions: anf-netmgt@netcom.com

The Appletalk/IP WG is currently inactive. The old mailing list (apple-ip@cayman.com) was replaced with the AppleTalk general list, anf-disc@netcom.com, and the network management issues list, anf-netmgt@netcom.com. The AppleTalk MIB (RFC 1243) is currently eligible for advancement. The April 30 draft is pending review for submittal as a Proposed standard.

AToM MIB Working Group

Submissions: atommib@thumper.bellcore.com

The AToMMIB WG met in Amsterdam to discuss the ATM MIB and the SONET MIB. As a result, new versions will be posted as an I-Ds.

Compatibility of the ATM MIB with the ILMI MIB of the ATM Forum was discussed. The scope of the ATM MIB is beyond that of ATM Forum's ILMI MIB, which is limited to the local interface. The suggestion is to maintain ILMI compatibility and semantics as much as possible.

A discussion on how the ATM protocol stack must be represented with ifEntries concluded that potentially 3 ifEntries apply to an ATM interface, i.e., for the general ATM layer, for all combined AAL3/4 and AAL5 interfaces, and for all combined AAL1 and AAL2 interfaces. As per the Interface Extensions WG, no ifEntries will be kept for virtual connections.

The draft ATM MIB proposes for each VC and VP interface a counter for the number of received and transmitted cells, and the number of discards due to traffic policing and shaping. Other suggestions included not to have statistics at all for VC or VP interfaces (suggesting that hardware costs outweigh the benefits), or to define a test capability that can measure these statistics for specific VP/VC interfaces for a short amount of time.

Ken Rodemann gave a presentation on a generic approach to the management of virtual connections, suggesting a common approach for Frame Relay, X.25, and ATM. The contents of the interface specific tables would not be affected by adoption of the generic approach. Discussion of this topic was, due to lack of time, deferred to the Frame Relay Service and Interface WG meetings.

On the specifics of the ATM connection table, it was agreed that a connection table should work for both end systems and intermediate systems. The draft allows creation of a new association between an ingress and egress with a single table row. A small group was tasked to review connection table alternatives.

NM needs for SVCs were not yet discussed. However, a discussion took place on the scope of the connection table. In general the observation was supported that the connection table should not state whether it applies to SVCs and/or PVCs, leaving it to implementation as to how the table is applied.

The existing SONET MIB I-D is compatible with other trunk MIBs. The mailing list has pointed out that the Interval and Total tables are redundant, since they can be deduced from the Current and first Interval tables. Given that some implementations may already exist of these tables, a discussion on the mailing list should confirm whether deletion of these tables is appropriate. Discussion of the use of ifTable for SONET concluded to have ifEntries for the combined SONET photonic, section and line layers, one ifEntry for each SONET path (if used on that port), and one ifEntry for each SONET VT (if used on that port).

BGP Working Group

Submissions: iwg@ans.net

There is an I-D for the BGP-4 MIB, which is a work item of the BGP WG. At the Amsterdam meeting, it was reported that several implementations of this MIB are currently underway. Once implementations appear, the required “experience report” will be written, and the specification will be recommended to progress to Proposed standard.

Bridge MIB Working Group

Submissions: bridge-mib@nsl.dec.com

The Bridge MIB WG has republished the 802.1 portion of its MIB as RFC 1493, a Draft standard. The Source Routing portion was updated per discussions with IEEE 802, and was recently reviewed by the NM-Directorate. After these comments are incorporated, the draft will be considered for publication as a Proposed standard.

Character MIB Working Group

Submissions: char-mib@decwrl.dec.com

Anticipating promotion to Draft Standard, the chair solicited implementation experience for RFCs 1316, 1317, and 1318. There were no public responses. The chair then submitted a list of issues for discussion. There were no public responses.

A question about linking the RS-232 MIB, RFC 1317, to other MIBs was referred to the current I-D from the Interface WG.

Chassis MIB Working Group

Submissions: chassismib@cs.utk.edu

The Chassis MIB WG has concluded. Despite some hard work, the WG was unable to reach consensus on a document by July 5. (This was an extension of an original target.) At the beginning of 1994, the NM Area Director will make a determination on whether or not a new WG should be chartered. In the meantime the mailing list will remain active.

Some issues discussed on the mailing lists:

- Different planned uses for a Chassis MIB. E-mail described two camps: 1) those who want to use a chassis MIB to retrieve inventory data and "point" to other agents in the chassis, and 2) those who want to use the chassis MIB to manage arbitrary logical and physical entities within the chassis.
- Increased complexity of each successor draft.
- Disagreement on the overall modeling of a chassis, its elements (modules, resources, and entities) and their relationship or mappings.

DECnet Phase IV MIB Working Group

Submissions: phiv-mib@jove.pa.dec.com

The DECNET Phase IV MIB WG has posted a recent draft incorporating field experience. The NM-Directorate is now reviewing the draft prior to consideration as a Draft standard.

FDDI MIB Working Group

Submissions: fddi-mib@cs.utk.edu

The WG met briefly in Amsterdam. Status of the current MIB draft was discussed. The current draft is progressing through to IESG review. RFC 1285 will continue to be a Proposed standard for management of ANSI 6.2-based FDDI network devices; the new document will be for the management of ANSI 7.3-based FDDI network devices. The WG also reviewed the need for a traps document and reached the consensus that no traps document is needed; however, the final decision is pending e-mail discussion with the full WG.

Frame Relay Service MIB Working Group

Submissions: frftc@nsco.network.com

The Frame Relay Service MIB is a relatively new type of MIB, and is under development jointly between the Frame Relay Forum and the IETF. It follows the model of the AToM MIB, discussed earlier. The MIB attempts to treat the Frame Relay network as a virtual device from the perspective of the manager, using MIB views to limit the information to which a manager has access to, to that which directly concerns it. Rather than managing "boxes" (frame relay switches), customer network management is achieved using a proxy system.

The WG met twice in Amsterdam to discuss both the service model and several managed objects.

Host Resources MIB Working Group

Submissions: hostmib@andrew.cmu.edu

The Host Resources MIB WG is active, but is waiting for NM-Directorate feedback on its final draft.

IEEE 802.3 Hub MIB Working Group

Submissions: hubmib@synoptics.com

The WG completed a revision to RFC 1368, and is waiting for the NM-Directorate feedback. The WG also completed a new MIB module which defines managed objects for MAUs (media access units), and this too is being reviewed by the NM-Directorate.

IDPR Working Group

Submissions: idpr-wg@bbn.com

No SNMP-related traffic to report.

IDRP for IP Working Group

Submissions: idrp-for-ip@merit.edu

No SNMP-related traffic to report.

Interfaces MIB Working Group

Submissions: if-mib@thumper.bellcore.com

The Interfaces MIB WG met at the Amsterdam IETF to discuss models and managed objects. Many key issues have been settled and the next draft is in the works.

IPLPDN Working Group

Submissions: iplpdn@nri.reston.va.us

A slightly updated version of RFC 1315 (the Frame Relay DTE MIB) is being worked on by the WG.

IS-IS Working Group

Submissions: isis@merit.edu

No SNMP-related traffic to report.

Mail and Directory Management Working Group

Submissions: mailserv@innosoft.com

The WG met at Amsterdam and all three I-Ds were reviewed: network services monitoring, mail monitoring, and directory monitoring. The first two will be submitted with a recommendation for Proposed standard, following editorial changes. The third document is expected to be submitted following online review and some editorial changes. There was no input on the proposed Message Store MIB work item, and this will be dropped unless there is substantive input.

Modem Management Working Group

Submissions: majordomo@telebit.com

The WG met at the Amsterdam IETF to review an initial draft. An I-D should be posted soon.

NOCtools Working Group

Submissions: noctools@merit.edu

The WG produced RFC 1470 which updates RFC 1147. RFC 1470 provides practical information to site administrators and network managers. New and/or updated tools are listed in this RFC. Additional descriptions are welcome, and should be sent to: noctools-entries@merit.edu.

OSPF Working Group

Submissions: ospfigp@gated.cornell.edu

No SNMP-related traffic to report.

PPP Working Group

Submissions: ietf-ppp@ucdavis.edu

Four MIB modules were recently published as Proposed standards: RFC 1471 manages the Link Control Protocol, RFC 1472 manages the Security protocols, RFC 1473 manages the IP over PPP, and, RFC 1474 manages Bridging over PPP.

RIP Working Group

Submissions: ietf-rip@xylogics.com

The RIP-II effort has been stymied by a difference of opinion regarding the implementation of Routing Domains as found in IGRP. If they are implemented, the MIB needs substantial redefinition, and the basic RFC needs additional text to define them. If they are not, the MIB and RFC need only minor changes to accommodate unnumbered links.

The current direction of the WG, decided at the Amsterdam IETF, is to remove references to Routing Domains.

Remote Monitoring (RMON) Working Group

Submissions: rmonmib@jarthur.claremont.edu

See Token Ring Remote Monitoring Working Group.

SNA DLC Services MIB Working Group

Submissions: snadlcmib@apertus.com

The SNA DLC MIB WG has begun working on an initial draft. Discussion on the mailing list started with some e-mail in the basic rules (e.g., keeping the mib small, keeping things simple, etc.). Comments on the list encouraged development of a "vanilla" mib, keeping only the non-implementation specific objects from the available collection of private mibs. Recent mail includes comments on the working draft, such as the merit/lack of merit of large tables versus multiple smaller tables.

SNA NAU Services MIB Working Group

Submissions: snanaumib@thumper.bellcore.com

The SNA NAU MIB WG has completed an initial I-D which reflects changes discussed at the July 14 meeting plus input received via the list. This I-D identifies managed objects for SNA LUs and PUs, which are two types of Network Addressable Units (NAUs) in the logical structure of an SNA network. NAUs are the

origination or destination points for SNA data streams. An interim meeting is planned for the late September time-frame in North Carolina.

SNMPv2 Working Group

Submissions: snmp2@thumper.bellcore.com

The SNMPv2 MIB WG is inactive, waiting in the wings to reactivate for work on moving RFCs 1441–1452 to Proposed standard.

SNMPv2 implementations were announced for the openly-available 4BSD/ISODE 8.0 release and the Carnegie Mellon University (CMU) SNMPv2 release! Both of these implementations are copyrighted, but freely-available.

Of interest to the WG is the I-D describing "Algorithms for Automating Administration in SNMPv2 Managers". This I-D suggests one method for minimizing the information that a user must supply to establish/maintain SNMPv2 communications. Expect to see more on this and other suggestions for making SNMPv2 administration easier for the user.

The "dueling-managers-leapfrog-clock" problem was described. This problem may arise when two SNMPv2 management entities sharing the same set of MD5 parties talk to an agent, and race or accelerate clock values by returning clock values that are greater than the current, actual value. During the clock synchronization process, the entities involved continuously advanced the clock values at an artificially accelerated rate. These messages served as a warning to implementors to double-check their clock synchronization code.

There was a question on identifying destinations for sending traps and informs. RFC 1448 states that the destinations to which an `inform` will be sent are determined by inspecting the `snmpEventNotifyTable`. This table only provides the context to which a notification should be sent; a check of the `aclTable` is required to find the destination parties. For traps, an OID points to a MIB view, which in turn identifies a context. With the `snmpEventNotifyTable`, note that the context is supplied directly. RFC 1448 also notes that the

> "destination(s) ... is determined by inspecting the `snmpEventNotifyTable` or as specified by the requesting application."

which allows management applications to use other mechanisms to determine the destinations of informs.

Questions related to `StorageType` were raised. One person was troubled that a user requesting non-volatile status for a table entries will require access to a file system each time something changes for any of those entries and wondered if there is some way to synchronize all file system access. That is, have the user make all his changes and then set something else to signal the changes to be written out. A response acknowledges this as a legitimate concern, for file systems and for things like flash cards and EEPROM that have a limit on the times you can write them. The respondent further indicated that this requires an implementation-specific, but that a reasonable solution is use of a hold-down timer.

When processing an incoming message, a question was raised as to whether `snmpStatsBadOperations` or `snmpStatsUnknownContexts` should be incremented when there is no access control entry in the local database. The response was that this should probably be explicitly stated in the RFC to avoid confusion. The current interpretation is that if no ACL entry is found then treat the privilege as zero and proceed through steps 15 and on. Thus, `snmpStatsBadOperations` is incremented when a response, trap, or inform is received and an `authorizationError` response when it is one of the other management-communication classes.

A question was raised on the creation requirements for conceptual rows in the `partyTable`. The description for `partyStatus` indicates that a party is not qualified for activation until instances of ALL columns of its `partyEntry` row have been populated. Both of the conditions identified in the description must be met.

There was also a question on the relation of changes to `contextViewIndex` and `contextProxyDstParty`. If `contextViewIndex` is changed to a non-negative number, the `contextProxyDstParty` instance will be 0.0.

TCP Client Identity Protocol

Submissions: ident@nri.reston.va.us

No SNMP-related traffic to report.

Token Ring Remote Monitoring Working Group

Submissions: rmonmib@jarthur.claremont.edu

The WG completed a Token Ring RMON draft with a recommendation that it be advanced as a Proposed standard. The NM-Directorate is currently reviewing this draft.

The WG met at the Amsterdam IETF for the purpose of discussing the advancement process for RFC 1271 (Ethernet RMON), once the Token Ring RMON draft is advanced.

Trunk MIB Working Group

Submissions: `trunk-mib@saffron.acc.com`

No SNMP-related traffic to report.

Uninterruptable Power Supply Working Group

Submissions: `ups-mib@cs.utk.edu`

The UPS WG has completed an initial I-D which reflects input discussed at the July 12 meeting, plus e-mail comments. The WG is targeting to reach consensus on this draft by October 25 prior to the next IETF meeting in Houston.

X.25 MIB Working Group

Submissions: `x25mib@dg-rtp.dg.com`

No SNMP-related traffic to report.

Activities Calendar

- INTEROP August 93
  August 23–27, San Francisco, CA
  For information: +1 415 941 3399
- INTEROP Europe
  October 25–29, Paris, France
  For information: +33 1 46 39 56 33
- 28th Meeting of the IETF
  November 1–5, Houston, TX
  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.
Featured Columnists:	
Frederick J. Baker	Advanced Computer Communications
Dr. Jeffrey D. Case	SNMP Research, Inc.
	University of Tennessee
Deirdre C. Kostick	Bell Communications Research
Keith McCloghrie	Hughes LAN Systems, Inc.
David T. Perkins	SynOptics Communications, Inc.
Kaj Tesink	Bell Communications Research
Steven L. Waldbusser	Carnegie Mellon University

Contact Information

Postal: ***The Simple Times***
c/o Dover Beach Consulting, Inc.
420 Whisman Court
Mountain View, CA 94043–2186

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 only to 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 8-mm `tar` tape, $\frac{1}{4}$-in `tar` cartridge-tape, or $3\frac{1}{2}$-in MS-DOS floppy-diskette).

Each submission must include the author's full name, title, affiliation, postal and electronic mail addresses, 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 three editions: *PostScript*, *MIME* (the multi-media 822 mail format), and *richtext* (a simple page description language). 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).