1. Протокол управления и мониторинга SNMP

SNMP (Simple Network Management Protocol, *Простой Протокол Сетевого Управления*) Протокол создан в 1988 г. с целью управления большим количеством сетевых устройств. С того момента протокол набрал соответствующую популярность и стал стандартом. С момента разработки протокол SNMP был 3 раза переработан с версии SNMPv1**,** SNMPv2 и SNMPv3. На самом деле, версий было больше, например v2 была пересмотрена 2 раза (и более). Кроме управления устройствами, очень часто SNMP используют для мониторинга. SNMP может получать различную информацию от любых сетевых устройств, будь то маршрутизатор, коммутатор или просто компьютер (в которых имеется поддержка данного протокола (читай - запущен агент SNMP). Содержимое получаемой информации может быть очень разнообразно, например: время с момента загрузки, различные счетчики производительности CPU, сетевые параметры устройств и пр.

1.1 Архитектура протокола SNMP

Сеть, использующая SNMP для управления и мониторинга, содержит три основных компонента:

- SNMP менеджер - ПО, устанавливаемое на ПК администратора (системы мониторинга)
- SNMP агент - ПО, запущенное на сетевом узле, за которым осуществляется мониторинг.
- SNMP MIB - MIB это Management information base. Этот компонент системы обеспечивает структурированность данных, которыми обмениваются агенты и менеджеры. По сути - это некая база данных в виде текстовых файлов.

1.2 SNMP менеджер и SNMP агент

Для понимания назначения компонентов, можно сказать, что SNMP менеджер является прослойкой (интерфейсом) между оператором(администратором) и сетевым узлом с запущенным SNMP агентом. Так же, можно сказать, что SNMP агент - это интерфейс между SNMP менеджером и оборудованием на сетевом узле. Если провести аналогию протокола SNMP с клиент-серверной архитектурой (например, веб-сервера) то веб-сервер работает как служба на некотором порту, а пользователь силами браузера обращается к веб-серверу как клиент. Это четко обозначенная архитектура с выраженным клиентом и сервером. В случае SNMP роли клиента и сервера несколько размыты. Например, SNMP агент является своего рода службой, работающей на устройстве (за которым производится мониторинг) и обрабатывает запросы на определенном порту **udp(161)**, то есть фактически является сервером. А SNMP менеджер является своего рода клиентом, который обращается к SNMP агенту.

В SNMP существует также **Trap**. Это уведомление от агента к менеджеру. При отправке данного уведомления, агент и менеджер "меняются ролями". То есть, менеджер в таком случае должен являться сервером, работающем на порту **udp(162)**, а агент является клиентом.

SNMP работает на **7 уровне модели OSI**, то есть является службой прикладного уровня. Взаимодействие агента и менеджера на уровне протокола SNMP организуется посредством объектов PDU (Protocol Data Unit), которые инкапсулируются в транспортный протокол. Хотя, SNMP поддерживает различные виды транспорта, в 99% случаев используется - UDP. При этом, каждое сообщение PDU содержит определенную команду (на чтение переменной, запись значения переменной, или ответ\trap агента). При использовании

шифрования в SNMP, по умолчанию используются порт для запросов\ответов **udp(161)**, а Trap отправляются на **udp(162)**.

Агенты, работающие на хостах, собирают информацию об устройствах и записывают собранные счетчики в значения переменных в базу данных MIB. Тем самым делая ее доступной для менеджеров. Рассмотрим схему взаимодействия SNMP-агент - SNMP-менеджер:

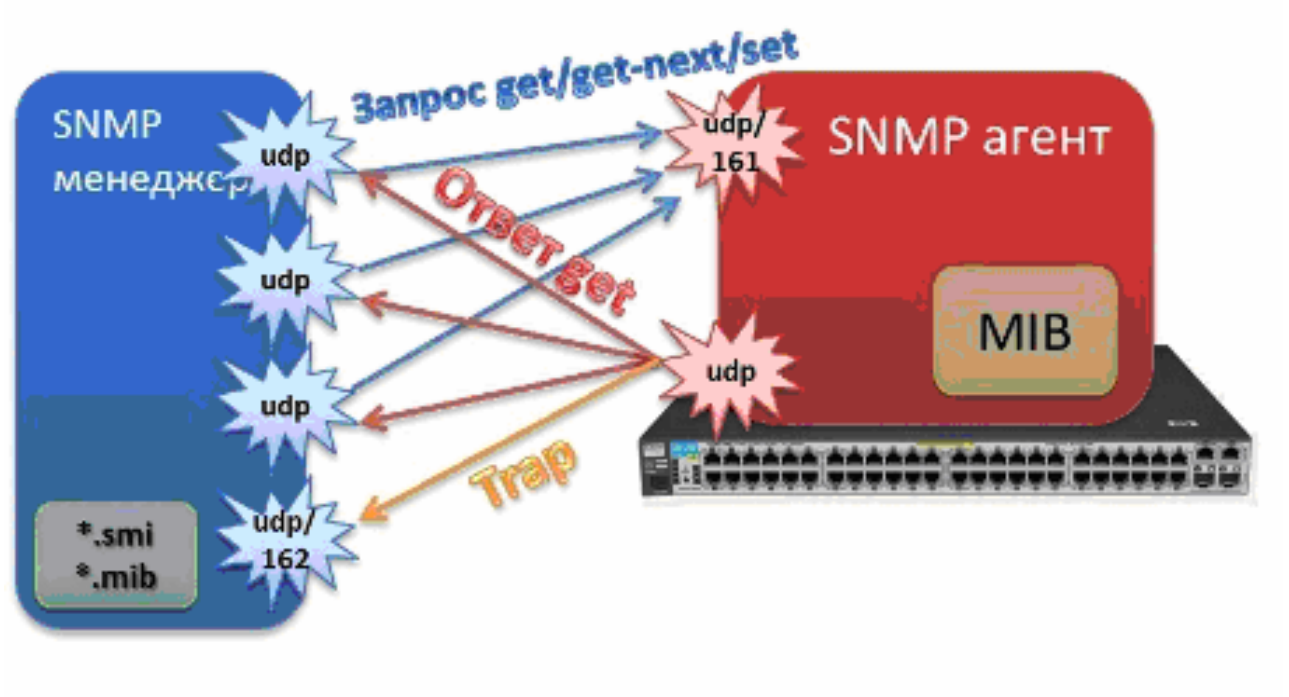

Рисунок 1. Взаимодействие SNMP-менеджера и SNMP-агента

SNMP менеджер отправляет запросы агенту на порт udp/161 (если конфигурационно в агенте не задан другой порт) с произвольного порта из эфемерного диапазона. В запросе SNMP менеджера указывается порт и адрес источника. Далее агент принимает пакет и обрабатывает. В процессе обработки, формируется ответ, который отправляется на порт взятый из исходного запроса. Ответ отправляется с udp/161 порта. Можно сказать, что SNMP агент таким образом предоставляет доступ SNMP менеджеру к данным, хранящимся в базе MIB. При этом, в момент отправки, SNMP менеджер вставляет в PDU некий ID (RequestID), а агент в ответном PDU вставляет данный ID без изменения, для того чтобы менеджер не путал пакеты от разных агентов. SNMP агент может быть настроен на посылку Trap уведомлений, которую он выполняет с эфимерного порта на udp/162 порт SNMP менеджера.

1.3 SNMP PDU (сообщения SNMP протокола)

Для обмена между агентом и менеджером используется определенный набор команд:

- Trap – одностороннее уведомление от SNMP агента –> менеджеру о каком-либо событии.
- GetReponse – ответ от агента к менеджеру, возвращающий запрошенные значения переменных.
- GetRequest - запрос к агенту от менеджера, используемый для получения значения одной или нескольких переменных.
- GetNextRequest - запрос к агенту от менеджера, используемый для получения следующего в иерархии значения переменной.
- SetRequest - запрос к агенту на установку значения одной или нескольких переменных.
- GetBulkRequest – запрос к агенту на получение массива данных (тюнингованная*GetNextRequest*). (Этот PDU был введен в SNMPv2.)
- InformRequest – одностороннее уведомление между менеджерами. Может использоваться, например для обмена информацией о MIB (соответствие символьных OID - цифровым). В ответ менеджер формирует аналогичный пакет в подтверждение того, что исходные данные получены. (Этот PDU был введен в SNMPv2.)

Как видно, в зависимости от версии протокола, набор команд разный (например, InformRequest и GetBulkRequest полноценно появился только во второй версии SNMP).

1.4 Структура PDU

Рассмотрение структуры PDU не обязательно, но это поможет более глубоко понять логику работы SNMP. Все PDU (кроме Trap) состоят из определенного набора полей, в которые записывается необходимая информация:

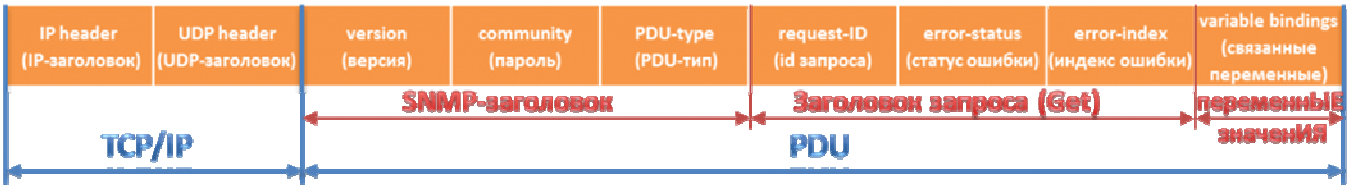

Рисунок 2. Структура PDU

Что данные поля обозначают:

- версия - содержит версию SNMP
- пароль (community) - содержит последовательность символов, описывающую принадлежность к группе
- тип PDU имеет цифровой идентификатор запроса (*Get,GetNext,Set,Responde,Trap*…)
- идентификатор запроса – набор символов, который связывает в единое целое запрос\ответ.
- Статус ошибки– это число, характеризующее характер ошибки:
- Для запросов (Get,Set,GetNExt и др.) - 0
- Для ответов:
- 0 (NoError) – Нет ошибок,
- 1 (TooBig) - Объект не вмещается в одно Response сообщение,
- 2 (NoSuchName) – не существующая переменная,

- 3 (BadValue) – при попытке установить значение задано недопустимое значение или совершена синтаксическая ошибка,
- 4 (ReadOnly) – при Set запросе была попытка изменить read-only переменную,
- 5 (GenErr) – другие ошибки.
- Индекс ошибки – содержит некий индекс переменной, к которой относится ошибка
- Поле связанные переменные может содержать одну или более переменную (для запросов Get – это только имя переменных, для Set – имя и устанавливаемое значение, для Response – имя и запрошенное значение).

1.5 Особенности работы SNMP

Некоторые общие особенности работы протокола SNMP, которые стоит учитывать:

- Принимающая сторона первым делом пытается декодировать сообщение. Если принимающей стороне не удается раскодировать PDU, то пакет отбрасывается без каких-либо действий.
- Если версия SNMP в пришедшем пакете не соответствует версии сервера, то пакет так же отбрасывается.
- После этого, сверяется аутентификационная информация (либо значение строки community, либо пользовательская информация). Могут применяться внешние модули для аутентификации
- Далее, происходит обработка сообщения. Если необходимо - генерируется ответ.

1.6 SNMP MIB

Management information base, то есть база управляющей информации. Каждый сетевой узел, имеющий SNMP агента (поддерживающий протокол SNMP) предоставляет свой набор данных, тот, который в него «вложили» разработчики при проектировании оборудования/программы. То есть в каждом сетевом устройстве с поддержкой SNMP имеется своя база MIB со строго обозначенным набором переменных. Каждая база MIB имеет древовидную структуру, каждый объект в которой характеризуется уникальным идентификатором объекта (Object Identifier, OID)**.**

Каждая ветка MIB-иерархии оканчивается некоторой переменной (так же имеющей свой OID), содержащей в себе определенное значение, которое записывается в переменную SNMP агентом, работающем на хосте. Данное значение неким образом характеризует данный хост (например, содержит информацию о загрузке сетевого интерфейса).

Имеется некая единая общая структура дерева MIB, а так же, имеются единые стандарты и принципы дальнейшего формирования данного дерева, его переменных, др. параметров, за эти правила отвечает стандарт под названием **Структура Информации Управления** (SMI, Structure of Management Information). Так же, имеется некий стандарт, называемый абстрактный синтаксис нотаций - ASN.1**,** готорый тоже участвует в описании протокола SNMP и базы MIB. А еще имеется базовые правила кодирования BER (Basic Encoding Rules), определяющие кодирование сущностей, применяемых в SNMP.

Кроме того, существует всемирное дерево регистрации стандартов ISO, содержащее базовую структура дерева MIB (точнее этих структур существует несколько, они формировались вместе с совершенствованием версий SNMP). Составное числовое имя объекта SNMP MIB соответствует полному имени этого объекта в дереве регистрации

объектов стандартизации ISO. За данную древовидную структуру отвечает и контролирует организация IANA (и некоторые другие).

Рассмотрим типичное дерево MIB на рисунке 3:

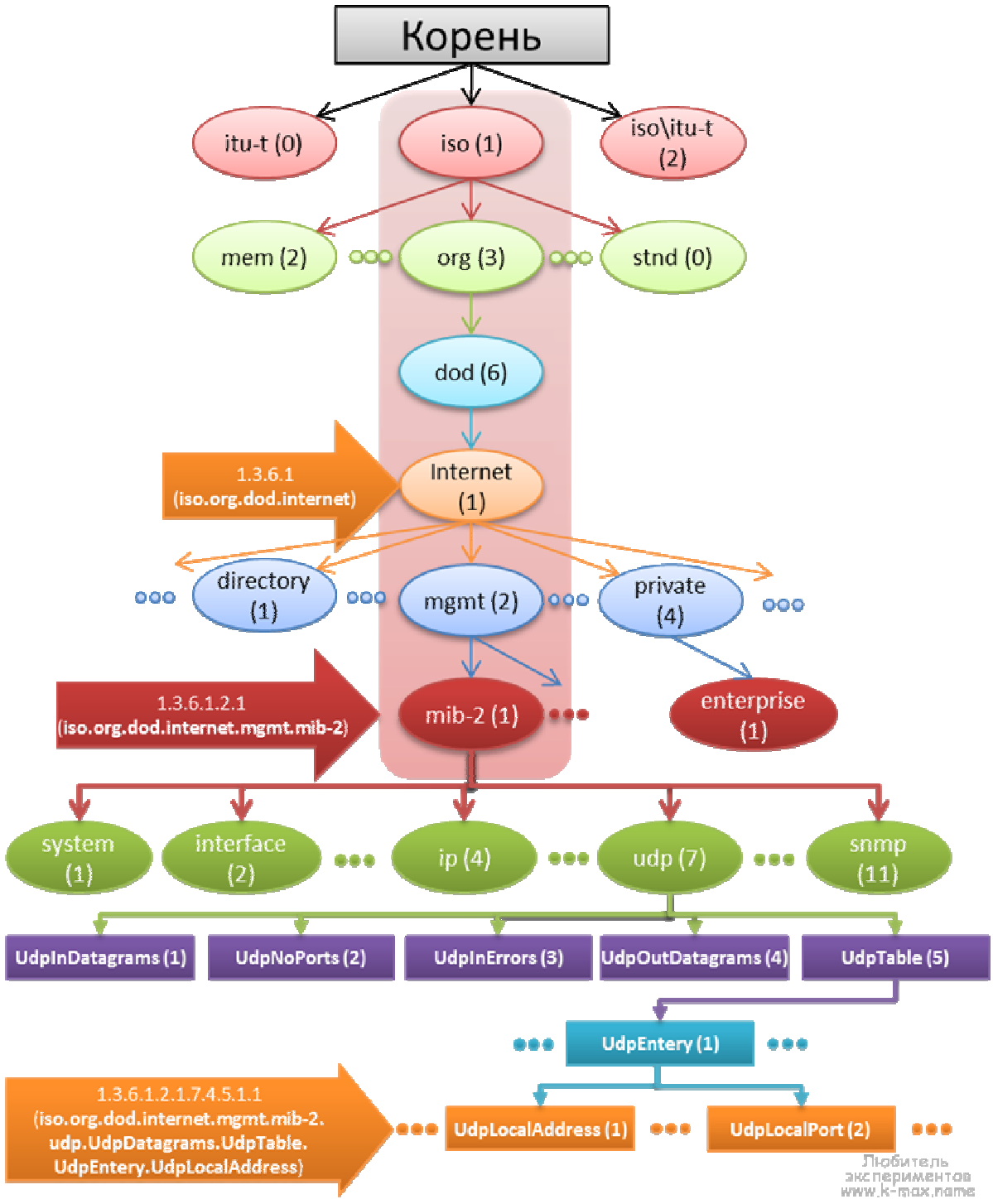

Рисунок 3. Типичное дерево MIB

Дерево объектов MIB имеет символьные имена (аналогично NS имени) и называемые ASN.1 нотацией, и соответствующие им числовые значения. Называемые dot нотацией. Каждый объект в MIB состоит из нескольких чисел, разделенных точками. Числам

соответствуют текстовые наименования. Все разрешения имен из символов в числа происходят силами SNMP менеджера (в зависимости от того, какие MIB загружены в менеджер). Весь обмен между узлами SNMP происходит только в числовом виде. В символьном виде, наименования используются только для отображения на экране или в документации.

Итак, **структура OID в дереве MIB**:

В вершине дерева – корень (точка), от которого ответвляются ветви. Во многих схемах приведены некоторые ветви верхнего уровня (например *itu-t, iso\itu-t* и другие ниже), но информации о их назначении я не нашел… Посему, нас интересует вертка **iso (0),** которая хранит в себе следующие значения до **internet (1)**: **iso.org.dod.internet**, что соответствует числовому ID **.1.3.6.1**.

Данный раздел (**iso.org.dod.internet**) разветвляется на подразделы, которые в большинстве своем контролируются IANA и состоят (согласно RFC1155) из:

- **directory** OID=**1.3.6.1.1** (*iso.org.dod.internet.directory*) – зарезервировано для использования в будущем
- **mgmt** OID=**1.3.6.1.2** (*iso.org.dod.internet.mgmt*) - в этой ветке находится стандартные ответвления объектов.
- **experimental** OID=**1.3.6.1.3** (*iso.org.dod.internet.experimental*) - это ветка для разработчиков. (*не отображено на схеме*)
- **private** OID=**1.3.6.1.4** (*iso.org.dod.internet.private*) – раздел предназначен для использования производителями оборудования.

Далее, необходимо рассмотреть отдельным пунктом ветку **1.3.6.1.2** (*iso.org.dod.internet.mgmt*), состоящую из подветки **mib-2 (1)**, а так же **reserved(0)**, **pib(2)**, **http(9)** и некоторых других. Стоит отметить, что в зависимости от версии протокола (SNMPv1 vs SNMPv2) на месте данной ветки могут быть «прилинкованы» разные поддеревья в целом имеющие примерно одинаковую структуру и одинаковые идентификаторы, но в более новой версии протокола – поддерево более расширено. (наверно, в этом и состоит несовместимость версий протокола…)

Итак **iso.org.dod.internet.mgmt.mib-2 (1.3.6.1.2.1)**: данная ветка является базовой для большинства сетевых устройств и содержится практически в любом устройстве. Ветка поделена на базовые группы (которых на текущий момент более 170), характерные для любого сетевого оборудования. Давайте рассмотрим наиболее применяемые:

1. **iso.org.dod.internet.mgmt.mib-2.system (1.3.6.1.2.1.1)** - ветка содержит в себе несколько объектов, характеризующих хост (аптайм, версия ОС и др.), описана в RFC1213.
2. **iso.org.dod.internet.mgmt.mib-2.interface (1.3.6.1.2.1.2)** - содержит объекты, описывающие сетевую подсистему хоста (количество интерфейсов, размер MTU, скорость передачи, физические адреса и т.п.), описана в RFC2863.
3. **iso.org.dod.internet.mgmt.mib-2.ip (1.3.6.1.2.1.4)** – содержит набор объектов, описывающих данные о проходящих IP пакетах (количество запросов, ответов, отброшенных пакетов и мн.др). Описана в RFC1213.

4. **iso.org.dod.internet.mgmt.mib-2.tcp (1.3.6.1.2.1.6)** и **iso.org.dod.internet.mgmt.mib-2.udp (1.3.6.1.2.1.7)** - содержат объекты, хранящие в себе информацию, касающуюся соответствующего транспортного протокола. Описана в RFC1213.

Отдельного абзаца так же достоин подраздел **iso.org.dod.internet.private (1.3.6.1.4)**, содержащий в себе поддерево **enterprise (1)**. Данная ветвь контролируется IANA и содержит в себе более 40000 поддеревьев (актуальный список http://www.iana.org/assignments/enterprise-numbers). В данной ветке регистрируют свои поддеревья – производители оборудования. Вот пример ответвления (рис. 4):

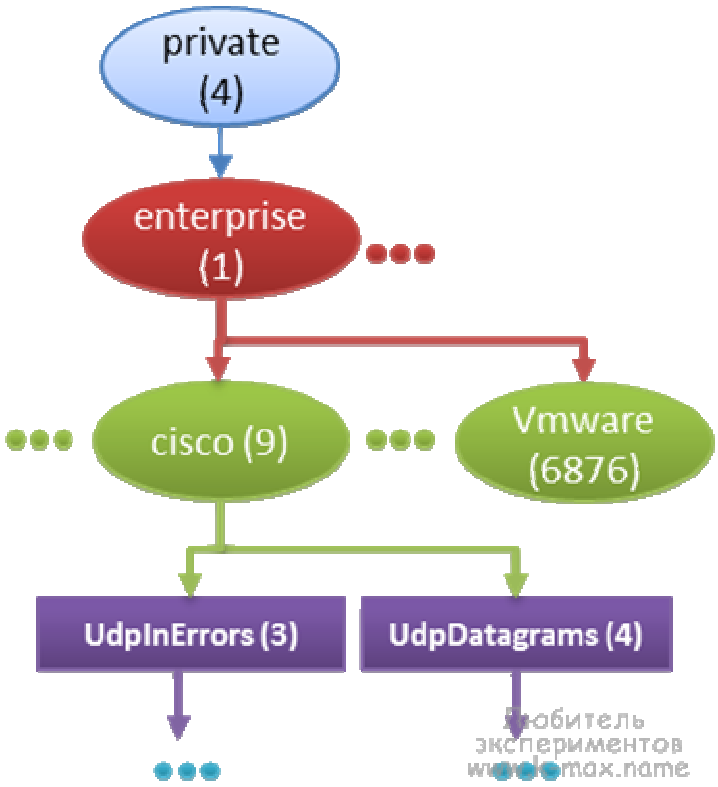

Рис. 4. Фрагмент MIB

Ниже структура аналогичная всем остальным разделам – древовидна. В каждом поддереве соответствующий производитель оборудования в праве сам регистрировать свои ветвления и переменные.

Так же, важным моментом, является то, что любая ветвь базы MIB оканчивается переменной, в которую записывается некоторое значение. При этом, в MIB существует несколько типов переменных. Во-первых, они делятся на переменные «только для чтения» и доступные для изменения, которые не позволяют выполнять запрос SetRequest и позволяют выполнять, соответственно. Во-вторых, имеются строковые переменные, табличные и мн.др., значения которых так же идентифицируются по OID.

1.7. Безопасность протокола SNMP

Безопасность протокола SNMP - это самый изменяемый раздел спецификации протокола со времени его создания. С каждой версией SNMP, производились попытки усилить безопасность. Первая версия протокола **SNMPv1** была самой простой и небезопасной. Сообщения протокола могли быть подвержены возможности модификации, подмене или прослушиванию. Безопасность протокола базировалась на **модели безопасности на основе сообществ** (т.н. **Community-based Security Model**), что

подразумевало аутентификацию на основе единой текстовой строки - своеобразного пароля (т.н. **community-sting**), которая передавалась в теле сообщения в открытом виде. Хотя, данная версия протокола самая незащищенная, она довольно часто применяется в современных сетях, т.к. является самой простой.

В одной из вторых версий **SNMP (SNMPv2p)** была попытка реализовать **аутентификацию на основе сторон** (т.н.**Party-based Security Model**). Технология кроме аутентификации, так же поддерживала возможность шифрования трафика. Данная технология не прижилась, как "сложная и запутанная") и в данный момент не используется. После чего SNMP второй версии вернулась к **Community-based Security** и стала именоваться **SNMPv2c** и применяется по сей день. **SNMPv2** была переписана чуть более чем полностью, в результате чего существенно повышено быстродействие протокола, безопасность.

Третья версия протокола (**SNMPv3**) была более удачно доработана и поддерживает как **аутентификацию на основе имени пользователя** (т.н. **User-based Security Model**), так и **шифрование трафика**. Хотя их можно и не использовать.

Версии протокола между собой **не совместимы**. Несовместимость заключается в разнице **пакетов PDU**, в наличии дополнительных команд в более новых версиях протокола. В RFC 2576 имеется некоторая информация, позволяющая организовать возможность совместного использования **SNMPv1** и **v2**. Для этого есть 2 пути: 1. Использование прокси-агентов (агент преобразует **PDU SNMPv2** в **PDU SNMPv1**), 2. Использование менеджера с поддержкой 2х версий (менеджер для каждого хоста должен помнить версию агента).