17

ILMI

ATM Forum ILMI specification 4.0 1996-0

IETF RFC 115, May 1990; RFC 1213, March 1991; RFC 1151, May 1990 ATM Forum UNI 3.0; «Managing Internetworks with SNMP», by Mark A. Miller, M&T Books, 1990

Каждое оконечное устройство ATM в соответствии с протоколом SNMP (Simple Network Management Protocol - простой протокол управления сетью) и ATM UNI MIB (Management Information Base - информационная база данных управления) должно поддерживать информацию о конфигурации и состоянии виртуальных путей и каналов, доступных через интерфейс UNI. Помимо управления, данная информация может использоваться при выполнении диагностических процедур UNI.

Протокол ILMI (Interim Local Management Interface – временный интерфейс локального управления) поддерживает двунаправленный обмен управляющей информацией между системами, использующими для связи протокол UME (UNI Management Entities - объекты управления UNI). Оба объекта UME должны поддерживать идентичные MIB, даже если семантика некоторых объектов MIB интерпретируется по-разному. Протокол ILMI используется в различных типах оборудования, (коммутаторы верхних уровней, рабочие станции, компьютеры с интерфейсом ATM, сетевые коммутаторы ATM и др.).

Имена MIB

Ниже приведен список имен, используемых в MIB.

enterprises
353 atmForum
1 atmForumAdmin
2 atmfTransmissionTypes
1 atmfUnknownType
2 atmfSonetSTS3c
3 atmfDs3
4 atmf4B5B
5 atmf8B10B
3 atmfMediaTypes
1 atmfMediaUnknownType
2 atmfMediaCoaxCable
3 atmfMediaSingleMode
4 atmfMediaMultiMode
5 atmfMediaStp
6 atmfMediaUtp
4 atmTrafficDescrTypes
1 atmfNoDescriptor
2 atmfPeakRate
3 atmfNoClpNoScr
4 atmfClpNoTaggingNoScr
5 atmfClpTaggingNoScr
6 atmfNoClpScr
7 atmfClpNoTaggingScr
8 atmfClpTaggingScr
5 atmfSrvcRegTypes
1 atmfSrvcRegLecs
2 atmForumUni
1 atmfPhysicalGroup
1 atmfPortTable
1 atmfPortEntry
1 atmfPortIndex
2 atmfPortAddress
3 atmfPortTransmissionType
4 atmfPortMediaType
5 atmfPortOperStatus
6 atmfPortSpecific
2 atmfAtmLayerGroup
1 atmfAtmLayerTable
1 atmfAtmLayerEntry
1 atmfAtmLayerIndex
2 atmfAtmLayerMaxVPCs
3 atmfAtmLayerMaxVCCs
4 atmfAtmLayerConfiguredVPCs
5 atmfAtmLayerConfiguredVCCs

6 atmfAtmLayerMaxVpiBits
7 atmfAtmLayerMaxVciBits
8 atmfAtmLayerUniType
3 atmfAtmStatsGroup
1 atmfAtmStatsTable
1 atmfAtmStatsEntry
1 atmfAtmStatsIndex
2 atmfAtmStatsReceivedCells
3 atmfAtmStatsDroppedReceivedCells
4 atmfAtmStatsTransmittedCells
4 atmfVpcGroup
1 atmVpcTable
1 atmVpcEntry
1 atmVpcPortIndex
2 atmfVpcVpi
3 atmfVpcOperStatus
4 atmfVpcTransmitTrafficDescriptorType
5 atmfVpcTransmitTrafficDescriptorParam1
6 atmfVpcTransmitTrafficDescriptorParam2
7 atmfVpcTransmitTrafficDescriptorParam3
8 atmfVpcTransmitTrafficDescriptorParam4
9 atmfVpcTransmitTrafficDescriptorParam5
10 atmfVpcReceiveTrafficDescriptorType
11 atmfVpcReceiveTrafficDescriptorParam1
12 atmfVpcReceiveTrafficDescriptorParam2
13 atmfVpcReceiveTrafficDescriptorParam3
14 atmfVpcReceiveTrafficDescriptorParam4
15 atmfVpcReceiveTrafficDescriptorParam5
16 atmfVpcQoSCategory
17 atmfVpcTransmitQoSClass
18 atmfVpcReceiveQoSClass
5 atmfVccGroup
1 atmfVccTable
1 atmfVccEntry
1 atmVccPortIndex
2 atmfVccVpi
3 atmfVccVci
4 atmfVccOperStatus
5 atmfVccTransmitTrafficDescriptorType
6 atmfVccTransmitTrafficDescriptorParam1
7 atmfVccTransmitTrafficDescriptorParam2
8 atmfVccTransmitTrafficDescriptorParam3
9 atmfVccTransmitTrafficDescriptorParam4
10 atmfVccTransmitTrafficDescriptorParam5
11 atmfVccReceiveTrafficDescriptorType
12 atmfVccReceiveTrafficDescriptorParam1
13 atmfVccReceiveTrafficDescriptorParam2
14 atmfVccReceiveTrafficDescriptorParam3
15 atmfVccReceiveTrafficDescriptorParam4
16 atmfVccReceiveTrafficDescriptorParam5

17 atmfVccQoSCategory
18 atmfVccTransmitQoSClass
19 atmfVccReceiveQoSClass
8 atmfSrvcRegistryGroup
1 atmfSrvcRegTable
1 atmfSrvcRegEntry
1 atmfSrvcRegPort
1 atmfSrvcRegServiceID
1 atmfSrvcRegATMAddress
1 atmfSrvcRegAddressIndex

SNMP

RFC 1157: http://www.cis.ohio-state.edu/htbin/rfc/rfc1157.html

ILMI использует протокол SNMP, разработанный в качестве простого средства управления сетевыми устройствами с четкой структурой. Сообщения протокола SNMP делятся на две части - идентификатор версии с именем сообщества (community name), затем данные (PDU).

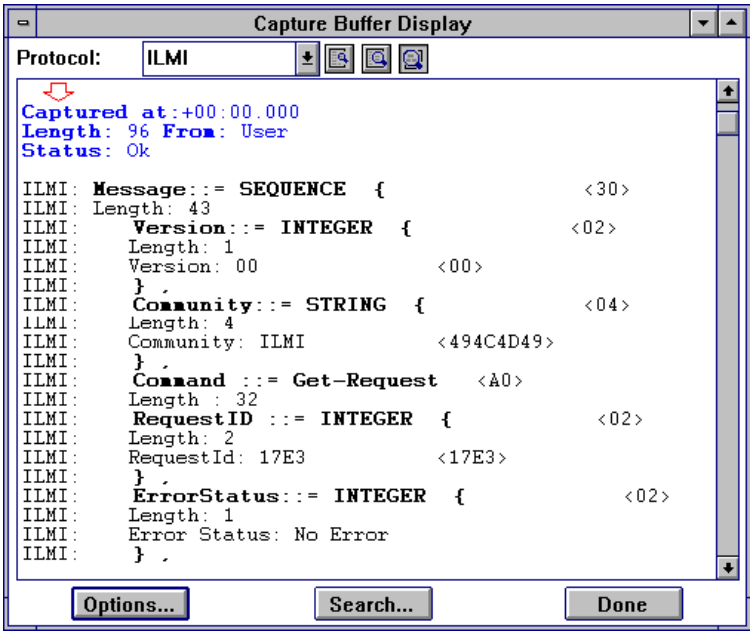
Идентификатор версии и имя сообщества иногда называют заголовком аутентификации (authentication header). Идентификатор версии обеспечивает использование одной версии протокола SNMP в управляющих станциях и агентах (управляемое устройство). Сообщения между станцией управления и агентом, содержащие другой номер версии, отбрасываются без дальнейшей обработки. Имя сообщества удостоверяет права управляющей станции на доступ к агенту. Имя сообщества совместно с IP-адресом станции управления сохраняются в community-профайле агента. Если имена сообществ различаются для агента и станции управления, агент посылает станции управления сообщение об ошибке аутентификации (authentication failure trap message).

GetRequest и GetResponse.

Станция управления использует пакеты GetRequest для получения значений одного или нескольких объектов MIB от агента. При отсутствии ошибок агент генерирует пакет GetResponse в ответ на полученный запрос. В обоих типах пакетов присутствует поле Request Index, которое служит для установки соответствия между запросами и откликами на них. Кроме того, в обоих типах сообщений используются поля Error Status (оно имеет значение noError) и Error Index (0). В рассмотренном процессе возможно возникновение четырех типов ошибок:

1. Если запрашиваемая переменная не имеет точного соответствия с существующими объектами, агент возвращает сообщение GetResponse с Error Status=NoSuchName и полем Error Index, значение которого совпадает с индексом переменной в запросе.
2. Если переменная имеет совокупный (aggregate) тип, ответ формируется таким же образом (см. пункт 1).
3. Если размер пакета GetResponse превышает локальные ограничения, агент возвращает сообщение GetResponse идентичной формы с Error Status = tooBig и Error Index = 0.
4. Если значение запрошенной переменной не может быть возвращено по иным причинам, агент возвращает сообщение GetResponse с Error Status = genErr и полем Error Index, содержащим значение индекса переменной в запросе.

На приведенном ниже рисунке содержится пример декодирования пакета GetRequest:



Пример декодирования пакета GetRequest.

GetNextRequest

Пакет типа GetNextRequest используется для получения от агента значения одного или нескольких объектов. При отсутствии ошибок агент генерирует сообщение GetResponse, для которого значение Request Index совпадает с аналогичным полем полученного запроса GetNextRequest. Поле Variable Bindings содержит имя и значение лексического потомка для объекта, идентификатор которого (OID - object identifier) указан в запросе GetNextRequest. Основная разница между пакетами GetRequest и GetNextRequest заключается в том, что по запросам GetNextRequest возвращается значение следующего объекта в MIB агента. В процессе обработки запросов возможны три типа ошибок:

1. Если поле Variable Bindings содержит переменную, которая не соответствует ни одному имени объекта в MIB, агент возвращает сообщение GetResponse с Error Status = noSuchName и полем Error Index, содержащим значение переменной в запросе.
2. Если размер пакета GetNextResponse выходит за пределы локальных ограничений, агент возвращает сообщение GetResponse идентичной формы с Error Status = tooBig и Error Status = 0.
3. Если значение лексического потомка запрашиваемой переменной по каким-либо причинам не может быть получено, агент возвращает

сообщение GetResponse с Error Status = genErr и полем Error Index, содержащим значение индекса переменной в запросе.

SetRequest

Сообщения SetRequest служат для задания значений объектов MIB, хранящихся в агенте. Когда агент получает пакет SetRequest, он устанавливает для заданного объекта значение, содержащееся в сообщении. При отсутствии ошибок агент возвращает сообщение GetResponse идентичной формы, устанавливая в поле типа пакета значение 2. В рассматриваемом процессе возможно четыре типа ошибок:

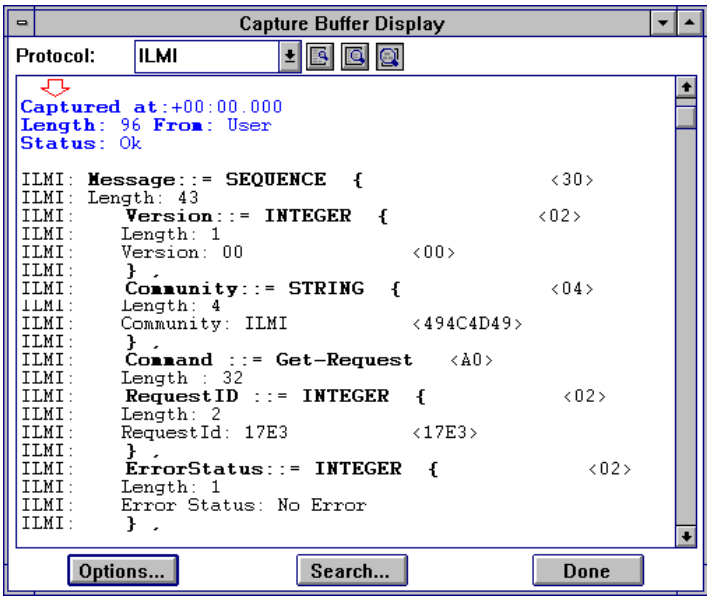
1. В том случае, если переменная не может быть использована для присвоения значений объект MIB, агент возвращает сообщение GetResponse с Error Status = NoSuchName (или ReadOnly) и полем Error Index, содержащим значение индекса переменной в запросе.
2. Если переменная по типу, длине или значению не соответствует синтаксису ASN.1, агент возвращает сообщение GetResponse с полями Error Status и Error Index, содержащими значение badValue.
3. Если размер пакета SetResponse превышает локальные ограничения, агент возвращает сообщение GetResponse идентичной формы, с Error Status = tooBig и Error Status = 0.
4. Если значение переменной не может быть обработано по каким-либо иным причинам, агент возвращает сообщение GetResponse с Error Status = genErr и полем Error Index, содержащим значение индекса переменной в запросе.

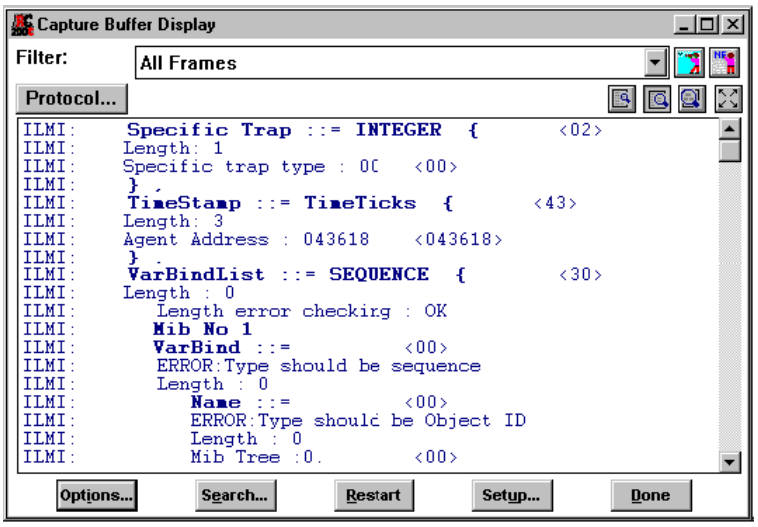
Trap

Последний тип сообщений протокола SNMP - это прерывания (Trap). Сообщения этого типа имеют формат, отличный от формата остальных четырех типов, и содержат следующие поля:

- Поле Enterprise (Предприятие) идентифицирует организацию, для которой были определены регистрационные полномочия.
- Прерывания общего типа (Generic Trap Type) обеспечивают более детальную информацию о событии. Для этого поля существует семь уникальных значений: coldStart («холодная» перезагрузка), warmStart («теплая» перезагрузка), linkDown (разрыв соединения), LinkUp (организация соединения), authentication Failure (ошибка аутентификации), egpNeighborLoss (потеря соседа в EGP) и enterpriseSpecific (фирменное значение для предприятия).
- Поле Specific Trap Type идентифицирует конкретное прерывание.
- Поле timestamp - содержит временную метку, содержащую значение времени, прошедшего с момента инициализации агента до генерации данного прерывания.
- Значения переменных.

На рисунке показан пример декодирования прерываний.





Пример декодирования пакета Trap

SMI

Общее описание

SMI (Structure of Management Information - структура управляющей информации) является стандартом, служащим для определения правил идентификации объектов управления (management objects). SMI организует, именует и описывает информацию для обеспечения логического доступа к ней. В соответствии с требованиями SMI каждый объект управления должен иметь имя, синтаксис и способ кодирования (представления). Имя или OID является уникальным идентификатором объекта. Синтаксис определяет тип данных (например, целое число или строка октетов). Тип представления описывает способ преобразования сведений об управляемом объекте в последовательную форму для передачи между машинами.

SMI определяет синтаксис обмена информацией и ее логической группировки, а также механизм именования (идентификации) объектов управления. Последний используется для идентификации каждого управляемого объекта. Возможно расширение SMI за счет включения баз данных MIB. Доступ к управляемым объектам осуществляется через MIB. Объекты MIB определяются в соответствии с ASN.1 (Abstract Syntax Notation One). Каждый тип объекта (object type) имеет имя, синтаксис и способ представления. Имя является уникальным и служит идентификатором объекта (OBJECT IDENTIFIER), который назначается администратором. Синтаксис определяет абстрактную структуру данных, соответствующую типу объекта. Примерами структуры объектов могут быть целое число (INTEGER) или строка октетов (OCTET STRING). Способ представления типа объектов описывает представление экземпляров данного типа.

Идентификатор объекта

Идентификатор объекта (object identifier) - это последовательность целых чисел, образованная пересечением глобального дерева MIB по вертикали. Дерево состоит из корня (root), с которым связаны ветвями (edge) маркированные (нумерованные) узлы. Каждый узел может служить корнем дочернего дерева (такой узел можно назвать субдеревом). Узлы субдерева, в свою очередь, могут порождать новые субдеревья и т.д. Количество уровней глобального дерева не ограничено.

У корневого узла нет номера, но он имеет по крайней мере три дочерних узла, напрямую связанных с корнем. Один из этих узлов администрируется организацией Международным комитетом по стандартизации (International Organization for Standartization - ISO) и обозначается iso (1), второй - управляется Международным телекоммуникационным союзом (International Telegraph and Telephone Consultative Committee или CCITT - современное название UTU-T) и обозначается ccitt (0), третий узел администрируется совместно ISO и CCITT и обозначается joint-iso-ccitt (2). Для узла iso (1) Международный комитет по стандартизации определил одно поддерево, используемое другими национальными и международными организациями, org (3). Из дочерних узлов этого узла два были выделены Американскому

институту стандартов и технологий (US National Institute of Standards and Technology - NIST). Затем один из этих узлов - dod (6) - был передан NIST министерству обороны США (DoD), а последнее организовало дочерний узел для Internet. Этот узел администрируется организацией IAB (Internet Activities Boards) следующим образом:

internet Идентификатор объекта::= {iso org(3) dod (6) 1} -> 1.3.6.1

В этом субдереве присутствуют четыре узла:

directory Идентификатор объекта::= { internet 1 }

mgmt Идентификатор объекта::= { internet 2 }

experimental Идентификатор объекта::= { internet 3 }

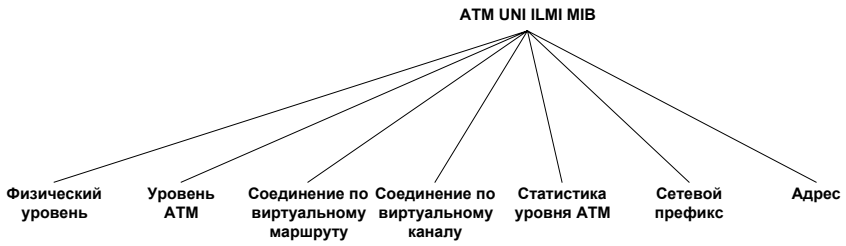
private Идентификатор объекта::= { internet 4 }

Субдерево private(4) используется для идентификации объектов, определяемых в одностороннем порядке. Администрирование субдерева private(4) передано IAB в IANA (Internet к Internet Assigned Numbers Authority). Изначально это дерево имеет по крайней мере один дочерний узел:

enterprises Идентификатор объекта::= { private 1 }

Субдерево enterprises(1) передается организациям, производящим сетевое оборудование для использования в их продукции.

Некоторые организации разрабатывают собственные субдеревья для использования в своей продукции. Примером такого дерева может служить ATM UNI MIB. Производители оборудования могут определять собственные расширения ATM UNI MIB для поддержки дополнительных (фирменных) возможностей выпускаемой продукции. Объекты MIB определяются с использованием стандарта ASN.1, разработанного SMI. Синтаксис типа объекта задает абстрактную структуру данных, соответствующую типу определяемого объекта. Для формализации таких определений служит язык ASN.1. SMI в целях упрощения ограничивает перечень используемых конструкций ASN.1.



На рисунке показана в качестве примера структура ATM UNI ILMI MIB.

Группа дерева может быть необязательной, условной и обязательной. Если группа является обязательной, каждый элемент в группе также относится к числу обязательных. Если группа является условной, каждый элемент в группе в случае его реализации является обязательным.

Ограничения протокола

В приведенном ниже списке перечислены некоторые ограничения для протокола SNMP:

- Сообщения ATM должны форматироваться в соответствии с SNMP версии 1, но не версии 2.
- Все сообщения SNMP должны использовать имя сообщества ILMI.
- Во всех прерываниях SNMP поле адреса агента должно содержать IP-адрес 0.0.0.0.
- Поддерживаемые типы пакетов Trap - coldStart и enterpriseSpecific.
- Во всех пакетах Trap SNMP поле timestamp (временная метка) содержит значение объекта MIB sysUpTime на момент генерации прерывания. Во всех стандартных прерываниях Trap поле enterprise содержит значение объекта sysObjectID.
- Размер сообщения не может превышать 484 октетов.