

sFlow Version 5

Протокол sFlow версии 5

Авторские права

Copyright (C) sFlow.org (2004). All Rights Reserved.

Тезисы

Этот документ определяет систему sFlow от sFlow.org, которая обеспечивает технологию мониторинга трафика в сетях передачи данных с коммутаторами и маршрутизаторами. В частности, определяются механизмы выборки трафика, реализованные в агентах sFlow, база sFlow MIB для настройки таких агентов и формат дейтаграмм sFlow для передачи данных измерения от агентов к коллектору sFlow.

Оглавление

1. Обзор.....	1
2. Терминология и архитектура.....	2
2.1 Термины.....	2
2.2 Эталонная модель sFlow.....	2
3. Механизмы отбора.....	3
3.1 Выборка из потока пакетов.....	3
3.2 Выборка счетчиков.....	4
4. sFlow MIB.....	4
4.1 Схема управления SNMP.....	4
4.2 Структура модуля SFLOW MIB.....	4
4.2.1 Группа Receiver.....	4
4.2.2 Группа Flow Sampling.....	5
4.2.3 Группа Counter Polling.....	5
4.3 Определения.....	5
5. Формат дейтаграмм sFlow.....	13
6. Вопросы безопасности.....	24
6.1 Конфигурация.....	24
6.2 Транспорт.....	24
6.3 Конфиденциальность.....	25
7. Литература.....	25
8. Адреса авторов.....	26
Приложение А. Различия sFlow версий 4 и 5.....	26
Приложение В. Генерация случайных чисел.....	26

1. Обзор

sFlow является технологией мониторинга трафика в сетях, содержащих коммутаторы и маршрутизаторы.

Система мониторинга sFlow включает агенты (sFlow Agent - встроенный в коммутатор/маршрутизатор или автономный датчик) и центральный сборщик данных sFlow Collector. Архитектура и методы выборки sFlow разработаны для непрерывного мониторинга трафика в масштабе сайта (и предприятия) высокоскоростных сетей с коммутацией и маршрутизацией. В частности, было уделено специальное внимание:

- точности мониторинга при гигабитных и более высоких скоростях;
- расширяемости системы до десятков тысяч агентов на одие коллектор sFlow;
- обеспечения максимально низкой стоимости реализации агентов sFlow.

Агент sFlow использует технологию выборки для сбора статистики трафика с обслуживаемого агентом устройства. Для незамедлительной пересылки собранных данных для анализа в коллекторе sFlow используются специальные дейтаграммы sFlow.

В документе описаны механизмы отбора, применяемые агентами sFlow, база SFLOW MIB, используемая коллектором для управления агентами, а также формат дейтаграмм sFlow, служащих для передачи данных коллектору.

В документе описан протокол sFlow версии 5, который служит заменой sFlow версии 4, описанному в RFC 3176 [1]. Различия между версиями протокола перечислены в Приложении А.

2. Терминология и архитектура

В этом разделе определены элементы системы sFlow.

2.1 Термины

Ниже приведены определения основных терминов, используемых при описании архитектуры sFlow.

Network Device - сетевое устройство

Элемент сетевого оборудования, пересылающий пакеты данных (например, коммутатор или маршрутизатор).

Data Source - источник данных

Источник данных указывает место в сетевом устройстве, где можно выполнить измерения трафика. Источниками данных могут служить интерфейсы, элементы сетевого устройства (например, системная магистраль или VLAN). Каждый источник данных имеет доступ к подмножеству трафика, проходящего через устройство.

Тип и число источников данных, которые нужны для полного мониторинга устройства, зависят от внутренней архитектуры устройства. Обычно источник данных определяется на каждом физическом интерфейсе устройства, поскольку это позволяет наблюдать каждый пакет, проходящий через устройство.

Packet Flow - поток пакетов

Поток пакетов определяется как путь или траектория прохождения пакетов через сетевое устройство (т. е. путь от приема пакетов на одном интерфейсе через принятие решения о коммутации или маршрутизации до передачи через выходной интерфейс). При этом входной и выходной интерфейс могут совпадать, а для групповых и широкополосных пакетов используется множество выходных интерфейсов.

Packet Flow Sampling - выборка из потока пакетов

Выборка из потока пакетов означает произвольный (случайный) отбор части потока, наблюдаемой в источнике данных.

Sampling Rate - период выборки

Период выборки определяется отношением числа пакетов, наблюдаемых в источнике данных, к числу отобранных. Например, при периоде 100 будет выбираться в среднем 1 пакет из 100 наблюдаемых.

Packet Flow Record - запись для потока пакетов

Запись для потока пакетов описывает атрибуты потока. В записи имеется два типа информации:

1. информация о самом пакете (обычно заголовок, размер и инкапсуляция);
2. информация о пути пакета через устройство, включая данные, относящиеся к выбору пути пересылки.

Counter Sampling - выборка счетчика

Периодическая выборка или опрос счетчиков, связанных с источником данных.

Sampling Interval - интервал выборки

Интервал времени между последовательными выборками счетчика.

Counter Record - запись для счетчика

Запись, содержащая значения счетчика, связанные с источником данных в конце интервала выборки.

sFlow Instance - экземпляр sFlow

Экземпляр sFlow - это измерительный процесс, связанный с источником данных. В одном источнике данных может присутствовать множество экземпляров sFlow.

Каждый экземпляр sFlow работает независимо от других экземпляров sFlow, например, экземпляры выборки из потоков отбирают данные со своими периодами, а экземпляры выборки счетчиков - со своими интервалами.

sFlow Agent - агент sFlow

Агент sFlow обеспечивает интерфейс для настройки экземпляров sFlow на устройстве. Агент может поддерживать настройку из командной строки и/или с помощью протокола SNMP.

Агент также отвечает за поддержку сеансов измерения с коллекторами sFlow. Он собирает данные в дейтаграммы sFlow для передачи коллекторам. По завершении сессии агент sFlow освобождает ресурсы.

sFlow Sub-Agent - субагент sFlow

При реализации sFlow на устройствах с распределенной архитектурой может оказаться желательным распределение функциональности агентов sFlow. Каждый субагент sFlow отвечает за определенное подмножество источников данных.

sFlow Collector - коллектор sFlow

Коллектор sFlow получает дейтаграммы sFlow от одного или множества агентов sFlow. Коллектор может также настраивать экземпляры sFlow с использованием механизмов, предоставляемых агентами sFlow.

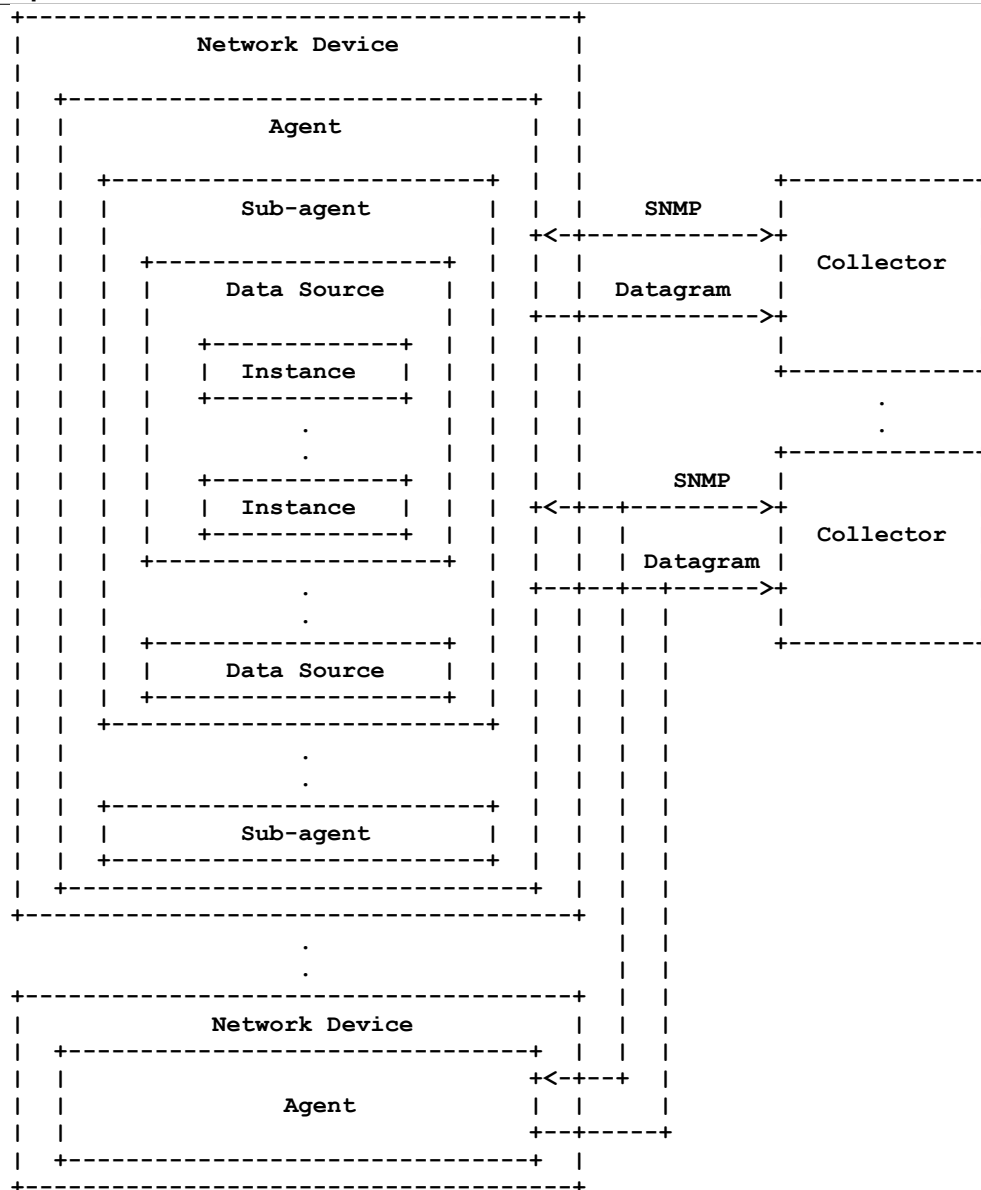
sFlow Datagram - дейтаграмма sFlow

Дейтаграмм sFlow представляет собой дейтаграмму UDP, содержащую результаты измерения вместе с информацией об источнике и измерительном процессе. Формат дейтаграмм sFlow определен в этом документе.

2.2 Эталонная модель sFlow

Связи между различными элементами системы sFlow показаны на рисунке.

Коллектор sFlow использует протокол SNMP для взаимодействия с агентом sFlow, чтобы настроить мониторинг sFlow на сетевом устройстве. База sFlow MIB описывает управляемые объекты, требуемые для настройки агента sFlow. Выборки счетчиков и потоков выполняются экземплярами sFlow, связанными с отдельными источниками данных в агенте sFlow. Для выполнения выборки из потока пакетов на экземпляре sFlow задается период выборки (Sampling Rate). Процесс выборки из потока пакетов приводит к генерации записей Packet Flow Record. Для выборки счетчика на экземпляре sFlow задается интервал выборки (Sampling Interval). Агент sFlow собирает записи для счетчиков и потоков, а затем передает их в дейтаграммах sFlow коллекторам sFlow.



3. Механизмы отбора

Агенты sFlow используют две формы выборки - статистическая выборка из коммутируемых или маршрутизируемых потоков пакетов и выборка значений счетчиков по времени.

3.1 Выборка из потока пакетов

Механизм выборки из потока, выполняемой каждым экземпляром sFlow, должен обеспечивать равную вероятность отбора любого наблюдаемого в источнике данных пакета, независимо от потоков, к которым пакет относится.

Выборка пакетов из потока выполняется в процессе их обработки. При поступлении пакета на интерфейс сетевое устройство применяет фильтры, что может приводить к отбрасыванию некоторых пакетов. Если пакет не отбрасывается, для него выбирается выходной интерфейс с помощью функции коммутации или маршрутизации. В этот момент принимается и решение о выборке. Механизм отбора пакетов включает счетчик, значение которого декрементируется с каждым пакетом, и при достижении нулевого значения происходит выборка пакета. Значение счетчика `Total_Packets` инкрементируется независимо от того, был ли пакет отобран, т. е. `Total_Packets` учитывает все пакеты, которые могли быть отобраны.

Выборка пакета включает копирование его заголовка или извлечение информации из пакета. Различные формы выборки рассмотрены в описании дейтаграмм sFlow.

При каждой выборке пакета значение счетчика `Total_Samples` инкрементируется, т. е. этот счетчик указывает общее число отобранных образцов. Выборки передаются экземпляром sFlow агенту sFlow для обработки. Выборка включает информацию о пакете, а также значения счетчиков `Total_Packets` и `Total_Samples`. Агент sFlow может использовать полученные образцы для получения дополнительной информации о прохождении пакета через устройство. Такая информация зависит от функций пересылки пакетов в устройстве. Примерами информации о прохождении пакета через устройство являются входной и выходной интерфейсы, исходная и целевая VLAN, подсеть следующего интервала (`next hop`), полный путь AS. Более подробно это рассмотрено в описании формата дейтаграмм sFlow.

При выполнении выборки значение счетчика пропускаемых до следующей выборки пакетов следует установить заново. Для этого счетчика следует устанавливать случайные целочисленные значения, которые обеспечат в среднем выполнение условия

$$\text{Total_Packets} / \text{Total_Samples} = \text{Sampling Rate} \quad (1)$$

Другим вариантом отбора образцов из потока пакетов является генерация для каждого пакета случайного числа, сравнение этого числа с заданным заранее пороговым значением и выборка пакета, если случайное число для него

меньше заданного порога. Расчет порогового значения зависит от характеристик генератора случайных чисел, однако результат должен удовлетворять уравнению (1).

Требования к генератору случайных чисел приведены в Приложении В.

3.2 Выборка счетчиков

Основной целью выборки значений счетчиков является эффективный периодический экспорт значений, связанных с источниками данных.

Обычно источник данных связывается с каждым интерфейсом устройства, который может служить входом или выходом для потока пакетов. Эти источники данных будут использоваться для экспорта значений счетчиков, относящихся к интерфейсам.

Для каждого экземпляра sFlow, связанного с интерфейсным источником данных, задается максимальный интервал выборки, но агент sFlow может указать свой интервал опроса для обеспечения максимальной эффективности.

Выборки из потока пакетов и счетчиков являются частями интегрированной системы. Оба типа образцов передаются в дейтаграммах sFlow. Поскольку выборки из потоков создают случайный, но устойчивый поток дейтаграмм sFlow, передаваемых коллектору, выборки счетчиков могут выполняться произвольно для заполнения этих дейтаграмм.

Одна из стратегий выборки счетчиков заключается в поддержке агентом sFlow списка опрашиваемых счетчиков. При генерации выборки из потока агент sFlow проверяет список и добавляет значения счетчиков в дейтаграмму выборки из потока, беря из списка счетчики, которые опрашивались наиболее давно. При этом в дейтаграмму добавляются лишь те счетчики, для которых отклонение от заданного интервалом (см. `sFlowCpInterval`¹ в SFLOW MIB) момента выборки невелико (скажем, 5 секунд). Всякий раз при добавлении счетчика в дейтаграмму выборки обновляется время выборки для этого счетчика и счетчик перемещается в конец очереди на опрос. Периодически (например, каждую секунду) агент sFlow просматривает список счетчиков и передает контроллеру значения всех счетчиков, для которых истекает интервал выборки.

Если агент sFlow выбирает регулярный опрос счетчиков, ему следует задать для каждого счетчика свое начало отсчета времени (предпочтительно случайное), чтобы выборки счетчиков не происходили синхронно в рамках одного или множества агентов.

4. sFlow MIB

База sFlow MIB обеспечивает стандартный механизм для удаленного управления и настройки агентов sFlow.

4.1 Схема управления SNMP

Модель управления SNMP в настоящее время включает пять основных компонент, перечисленных ниже.

- Общая архитектура, описанная в RFC 2571 [2].
- Механизмы именования и описания объектов и событий для целей управления. Первая версия структуры данных управления (SMI²) называется SMIv1 и описана в STD 16 RFC 1155 [3], STD 16 RFC 1212 [4] и RFC 1215 [5]. Вторая версия - SMIv2 описана в STD 58 RFC 2578 [6], STD 58 RFC 2579 [7] и STD 58 RFC 2580 [8].
- Протоколы сообщений для обмена данными управления. Первая версия протокола сообщений SNMP под названием SNMPv1 была описана в STD 15 RFC 1157 [9]. Вторая версия, которая не стала стандартом Internet, называется SNMPv2c и описана в RFC 1901 [10] и RFC 1906 [11]. Третья версия протокола называется SNMPv3 и описана в RFC 1906 [11], RFC 2572 [12] и RFC 2574 [13].
- Протокольные операции для доступа к данным управления. Первый набор протокольных операций и форматов связанных с ними PDU описан в STD 15 RFC 1157 [9]. Второй набор операций и форматов PDU описан в RFC 1905 [14].
- Набор базовых приложений описан в RFC 2573 [15], а механизмы контроля доступа на основе представлений - в RFC 2575 [16].

Дополнительную информацию о текущей модели сетевого управления SNMP можно найти в RFC 2570 [17].

Доступ к управляемым объектам осуществляется через виртуальное хранилище информации MIB³. Объекты в MIB определяются с использованием механизмов, заданных в SMI.

Этот документ задает модуль MIB, соответствующий SMIv2. Модуль MIB, соответствующий SMIv1, можно получить с помощью подходящей трансляции. Транспирированный модуль MIB должен быть семантически эквивалентен, за исключением случаев, когда объекты или событие пропускаются по причине невозможности трансляции (использование Counter64). Часть машиночитаемой информации SMIv2 будет при трансляции преобразована в текстовые описания SMIv1. Однако потеря машиночитаемой информации не считается изменением семантики MIB.

4.2 Структура модуля SFLOW MIB

MIB состоит из трех групп объектов - группа получателей, группа выборки из потоков и группа опроса счетчиков.

4.2.1 Группа Receiver

Группа получателей определяет набор объектов, используемых для поддержки сессии sFlow между агентом и коллектором sFlow.

Перед тем или иным изменением конфигурации коллектор sFlow должен сначала найти свободную строку в таблице получателей, а затем заявить ее путем записи своей строки владельца и времени резервирования в эту пустую строку.

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.

²Structure of Management Information.

³Management Information Base - база управляющей информации.

Сессия будет автоматически завершаться по тайм-ауту с освобождением всех связанных с ней ресурсов, если запись не будет периодически обновляться коллектором. Периодически обновляя свою запись в таблице получателей, коллектор sFlow обеспечивает получение его адреса (обучение) всеми мостами на пути к агенту sFlow, что сводит к минимуму вероятность лавинной рассылки дейтаграмм sFlow мостами.

Записи не могут добавляться в таблицу получателей или удаляться из нее. Разработчикам агентов sFlow следует ограничивать число записей, чтобы число одновременных сессий sFlow не превысило возможностей устройства.

Получив строку в таблице получателей, коллектор sFlow задает адрес и порт, которые он будет использовать для приема дейтаграмм sFlow. Для предотвращения фрагментации пакетов может быть установлен максимальный размер дейтаграмм sFlow.

4.2.2 Группа Flow Sampling

Группа выборки из потоков определяет набор мест (источников данных) в устройстве, которые способны выполнять выборку из потока пакетов. Источники данных могут соответствовать интерфейсам, VLAN и другим элементам устройства. Набор источников данных, анонсируемый агентом sFlow, зависит от архитектуры устройства. Например, устройство может включать средства отбора пакетов, интегрированные в интерфейсные ASIC, и в этом случае будут анонсироваться источники данных для каждого интерфейса. Программный маршрутизатор может иметь просто один источник данных, связанный с модулем маршрутизации.

Каждый источник данных может быть способен поддерживать более одного независимого процесса выборки и тогда с ним будет связано множество экземпляров sFlow, каждый из которых может свой период выборки.

Даже если оборудование источника данных способно генерировать лишь один поток образцов пакетов, агент sFlow может использовать субвыборки для создания множества экземпляров sFlow на одном источнике данных. Предположим в качестве примера, два экземпляра sFlow, у одного из которых Sampling Rate составляет 512, а у другого - 1024. Оборудование можно настроить для выборки с периодом 512, а затем делать субвыборки на программном уровне с интервалом 1024. Привлекательным является использование периодов в виде степени 2, поскольку это позволяет задать на аппаратном уровне наименьшее значение Sampling Rate, а остальные скорости реализовать субвыборками на программном уровне.

Экземпляр выборки из потока может иметь минимальное разрешенное значение Sampling Rate. Установка консервативного значения, при котором агент sFlow никогда не будет перегружаться выборками из потока пакетов даже при худшей ситуации, не рекомендуется. Это привело бы к слишком высоким минимальным значениям Sampling Rate, которые не отражали бы реальную картину трафика. Агент может реализовать автоматическое одностороннее снижение скорости отбора, которое запускается при генерации избыточного числа пакетов в секунду. При срабатывании триггера агент может удваивать значение Sampling Rate. При следующем превышении порога Sampling Rate удваивается еще раз. Значение Sampling Rate будет быстро приходить к установившемуся уровню. После этого значение скорости отбора должно сохраняться без возврата к заданному изначально. Начальное значение может быть возвращено вручную с помощью командного интерфейса или sFlow MIB. Такая схема защищает от неудачного выбора Sampling Rate и неожиданных изменений при пиках трафика, но позволяет без опаски устанавливать низкие значения Sampling Rate, когда это приемлемо.

4.2.3 Группа Counter Polling

Группа опроса счетчиков определяет места (источники данных) в устройстве, где можно получить показания счетчиков.

Обычно источниками данных служат интерфейсы устройства. Каждый источник данных может быть способен поддерживать более одного независимого процесса опроса. В таких случаях с источником данных будет связано множество экземпляров опроса счетчиков, каждый из которых может использовать свой интервал опроса.

4.3 Определения

```
SFLOW-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```

MODULE-IDENTITY, OBJECT-TYPE, Integer32, enterprises
    FROM SNMPv2-SMI
TEXTUAL-CONVENTION
    FROM SNMPv2-TC
SnmpAdminString
    FROM SNMP-FRAMEWORK-MIB
OwnerString
    FROM RMON-MIB
InetAddressType, InetAddress
    FROM INET-ADDRESS-MIB
MODULE-COMPLIANCE, OBJECT-GROUP
    FROM SNMPv2-CONF;

sFlowMIB MODULE-IDENTITY
    LAST-UPDATED "200309240000Z"    -- 24 сентября 2003 г.
    ORGANIZATION "sFlow.org"
    CONTACT-INFO
        "Peter Phaal
         sFlow.org
         http://www.sflow.org/

         Tel: +1-415-283-3260
         Email: peter.phaal@sflow.org"

```

DESCRIPTION

"MIB для управления генерацией и транспортировкой записей sFlow."

--

-- История выпусков

--

REVISION "200310180000Z" -- 18 ноября 2003 г.

DESCRIPTION

"Версия 1.3 (вариант 5)

Разрешает устанавливать SflowReceiver, если это не меняет значение."

REVISION "200309240000Z" -- 24 сентября 2003 г.

DESCRIPTION

"Версия 1.3 (вариант 4)

По умолчанию для sFlowRcvrAddress следует использовать шестнадцатеричное значение 00000000, для sFlowCpReceiver - 0."

REVISION "200304080000Z" -- 8 апреля 2003 г.

DESCRIPTION

"Версия 1.3 (вариант 3)

Разъяснена семантика интервала опроса счетчиков sFlowCpInterval."

REVISION "200209170000Z" -- 17 сентября 2002 г.

DESCRIPTION

"Версия 1.3 (вариант 2)

Добавлена поддержка множества сборщиков sFlow на sFlowDataSource. Номер организации смнен на sflow.org. Разделены выборки из потока и счетчиков, а также спецификация получателя в разные таблицы."

REVISION "200107310000Z" -- 31 июля 2001 г.

DESCRIPTION

"Версия 1.2

Преобразование структуры MIB в соответствии с SMIv2."

REVISION "200105010000Z" -- 1 мая 2001 г.

DESCRIPTION

"Версия 1.1

Добавлена переменная sfDatagramVersion."

::= { enterprises sflow(14706) 1 }

sFlowAgent OBJECT IDENTIFIER ::= { sFlowMIB 1 }

SFlowDataSource ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Указывает источник данных sFlow.

Ниже перечислены определенные в настоящее время источники.

- ifIndex.<I>

SflowDataSource этой традиционной формы называются port-based. Идеальный элемент отбора будет делать выборки из всех потоков, начинающихся или завершающихся на заданном интерфейсе. Однако, если архитектура коммутатора разрешает лишь входную и выходную выборку, агент выборки сможет брать образцы лишь из входных или выходных потоков. Каждый пакет для выборки должен рассматриваться лишь однократно, независимо от числа портов, куда он пересылается. Примечание. Порт 0 служит для указания всех портов устройства как одного источника данных. sFlowFsPacketSamplingRate применяется ко всем портам устройства, способным отбирать пакеты.

- smonVlanDataSource.<V>

SFlowDataSource для этой формы указывает Packet-based VLAN и называется источником данных VLAN-based. <V> указывает идентификатор VLAN, соответствующий стандарту IEEE 802.1Q. Он может принимать значения от 1 до 4094, включительно и представляет 802.1Q VLAN-ID со значимостью в масштабе домена на базе мостов.

Выборка выполняется для всех пакетов, относящихся к заданной VLAN (порт не имеет значения). Каждый пакет учитывается при выборке лишь один раз, независимо от числа выходных портов.

- entPhysicalEntry.<N>

SFlowDataSource для этой формы указывает на физический элемент в агенте (например, entPhysicalClass = backplane(4)) и называется источником данных entity-based. Выборка выполняется из всех пакетов, входящих в ресурс (например, при выборке из системной шины все пакеты, проходящие через магистраль, будут рассматриваться при отборе образцов, независимо от номеров портов).

Примечание. Поскольку каждый источник функционирует независимо, пакеты, проходящие через несколько источников, могут создавать множество записей для потоков."

SYNTAX OBJECT IDENTIFIER

SFlowInstance ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"При наличии более одного отборщика проб sFlow для данного SFlowDataSource эти отдельные отборщики различаются значением SFlowInstance. Значения переменной берутся из диапазона от 1 до n, где n - число отборщиков, связанный с источником данных.

Примечание. Каждый экземпляр отборщика sFlow должен работать независимо от всех других экземпляров. Установка атрибута для одного экземпляра не должна менять поведение и настройки других экземпляров."

SYNTAX Integer32 (1..65535)

SFlowReceiver ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Указывает получателя sFlow, связанного с этим ресурсом.

Нулевое значение говорит о доступности ресурса, отличные от 0 значения должны соответствовать действительным sFlowRcvrIndex.

Если в настоящий момент установлено значение 0, можно задать значение любой активной записи из sFlowRcvrTable. Для ненулевых значений будет возникать ошибка SNMP (bad value) при задании значения, отличного от 0 и текущего значения.

Установка значения 0 освобождает ресурс и возвращает для записи принятые по умолчанию значения.

Если срок действия записи в sFlowRcvrTable завершается в результате установки для sFlowRcvrOwner пустой строки или обнуления таймера sFlowRcvrTimeout, агент должен отметить все связанные с ней ресурсы как доступные (установив 0 в соответствующей записи SflowReceiver), а для записи должны быть установлены принятые по умолчанию значения.

Этот механизм не включает принудительного выполнения и опирается на кооперацию элементов управления для разрешения конфликтов при доступе к ресурсам. Элементам управления не следует менять какие-либо ресурсы без предварительного их приобретения путем записи своего значения sFlowRcvrIndex в качестве SFlowReceiver."

SYNTAX Integer32

sFlowVersion OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Однозначно указывает версию и реализацию этого модуля MIB.

Строка версии должна иметь структуру

<MIB Version>;<Organization>;<Software Revision>

где

<MIB Version> - должно быть 1.3, номер версии данного MIB.

<Organization> - название организации, ответственной за реализацию агента.

<Revision> - выпуск данного агента.

Например, строка '1.3;InMon Corp.;2.1.1' указывает, что агент

реализует SFLOW MIB версии '1.2', разработанный 'InMon Corp.'
с номером выпуска '2.1.1'.

Версия MIB будет меняться в каждом новом выпуске SFLOW MIB.

Элементы управления должны проверять версию MIB и не пытаться
управлять агентами, с номером версии больше того, для которого
модуль MIB был разработан.

Примечание. Формат дейтаграмм sFlow использует свой номер версии,
который может меняться независимо от <MIB Version>. Версия MIB
относится лишь к структуре и семантике SFLOW MIB."

```
DEFVAL { "1.3;;" }
 ::= { sFlowAgent 1 }
```

sFlowAgentAddressType OBJECT-TYPE

```
SYNTAX      InetAddressType
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
```

"Тип адреса, связанного с агентом. Поддерживаются лишь ipv4 и ipv6."

```
 ::= { sFlowAgent 2 }
```

sFlowAgentAddress OBJECT-TYPE

```
SYNTAX      InetAddress
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
```

"IP-адрес, связанный с агентом. Для многодомных агентов следует
указывать loopback-адрес агента. Адрес sFlowAgent должен
обеспечивать SNMP-связность с агентом. Поле адреса следует сохранять
при перенастройке, включении, отключении, добавлении и удалении
интерфейсов. Менеджер должен иметь возможность использовать
sFlowAgentAddress для однозначного указания агента в течение срока
поддержки истории работы."

```
 ::= { sFlowAgent 3 }
```

--

-- Таблица получателей

--

sFlowRcvrTable OBJECT-TYPE

```
SYNTAX      SEQUENCE OF SFlowRcvrEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
```

"Таблица получателей информации sFlow."

```
 ::= { sFlowAgent 4 }
```

sFlowRcvrEntry OBJECT-TYPE

```
SYNTAX      SFlowRcvrEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
```

"Атрибуты получателя sFlow."

```
INDEX { sFlowRcvrIndex }
```

```
 ::= { sFlowRcvrTable 1 }
```

SFlowRcvrEntry ::= SEQUENCE {

```
  sFlowRcvrIndex          Integer32,
  sFlowRcvrOwner          OwnerString,
  sFlowRcvrTimeout        Integer32,
  sFlowRcvrMaximumDatagramSize Integer32,
  sFlowRcvrAddressType    InetAddressType,
  sFlowRcvrAddress        InetAddress,
  sFlowRcvrPort           Integer32,
  sFlowRcvrDatagramVersion Integer32
```

}

sFlowRcvrIndex OBJECT-TYPE

```
SYNTAX      Integer32 (1..65535)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
```

"Индекс в таблице sFlowReceiverTable."

```
 ::= { sFlowRcvrEntry 1 }
```



```
sFlowRcvrOwner OBJECT-TYPE
    SYNTAX      OwnerString
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Элемент, использующий эту запись sFlowRcvrTable. Пустая строка
        говорит, что элемент не заявлен. Элемент, желающий заявить
        запись sFlowRcvrTable, должен убедиться, что эта запись свободна.
        Запись заявляется установкой строки владельца. Запись должна быть
        заявлена до того, как могут быть внесены какие-либо изменения в
        другие объекты отборщика.

        Для предотвращения конфликтов объект, контролирующий отборщик,
        должен установить владельца и значения для sFlowRcvrTimeout
        в одном запросе SNMP.

        Когда элемент управления завершает использование отборщика, ему
        следует снова установить пустое значения для sFlowRcvrOwner.
        Агент должен восстановить для всех других элементов этой строки
        принятые по умолчанию значения при удалении владельца. Он также
        должен освободить все другие ресурсы, связанные с записью
        sFlowRcvrTable.

        Этот механизм не включает принудительного выполнения и опирается
        на кооперацию элементов управления для разрешения конфликтов при
        доступе к ресурсам. "
    DEFVAL { "" }
    ::= { sFlowRcvrEntry 2 }
```

```
sFlowRcvrTimeout OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Число секунд до освобождения отборщика и остановки выборки.
        При установленном значении владелец организует контроль времени.
        При считывании возвращается оставшееся время.

        Элемент управления, желающий контролировать отборщик, отвечает
        за установку нового значения до истечения прежнего.

        По завершении интервала агент отвечает за восстановление для всех
        других элементов принятых по умолчанию значений. Он должен также
        освободить все ресурсы, связанные с этой записью sFlowRcvrTable."
    DEFVAL { 0 }
    ::= { sFlowRcvrEntry 3 }
```

```
sFlowRcvrMaximumDatagramSize OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Максимальное число байтов данных, которые могут быть переданы в
        одной дейтаграмме выборки. Менеджеру следует установить значение,
        предотвращающее фрагментирование дейтаграмм sFlow."
    DEFVAL { 1400 }
    ::= { sFlowRcvrEntry 4 }
```

```
sFlowRcvrAddressType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Тип адреса sFlowRcvrCollectorAddress."
    DEFVAL { ipv4 }
    ::= { sFlowRcvrEntry 5 }
```

```
sFlowRcvrAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "IP-адрес коллектора sFlow.
        При установке 0.0.0.0 дейтаграммы sFlow не будут передаваться."
    DEFVAL { '00000000'h } -- 0.0.0.0
    ::= { sFlowRcvrEntry 6 }
```

```
sFlowRcvrPort OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Порт получателя дейтаграмм sFlow."
    DEFVAL { 6343 }
    ::= { sFlowRcvrEntry 7 }

sFlowRcvrDatagramVersion OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Версия дейтаграмм sFlow, которые следует передавать.

        При передаче не поддерживаемого агентом значения агенту следует
        указать максимальный поддерживаемый номер (меньше предложенного)
        или вернуть ошибку SNMP bad value, если такого значения нет."
    DEFVAL { 5 }
    ::= { sFlowRcvrEntry 8 }

--
-- Таблица выборки из потоков
--

sFlowFsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SFlowFsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Таблица отборщиков потока в устройстве."
    ::= { sFlowAgent 5 }

sFlowFsEntry OBJECT-TYPE
    SYNTAX      SFlowFsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Атрибуты отборщика потоков."
    INDEX { sFlowFsDataSource, sFlowFsInstance }
    ::= { sFlowFsTable 1 }

SFlowFsEntry ::= SEQUENCE {
    sFlowFsDataSource          SFlowDataSource,
    sFlowFsInstance           SFlowInstance,
    sFlowFsReceiver           SFlowReceiver,
    sFlowFsPacketSamplingRate Integer32,
    sFlowFsMaximumHeaderSize Integer32
}

sFlowFsDataSource OBJECT-TYPE
    SYNTAX      SFlowDataSource
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "sFlowDataSource для данного отборщика потоков."
    ::= { sFlowFsEntry 1 }

sFlowFsInstance OBJECT-TYPE
    SYNTAX      SFlowInstance
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Экземпляр sFlow для этого отборщика потоков."
    ::= { sFlowFsEntry 2 }

sFlowFsReceiver OBJECT-TYPE
    SYNTAX      SFlowReceiver
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Получатель SFlowReceiver для этого отборщика потоков."
    DEFVAL { 0 }
    ::= { sFlowFsEntry 3 }
```

```

sFlowFsPacketSamplingRate OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Статистический период выборки для пакетов из этого источника.

        Значение N задает выборку 1/N пакетов из контролируемых потоков.
        Агенту следует выбрать свой алгоритм для добавления вариаций к
        периоду выборки, чтобы не отбирался в точности каждый N-й пакет.
        Значение 1 задает выборку каждого пакета, 0 отключает выборку.

        Агенту разрешено иметь минимальное и максимальное разрешенное
        значение для периода выборки. Минимальное значение ограничивает
        издержки, связанные с отбором пакетов, а максимальное может
        быть связано с аппаратными ограничениями (например, размер
        счетчика). Кроме того, не все разрешения между минимумом и
        максимумом могут использоваться для периода отбора (это
        вопрос реализации).

        Когда период выборки установлен, агент может регулировать его в
        диапазоне от минимального до максимального значения, выбирая
        ближайшее приемлемое значение.

        При чтении агент должен возвращать используемое значение периода
        отбора (после своей регулировки). Алгоритм отбора должен
        сходиться так, чтобы в среднем отбиралось 1/N пакетов из общего
        числа в контролируемом потоке."
    DEFVAL { 0 }
    ::= { sFlowFsEntry 4 }

sFlowFsMaximumHeaderSize OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Максимальное число байтов, которые следует копировать из
        выбранного пакета. Агент может иметь внутренние ограничения
        минимального и максимального размера. При попытке выйти за
        пределы разрешенных значений агенту следует установить
        ближайшее разрешенное значение."
    DEFVAL { 128 }
    ::= { sFlowFsEntry 5 }

--
-- Таблица опроса счетчиков
--

sFlowCpTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SFlowCpEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Таблица считывателей счетчиков в устройстве."
    ::= { sFlowAgent 6 }

sFlowCpEntry OBJECT-TYPE
    SYNTAX      SFlowCpEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Атрибуты считывателя счетчиков."
    INDEX { sFlowCpDataSource, sFlowCpInstance }
    ::= { sFlowCpTable 1 }

SFlowCpEntry ::= SEQUENCE {
    sFlowCpDataSource          SFlowDataSource,
    sFlowCpInstance           SFlowInstance,
    sFlowCpReceiver           SFlowReceiver,
    sFlowCpInterval           Integer32
}

sFlowCpDataSource OBJECT-TYPE
    SYNTAX      SFlowDataSource
    MAX-ACCESS  not-accessible
    STATUS      current

```

```

DESCRIPTION
    "Указывает источник данных для считывателя счетчиков."
    ::= { sFlowCpEntry 1 }

sFlowCpInstance OBJECT-TYPE
    SYNTAX      SFlowInstance
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Экземпляр sFlowInstance для считывателя счетчиков."
    ::= { sFlowCpEntry 2 }

sFlowCpReceiver OBJECT-TYPE
    SYNTAX      SFlowReceiver
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Получатель SflowReciever, связанный со считывателем счетчиков."
    DEFVAL { 0 }
    ::= { sFlowCpEntry 3 }

sFlowCpInterval OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION1
        "Максимальное число секунд между последовательными опросами
        счетчиков, связанных с этим источником данных. Значение 0
        отключает выборку счетчиков.

        Агенту разрешено иметь минимальное и максимальное разрешенное
        значение для интервала опроса. Минимальное значение ограничивает
        издержки, связанные с опросом счетчиков, а максимальное может
        быть связано с аппаратными ограничениями (например, размер
        счетчика). Кроме того, не все разрешения между минимумом и
        максимумом могут использоваться для периода отбора (это
        вопрос реализации).

        При чтении агент должен возвращать используемое значение периода
        отбора (после своей регулировки, как указано выше).

        DEFVAL { 0 }
        ::= { sFlowCpEntry 4 }

--
-- Заявления о соответствии
--

sFlowMIBConformance OBJECT IDENTIFIER ::= { sFlowMIB 2 }
sFlowMIBGroups       OBJECT IDENTIFIER ::= { sFlowMIBConformance 1 }
sFlowMIBCompliances OBJECT IDENTIFIER ::= { sFlowMIBConformance 2 }

sFlowCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
        "Заявления о совместимости для агента sFlow."

    MODULE - данный модуль
        MANDATORY-GROUPS { sFlowAgentGroup }

        OBJECT      sFlowAgentAddressType
        SYNTAX      InetAddressType { ipv4(1) }
        DESCRIPTION
            "От агента нужна лишь поддержка ipv4."

        OBJECT sFlowRcvrAddressType
        SYNTAX      InetAddressType { ipv4(1) }
        DESCRIPTION
            "От агента нужна лишь поддержка ipv4."
    ::= { sFlowMIBCompliances 1 }
sFlowAgentGroup OBJECT-GROUP
    OBJECTS { sFlowVersion, sFlowAgentAddressType, sFlowAgentAddress,
              sFlowRcvrOwner, sFlowRcvrTimeout,
              sFlowRcvrMaximumDatagramSize, sFlowRcvrAddressType,
              sFlowRcvrAddress, sFlowRcvrPort,

```

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.

```

sFlowRcvrDatagramVersion, sFlowFsReceiver,
sFlowFsPacketSamplingRate, sFlowFsMaximumHeaderSize,
sFlowCpReceiver, sFlowCpInterval }
STATUS current
DESCRIPTION
  "набор объектов для управления генерацией и транспортировкой
  записей sFlow."
:= { sFlowMIBGroups 1 }

```

END

sFlow MIB ссылается на определения из других RFC, включая [18], [19], [20] и [21].

5. Формат дейтаграмм sFlow

Формат дейтаграмм sFlow определяет стандартный формат, используемый агентом sFlow для передачи данных выборки удаленному коллектору sFlow.

Формат дейтаграмм задается с использованием стандарта XDR [32]. Это обеспечивает более компактное по сравнению с ASN.1 представление и упрощает кодирование в агентах sFlow и декодирование в коллекторах.

Выборки передаются в пакетах UDP с использованием адреса и порта, заданных в SFLOW MIB. Для протокола sFlow задан (и по умолчанию используется в SFLOW MIB) порт 6343. Всем агентам sFlow следует по умолчанию использовать порт UDP 6343. Стандартный номер порта позволяет избавиться от проблем несогласованности агентов и коллекторов sFlow, упрощает идентификацию пакетов sFlow, а также пересылку таких пакетов через промежуточные устройства (например, межсетевые экраны).

Недостаточная надежность транспорта UDP не оказывает значительного влияния на точность измерений агента sFlow.

- Если будет потеряна выборка счетчика, при следующем опросе будут переданы новые значения. Вероятность перехода через 0 в течение этого времени пренебрежимо мала. В дейтаграммах sFlow передаются 64-битовые счетчики, а интервал опроса составляет обычно от 20 до 120 секунд, поэтому вероятность потери значительного числа дейтаграмм с переходом счетчика через 0 очень мала.
- Потери выборок из пакетов в конечном итоге приводят лишь к незначительному снижению частоты отбора.

Использование UDP снижает объем памяти для буферизации данных, а также обеспечивает надежные способы своевременной доставки трафика в периоды высокой нагрузки (например, во время DoS-атаки). Протокол UDP более устойчив к отказам по сравнению с транспортом с гарантированной доставкой, поскольку единственным результатом перегрузок является незначительный рост задержки при передаче и числа теряемых пакетов, что не оказывает существенного влияния на работу системы мониторинга sFlow. Использование механизмов гарантированной доставки будет приводить к значительным задержкам при перегрузке и существенному росту размера буферов у агента.

Хотя структура дейтаграмм sFlow позволяет включать множество выборок в одну дейтаграмму, агент sFlow не обязан ждать заполнения буфера образцами и может передать дейтаграмму sFlow раньше. Протокол sFlow предназначен для своевременной доставки информации о трафике. Агент sFlow может задерживать выборку не более чем на 1 сек.

Агенту sFlow следует пытаться добавлять выборки счетчиков в поток дейтаграмм от выборки из пакетов. Перед отправкой дейтаграммы свободное пространство буфера может быть дополнено выборками счетчиков. Агент sFlow имеет право сам определять время опроса счетчиков, а значение sFlowCounterSamplingInterval указывает максимальный интервал между выборками. Если значения счетчиков должны быть переданы в соответствии с максимальным интервалом выборки, должна передаваться дейтаграмма со значениями соответствующих счетчиков.

Ниже приведено описание дейтаграмм sFlow в формате XDR.

```

/* Формат дейтаграмм sFlow версии 5 (вариант 10) */

/* История выпусков
  - в версии 5 добавлена поддержка:
    расширенного кодирования для flow_sample и counter_sample;
    неизвестных типов адресов;
    субагентов;
    информации об отбрасывании пакетов;
    фирменных расширений;
    информации о размере полей данных;
    информации о размере типов выборки;
    семантики сброса порядковых номеров flow_sample, counter_sample.
    определение дейтаграмм отделено от определения данных потоков и счетчиков.
    Примечание. История выпускной определений данных sFlow сейчас является
    частью документа с определениями данных. */

/* Типы адресов */

typedef opaque ip_v4[4];
typedef opaque ip_v6[16];

enum address_type {
    UNKNOWN = 0,
    IP_V4   = 1,
    IP_V6   = 2
}

```

¹Denial of service attack - атака на отказ служб.

```
union address (address_type type) {
    case UNKNOWN:    void;
    case IP_V4:      ip_v4;
    case IP_V6:      ip_v6;
}

```

/* Формат данных

Значение data_format однозначно указывает формат структуры ораque в спецификации sFlow. Синтаксис data_format показан ниже.

- старшие 20 битов соответствуют коду SMI Private Enterprise для элемента, отвечающего за определение структуры. Нулевое значение указывает стандартные структуры, определенные sflow.org;
- младшие 12 указывают номер формата структуры, определенный организацией, который должен быть уникальным.

В настоящее время определены 3 структуры ораque со значениями data_format:

1. sample_data;
2. counter_data;
3. flow_data.

Один и тот же номера формата можно использовать в разных контекстах. Например, (inmon,1) может указывать определенный набор счетчиков для описания counter_data и набор атрибутов потока для описания flow_data.

Разработчикам sFlow следует по-возможности использовать стандартные структуры даже при их частичном заполнении. Фирменные структуры разрешены, но применять их следует лишь в дополнение к имеющимся или для передачи информации, которая еще не стандартизована.

Организациям рекомендуется публиковать определения структур в формате XDR на сайте www.sflow.org. В документ с определением структур следует включать определение XDR с комментарием, указывающим структуры, к которым он относится, номером организации и номером структуры. Примерами могут служить определения counter_sample и flow_sample.

Примечание. Организациям, определяющим структуры sFlow, можно расширять такие определения структур в конце без смены номера структуры. Все изменения полей в опубликованных определениях структур должны выполняться со сменой номера структуры. Это правило позволяет добавлять данные в структуры, сохраняя совместимость с прежними версиями. Приложения, получающие данные sFlow всегда должны использовать данные о размере при декодировании структур ораque<>, чтобы расширенные структуры не вызвали ошибок. Отметим, что эти правила относятся и к стандартным структурам. */

```
typedef unsigned int data_format;
```

/* Кодирование sFlowDataSource показано ниже.

Старший байт source_id служит для указания типа sFlowDataSource:

- 0 = ifIndex
- 1 = smonVlanDataSource
- 2 = entPhysicalEntry

Три младших байта содержат значение подходящего индекса. */

```
typedef unsigned int sflow_data_source;
```

/* Входной/выходной порт.

Кодирует интерфейсы на пути пакета через устройство.

0 - интерфейс не известен.

Два старших бита указывают формат 30-битового значения.

- format = 0 - один интерфейс

Значением служит ifIndex этого интерфейса. Максимальное значение 0x3FFFFFFF указывает отсутствие входного или выходного интерфейса (в зависимости от поля, где указано).

Это используется для описания трафика, который не пересылается устройством, но отслеживается агентом, поскольку адресован устройству или создается им. Во входном поле это значение указывает пакеты, созданные устройством (например, пакет RIP, созданный маршрутизатором). В выходном поле это значение указывает пакеты, адресованные устройству (например, индивидуальный или групповой отклик RIP, принятый маршрутизатором IP).

- format = 1 - пакет отброшен

Значение указывает код причины:

- 0 - 255 используются коды ICMP Destination Unreachable, список которых доступен на сайте www.iana.org. В параграфе 5.2.7.1 RFC 1812 описаны текущие коды. Отметим, что применение этих кодов не предполагает, что пакет относится к протоколу IP, а для пакетов IP не предполагается генерация каких-либо сообщений ICMP.
- 0 сеть не доступна
- 1 хост не доступен
- 2 протокол не доступен
- 3 порт не доступен
- 4 нужна фрагментация, но установлен флаг DF
- 5 отказ на заданном источнике маршрута
- 6 целевая сеть не известна
- 7 целевой хост не доступен
- 8 исходных хост изолирован
- 9 связь с целевой сетью административно запрещена
- 10 связь с целевым хостом административно запрещена
- 11 целевая сеть не доступна для ToS
- 12 целевой хост не доступен для ToS
- 13 связь административно запрещена
- 14 нарушение предпочтений хоста
- 15 действует отсечка предпочтений
- 256 = неизвестно
- 257 = завершилось время жизни
- 258 = ACL
- 259 = нет места в буфере
- 260 = RED
- 261 = формирование трафика или ограничение скорости
- 262 = слишком большой пакет (для протоколов без фрагментирования)

Примечание. С течением времени могут публиковаться дополнительные коды причин. Приложения, получающие sFlow, должны быть готовы воспринимать дополнительные коды. Актуальный список кодов будет поддерживаться на сайте www.sflow.org.

- format = 2 - множество интерфейсов-получателей

Значение указывает число интерфейсов. 0 указывает неизвестное число интерфейсов больше 1.

Примечание. Форматы 1 и 2 применимы только к выходному интерфейсу. Пакет всегда принимается одним (иногда неизвестным) интерфейсом.

Примеры:

```
0x00000002 указывает ifIndex = 2;
0x00000000 ifIndex не известен;
0x40000102 пакет отброшен на основании ACL1;
0x80000007 пакет передан в 7 интерфейсов;
0x80000000 пакет передан в неизвестное число интерфейсов > 1. */
```

```
typedef unsigned int interface;
```

```
/* Форматы выборки счетчиков и потоков
```

Определены компактные и расширенные формы образцов для счетчиков и потоков. Агенту недопустимо смешивать компактное и расширенное кодирование. Если агент никогда не будет использовать значения ifIndex $\geq 2^{24}$, он должен применять компактное представление для всех интерфейсов, в остальных случаях должно использоваться расширенное представление.

Хотя теоретический диапазон значений ifIndex составляет 2^{32} , RFC 2863 рекомендует выделять номера с использованием небольших целых чисел, начиная с 1. Для большинства реализаций агентов поддерживается 2^{24} значений ifIndex для компактного представления, которое более адекватно и экономит пропускную способность. Расширенное кодирование позволяет использовать полный диапазон значений ifIndex, хотя большие значения ifIndex не приветствуются. */

```
struct flow_record {
    data_format flow_format;          /* Формат sflow_data */
    opaque flow_data<>;              /* Данные потока однозначно
                                     определяются flow_format. */
}
```

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.

```

struct counter_record {
    data_format counter format; /* формат counter_data */
    opaque counter_data<>; /* Блок счетчиков однозначно
                             определен counter_format. */
}

/* Компактная форма выборки счетчиков и потоков
   Если ifIndex всегда < 224, должна применяться эта форма. */

/* Формат одной выборки из потока. */
/* opaque = sample_data; enterprise = 0; format = 1 */

struct flow_sample {
    unsigned int sequence_number; /* Инкрементируется для каждой выборки
                                   из потока с этим экземпляром sFlow1.
                                   Примечание. Если агент сбрасывает
                                   sample_pool, он должен
                                   сбрасывать и sequence_number.*/
    sflow_data_source source_id; /* sFlowDataSource */
    unsigned int sampling_rate; /* sFlowPacketSamplingRate */
    unsigned int sample_pool; /* Общее число пакетов, которые могут
                               быть отобраны (т. е. пропущенные
                               пакеты и общее число выборок) */
    unsigned int drops; /* Общее число случаев, когда агент sFlow
                        обнаружил отбрасывание помеченного для
                        выборки пакета по причине нехватки
                        ресурсов. Счетчик указывает общее число
                        обнаруженных фактов с момента сброса
                        агента. Большое число отбрасываний
                        показывает, что агент управления не
                        способен обрабатывать выборки со
                        скоростью их создания оборудованием.
                        Увеличение sampling_rate снизит скорость
                        отбрасывания. Если агент не может
                        обнаружить отбрасывание, он сообщает 0. */
    interface input; /* Интерфейс, принявший пакет. */
    interface output; /* Интерфейс, передавший пакет. */
    flow_record flow_records<>; /* Информация о выбранном пакете. */
}

/* Формат одной выборки счетчика. */
/* opaque = sample_data; enterprise = 0; format = 2 */

struct counters_sample {
    unsigned int sequence_number; /* Инкрементируется с каждой выборкой
                                   для этого экземпляра sFlow1.
                                   Примечание. Если агент сбрасывает любой
                                   из счетчиков, он должен также
                                   сбрасывать sequence_number. Для
                                   source_id на базе ifIndex порядковый
                                   номер должен сбрасываться при каждом
                                   изменении ifCounterDiscontinuityTime. */
    sflow_data_source source_id; /* sFlowDataSource */
    counter_record counters<>; /* Счетчики, опрашиваемые для этого источника. */
}

/* Расширенный формат выборки счетчиков и потоков.
   Если ifIndex может быть >= 224, должен применяться этот формат. */

struct sflow_data_source_expanded {
    unsigned int source_id_type; /* Тип sFlowDataSource */
    unsigned int source_id_index; /* Индекс sFlowDataSource */
}

struct interface_expanded {
    unsigned int format; /* формат интерфейса */
    unsigned int value; /* значение интерфейса */
}

/* Формат одной расширенной выборки из потока. */
/* opaque = sample_data; enterprise = 0; format = 3 */

```

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.


```

struct flow_sample_expanded {
    unsigned int sequence_number; /* Инкрементируется для каждой выборки
                                  из потока с с этим экземпляром sFlow1.
                                  Примечание. Если агент сбрасывает
                                  sample_pool, он должен
                                  сбрасывать и sequence_number.*/
    sflow_data_source_expanded source_id; /* sFlowDataSource */
    unsigned int sampling_rate; /* sFlowPacketSamplingRate */
    unsigned int sample_pool; /* Общее число пакетов, которые могут
                               быть отобраны (т. е. пропущенные
                               пакеты и общее число выборок) */
    unsigned int drops; /* Общее число случаев, когда агент sFlow
                        обнаружил отбрасывание помеченного для
                        выборки пакета по причине нехватки
                        ресурсов. Счетчик указывает общее число
                        обнаруженных фактов с момента сброса
                        агента. Большое число отбрасываний
                        показывает, что агент управления не
                        способен обрабатывать выборки со
                        скоростью их создания оборудованием.
                        Увеличение sampling_rate снизит скорость
                        отбрасывания. Если агент не может
                        обнаружить отбрасывание, он сообщает 0. */
    interface_expanded input; /* Интерфейс, принявший пакет. */
    interface_expanded output; /* Интерфейс, передавший пакет. */
    flow_record flow_records<>; /* Информация о выбранном пакете. */
}

/* Формат одной расширенной выборки счетчика. */
/* opaque = sample_data; enterprise = 0; format = 4 */
struct counters_sample_expanded {
    unsigned int sequence_number; /* Инкрементируется с каждой выборкой
                                  для этого экземпляра sFlow1.
                                  Примечание. Если агент сбрасывает любой
                                  из счетчиков, он должен также
                                  сбрасывать sequence_number. Для
                                  source_id на базе ifIndex порядковый
                                  номер должен сбрасываться при каждом
                                  изменении ifCounterDiscontinuityTime. */
    sflow_data_source_expanded source_id; /* sFlowDataSource */
    counter_record counters<>; /* Счетчики, опрашиваемые для этого источника. */
}

/* Формат дейтаграммы с выборкой. */

struct sample_record {
    data_format sample_type; /* Указывает тип данных выборки. */
    opaque sample_data<>; /* Структура, соответствующая sample_type. */
}

/* Информация заголовка для дейтаграмм sFlow версии 5.

```

Поле субагента используется в тех случаях, когда агент sFlow реализован в распределенной архитектуре и для передачи непрактичны выборки в одной точке.

Однако настоятельно рекомендуется не применять механизм субагентов без крайней необходимости. Если в устройстве доступно множество процессоров, разные задачи, связанные с выборками из потоков и счетчиков, можно распределять между этими процессорами. Однако архитектуру агента следует организовать так, чтобы все образцы отправлялись в один поток дейтаграмм. финальная задача отправки образцов включает минимальную обработку, но может существенно влиять на расширяемость системы sFlow. За счет снижения числа пакетов UDP и потоков, связанные с sFlow издержки на приемной стороне существенно снижаются.

Каждый источник sFlowDataSource должен быть связан лишь с одним субагентом. Привязка sFlowDataSource к субагенту должна сохраняться в течение всей сессии sFlow. */

```

struct sample_datagram_v5 {
    address agent_address /* IP-адрес агента выборки sFlowAgentAddress. */
    unsigned int sub_agent_id; /* Служит для обозначения потоков дейтаграмм от
                               разных элементов агента в устройстве. */
    unsigned int sequence_number; /* Инкрементируется для каждой дейтаграммы выборки
                                   созданной субагентом в рамках агента. */
    unsigned int uptime; /* Текущее время (в миллисекундах с момента

```

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.

загрузки устройства). Следует устанавливать время, близкое к моменту передачи дейтаграммы.

Примечание. Хотя субагенту нужно пытаться отслеживать глобальное значение sysUptime, получателю пакетов sFlow недопустимо предполагать синхронность субагентов. */
 sample_record samples<>; /* Массив записей с выборками. */

```

}

enum datagram_version {
  VERSION5 = 5
}

union sample_datagram_type (datagram_version version) {
  case VERSION5:
    sample_datagram_v5 datagram;
}

struct sample_datagram {
  sample_datagram_type version;
}

```

Дейтаграмма sFlow содержит список записей из потоков и счетчиков. Формат каждой записи Packet Flow указан значением data_format. Пространство значений data_format может расширяться для добавления к стандартным типам фирменных расширений производителей.

Число стандартных типов записей определено. Однако агенты sFlow не обязаны поддерживать все типы записей и могут ограничиться лишь теми, которые применимы к трактовкам конкретных пакетов, о которых сообщает агент. Например, коммутатор L2 не будет информировать о подсетях, поскольку он не выполняет маршрутизации. Коммутатор L2/3 будет сообщать данные L2 для коммутируемых пакетов и данные L2 и L3 для маршрутизируемых.

Ниже приведено описание наборов данных дейтаграмм sFlow в формате XDR.

```

/* Предложенные стандартные форматы данных sFlow (вариант 14) */
/* История выпусков
- версия 5
    уточнены определения extended_switch для портов без тега;
    добавлена информация о загрузке CPU и памяти;
    добавлена информация о туннелях mpls, виртуальных каналах и fec;
    уточнены определения next_hop;
    добавлен урезанный счетчик в sampled_header;
    добавлен POS header_protocol;
    добавлен BGP next hop router;
    уточнено определение размера пакета;
    снято ограничение для размера заголовка пакетов;
    добавлено поле host к расширению URL и уточнено url_direction;
    определен url как строка запроса http
    добавлена информация о кодировке символов для данных пользователя;
    добавлена поддержка NAT;
    добавлена информация MPLS.
- в версии 4 добавлена поддержка групп BGP.
- в версии 3 добавлена поддержка extended_url. */

```

```

/* Enterprise = 0 указывает стандартные структуры sFlow. Разработчикам
sFlow следует применять стандартные структуры, когда это возможно, даже
если они не используются полностью. Фирменные структуры разрешены, но
их следует применять лишь в дополнение к имеющимся структурам или для
передачи информации, которая еще не стандартизована.

```

Приведенные ниже значения следует применять для неизвестных полей (если в определении структуры не указано иное).

- Неизвестное целое число. Используйте значение 0.
- Неизвестный счетчик. Используйте максимальное значение счетчика для индикации его недоступности. В любой данной сессии sFlow конкретный счетчик должен быть всегда доступным или всегда недоступным. Доступные счетчики могут временно иметь максимальное значение перед сбросом в 0. Это разрешено.
- Неизвестная строка. Используйте пустую строку (размера 0). */

```

/* Типы данных потока.

```

В flow_sample должна содержаться информация о заголовке пакета. Предпочтителен формат sampled_header, однако при недоступности заголовка для процесса выборки можно использовать 1 или несколько sampled_ethernet, sampled_ipv4, sampled_ipv6. */

```
/* Данные заголовка пакета. */
```

```
/* Нумерация header_protocol может со временем расширяться.
Приложения, получающие sFlow, должны быть готовы воспринимать
структуры sampled_header с неизвестными значениями.
```

```
Актуальный список номеров поддерживается на сайте www.sflow.org. */
```

```
enum header_protocol {
    ETHERNET-ISO88023      = 1,
    ISO88024-TOKENBUS     = 2,
    ISO88025-TOKENRING    = 3,
    FDDI                  = 4,
    FRAME-RELAY           = 5,
    X25                   = 6,
    PPP                   = 7,
    SMDS                  = 8,
    AAL5                  = 9,
    AAL5-IP               = 10, /* например, Cisco AAL5 мультиплекс */
    IPv4                  = 11,
    IPv6                  = 12,
    MPLS                  = 13,
    POS                   = 14 /* RFC 1662, 2615 */
}
```

```
/* Заголовок необработанных (Raw) пакетов */
/* opaque = flow_data; enterprise = 0; format = 1 */
```

```
struct sampled_header {
    header_protocol protocol; /* Формат выбранного заголовка. */
    unsigned int frame_length; /* Исходный размер пакета до выборки.
Примечание. Для header_protocol
канального уровня - общее число
октетов данных, полученных из
сети (без битов кадрирования, но
с учетом FCS). Аппаратные
ограничения могут препятствовать
указанию точно размера кадра на
канальном уровне, но агенту следует
по-возможности соблюдать точность.
Оклеты, добавленные к frame_length
для компенсации удаленной инкапсуляции,
должны учитываться в числе вырезанных. */
    unsigned int stripped; /* Число октетов, удаленных из пакета до
извлечения октетов header<>. Данные
трейлера, соответствующего удаленной
инкапсуляции, также должны вырезаться.
Данные трейлера для внешнего протокола,
включенного в выборку, должны вырезаться.

Для пакета 802.3 без инкапсуляции вырезается
не менее 4 октетов, поскольку в дополнение к
FCS может вырезаться тег VLAN.

Внешняя инкапсуляция, которая не однозначна
или не относится к стандартным header_protocol,
должна вырезаться. */
    opaque header<>; /* Байты заголовка. */
}
```

```
typedef opaque mac[6];
```

```
/* Данные кадра Ethernet. */
/* opaque = flow_data; enterprise = 0; format = 2 */
```

```
struct sampled_ethernet {
    unsigned int length; /* Размер кадра MAC, полученного из сети, без
учета инкапсуляции нижележащих уровней и
кадрирования, но с учетом октетов FCS. */
    mac src_mac; /* MAC-адрес отправителя. */
    mac dst_mac; /* MAC-адрес получателя. */
    unsigned int type; /* Тип пакета Ethernet. */
}
```

```
/* Данные пакета IPv4. */
/* opaque = flow_data; enterprise = 0; format = 3 */
```

```

struct sampled_ipv4 {
    unsigned int length;          /* Размер пакета IP без учета инкапсуляции
                                  нижележащих уровней. */
    unsigned int protocol;       /* Тип протокола IP (например, TCP = 6, UDP = 17) */
    ip_v4_src_ip;               /* IP-адрес отправителя. */
    ip_v4_dst_ip;               /* IP-адрес получателя. */
    unsigned int src_port;       /* Порт отправителя TCP/UDP или его эквивалент. */
    unsigned int dst_port;       /* Порт получателя TCP/UDP или его эквивалент. */
    unsigned int tcp_flags;      /* Флаги TCP. */
    unsigned int tos;            /* IP TOS. */
}

/* Данные пакета IPv6. */
/* opaque = flow_data; enterprise = 0; format = 4 */

struct sampled_ipv6 {
    unsigned int length;          /* Размер пакета IP без учета инкапсуляции
                                  нижележащих уровней. */
    unsigned int protocol;       /* IP next header (например, TCP = 6, UDP = 17) */
    ip_v6_src_ip;               /* IP-адрес отправителя. */
    ip_v6_dst_ip;               /* IP-адрес получателя. */
    unsigned int src_port;       /* Порт отправителя TCP/UDP или его эквивалент. */
    unsigned int dst_port;       /* Порт получателя TCP/UDP или его эквивалент. */
    unsigned int tcp_flags;      /* Флаги TCP. */
    unsigned int priority;       /* IP priority */
}

/* Расширенные данные потока.

Расширенные типы данных обеспечивают дополнительную информацию о
выбранном пакете. В каждую выборку влудует включать все применимые
расширенные данные. */

/* Расширенные данные коммутатора. */
/* opaque = flow_data; enterprise = 0; format = 1001 */
/* Примечание. Для входных портов без тегов используются назначенные
значения vlan и приоритета в качестве src_vlan и src_priority.
Для выходных портов без тегов используются значения dst_vlan и
dst_priority, которые будут помещаться в тег 802.1Q и делать
порт членом VLAN. */

struct extended_switch {
    unsigned int src_vlan;        /* 802.1Q VLAN из входящего кадра.
                                   0xffffffff, если тег не известен. */
    unsigned int src_priority;    /* Приоритет 802.1p из входящего кадра.
                                   0xffffffff, если приоритет не известен. */
    unsigned int dst_vlan;        /* 802.1Q VLAN исходящего кадра.
                                   0xffffffff, если тег не известен. */
    unsigned int dst_priority;    /* Приоритет 802.1p исходящего кадра.
                                   0xffffffff, если приоритет не известен. */
};1

/* Следующий интервал маршрута IP.
ipForwardNextHop (RFC 2096) для маршрутов IPv4.
ipv6RouteNextHop (RFC 2465) для маршрутов IPv6. */

typedef next_hop address;

/* Расширенные данные маршрутизатора. */
/* opaque = flow_data; enterprise = 0; format = 1002 */

struct extended_router {
    next_hop nexthop;            /* IP-адрес следующего маршрутизатора. */
    unsigned int src_mask_len;   /* Число битов в маске префикса отправителя. */
    unsigned int dst_mask_len;   /* Число битов в маске префикса получателя. */
}

enum as_path_segment_type {
    AS_SET = 1,                  /* Неупорядоченный набор AS. */
    AS_SEQUENCE = 2              /* Упорядоченный набор AS. */
}

```

¹В оригинале допущена ошибка. См. <https://sflow.org/developers/errata.php>. Прим. перев.

```

union as_path_type (as_path_segment_type) {
    case AS_SET:
        unsigned int as_set<>;
    case AS_SEQUENCE:
        unsigned int as_sequence<>;
}

/* Расширенные данные шлюза. */
/* opaque = flow_data; enterprise = 0; format = 1003 */

struct extended_gateway {
    next_hop nexthop;          /* Адрес граничного маршрутизатора
                               для целевой сети. */
    unsigned int as;          /* Номер AS для маршрутизатора. */
    unsigned int src_as;      /* Номер AS источника */
    unsigned int src_peer_as; /* Номер AS партнера со стороны источника. */
    as_path_type dst_as_path<>; /* Путь AS к получателю */
    unsigned int communities<>; /* Группы, связанные с маршрутом. */
    unsigned int localpref;   /* LocalPref для маршрута. */
}

/* Кодировка символов
   Значение MIBEnum набора символов, используемого в строках, см. RFC 2978
   По-возможности следует применять кодировку UTF-8 (MIBEnum=106).
   0 указывает неизвестную кодировку. */

typedef unsigned int charset;

/* Расширенные данные пользователя. */
/* opaque = flow_data; enterprise = 0; format = 1004 */

struct extended_user {
    charset src_charset;      /* Кодировка для src_user. */
    opaque src_user<>;        /* Идентификатор пользователя, связанный
                               с источником пакета. */
    charset dst_charset;     /* Кодировка для dst_user. */
    opaque dst_user<>;        /* Идентификатор пользователя, связанный
                               с адресатом пакета. */
}

enum url_direction {
    src = 1,                  /* Источником является сервер. */
    dst = 2                   /* Получателем является сервер. */
}

/* Расширенные данные URL. */
/* opaque = flow_data; enterprise = 0; format = 1005 */

struct extended_url {
    url_direction direction; /* Направление соединения. */
    string url<>;            /* Строка запроса HTTP (см. RFC 2616). */
    string host<>;          /* Поле host из заголовка HTTP. */
}

/* Стек меток MPLS.
   - При неизвестном значении может возвращаться пустой стек.
   - Если известна лишь внешняя метка, стек может содержать 1 запись.
   - Кодирование меток описано в RFC 3032.
   - Метки используют сетевой порядок байтов. */
typedef int label_stack<>;

/* Расширенные данные MPLS. */
/* opaque = flow_data; enterprise = 0; format = 1006 */

struct extended_mpls {
    next_hop nexthop;        /* Адрес следующего интервала. */
    label_stack in_stack;    /* Стек меток принятого пакета. */
    label_stack out_stack;   /* Стек меток передаваемого пакета. */
}

/* Расширенные данные NAT.
   Записи для заголовка пакета указывают адрес, видимый в sFlowDataSource.
   Структура extended_nat указывает адрес отправителя и/или получателя
   после трансляции. Если адрес не меняется, он должен совпадать с
   указанным для заголовка. */
/* opaque = flow_data; enterprise = 0; format = 1007 */

```

```

struct extended_nat {
    address src_address;          /* Адрес отправителя. */
    address dst_address;         /* Адрес получателя. */
}

/* Расширенные данные туннеля MPLS. */
/* opaque = flow_data; enterprise = 0; format = 1008 */

struct extended_mpls_tunnel {
    string tunnel_lsp_name<>;    /* Имя туннеля. */
    unsigned int tunnel_id;      /* Идентификатор туннеля. */
    unsigned int tunnel_cos;     /* COS для туннеля. */
}

/* Расширенные данные MPLS VC. */
/* opaque = flow_data; enterprise = 0; format = 1009 */

struct extended_mpls_vc {
    string vc_instance_name<>;   /* Имя экземпляра VC. */
    unsigned int vll_vc_id;      /* Идентификатор экземпляра VLL/VC. */
    unsigned int vc_label_cos;   /* COS для VC Label. */
}

/* Расширенные данные MPLS FEC.
   - Определения из MPLS-FTN-STD-MIB mplsFTNTable */
/* opaque = flow_data; enterprise = 0; format = 1010 */

struct extended_mpls_FTN {
    string mplsFTNDescr<>;
    unsigned int mplsFTNMask;
}

/* Расширенные данные MPLS LVP FEC.
   - Определения из MPLS-LDP-STD-MIB mplsFecTable
   Примечание. Данные mplsFecAddrType, mplsFecAddr доступны из заголовка */
/* opaque = flow_data; enterprise = 0; format = 1011 */

struct extended_mpls_LDP_FEC {
    unsigned int mplsFecAddrPrefixLength;
}

/* Расширенные данные туннеля VLAN.
   Запись внешней инкапсуляции VLAN, которая была вырезана. Следует
   сообщать информацию extended_vlanntunnel лишь при выполнении
   всех перечисленных ниже условий.
   1. Пакет имеет вложенные теги VLAN.
   2. Рапортующее устройство осведомлено о VLAN.
   3. Один или несколько тегов VLAN были вырезаны в результате
   использования фирменной инкапсуляции или по причине того,
   что оборудование коммутатора автоматически вырезает внешнюю
   инкапсуляцию VLAN.
   Передача информации extended_vlanntunnel не замещает передачу
   extended_switch. Данные extended_switch должны передаваться всегда
   для описания входящей/исходящей информации VLAN для пакета.
   Данные extended_vlanntunnel применимы лишь к вложенным VLAN и только
   при вырезании одного или нескольких тегов VLAN. */
/* opaque = flow_data; enterprise = 0; format = 1012 */

extended_vlanntunnel {
    unsigned int stack<>; /* Список вырезанных уровней 802.1Q TPID/TCI. Каждая
                           пара TPID,TCI представляется 32-битовым целым числом.
                           Уровни указываются от внешнего к внутреннему. */
}

/* Типы данных счетчиков

   По возможности следует включать блок if_counters. Могут включаться также
   связанные со средой счетчики. */

/* Базовые счетчики интерфейсов - RFC 2233 */
/* opaque = counter_data; enterprise = 0; format = 1 */

struct if_counters {
    unsigned int ifIndex;
    unsigned int ifType;
    unsigned hyper ifSpeed;
}

```

```

unsigned int ifDirection;      /* Взято из MAU MIB (RFC 2668)
                                0 = неизвестно, 1=полнодуплексный,
                                2=полудуплексный, 3 = вход, 4=выход */

unsigned int ifStatus;        /* битовое поле
                                бит 0 = ifAdminStatus (0 = down, 1 = up)
                                бит 1 = ifOperStatus (0 = down, 1 = up) */

unsigned hyper ifInOctets;
unsigned int ifInUcastPkts;
unsigned int ifInMulticastPkts;
unsigned int ifInBroadcastPkts;
unsigned int ifInDiscards;
unsigned int ifInErrors;
unsigned int ifInUnknownProtos;
unsigned hyper ifOutOctets;
unsigned int ifOutUcastPkts;
unsigned int ifOutMulticastPkts;
unsigned int ifOutBroadcastPkts;
unsigned int ifOutDiscards;
unsigned int ifOutErrors;
unsigned int ifPromiscuousMode;
}

/* Счетчики интерфейса Ethernet, см. RFC 2358 */
/* opaque = counter_data; enterprise = 0; format = 2 */

struct ethernet_counters {
    unsigned int dot3StatsAlignmentErrors;
    unsigned int dot3StatsFCSErrors;
    unsigned int dot3StatsSingleCollisionFrames;
    unsigned int dot3StatsMultipleCollisionFrames;
    unsigned int dot3StatsSQETestErrors;
    unsigned int dot3StatsDeferredTransmissions;
    unsigned int dot3StatsLateCollisions;
    unsigned int dot3StatsExcessiveCollisions;
    unsigned int dot3StatsInternalMacTransmitErrors;
    unsigned int dot3StatsCarrierSenseErrors;
    unsigned int dot3StatsFrameTooLongs;
    unsigned int dot3StatsInternalMacReceiveErrors;
    unsigned int dot3StatsSymbolErrors;
}

/* Счетчики Token Ring, см. RFC 1748 */
/* opaque = counter_data; enterprise = 0; format = 3 */

struct tokenring_counters {
    unsigned int dot5StatsLineErrors;
    unsigned int dot5StatsBurstErrors;
    unsigned int dot5StatsACErrors;
    unsigned int dot5StatsAbortTransErrors;
    unsigned int dot5StatsInternalErrors;
    unsigned int dot5StatsLostFrameErrors;
    unsigned int dot5StatsReceiveCongestions;
    unsigned int dot5StatsFrameCopiedErrors;
    unsigned int dot5StatsTokenErrors;
    unsigned int dot5StatsSoftErrors;
    unsigned int dot5StatsHardErrors;
    unsigned int dot5StatsSignalLoss;
    unsigned int dot5StatsTransmitBeacons;
    unsigned int dot5StatsRecoverys;
    unsigned int dot5StatsLobeWires;
    unsigned int dot5StatsRemoves;
    unsigned int dot5StatsSingles;
    unsigned int dot5StatsFreqErrors;
}

/* Счетчики интерфейса 100BaseVG, см. RFC 2020 */
/* opaque = counter_data; enterprise = 0; format = 4 */

struct vg_counters {
    unsigned int dot12InHighPriorityFrames;
    unsigned hyper dot12InHighPriorityOctets;
    unsigned int dot12InNormPriorityFrames;
    unsigned hyper dot12InNormPriorityOctets;
    unsigned int dot12InIPMErrors;
    unsigned int dot12InOversizeFrameErrors;
    unsigned int dot12InDataErrors;
}

```

```

unsigned int dot12InNullAddressedFrames;
unsigned int dot12OutHighPriorityFrames;
unsigned hyper dot12OutHighPriorityOctets;
unsigned int dot12TransitionIntoTrainings;
unsigned hyper dot12HCInHighPriorityOctets;
unsigned hyper dot12HCInNormPriorityOctets;
unsigned hyper dot12HCOutHighPriorityOctets;
}

/* Счетчики VLAN */
/* opaque = counter_data; enterprise = 0; format = 5 */

struct vlan_counters {
    unsigned int vlan_id;
    unsigned hyper octets;
    unsigned int ucastPkts;
    unsigned int multicastPkts;
    unsigned int broadcastPkts;
    unsigned int discards;
}

/* Проценты указываются в сотых долях, т. е. 100 = 1%.
   Если значение не известно, используется -1. */

typedef int percentage;

/* Данные о процессоре */
/* opaque = counter_data; enterprise = 0; format = 1001 */

struct processor {
    percentage 5s_cpu;           /* средняя загрузка CPU в течение 5 секунд */
    percentage 1m_cpu;          /* средняя загрузка CPU в течение 1 минуты */
    percentage 5m_cpu;          /* средняя загрузка CPU в течение 5 минут */
    unsigned hyper total_memory /* размер памяти (в байтах) */
    unsigned hyper free_memory /* свободная память (в байтах) */
}

```

В спецификациях диаграмм sFlow и записей данных используются определения из множества опубликованных RFC, включая [22], [23], [24], [25], [26], [27], [28], [29], [30], [31].

6. Вопросы безопасности

При развертывании системы мониторинга трафика возникает ряд вопросов, связанных с безопасностью. sFlow не обеспечивает каких-либо механизмов защиты, опираясь на продуманное развертывание и настройку конфигурации для обеспечения адекватного уровня защиты.

Хотя развертывание системы мониторинга связано с некоторым риском, этот мониторинг обеспечивает также мощные средства обнаружения и отслеживания несанкционированных действий в сети.

В этом разделе представлена информация, которая поможет оценить возможные риски и параметры конфигурации, позволяющие снизить риск.

6.1 Конфигурация

sFlow MIB применяется для настройки агентов sFlow. Защита SNMP со списками управления доступом обычно считается адекватной для корпоративных сред. Однако в некоторых ситуациях эти меры защиты не эффективны (например, при настройке маршрутизаторов ядра Internet) и настройку по протоколу SNMP отключают.

При отключенной поддержке SNMP обычно используется командный интерфейс через консоль управления. Аргументы, используемые для настройки выборки sFlow на интерфейсе, приведены ниже.

```

-SFlowDataSource           <source>
-sFlowFsPacketSamplingRate <rate>
-sFlowFsMaximumHeaderSize  <header size>
-sFlowCpInterval           <interval>
-sFlowRcvrMaximumDatagramSize <datagram size>
-sFlowRcvrCollectorAddress <address>
-sFlowRcvrCollectorPort    <port>

```

Если командный интерфейс используется вместе с SNMP для настройки агента sFlow, разработчикам агента следует обеспечить независимую работу этих механизмов или передачу внесенных с помощью команд изменений в sFlow MIB. Обычно внесенные с помощью команд изменения имеют более высокий приоритет, нежели изменения по протоколу SNMP. Записи, созданные с помощью команд, следует делать неизменяемыми для SNMP. Тогда строки в каждой из таблиц SNMP будут либо принадлежать командному интерфейсу (и не могут быть изменены с помощью SNMP), либо будут относиться к агенту SNMP с возможностью изменения по протоколу SNMP.

6.2 Транспорт

Информация о трафике передается от агентов sFlow коллекторам через сеть без шифрования, что делает возможным ее перехват. Риск можно снизить путем организации защищенной измерительной сети и передачи диаграмм sFlow

через эту сеть. Выбор технологии для защищенной измерительной сети определяется реализацией и может включать VLAN и/или туннели VPN.

Коллекторы sFlow уязвимы для атак с использованием фиктивных дейтаграмм sFlow. Для снижения риска коллекторам sFlow следует проверять порядковые номера дейтаграмм и адреса их отправителей. При использовании защищенной измерительной сети следует обрабатывать лишь дейтаграммы sFlow, принятые из этой сети.

6.3 Конфиденциальность

Данные о трафике могут раскрывать конфиденциальную информацию об отдельных пользователях сети. Степень раскрытия данных прикладного уровня можно контролировать путем ограничения размера заголовков, собираемых агентами sFlow. Кроме того, механизмы выборки делают практически невозможным захват последовательных пакетов из отдельной транзакции.

Картинки трафика, различаемые при декодировании дейтаграмм sFlow в коллекторе, могут раскрывать детали действий конкретных пользователей, поэтому требуется защитить доступ к коллекторам sFlow.

7. Литература

- [1] Phaal, P., Panchen, S., and N. McKee, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks", RFC 3176, September 2001.
- [2] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.
- [3] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.
- [4] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.
- [5] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.
- [6] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [7] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [8] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [9] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, [RFC 1157](#), May 1990.
- [10] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.
- [11] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.
- [12] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.
- [13] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.
- [14] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.
- [15] Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.
- [16] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.
- [17] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.
- [18] S. Waldbusser, "Remote Network Monitoring Management Information Base", RFC 2819, May 2000.
- [19] Waterman, R., Lahaye, B., Romascanu, D., and S. Waldbusser, "Remote Network Monitoring MIB Extensions for Switched Networks Version 1.0", RFC 2613, June 1999.
- [20] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 2851, June 2000.
- [21] N. Brownlee, "Traffic Flow Measurement: Meter MIB", RFC 2720, October 1999.
- [22] Smith, A., Flick, J., de Graaf, K., Romascanu, D., McMaster, D., McCloghrie, K., and S. Roberts, "Definition of Managed Objects for IEEE 802.3 Medium Attachment Units (MAUs)", RFC 2668, August 1999.
- [23] McCloghrie, K., and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [24] Flick, J., and J. Johnson, "Definition of Managed Objects for the Ethernet-like Interface Types", RFC 2358, June 1998.
- [25] J. Case, "FDDI Management Information Base", RFC 1512, September 1993.
- [26] McCloghrie, K., and E. Decker, "IEEE 802.5 MIB using SMIv2", RFC 1748, December 1994.
- [27] J. Flick, "Definitions of Managed Objects for IEEE 802.12 Interfaces", RFC 2020, October 1996.

[28] Willis, S., Burruss, J., and J. Chu, "Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIv2", RFC 1657, July 1994.

[29] Baker, F. (Editor), "Requirements for IP Version 4 Routers", [RFC 1812](#), June 1995.

[30] F. Baker, "IP Forwarding Table MIB", RFC 2096, January 1997. [31] D. Haskin and S. Onishi, "Management Information Base for IP Version 6", RFC 2465, December 1998.

[32] R. Srinivasan, "XDR: External Data Representation Standard", RFC 1832, August 1995.

8. Адреса авторов

Peter Phaal

InMon Corp.

580 California Street, 5th Floor

San Francisco, CA 94104

Phone: (415) 283-3263

E-Mail: peter.phaal@inmon.com

Marc Lavine

Foundry Networks

2100 Gold Street

P.O. Box 649100

San Jose, CA 95164-9100

Phone: (408) 586-1700

E-Mail: mlavine@foundrynet.com

Перевод на русский язык

Николай Малых

nmalykh@gmail.com

Приложение А. Различия sFlow версий 4 и 5

Архитектура sFlow не изменилась, но было внесено множество изменений в формат дейтаграмм и sFlow MIB. Эти изменения подробно описаны в истории изменений каждой из этих спецификаций.

Главным отличием sFlow MIB версии 5 является деление sFlowTable на три отдельных таблицы:

- sFlowRcvrTable для поддержки сессии с коллектором sFlow;
- sFlowFsTable для настройки выборки пакетов;
- sFlowCpTable для настройки опроса счетчиков интерфейсов.

Основные изменения в формате дейтаграмм sFlow связаны с расширяемостью. Записи в дейтаграммах теперь включают TLV¹. Определено глобальное пространство имен для типов записей, позволяющее разработчикам создавать свои типы записей. Это также позволяет добавлять новые типы записей без изменения опубликованного протокола sFlow.

Приложение В. Генерация случайных чисел

Важным свойством генератора случайных чисел является сходимости среднего значения генерируемых значений к требуемой частоте выборки.

Генераторы с однородным распределением очень эффективны. Диапазон числа пропусков (дисперсия) не оказывает существенного влияния на результат и даже небольших случайных возмущений достаточно для того, чтобы процесс выборки не синхронизировался с периодичностью в потоке пакетов.

Генератор случайных чисел должен гарантировать возможность выдачи всех значений между минимальным и максимальным. Генераторы, способные выдавать лишь четные числа или числа с каким-либо общим делителем, не подходят.

Новое значение пропуска требуется лишь при выполнении каждой выборки.

¹Tag, length and value - тег, размер и значение.