УТВЕРЖДЕН

ЛКНВ.466217.002 Д90-ЛУ

МЕЖСЕТЕВОЙ ЭКРАН ИВК КОЛЬЧУГА-К
(МЭ ИВК КОЛЬЧУГА-К)

Руководство администратора

ЛКНВ.466217.002 Д90

Листов 478

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2023

Литера О

АННОТАЦИЯ

В настоящем документе описывается информация по работе с изделием «Межсетевой экран ИВК КОЛЬЧУГА-К» ЛКНВ.466217.002, предназначенным для реализации контроля за информацией, поступающей в автоматизированную систему (АС) и (или) выходящей из АС, и обеспечивающий защиту АС посредством фильтрации информации, т. е. ее анализа по совокупности критериев и принятия решения о ее распространении в (из) АС.

Версия документа **1.4**.

В документе приведены сведения, необходимые для выполнения операций администрирования:

- режимы работы изделия;
- функций и интерфейсы функций изделия, доступные каждой роли пользователей;
- параметры (настроек) безопасности изделия, доступные каждой роли пользователей, и их безопасные значения;
- типы событий безопасности, связанные с доступными пользователю функциями изделия;
- действия после сбоев и ошибок эксплуатации изделия;
- действия по приемке поставленного изделия;
- действия по безопасной установке и настройке изделия;
- требования к среде функционирования изделия.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Обозначение и наименование

Полное наименование изделия – «Межсетевой экран ИВК КОЛЬЧУГА-К».

Сокращенное наименование изделия – МЭ ИВК КОЛЬЧУГА-К.

Обозначение изделия – ЛКНВ.466217.002.

1.2. Назначение и функции МЭ ИВК КОЛЬЧУГА-К

МЭ ИВК КОЛЬЧУГА-К предназначен для реализации контроля за информацией, поступающей в АС и (или) выходящей из АС, и обеспечении защиты АС посредством фильтрации информации, т. е. ее анализа по совокупности критериев и принятия решения о ее распространении в (из) АС.

МЭ ИВК КОЛЬЧУГА-К может применяться для обеспечения защиты распространения информации АС по каналам компьютерных сетей в условиях автономной работы в круглосуточном режиме эксплуатации.

Возможности, заложенные в МЭ ИВК КОЛЬЧУГА-К, могут служить основой для разработки политики безопасности АС в области организации защиты сетей предприятий, организаций и учреждений всех возможных типов, направлений и форм собственности, где должна быть обеспечена защита данных на конфиденциальном уровне.

Комплекс средств защиты (КСЗ) МЭ ИВК КОЛЬЧУГА-К осуществляет решение задач по выполнению функций защиты информации для межсетевых экранов (МЭ) в объеме требований профиля защиты МЭ типа «А» четвертого класса защищенности ИТ.МЭ.А4.ПЗ (далее – ИТ.МЭ.А4.ПЗ) и для систем обнаружения вторжений (СОВ) в объеме требований профиля защиты СОВ уровня сети четвертого класса защиты ИТ.СОВ.С4.ПЗ (далее – ИТ.СОВ.С4.ПЗ).

МЭ ИВК КОЛЬЧУГА-К включает следующую функциональность:

- обеспечивает мультизадачность процессов;
- обеспечивает многопользовательский режим эксплуатации;
- обеспечивает поддержку мультипроцессорных систем;

- обеспечивает сетевую обработку данных;
- обеспечивает выполнение функций защиты информации от несанкционированного доступа в объеме требований ИТ.МЭ.А4.ПЗ, ИТ.СОВ.С4.ПЗ;
- обладает устойчивостью работы.

Основное предназначение МЭ ИВК КОЛЬЧУГА-К фильтрация и обработка пакетов, проходящих через сеть, является стеком стандартных сетевых протоколов. МЭ ИВК КОЛЬЧУГА-К обеспечивает полное и гибкое управление пакетами, при соответствующей реализованной логике работы с ними в управляющем модуле iptables, а также сбор и анализ информации об информационных потоках, передаваемых в рамках контролируемого сегмента информационной системы, заданными методами с целью вынесения решения об обнаружении вторжения.

1.3. Требования к квалификации персонала

Сотрудники, выполняющие развертывание МЭ ИВК КОЛЬЧУГА-К, должны иметь навыки по обслуживанию вычислительной техники и иметь навыки настройки оборудования для работы в локальной сети.

Администратор МЭ ИВК КОЛЬЧУГА-К должен иметь:

- общее представление об установке правил фильтрации сетевого трафика,создании правил СОВ;
- знания и навыки по техническим и криптографическим аспектам обеспечения информационной безопасности;
- навыки настройки оборудования для работы в локальной сети.

К эксплуатации МЭ ИВК КОЛЬЧУГА-К допускается персонал, ознакомленный с данным руководством.

2. ОБЩИЕ ПОЛОЖЕНИЯ

2.1. Условия поставки изделия

Комплектность МЭ ИВК КОЛЬЧУГА-К, в зависимости от варианта комплектации, представлена в документе «Формуляр. ЛКНВ.466217.002 ФО», входящий в комплект поставки.

2.2. Внешний вид МЭ ИВК КОЛЬЧУГА-К

Внешний вид изделия МЭ ИВК КОЛЬЧУГА-К:

- комплектация К01/ К04: рис. 1, рис. 2;
- комплектация К02: рис. 3, рис. 4;
- комплектация К03: рис. 5, рис. 6.

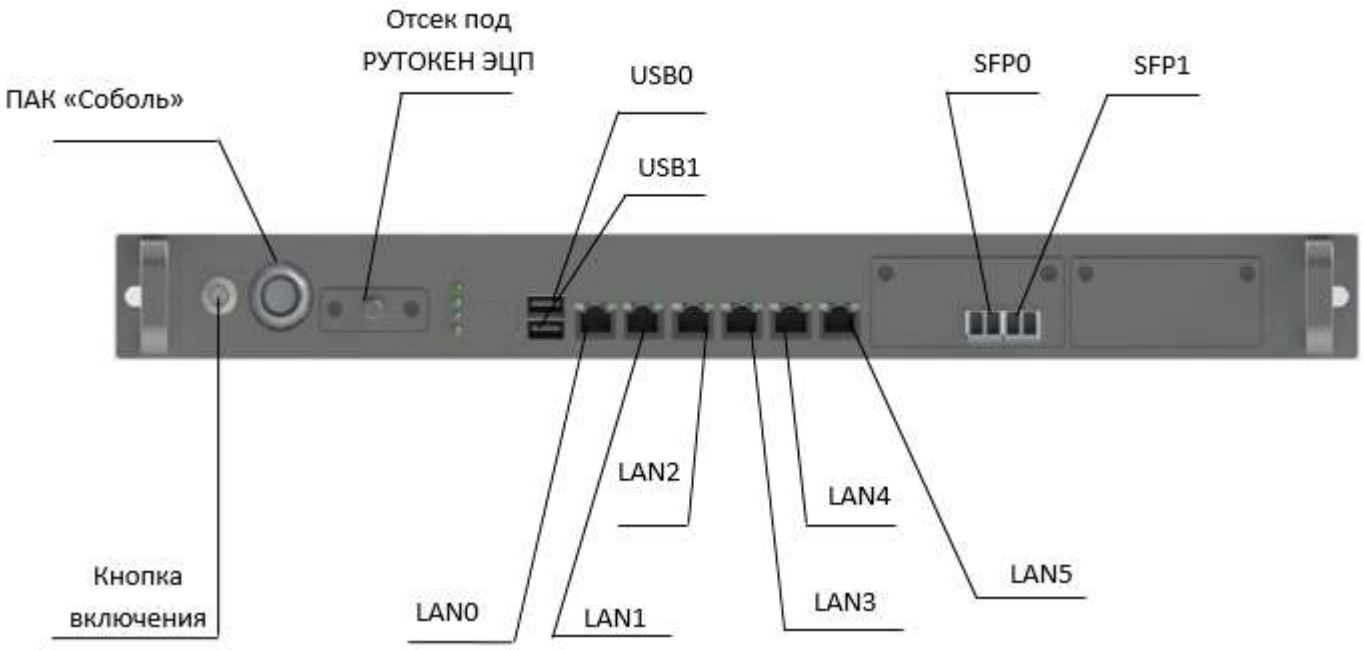


Рис. 1 – Передняя панель. Комплектация К01/К04

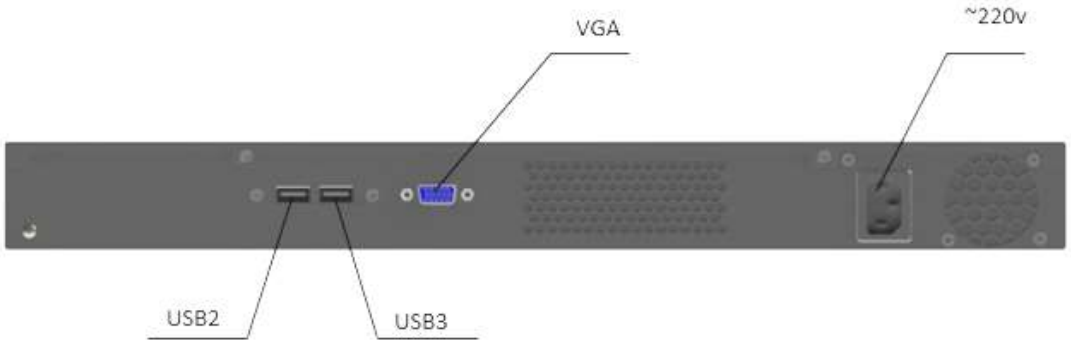


Рис. 2 – Задняя панель. Комплектация К01/К04

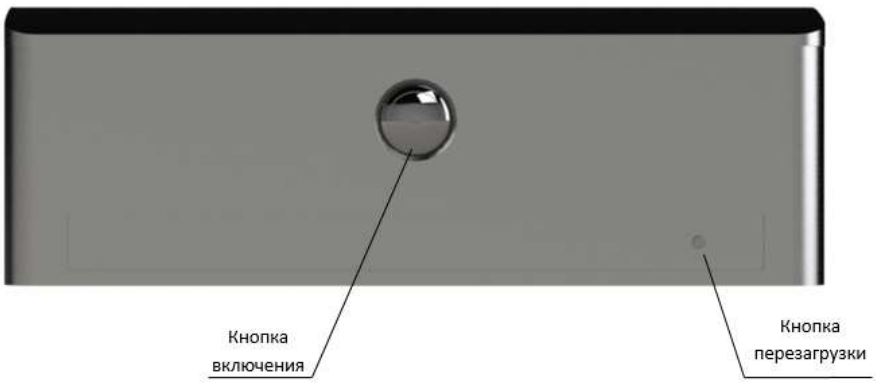


Рис. 3 – Передняя панель. Комплектация К02

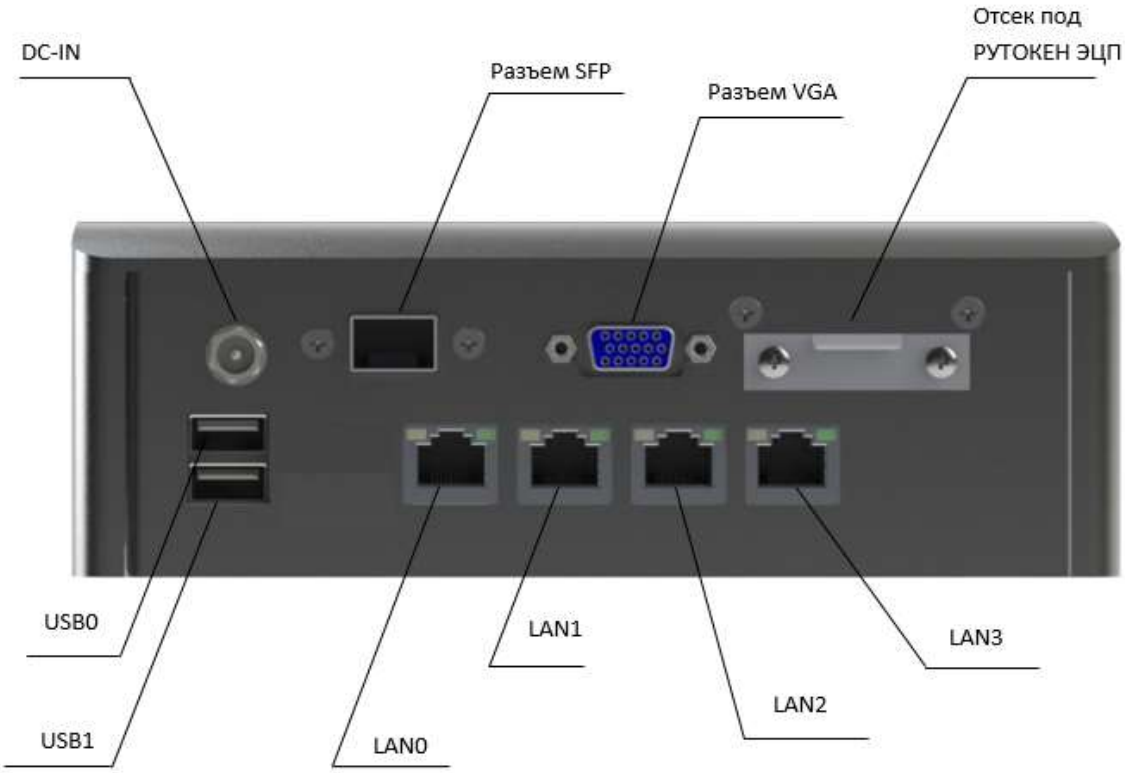


Рис. 4 – Задняя панель. Комплектация К02

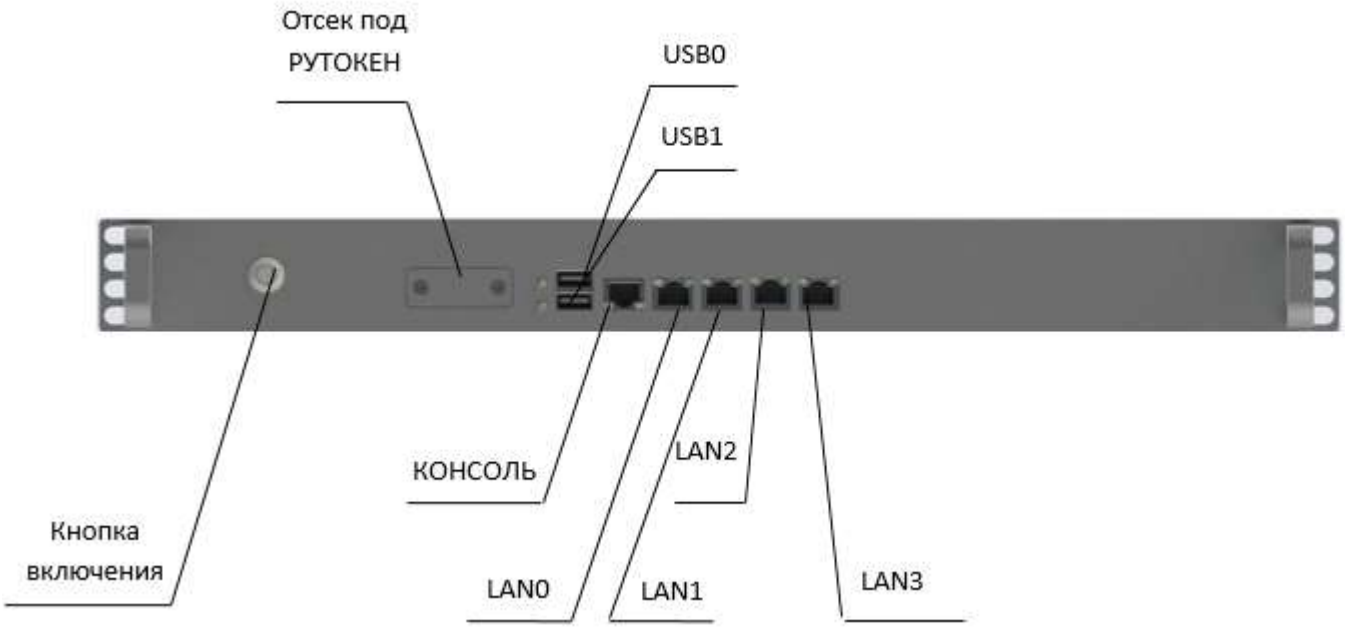


Рис. 5 – Передняя панель. Комплектация К03

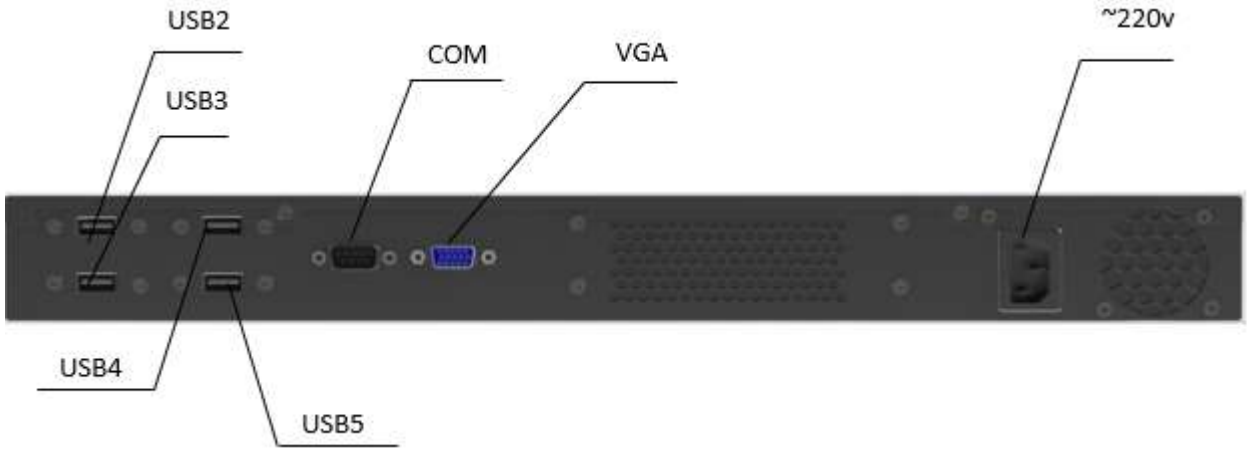


Рис. 6 – Задняя панель. Комплектация К03

Описание элементов изделия приведено в таблице 1.

Т а б л и ц а 1

Элемент	Описание
Кнопка включения	Кнопка включения изделия.
ПАК «Соболь»	Считыватель для идентификатора iButton ПАК «Соболь».
Отсек под РУТОКЕН ЭЦП	Подключение СКЗИ «Рутокен ЭЦП 2.0»/ «Рутокен ЭЦП 2.0 Flash»/ «Рутокен ЭЦП 3.0».
USB0 – USB5	Разъемы: USB2.0, USB3.0, DUSB.
LAN0 – LAN5	LAN0~5 – шесть 10/100/1000 Mbps RJ45 портов, LAN0 и LAN1 составляют первую группу bypass функции, LAN2 и LAN3 составляют вторую группу, LAN4 и LAN5 составляют третью группу. LED_индикаторы (рис. 7) по краям разъема RJ45 информируют о состоянии и скорости соединения, согласно таблице 2.
SFP0, SFP1 (количество опционально)	Оптический модуль SFP – специальное оптическое устройство для приема и передачи данных.
VGA	Разъем для подключения мониторов по стандарту видеоинтерфейса VGA.
COM	Разъем подключения периферийных устройств.
~220v/DC-IN	Разъем подключения силового кабеля к изделию или блока питания.

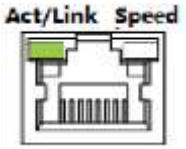


Рис. 7

Т а б л и ц а 2 – LED_индикаторы

Atc/Link индикатор	Состояние
Мигает	Передача данных
Зеленый	Подключено

Speed индикатор	Скорость LAN
Зеленый	1000 Mbps
Желтый	100 Mbps
Выкл	10 Mbps

2.3. Интерфейсы МЭ ИВК КОЛЬЧУГА-К

Предоставляются следующие внешние интерфейсы взаимодействия с МЭ ИВК КОЛЬЧУГА-К, приведенные в таблице 3.

Таблица 3

Интерфейсы функций безопасности	Краткое описание
ИФБО.1 Сетевой интерфейс	Внешний интерфейс, осуществляющий прием форматированных блоков информации, передаваемых по компьютерной сети, структура которых определена стеком стандартных сетевых протоколов.
ИФБО.2 Графический интерфейс	Внешний интерфейс, представляющий программные функции графическими элементами экрана, при помощи которых пользователь взаимодействует с различными программами и устройствами.
ИФБО.3 Локальный интерфейс управления	Внешний интерфейс, представляющий собой инструкции компьютеру, передающиеся путем ввода с клавиатуры текстовых строк (команд), при помощи которых пользователь взаимодействует с различными программами и устройствами.
ИФБО.4 Удаленный консольный интерфейс	Внешний интерфейс, представляющий собой инструкции компьютеру, передающиеся через удаленный доступ путем ввода с клавиатуры текстовых строк (команд), при помощи которых пользователь взаимодействует с различными программами и устройствами.
ИФБО.5 Интерфейс прикладного взаимодействия	Внешний интерфейс, предназначенный для передачи информации, необходимой для взаимодействия элементов межсетевого экрана и внешнего окружения: для реализации кластеризации, взаимодействия с внешней системой мониторинга, работы системы анализа пропускной способности, динамической маршрутизации.

3. СТРУКТУРА МЭ ИВК КОЛЬЧУГА-К

В составе МЭ ИВК КОЛЬЧУГА-К выделены следующие функциональные возможности безопасности:

- межсетевой экран – контроль и фильтрация;
- идентификация и аутентификация;
- регистрация событий безопасности (аудит);
- обеспечение бесперебойного функционирования и восстановление;
- тестирование и контроль целостности;
- преобразование сетевых адресов;
- маскирование;
- приоритизация информационных потоков;
- управление (администрирование);
- взаимодействие с другими средствами защиты информации;
- функции системы обнаружения вторжений.

3.1. Межсетевой экран

Сопоставление с функциональными требованиями безопасности (ФТБ): FDP_IFC.2, FDP_IFF.1, FDP_ITC.1, FDP_ETC.1, FMT_MTD.1, FMT_MTD.2, FMT_MTD.3, FMT_MTD_EXT.5.

МЭ ИВК КОЛЬЧУГА-К с помощью NETFLOW iptables осуществляет фильтрацию для отправителей информации, получателей информации, сетевого трафика и всех операций перемещения контролируемой МЭ ИВК КОЛЬЧУГА-К информации сетевого трафика к узлам информационной системы и от них, на которые распространяется политика управления информационными потоками, в том числе с учетом управляющих команд от взаимодействующих с МЭ ИВК КОЛЬЧУГА-К средств защиты информации других видов (систем обнаружения вторжений). Управляющие команды передаются на МЭ ИВК КОЛЬЧУГА-К в формате iptables при наличии установленного SSH-соединения.

Фильтры состоят из правил. Каждое правило – это строка, содержащая в себе критерии, определяющие, подпадает ли пакет под заданное правило, и действие, которое необходимо выполнить в случае удовлетворения критерия. Функции безопасности (ФБ) обеспечивают распространение фильтрации на все операции перемещения через МЭ ИВК КОЛЬЧУГА-К информации к узлам информационной системы и от них.

Когда пакет приходит на сетевое устройство (рис. 8), он обрабатывается соответствующим драйвером и далее передается в фильтр в ядре операционной системы (ОС). Далее пакет проходит ряд таблиц и затем передается либо локальному приложению, либо переправляется на другую машину. ФБ МЭ ИВК КОЛЬЧУГА-К осуществляют проверку пакета при его импорте из-за пределов МЭ ИВК КОЛЬЧУГА-К и экспорте за пределы МЭ ИВК КОЛЬЧУГА-К, тем самым, исключая прямое взаимодействие между узлами.

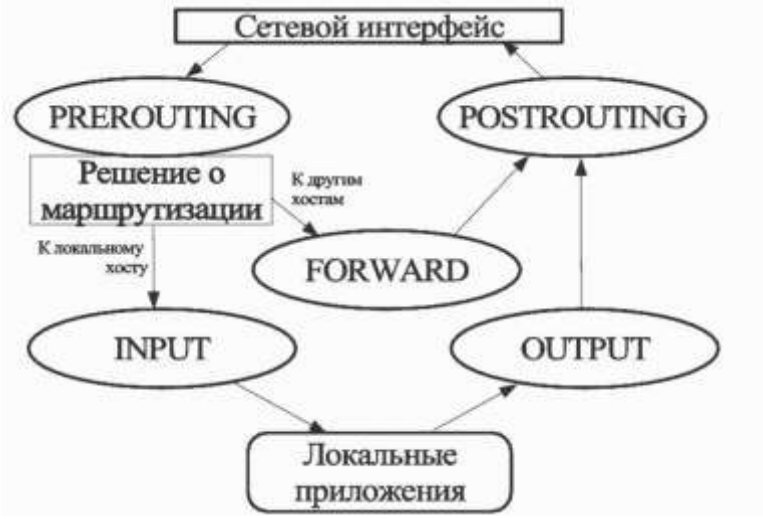


Рис. 8 – Схема движения пакетов в iptables

ФБ явно запрещают информационный поток при нарушении функционирования МЭ, в частности осуществляется автоматическое создание правила блокировки трафика в цепочке FORWARD при отключении сервисов аудита, удаленного управления, а также выгрузки модулей iptables: nf_conntrack_ipv4, xt_ndpi.

3.2. Идентификация и аутентификация

Сопоставление с ФТБ: FIA_UAU.2, FIA_UID.2, FIA_AFL.1, FIA_SOS.1.

ФБ МЭ ИВК КОЛЬЧУГА-К обеспечивают:

- идентификацию и аутентификацию администратора МЭ ИВК КОЛЬЧУГА-К при его запросах на доступ;
- возможность для аутентификации по паролю;
- невозможность доступа (выполнения любых действий) неидентифицированного субъекта или субъекта, подлинность идентификации которого при аутентификации не подтвердилась.

ФБ МЭ ИВК КОЛЬЧУГА-К осуществляют идентификацию каждого субъекта межсетевого взаимодействия до передачи МЭ ИВК КОЛЬЧУГА-К информационного потока получателю по его уникальному идентификатору субъекта.

3.3. Регистрация событий безопасности

Сопоставление с ФТБ: ФБО ФБ 3 удовлетворяет следующим ФТБ: FAU_ARP.1, FAU_GEN.1, FAU_GEN.2, FAU_SAR.1, FAU_SAR.2, FAU_SAR.3, FAU_SEL.1, FAU_STG.2, FAU_STG.4, FPT_STM.1.

Функция «Регистрация событий безопасности (аудит)» предоставляет МЭ ИВК КОЛЬЧУГА-К возможности создания, хранения и просмотра записей аудита. В соответствии с настройками и конфигурацией МЭ ИВК КОЛЬЧУГА-К вся активность администраторов отслеживается и соответствующие записи помещаются в файлы регистрации событий.

ФБ регистрируют в каждой записи аудита следующую информацию: дату и время события, тип события, идентификатор субъекта (если применимо) и результат события (успешный или неуспешный).

ФБ сигнализируют о попытках нарушения правил межсетевого экранирования при обнаружении критичных событий безопасности посредством вывода соответствующих уведомлений в графический интерфейс управления, а также

отправки электронного сообщения (SMTP) при блокировании правилом МЭ ИВК КОЛЬЧУГА-К.

Журналирование событий безопасности, связанных с правилами межсетевого экранирования, осуществляется посредством службы syslog.

Когда какая-то часть ядра генерирует часть сообщения аудита, эта часть будет немедленно послана в пользовательское пространство, и автоматически выставится флаг, указывающий, что этот системный вызов находится под аудитом. Таким образом, при выходе из системного вызова будет сформирована дополнительная информация (если включен аудит системных вызовов).

Настройка избирательного аудита осуществляется посредством конфигурирования утилит AUDITCTL и AUDITD, а также настройки `auditd.conf`.

Аудит в МЭ ИВК КОЛЬЧУГА-К производится по следующим правилам:

1) во время создания процесса, формируется контекст аудита и привязывается к структуре, описывающей процесс;
2) во время входа в системный вызов, заполняется следующая информация в контексте аудита, если он есть: номер системного вызова, дата и время, но не аргументы;
3) в ходе работы системного вызова перехватываются обращения к `getname()` и `path_lookup()`. Эти процедуры вызываются, когда ядро действительно собирается искать информацию, для принятия решения, будет ли системный вызов успешно выполнен или нет. Перехватывать вызовы нужно для того, чтобы не допустить копирование информации, которую генерирует `getname`, поскольку `getname` уже сделал приватную (для ядра) копию этой информации;
4) во время выхода из системного вызова генерируется та часть сообщения аудита, которая ответственна за информацию о системном вызове, включая имена файлов и номера `inode` (если доступны). Сообщение о системном вызове генерируется только если выставлен флаг, указывающий, что системный вызов находится под аудитом (он выставляется, например, часть ядра определяет, что должно

формироваться сообщение для аудита). Следует заметить, что полное сообщение аудита приходит в пользовательское пространство по частям, это позволяет не хранить сообщения неопределенный срок внутри ядра;

5) во время завершения процесса контекст аудита уничтожается;
6) во время шагов 1, 2 и 4 может быть выполнена простая фильтрация (например, для увеличения производительности – отключение аудита системных вызовов, выполняемых от имени пользователя, работающего с базой данных). Фильтрация может быть, как простой, так и сложной. Фильтрация реализована на столько полно на сколько возможно без существенного увеличения потребления ресурсов (например, `d_path()`).

ФБ предоставляют администратору МЭ ИВК КОЛЬЧУГА-К возможность читать как выборочно, так и всю информацию из записей аудита. Просмотр журнала аудита, в том числе выборочный, может осуществляться посредством утилит AUREPORT и AUSEARCH.

В МЭ ИВК КОЛЬЧУГА-К используется инструмент AUREPORT – это инструмент, который генерирует итоговые отчеты на основе логов службы аудита. AUREPORT может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов.

Программа AUSEARCH является инструментом поиска по журналу аудита. AUSEARCH может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов. Все условия, указанные в параметрах, объединяются логическим «И». К примеру, при указании `-m` и `-ui` в качестве параметров будут показаны события, соответствующие заданному типу и идентификатору пользователя.

Для информирования блокировки IP-адресов при превышении количества попыток совершения какого-либо действия (превышения ввода аутентификационных данных) используется программа fail2ban.

3.4. Обеспечение бесперебойного функционирования и восстановление

Сопоставление с ФТБ: FRU_FLT.2, FPT_RCV.1, FPT_FLS.1, FRU_RSA_EXT.3.

В случае сбоев МЭ ИВК КОЛЬЧУГА-К возможно восстановление работоспособности и настроек ФБ посредством реализованного изделии модуля резервного копирования и горячего резервирования.

ФБ МЭ ИВК КОЛЬЧУГА-К реализуют кластеризацию на основе модуля «горячего» резервирования (программа csync2), сохраняющего с основного на резервный хост настройки правил фильтрации, настройки DHCP, настройки DNS (перечень сохраняемых настроек определяется администратором). При отключении основного хоста адрес, указанный в «IP-адрес ресурса» будет поднят на резервном сервере.

ФБ МЭ ИВК КОЛЬЧУГА-К предоставляет возможность настройки автоматического ежедневного резервного копирования в заданное время либо ручной запуск процедуры копирования. Также можно указать конкретные каталоги для включения в процедуру резервирования и место, и период хранения резервных копий.

3.5. Тестирование и контроль целостности

Сопоставление с ФТБ: FPT_TST.1, FPT_AMT.1.

МЭ ИВК КОЛЬЧУГА-К выполняет пакет программ самотестирования при запуске для демонстрации правильного выполнения ФБ (настроек правил фильтрации и корректности запуска служб МЭ ИВК КОЛЬЧУГА-К).

ФБ предоставляют администраторам МЭ ИВК КОЛЬЧУГА-К возможность верифицировать целостность данных ФБ и программного кода ФБ. Контроль целостности в МЭ ИВК КОЛЬЧУГА-К осуществляется посредством программного комплекса Osec.

Защита ФБ обеспечивает целостность и управление механизмами, обеспечивающими ФБ. Пользователи должны быть аутентифицированы до

проведения каких-либо административных операций, которые могут быть проведены в МЭ ИВК КОЛЬЧУГА-К.

3.6. Преобразование сетевых адресов

Сопоставление с ФТБ: FDP_ITC.1, FDP_ETC.1, FDP_ETC_EXT.3.

В МЭ ИВК КОЛЬЧУГА-К используются следующие механизмы преобразования сетевых адресов как для импортируемых данных пользователя (входящего трафика), так и для экспортируемых данных (исходящего трафика):

- SNAT (Source Network Address Translation) для преобразования сетевых адресов, т. е. изменение исходящего IP-адреса в IP-заголовке пакета (см. п. 7.3.7);
- DNAT (Destination Network Address Translation) используется для преобразования адреса места назначения в IP-заголовке пакета (см. п. 7.3.8);
- MASQUERADE (см. п. 7.3.9) – в основе своей представляет то же самое, что и SNAT только не имеет ключа `--to-source`, так как маскировка может работать, например, с dialup подключением или DHCP, т. е. в тех случаях, когда IP-адрес присваивается устройству динамически. Если используется динамическое подключение, то нужно использовать маскировку, если же используется статическое IP-подключение, то лучшим выходом будет использование действия SNAT.

3.7. Маскирование

Данная группа элементов отвечает за подмену сетевого адреса на маскирующий.

Сопоставление с ФТБ: FDP_ETC_EXT.3.

Конфиденциальность данных функциональных возможностей МЭ ИВК КОЛЬЧУГА-К при передаче информации от МЭ обеспечивает маскирование наличия МЭ методом сохранения TTL проходящего IP-пакета, то есть сохранение числа итераций периода жизни IP-пакета.

3.8. Приоритизация информационных потоков

Сопоставление с ФТБ: FRU_PRS_EXT.3.

ФБ МЭ ИВК КОЛЬЧУГА-К осуществляют приоритизацию информационных потоков на основе установленных приоритетов значений атрибутов информационных потоков по субъекту и типу передаваемых данных. Реализация осуществляется модулем TC (Traffic Control). Данный модуль управления трафиком позволяет делать следующее:

- shaping – шейпинг – ограничение трафика, задержка пакетов с целью создания желаемой скорости передачи. Может использоваться не только для «сужения» исходящего канала, но и для сглаживания бросков во время пиковых нагрузок;
- scheduling – планирование – упорядочивание типов трафика в канале. Позволяет избегать задержек для критичных типов трафика (QoS);
- policing – политика входящего трафика. Позволяет ограничить входящий трафик путем уничтожения превысивших лимит пакетов.

3.9. Управление (администрирование)

Сопоставление с ФТБ: FMT_SMF.1, FMT_MTD.1, FMT_MTD.2, FMT_MTD.3, FMT_MTD_EXT.5, FMT_MOF.1, FMT_SMR.1, FMT_MSA.1, FMT_MSA.3.

Для управления МЭ ИВК КОЛЬЧУГА-К администраторы используют интерфейс управления (консольный и графический см. п. 2.3).

Все попытки администратора МЭ ИВК КОЛЬЧУГА-К на осуществление запроса, изменение или удаление атрибутов доступа (таких как имя пользователя, пароль, роль), данных функций безопасности (таких как данные журналов регистрации событий, данные сетевых сервисов, данные ФБ, конфигурационные настройки), а также непосредственно самих функций безопасности) осуществляются при непосредственном участии МЭ ИВК КОЛЬЧУГА-К.

ФБ МЭ ИВК КОЛЬЧУГА-К обеспечивают возможность централизованного управления компонентами МЭ ИВК КОЛЬЧУГА-К, в том числе, конфигурирования фильтров, проверки корректного ввода правил iptables, анализа регистрационной информации.

ФБ ведут таблицу состояний соединения. Для просмотра состояния соединений используется утилита ss из набора утилит iproute2.

Таблица включает:

- сетевой адрес (IP-адрес) источника;
- сетевой адрес (IP-адрес) получателя;
- номера портов;
- информацию состояния соединения.

В качестве основных состояний для сетевого трафика используются следующие:

- установление соединения;
- использование соединения;
- завершение соединения.

В состав iptables включен модуль, позволяющий администраторам наблюдать и ограничивать подключения к службам, работающим во внутренней сети, с помощью так называемого отслеживания соединений.

Подсистема отслеживания соединений запоминает соединения в таблице, благодаря чему администраторы могут разрешать или запрещать доступ, исходя из состояний соединения см. п. 7.9.1.

ФБ МЭ ИВК КОЛЬЧУГА-К позволяет использовать функциональность сохранения состояния, предлагаемую средством отслеживания соединений iptables с любым сетевым протоколом, даже если сам протокол состояние не поддерживает (как, например, UDP).

Состояния текущих соединений conntrack хранит в ядре. Их можно просмотреть в файле `/proc/net/nf_conntrack` (или `/proc/net/ip_conntrack`)

ФБ предоставляют администраторам МЭ ИВК КОЛЬЧУГА-К возможность осуществления настройки (модификации, удаления разрешительных

и (или) запретительных атрибутов безопасности) фильтрации (критерий `iptables -string` см. таблицу 15) для используемых пользователями отдельных команд, использования сетевых ресурсов, содержащих отдельные типы мобильного кода (Java, JavaScript, ActiveX, PDF, Postscript, Flash-анимация, VBScript), а также для прикладного программного обеспечения (ПО) (приложений) по значению поля «User-Agent».

Для пересылки HTTP-запросов во внешнюю сеть в ФБ МЭ ИВК КОЛЬЧУГА-К реализован прокси-сервер. Поступление запроса ожидается на определенном порту, который по умолчанию имеет стандартный номер 3128. Перед тем как выполнить перенаправление запроса, прокси-сервер проверяет принадлежность сетевого адреса хоста, с которого запрос был отправлен к группе внутренних сетевых адресов. Для того, чтобы запросы, отправленные из локальной сети, обрабатывались прокси-сервером, необходимо добавить соответствующую группу адресов (адрес подсети и адресную маску) в список внутренних сетей.

Прокси-сервер может работать в двух режимах: обычном и прозрачном. Обычный режим использования прокси-сервера требует изменения режима работы программ локальной сети, что может потребовать их ручной настройки. В прозрачном режиме все обращения из внутренней сети по зарегистрированным протоколам (портам) во внешнюю сеть автоматически перехватываются прокси-сервером при прохождении через шлюз. Программы в локальной сети при этом продолжают работать в обычном режиме, не требуя никакой специальной настройки. Недостатком прозрачного режима работы является невозможность идентификации пользователей – все запросы отправляются из локальной сети анонимно. Преимуществом непрозрачного режима работы, требующего перенастройки программ локальной сети, является возможность производить аутентификацию пользователей и контролировать их доступ во внешнюю сеть.

Политика доступа пользователей во внешнюю сеть формируется на основе групп пользователей и сетевых доменов. Для каждой группы пользователей может быть сформирован список доменов, к которым разрешается (или наоборот, запрещается) обращение.

3.10. Взаимодействие с другими средствами защиты информации

Сопоставление с ФТБ: FPT_TDC.1, FDP_IFC.2, FDP_IFF.1, FMT_MSA_EXT.6, FPT_RVM.1, FPT_SEP.

Данная группа элементов отвечает за согласованную интерпретацию функциями безопасности управляющих команд, полученных от взаимодействующих с МЭ ИВК КОЛЬЧУГА-К средств защиты обнаружения вторжений, например, «Система обнаружения вторжений ИВК СЕНСОР» ЛКНВ.15405-01 (СОВ ИВК СЕНСОР), осуществляется за счет использования универсальных команд формата iptables.

3.11. Функции СОВ

Сопоставление с ФТБ: FID_ANL_EXT.1, FID_COL_EXT.1, FID_CON_EXT.1, FID_INF_EXT.1, FID_MTH_EXT.1, FID_MTH_EXT.2, FID_PCL_EXT.1, FID_RCT_EXT.1, FID_UPD_EXT.1.

МЭ ИВК КОЛЬЧУГА-К реализует следующие основные функции СОВ:

- централизованное управление;
- контроль работы СОВ в режиме реального времени;
- регистрация событий управления и работы СОВ;
- создание правил СОВ для контроля трафика.

Основа СОВ МЭ ИВК КОЛЬЧУГА-К программный комплекс Suricata, который выполняет анализ сетевого трафика с использованием сигнатурного и эвристического методов обнаружения вторжений.

СОВ пропускает весь трафик через базу сигнатур, с которой сравниваются пакеты данных. Если содержимое пакета совпадает с сигнатурой, происходит назначенное правилом одно из действий для указанной сигнатуры, например:

- alert (оповещать);
- drop (блокировать);
- pass (пропустить).

Запись о событии, действии и другие данные о трафике заносятся в журнал, например: адрес и порт источника пакета (IPv4/IPv6), адрес и порт назначения (IPv4/IPv6) пакета, время срабатывания, причина срабатывания.

4. НАЧАЛО ИСПОЛЬЗОВАНИЯ

4.1. Подготовка к использованию МЭ ИВК КОЛЬЧУГА-К

4.1.1. Общий порядок подготовки к подключению

После транспортирования изделия в условиях отрицательных температур изделие следует извлечь из упаковки и выдержать в течение суток при нормальных климатических условиях: температуре плюс 25±10 °С, влажности 65±15 %, атмосферном давлении 750±30 мм рт. ст.

Внешний осмотр изделия:

1) проверьте соответствие комплектности МЭ ИВК КОЛЬЧУГА-К, указанной в документах «Формуляр. ЛКНВ.466217.002 ФО», «Паспорт. ЛКНВ.466217.002 ПС»;
2) проверьте внешний вид изделия на отсутствие видимых механических повреждений; чистоту гнезд, разъемов; четкость маркировок.

4.1.2. Подключение МЭ ИВК КОЛЬЧУГА-К

Подключите внешние дополнительные устройства для локального управления МЭ ИВК КОЛЬЧУГА-К: клавиатура, видеомонитор (при необходимости и их наличии).

Подключите сетевой кабель RJ45 к сетевому интерфейсу LAN0 (см. внешний вид п. 2.2).

Подсоедините кабели электропитания или адаптер питания МЭ ИВК КОЛЬЧУГА-К к сети электропитания.

Включите питание МЭ ИВК КОЛЬЧУГА-К (см. п. 2.2).

Для визуального контроля подключения МЭ ИВК КОЛЬЧУГА-К к сети предусмотрены световые индикаторы на корпусе или кнопке питания (см. п. 2.2).

4.2. Описание старта и процедура проверки правильности старта

После включения питания загрузка МЭ ИВК КОЛЬЧУГА-К осуществляется в следующем порядке:

1) загрузка базовой системы ввода-вывода (БСВВ), осуществляется автоматически;
2) предъявите выданный персональный идентификатор для АМПДЗ ПАК «Соболь» (опционально), загрузка далее осуществляется автоматически;
3) загрузка ПО МЭ ИВК КОЛЬЧУГА-К, осуществляется автоматически.

После подключения МЭ ИВК КОЛЬЧУГА-К имеет сетевой интерфейс с IP-адресом **192.168.0.254**.

Доступ к МЭ ИВК КОЛЬЧУГА-К осуществляется с учетом идентификационных данных через графический (п. 4.2.1) или один из консольных интерфейсов (п. 4.2.2).

4.2.1. Запуск графического интерфейса МЭ ИВК КОЛЬЧУГА-К

Для доступа к графическому интерфейсу (ГИ) веб-интерфейсу МЭ ИВК КОЛЬЧУГА-К (рис. 9) из локальной сети с любого компьютера:

1) запустить операционную систему;
2) открыть веб-браузер;
3) подключиться по адресу: **https://192.168.0.254**;
4) ввести идентификационные данные пользователя:
   - логин: **admin**
   - пароль по умолчанию: **chainmail1234**
5) нажать кнопку «Вход».

Примечания:

1. При необходимости проверить доступность и работоспособность выполните в консоли команду:

```
ping 192.168.0.254
```

2. Для доступа рекомендуется использовать веб-браузеры:
   - Mozilla Firefox с 52 версии;
   - Google Chrome с 55 версии;
   - Microsoft Edge с 15 версии;
   - Safari с 10 версии.

При первом подключении к ГИ МЭ ИВК КОЛЬЧУГА-К смените пароль по умолчанию в разделе «Пользователи» → «Локальные учетные записи» (см. п. 6.2).

Описание стартовой страницы ГИ МЭ ИВК КОЛЬЧУГА-К приведено в (п. 5.1).

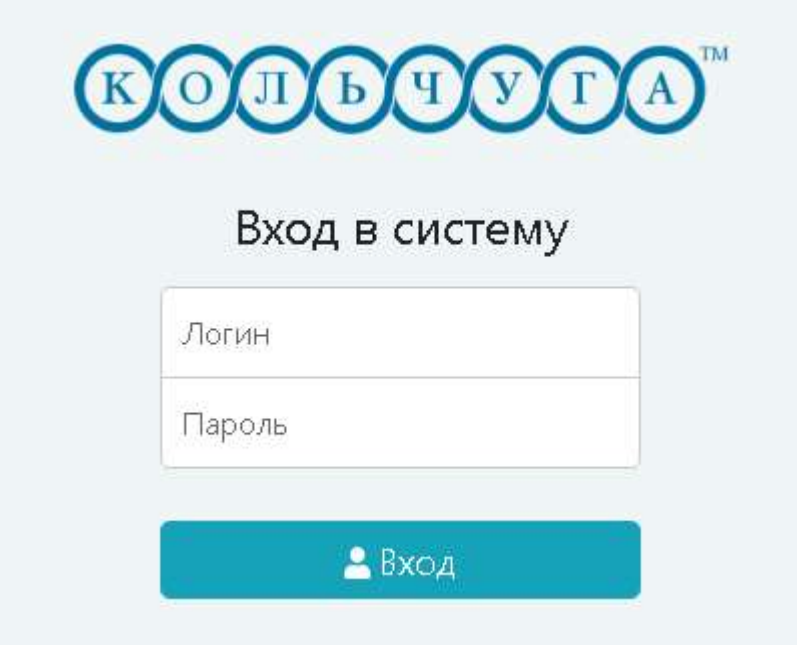


Рис. 9

4.2.2. Запуск консольного интерфейса

Для подключения к МЭ ИВК КОЛЬЧУГА-К через локальный интерфейс управления к изделию необходимо подключить монитор через порт VGA (см. рис. 2) и клавиатуру.

Для подключения к МЭ ИВК КОЛЬЧУГА-К через удаленный консольный интерфейс на автоматизированном рабочем месте (АРМ) откройте терминал и введите команду:

```
$ ssh admin@192.168.0.254
```

Примечание. Если пытаетесь подключиться удаленно к МЭ ИВК КОЛЬЧУГА-К первый раз, то утилита SSH также попросит подтвердить добавление нового устройства в свой список известных устройств; нужно набрать `yes` и нажать клавишу «Enter».

Далее при запросе ввести пароль пользователя **admin** (см. п. 4.2) и нажать клавишу «Enter».

Далее ввести команду:

```
$ sudo –s
```

Ввести пароль пользователя **admin**.

4.3. Верификация

Проверка поставленного потребителю изделия МЭ ИВК КОЛЬЧУГА-К производится путем подсчета контрольной суммы исполняемых файлов в соответствии с п. 9.6.1 через ГИ МЭ ИВК КОЛЬЧУГА-К (см. п. 4.2.1).

4.4. Информация о требованиях безопасности для среды функционирования

Для среды функционирования МЭ ИВК КОЛЬЧУГА-К предъявляются требования безопасности, приведенные таблице 4.

Т а б л и ц а 4

Требование	Описание
Обеспечение доверенного канала	Должен обеспечиваться доверенный канал передачи данных между защищаемой информационной системой и МЭ, а также между МЭ и терминалом, с которого выполняется управление им.
Обеспечение доверенного маршрута	Должен быть обеспечен доверенный маршрут между МЭ и его администраторами.
Обеспечение условий безопасного функционирования	Должно обеспечиваться исключение каналов связи защищаемой информационной системы с иными информационными системами в обход МЭ.
Физическая защита МЭ ИВК КОЛЬЧУГА-К	Должна обеспечиваться физическая защита МЭ и терминалов, с которых выполняется его управление.
Взаимодействие с доверенными продуктами информационных технологий	Должно обеспечиваться взаимодействие МЭ с сертифицированными на соответствие требованиям безопасности информации по соответствующему классу защиты средствами защиты информации (системами обнаружения вторжений, средствами антивирусной защиты и другими), от которых МЭ получает управляющие сигналы.
Соблюдение правил эксплуатации МЭ ИВК КОЛЬЧУГА-К	Должны быть обеспечены установка, конфигурирование и управление МЭ в соответствии с ЭД.

Окончание таблицы 4

Требование	Описание
Требование к персоналу	Персонал, ответственный за функционирование МЭ, должен обеспечивать функционирование изделия, руководствуясь ЭД.
Поддержка аудита	Должна быть обеспечена поддержка средств аудита, используемых в МЭ.
Исключение несертифицированных компонентов	Должна быть исключена возможность использования не прошедших сертификацию компонентов программно-технического средства, в котором интегрирован МЭ с иными видами средств защиты информации, при его эксплуатации.

Действия по реализации указанных функций безопасности среды функционирования средства:

- удаленное администрирование должно осуществляться по доверенным каналам передачи данных посредством применения средств криптографической защиты или организационно-технических мер, защищающих от перехвата и модификации передаваемых данных;
- должна обеспечиваться физическая защита МЭ и терминалов, с которых выполняется его управление;
- допускается обеспечение взаимодействия МЭ с сертифицированными на соответствие требованиям безопасности информации по соответствующему классу защиты средствами защиты информации (системами обнаружения вторжений), от которых МЭ получает управляющие сигналы;
- ответственные за функционирование МЭ ИВК КОЛЬЧУГА-К, должны обеспечивать установку, конфигурирование и управление изделием (в том числе мониторинг рабопоспособности, контроль журналов аудита) в соответствии с ЭД на МЭ ИВК КОЛЬЧУГА-К, а также документацией комплектующих изделий;
- установка других аппаратных компонентов и программных средств на МЭ ИВК КОЛЬЧУГА-К, в целях обеспечения безопасности информации АС, устойчивости работы и соблюдения нормальных условий функционирования

МЭ ИВК КОЛЬЧУГА-К запрещается, за исключением обновлений и дополнений, определяемых поставкой и прошедших сертификацию.

4.5. Описание механизмов устранения идентифицированных скрытых каналов

Подробнее об параметрах, критериях и опциях, используемых в правилах iptables см. в разделе 7.

1) Для предотвращения Timestamp Evaluation – отключить отметки времени TCP в МЭ ИВК КОЛЬЧУГА-К. Для этого выполнить следующие команды:

```
# echo 0 > /proc/sys/net/ipv4/tcp_timestamps
To make that change permenant though, you need to add the following line to /etc/sysctl.conf:
net.ipv4.tcp_timestamps = 0
```

также можно настроить правила iptables:

```
iptables -A INPUT -p icmp --icmp-type timestamp-request -j DROP
iptables -A OUTPUT -p icmp --icmp-type timestamp-reply -j DROP
```

2) Для предотвращения ISN Evaluation (оценка временной отметки) – использовать TCP/IP прокси (socks).

3) Для предотвращения TCP URG Pointer (указателя TCP URG) – настроить правила iptables:

```
iptables -N BADFLAGS
iptables -A BADFLAGS -j LOG --log-prefix "BADFLAGS: "
iptables -A BADFLAGS -j DROP
iptables -N TCP_FLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ACK,FIN FIN                  -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ACK,PSH PSH                  -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ACK,URG URG                  -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags FIN,RST FIN,RST              -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags SYN,FIN SYN,FIN              -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags SYN,RST SYN,RST              -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ALL     ALL                  -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ALL     NONE                 -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ALL     FIN,PSH,URG          -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ALL     SYN,FIN,PSH,URG      -j BADFLAGS
iptables -A TCP_FLAGS -p tcp --tcp-flags ALL     SYN,RST,ACK,FIN,URG  -j BADFLAGS
```

4) Для предотвращения IP ToS Evaluation (Оценки IP-ToS) – настроить способ обслуживания для telnet, ftp-control и ftp-data – выполнить команды:

```
# iptables -A PREROUTING -t mangle -p tcp --sport telnet \
  -j TOS --set-tos Minimize-Delay
# iptables -A PREROUTING -t mangle -p tcp --sport ftp \
  -j TOS --set-tos Minimize-Delay
# iptables -A PREROUTING -t mangle -p tcp --sport ftp-data \
  -j TOS --set-tos Maximize-Throughput
```

Эти правила прописываются на удаленном хосте и воздействуют на входящие, по отношению к компьютеру, пакеты. Для пакетов, отправляемых в обратном направлении, эти флаги устанавливаются автоматически. Настроить их можно, прописав следующие правила:

```
# iptables -A OUTPUT -t mangle -p tcp --dport telnet \
  -j TOS --set-tos Minimize-Delay
# iptables -A OUTPUT -t mangle -p tcp --dport ftp \
  -j TOS --set-tos Minimize-Delay
# iptables -A OUTPUT -t mangle -p tcp --dport ftp-data \
  -j TOS --set-tos Maximize-Throughput
```

Для противодействия данной атаке в командной строке выполнить следующие команды:

```
# Разрешить главные типы протокола ICMP (см. п. 7.9.3)
iptables -A OUTPUT -p icmp --icmp-type 0 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 3 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 4 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 11 -j ACCEPT
iptables -A OUTPUT -p icmp --icmp-type 12 -j ACCEPT
```

5) Для предовращения Initial Sequence Number hijacking and spoofing (урона и подделки исходного кода последовательности) – настроить правила iptables:

```
  # Защита от spoofing
  iptables -I INPUT -m conntrack --ctstate NEW,INVALID -p tcp --tcp-flags
SYN,ACK SYN,ACK -j REJECT --reject-with tcp-reset

  # Защита от SYN-флуда
  iptables -A INPUT -p tcp --syn -m limit --limit 10/s --limit-burst 50 -j
ACCEPT
  iptables -A INPUT -p udp -m limit --limit 10/s --limit-burst 50 -j ACCEPT
  iptables -A INPUT -p icmp -m limit --limit 10/s --limit-burst 50 -j ACCEPT
  iptables -A INPUT -j DROP

  # Отбрасывать ошибочные пакеты
  iptables -A INPUT -m state --state INVALID -j DROP
  iptables -I INPUT -m conntrack --ctstate INVALID -j DROP

  # Отбрасывать фрагментированные пакеты
  iptables -A INPUT -f -j DROP

  # Защита от попытки открыть входящее соединение TCP не через SYN
  iptables -I INPUT -m conntrack --ctstate NEW -p tcp ! --syn -j DROP

  # Защита от Ping of death
  iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit 10/s --
limit-burst 50 -j ACCEPT
  iptables -A INPUT -p icmp --icmp-type echo-request -j DROP

  # Защита от некорректных ICMP
  iptables -I INPUT -p icmp -f -j DROP

  # Отбросить ошибочные пакеты
  iptables -A FORWARD -m state --state INVALID -j DROP
  iptables -I FORWARD -m conntrack --ctstate INVALID -j DROP

  # Отбросить фрагментированные пакеты
  iptables -A FORWARD -f -j DROP
```

```
# Сбрасывать фрагментированные пакеты
iptables -A OUTPUT -f -j DROP
```

Дополнительно от администратора внести правки в файл `/etc/sysctl.conf`:

```
# vi /etc/systcl.conf
```

```
# Отбросить ICMP-redirect (против атак типа MITM)
net.ipv4.conf.all.accept_redirects=0
net.ipv6.conf.all.accept_redirects=0
```

```
# Включить механизм TCP syncookies
net.ipv4.tcp_syncookies=1
```

```
# Различные улучшения (защита от spoofing attack
# увеличение очереди «полуоткрытых» TCP-соединений и далее):
net.ipv4.tcp_timestamps=0
net.ipv4.conf.all.rp_filter=1
net.ipv4.tcp_max_syn_backlog=1280
kernel.core_uses_pid=1
```

4.6. Об изделии

После прохождения авторизации администратор может просмотреть сведения об изделии – нажать левой кнопкой мыши на имени пользователя, зарегистрировавшегося в ГИ на панели пользователя (рис. 10) и выбрать в выпадающем элементе пункт «О межсетевом экране».

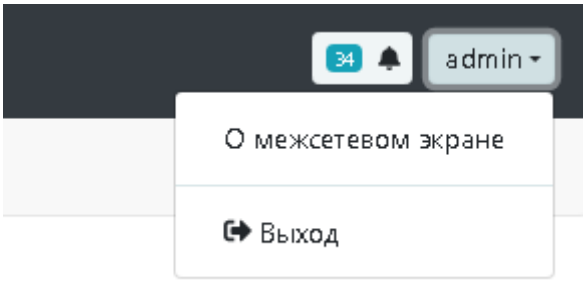


Рис. 10 – Панель пользователя МЭ ИВК КОЛЬЧУГА-К

На вкладке содержится следующая информация (рис. 11):

- наименования изделия, его версия, обозначение;
- контакты разработчика изделия;

- установленные обновления, с указанием даты установки и контрольной суммы (КС);
- интегральная КС исполняемых файлов изделия;
- возможность верефикации и проверки текущей КС изделия (см. п. 9.6.1).

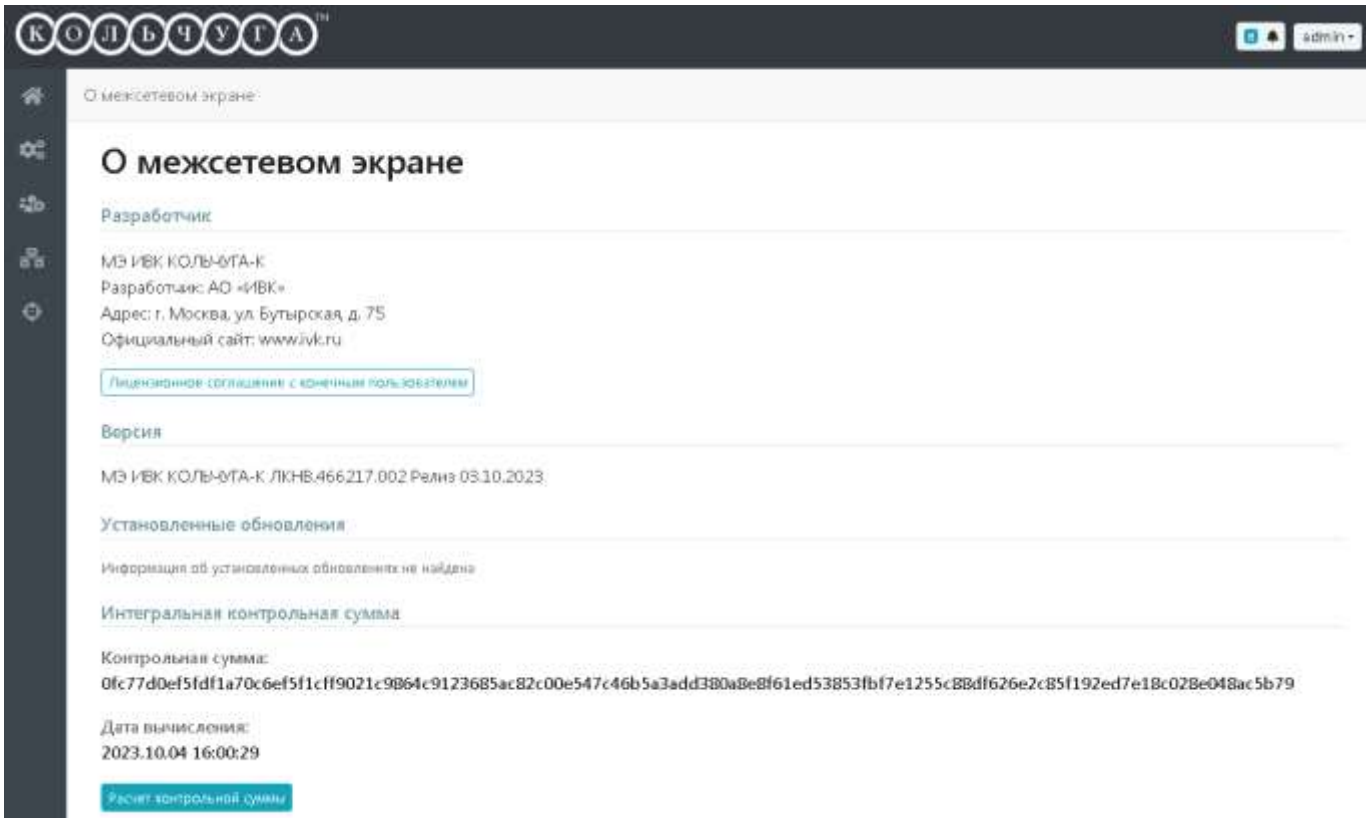


Рис. 11 – Пример вкладки «О межсетевом экране»

4.7. Роли МЭ ИВК КОЛЬЧУГА-К

Для работы с МЭ ИВК КОЛЬЧУГА-К выделены следующие роли пользователей:

- администратор – имеет доступ ко всем функциональным возможностям в графическом и консольном интерфейсах МЭ ИВК КОЛЬЧУГА-К;
- аудитор – имеет доступ к просмотру только графического интерфейса изделия, без возможности внесения изменений в конфигурацию МЭ ИВК КОЛЬЧУГА-К.

В МЭ ИВК КОЛЬЧУГА-К пользователь с именем **admin** и ролью администратор, пользователь с именем **auditor** и ролью аудитор существуют по умолчанию.

Пользователь аудитор (auditor) по умолчанию не активен (выключен) и пароль по умолчанию не установлен, устанавливается администратором (см. п. 6.2).

Добавление новых пользователей в МЭ ИВК КОЛЬЧУГА-К не предусмотрено.

4.8. Завершение работы

Перед выключением питания МЭ ИВК КОЛЬЧУГА-К завершить работу ПО изделия.

Для выхода из ГИ МЭ ИВК КОЛЬЧУГА-К нужно нажать левой кнопкой мыши на имени зарегистрировавшегося лица в панели пользователя (см. рис. 10) и выбрать в выпадающем элементе пункт «Выход».

Выключение или перезагрузка МЭ ИВК КОЛЬЧУГА-К осуществляется в соответствии с п. 9.5.

Для перезагрузки также можно воспользоваться сочетанием клавиш «Ctrl» + «Alt» + «Del».

5. ОПИСАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА МЭ ИВК КОЛЬЧУГА-К

5.1. Главная страница графического интерфейса МЭ ИВК КОЛЬЧУГА-К

ГИ МЭ ИВК КОЛЬЧУГА-К представляет набор инструментов администрирования для реализации функций межсетевого экранирования потоков информации в АС, сбора и аналитической обработки собранных СОВ данных.

ГИ МЭ ИВК КОЛЬЧУГА-К включает следующие функциональные блоки (рис. 12):

1) меню – панель разделов меню отображается на всех страницах интерфейса. Подробнее о разделах приведено в п. 5.3;
2) интерактивная панель мониторинга – для анализа информации о состоянии системы (см. раздел 8);
3) панель элементов мониторинга (см. пп. 8.2 – 8.10).
4) рабочая область – рабочая область межсетевого экрана (см. раздел 7), системы обнаружения вторжений (см. раздел 11), настроек системы;
5) панель пользователя (см. рис. 10).

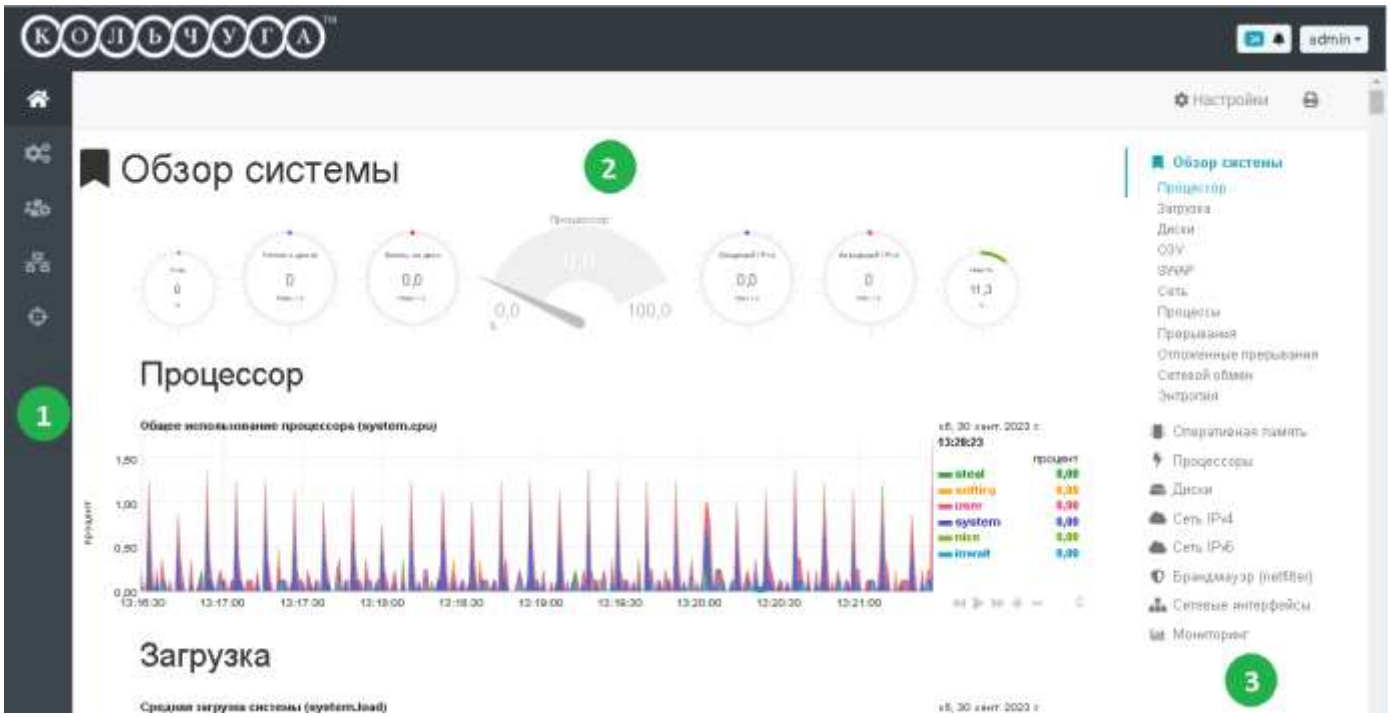


Рис. 12 – Главная страница веб-интерфейса МЭ ИВК КОЛЬЧУГА-К

5.2. Пользовательская информация

В правом верхнем углу отображается имя зарегистрировавшегося в ГИ пользователя и панель уведомлений (рис. 13).

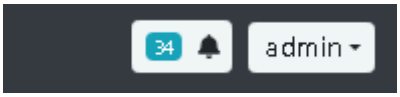


Рис. 13 – Панель пользователя

При появлении новых уведомлений счетчик в панели увеличится.

5.3. Меню

Основное меню МЭ ИВК КОЛЬЧУГА-К расположено слева (см. «1») на рис. 12).

Разделы меню (рис. 14):

-  «Главная» – переход на главную страницу ГИ МЭ ИВК КОЛЬЧУГА-К (п. 5.1, п. 7.9);
-  «Система» – раздел управления системными сервисами и служебными программами, а также просмотра системных журналов (раздел 9);
-  «Пользователи» – раздел управления учетными записями и правами доступа пользователей (раздел 6). Раздел доступен только пользователю с ролью администратор;
-  «Сеть» – раздел конфигурирования сетей и сетевых служб (раздел 10), работа с МЭ (раздел 7);
-  «СОВ» – раздел сбора и анализа информации о сетевом трафике, установки правил СОВ (раздел 11).

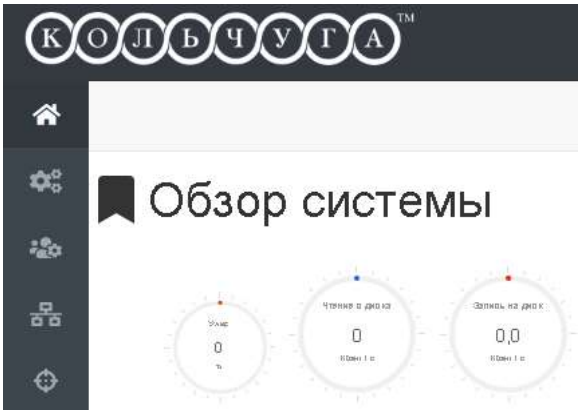
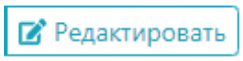
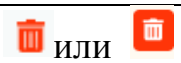
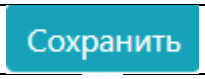


Рис. 14

5.4. Описание основных элементов

В таблице 5 приведено описание основных элементов/кнопок ГИ МЭ ИВК КОЛЬЧУГА-К.

Таблица 5

Элемент/Кнопка	Описание
	Отображает информацию о конфигурационном файле – размер, дата создания, дата изменения (рис. 15)
	Открыть дополнительное меню (рис. 16)
 или или 	Редактировать файл конфигурации или настройки (рис. 16)
	Подробный просмотр
	Удаление объекта
	Очистить
	Копирование объекта
	Перемещение объекта в списке, выставление нужной очередности
	Применение внесенных изменений
	Сбрасывает все не примененные настройки
	Добавление в список интерфейсов, адресов
	Исключение из списка интерфейсов, адресов
	Позволяет отметить статус подтверждения ознакомления администратора с содержанием уведомления
	Предупреждение об ошибках в файле конфигурации
	Правило отключено
	Запрет на редактирование правила
	Сохраняет внесенные изменения в конфигурации
	Объект активный
	Объект не активен, отключен

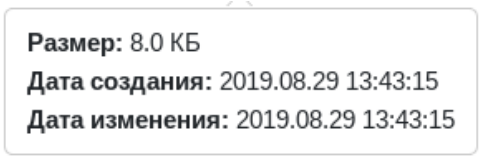


Рис. 15 – Информация о файле конфигурации

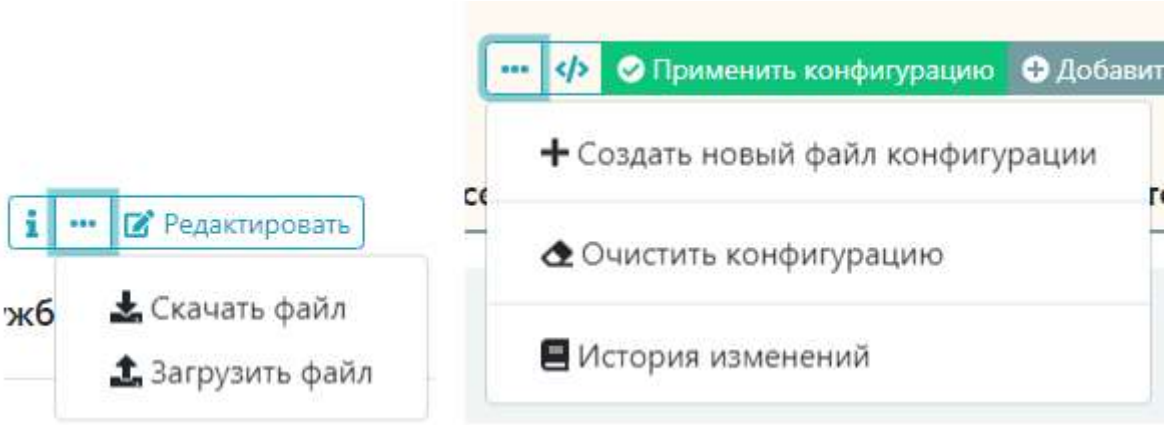


Рис. 16 – Дополнительное меню

6. ПОЛЬЗОВАТЕЛИ

Раздел «Пользователи» доступен только при входе от администратора в ГИ МЭ ИВК КОЛЬЧУГА-К и содержит два подраздела (рис. 17):

- «SSH ключи»;
- «Локальные учетные записи».

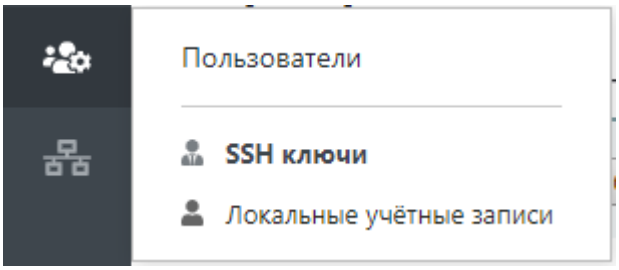


Рис. 17 – Меню раздела «Пользователи»

6.1. SSH ключи

В подразделе «SSH ключи» (рис. 18) осуществляется редактирование списка SSH-ключей администратора системы.

SSH (Secure Shell) – защищенный сетевой протокол для организации сеансов удаленного терминального доступа к серверу.

Протокол SSH (Secure Shell) реализует соединение с удаленным компьютером, защищая от:

- прослушивания данных, передаваемых по этому соединению;
- манипулирования данными на пути от клиента к серверу;
- подмены клиента, либо сервера путем манипулирования IP-адресами, DNS, либо маршрутизацией.

В дополнение к отличным характеристикам в области обеспечения безопасного клиент-серверного соединения, SSH обладает следующими возможностями:

- сжатие передаваемых данных;
- туннелирование каналов внутри установленного соединения;

- широкая распространенность – существуют реализации SSH для самых различных аппаратных платформ и операционных систем.

6.1.1. Добавление SSH ключа

1) Для добавления нового ключа нажмите на кнопку «Обзор».
2) Выберите файл созданного заранее ключа и нажмите кнопку «Добавить».
3) Новый ключ появится в строке разрешенных SSH ключей.

Для удаления ключа в конце строки нажмите кнопку  (рис. 18).

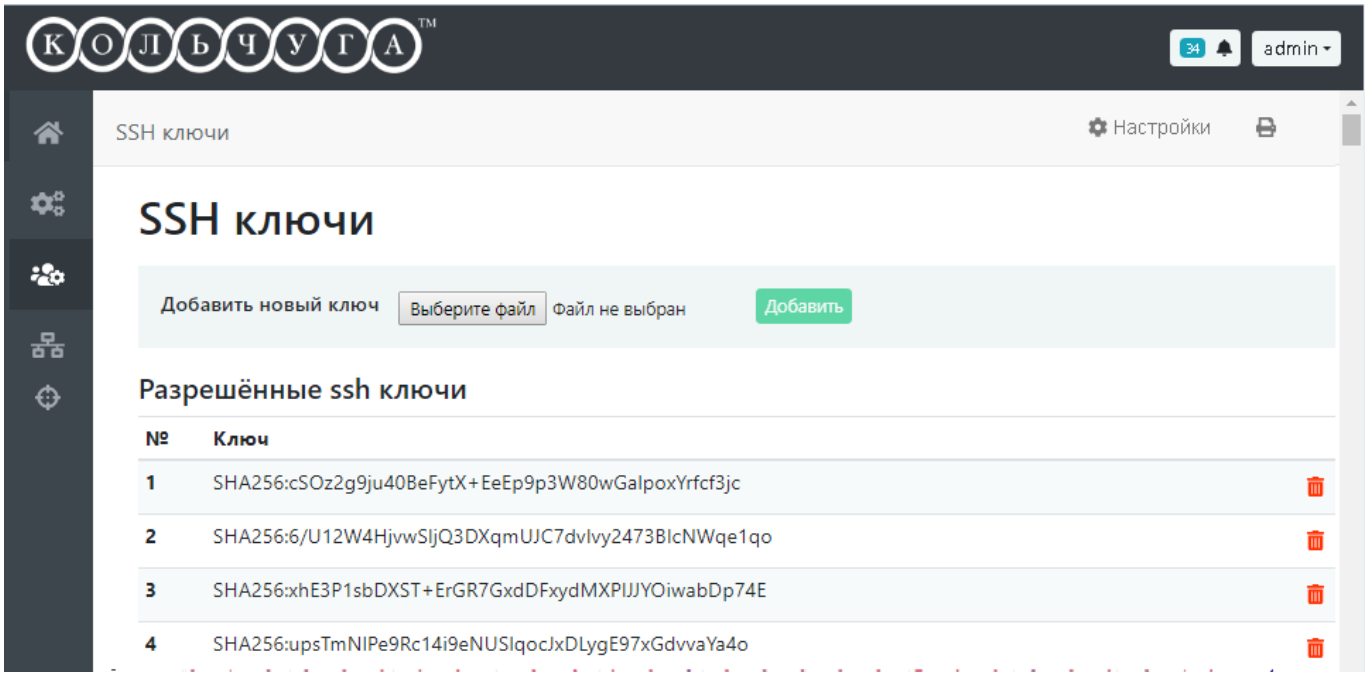


Рис. 18 – Подраздел «SSH ключи»

6.1.2. Настройка sshd

sshd (OpenSSH Daemon) – это программа-сервер, обслуживающая запросы программы-клиента ssh. Вместе эти программы обеспечивают защищенную и кодированную связь между двумя непроверенными компьютерами через незащищенную сеть.

sshd – это служба, принимающая запросы на соединения от клиентов. Для каждого нового соединения создается (с помощью вызова fork) новый экземпляр службы. Ответвленный экземпляр обрабатывает обмен ключами, кодирование, аутентификацию, выполнение команд и обмен данными.

Параметры определяются при помощи ключей командной строки или файла конфигурации (по умолчанию – `sshd_config`). Ключи командной строки имеют больший приоритет, чем значения, указанные в файле конфигурации. При получении сигнала отбоя SIGHUP перечитывает свой файл конфигурации путем запуска собственной копии с тем же самым именем, с которым был запущен.

Основные ключи:

- `-4` – использовать только адреса IPv4;
- `-6` – использовать только адреса IPv6;
- `-g время_задержки_регистрации` – определяет период, в течение которого клиент должен себя идентифицировать (по умолчанию – 120 секунд). Если клиент не смог идентифицировать себя в течение этого времени, экземпляр сервера прекращает свою работу. Значение равное нулю отменяет ограничение на время ожидания;
- `-p порт` – порт, на котором сервер будет ожидать соединения (по умолчанию – 22). Возможно указание нескольких ключей с разными портами. Если данный ключ указан, параметр Port файла конфигурации игнорируется, однако порты, указанные в ListenAddress имеют больший приоритет, чем указанные в командной строке.

Для настройки sshd также можно отредактировать конфигурационный файл `/etc/openssh/sshd_config`.

Основные параметры для изменения:

- `ListenAddress` – локальные IP-адреса, по которым sshd(8) будет ожидать соединения;
- `Port` – ограничение работы на определенном порту. Допустимо указание параметра несколько раз.

Для `ListenAddress` могут быть использованы следующие форматы записей:

```
ListenAddress
хост|адрес-IPv4|адрес-IPv6
ListenAddress
хост|адрес-IPv4:порт
ListenAddress
[хост|адрес-IPv6]:порт
```

Если порт не указан, sshd будет ожидать соединения на указанном адресе и на всех указанных ранее (но не после) в параметре `Port` портах. По умолчанию ожидается соединение на всех локальных адресах. Допустимо указание нескольких параметров.

Для контроля неудачных подключений:

- `LoginGraceTime` – период времени по истечению которого простаивающее подключение будет разорвано (в секундах), если пользователю не удалась регистрация в системе. Если стоит значение 0, то время ожидания не ограничено. Значение по умолчанию – 120 секунд;
- `MaxStartups` – ограничение на число одновременных соединений, в которых не был пройден этап аутентификации. Все последующие соединения не будут приниматься пока на уже существующем соединении не будет произведена аутентификация или не истечет время указанное в параметре `LoginGraceTime`. Значение по умолчанию – «10:30:20». Как альтернатива может быть задействован ранний случайный отказ в подключении путем указания трех разделенных через двоеточие значений «старт:норма:предел» (например, «10:30:60»).

6.2. Локальные учетные записи

Данная вкладка служит для управления учетными записями пользователей.

ВНИМАНИЕ!

При первом подключении к ГИ МЭ ИВК КОЛЬЧУГА-К:
- измените пароль по умолчанию для администратора;
- при необходимости активируйте пользователя аудитора (рис. 20) и установите пароль для него (рис. 21).

Информация о каждой учетной записи содержит (рис. 19):

- статус учетной записи;
- имя пользователя;
- описание пользователя – поле для ввода дополнительной информации о владельце учетной записи;
- признак наличия прав администратора;

- расположение домашнего каталога пользователя, в котором он будет иметь полные права. В случае регистрации в консоли работа начинается именно в этом каталоге. Обычно домашний каталог пользователя располагается в `/home/имя_пользователя`, где `имя_пользователя` – это имя учетной записи;
- интерпретатор команд – это командная оболочка, запускаемая по умолчанию при регистрации пользователя в текстовой консоли. Для администратора используется `/bin/bash`. Аудитору текстовая консоль недоступна.

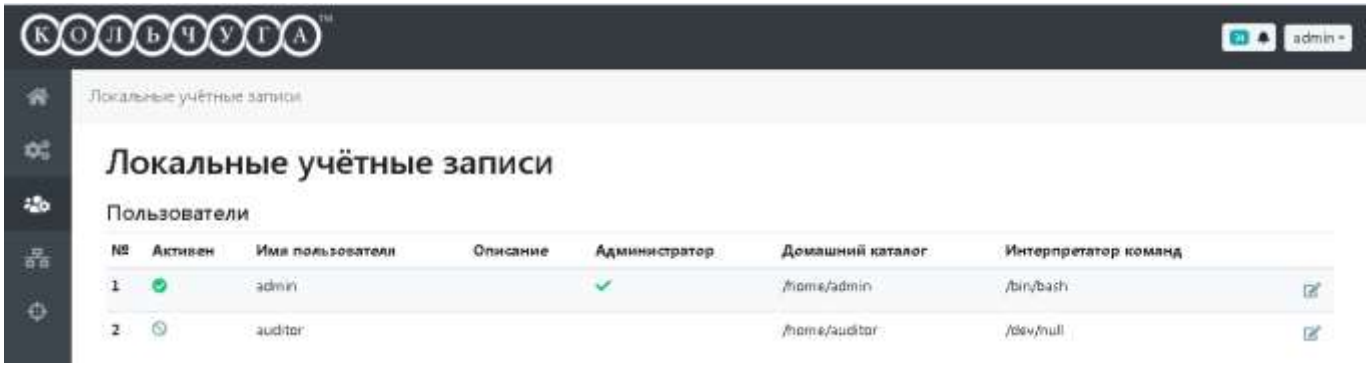


Рис. 19 – Список пользователей

Для дополнительных настроек учетной записи с помощью кнопки открывается окно редактирования данных выбранного пользователя (рис. 20).

Окно редактирования пользователя содержит две вкладки:

- «Данные пользователя»;
- «Изменение пароля».

Во вкладке «Данные пользователя» можно изменить имя и ввести описание учетной записи, а также статус активен/неактивен путем перемещения переключателя (см. рис. 20). При нажатии кнопки «Изменить» введенные данные будут сохранены.

Вкладка «Изменение пароля» (рис. 21) позволяет изменить пароль:

- ввести текущий пароль – всегда вводится пароль администратора;
- дважды ввести новый пароль пользователя;
- нажать на кнопку «Изменить».

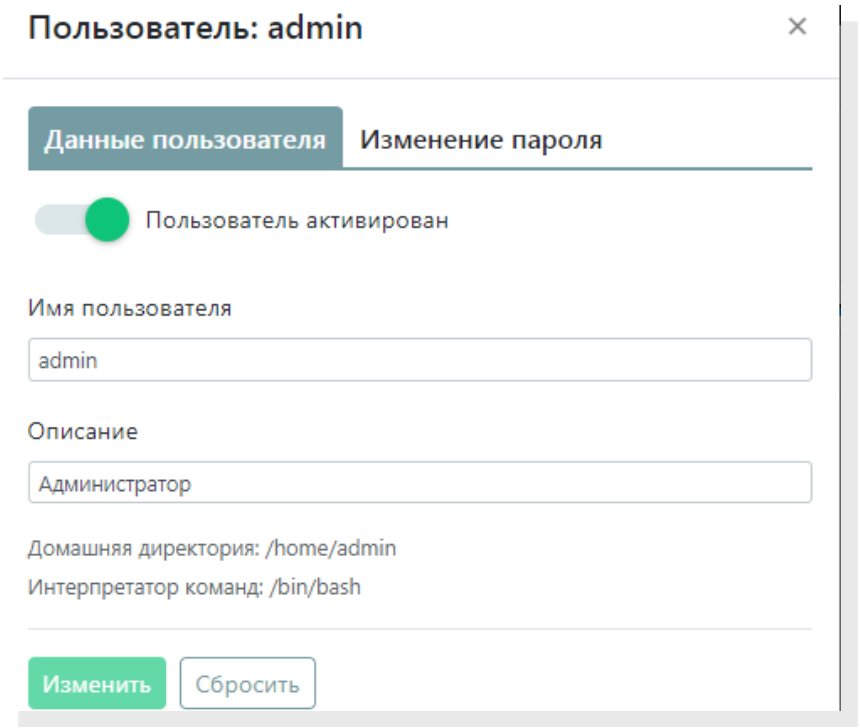


Рис. 20 – Окно «Данные пользователя»

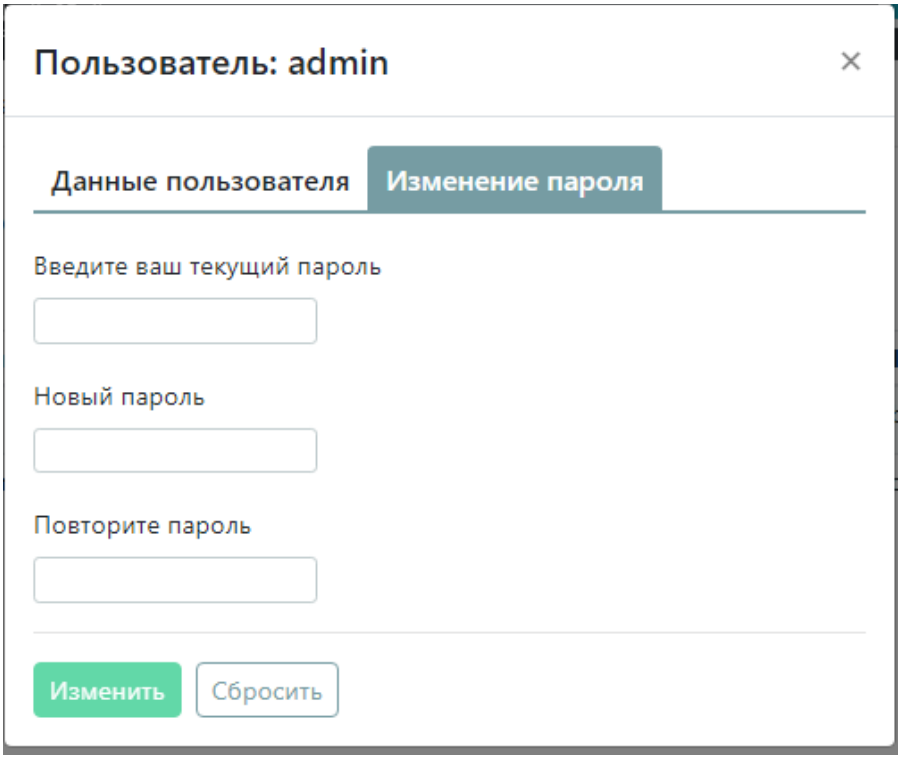


Рис. 21 – Окно «Изменение пароля» пользователя

7. МЕЖСЕТЕВОЙ ЭКРАН

7.1. Общие положения

Основной принцип манипуляции трафиком следующий: на основании заголовков пакетов принимается решение об их обработке. Модулем для модификации правил, по которым МЭ ИВК КОЛЬЧУГА-К обрабатывает пакеты, служит ПО iptables (IP-фильтр), которое реализует в себе всю логику межсетевого экранирования, используя стеки протоколов, вшитые в ядро ОС.

Назначение IP-фильтра – обработка потока данных, проходящих через стек сетевых протоколов ядра ОС по заданным критериям.

Фильтры состоят из правил. Каждое правило – это строка, содержащая в себе критерии, определяющие, подпадает ли пакет под заданное правило, и действие, которое необходимо выполнить в случае удовлетворения критерия.

Входящий пакет проходит по цепочке правил, устанавливаемых iptables. Каждое правило содержит *условие* и *цель* (действие). Если пакет *удовлетворяет условию,* то он *передается на цель*, в противном случае к пакету применяется следующее правило в цепочке. Если пакет не удовлетворил ни одному из условий в цепочке, то к нему применяется *действие_по_умолчанию*.

Для iptables в общем виде правила выглядят так:

```
iptables [-t таблица] команда [match] [target/jump]
```

Если в правило не включается спецификатор `[-t таблица]` (см. п. 7.4), то по умолчанию предполагается использование таблицы `filter` (см. п. 7.4.1); если же предполагается использование другой таблицы, то это требуется указать явно. Спецификатор таблицы так же можно указывать в любом месте строки правила, однако наиболее стандартным считается указание таблицы в начале правила.

Далее, непосредственно за именем таблицы, должна стоять `команда` управления фильтром. Если спецификатора таблицы (`[-t таблица]`) нет, то `команда` всегда должна стоять первой. `Команда` (см. п. 7.5.1) определяет действие

над правилом iptables, например: вставить правило, или добавить правило в конец цепочки, или удалить и т. п.

Тело `команды` в общем виде выглядит так:

`команда цепочка` или так `команда цепочка [номер-правила]`

`Цепочка` указывает в какую цепочку нужно добавить правило. Они находятся в таблицах фильтра. Не все таблицы содержат все стандартные цепочки. Подробнее о встроенных цепочках таблиц см. в п. 7.4.

`[match]` задает критерии проверки, по которым определяется подпадает ли пакет под действие этого правила или нет (см. подробнее п. 7.6). Здесь мы можем указать самые разные критерии; IP-адрес источника пакета или сети, IP-адрес места назначения, порт, протокол, сетевой интерфейс и т. д.

`[target/jump]` указывает, какое действие/цель или переход (см. п. 7.2, п. 7.3 и п. 7.7) должно быть выполнено при условии выполнения критериев (`[match]`) в правиле. Здесь можно передать пакет в другую цепочку правил, «сбросить» пакет и забыть про него, выдать на источник сообщение об ошибке и т. д.

Схематично обработку пакета можно изобразить следующим образом (рис. 22).

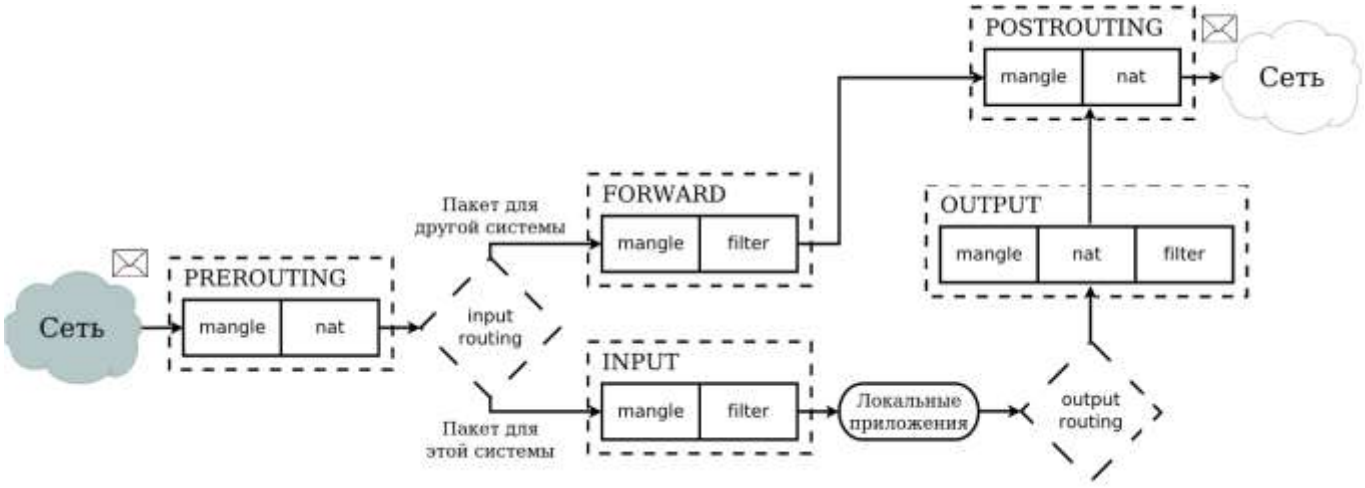


Рис. 22 – Схема обработки пакета

Входящий пакет начинает обрабатываться МЭ с цепочки PREROUTING в таблице `mangle` (см. п. 7.4.3). Затем он обрабатывается правилами цепочки PREROUTING таблицы `nat` (см. п. 7.4.2). На этом этапе проверяется, не требуется ли модификация назначения пакета (DNAT п. 7.3.8). Важно сменить назначение сейчас, потому что маршрут пакета определяется сразу после того, как он покинет цепочку PREROUTING. После этого он будет отправлен на цепочку INPUT (если целью пакета является этот компьютер) или FORWARD (если его целью является другой компьютер в сети).

Если целью пакета является другой компьютер, то пакет фильтруется правилами цепочки FORWARD таблиц `mangle` (см. п. 7.4.3) и `filter` (см. п. 7.4.1), а затем к нему применяются правила цепочки POSTROUTING. На данном этапе можно использовать SNAT (подмена источника см. п. 7.3.7)/MASQUARADE (маскировка см. п. 7.3.9). После этих действий пакет будет отправлен в сеть.

Если назначением пакета является сам компьютер с МЭ, то, после маршрутизации, он обрабатывается правилами цепочек INPUT таблиц `mangle` (см. п. 7.4.3) и `filter` (см. п. 7.4.1). В случае прохождения цепочек, пакет передается локальному приложению.

Когда приложение, на компьютере с МЭ, отвечает на запрос или отправляет собственный пакет, то он обрабатывается цепочкой OUTPUT таблицы `filter` (см. п. 7.4.1). Затем к нему применяются правила цепочки OUTPUT таблицы `nat` (см. п. 7.4.2), для определения, требуется ли использовать DNAT (преобразование адреса назначения см. п. 7.3.8), пакет фильтруется цепочкой OUTPUT таблицы `filter` (см. п. 7.4.1) и выпускается в цепочку POSTROUTING которая может использовать SNAT (см. п. 7.3.7). В случае успешного прохождения POSTROUTING пакет выходит в сеть (NET)(рис. 23).

Как видно из рис. 23, вся логика управления пакетами, обеспечивающая необходимый уровень безопасности, реализована в ПО iptables, которое активно взаимодействует со стеками протоколов, являющихся, буфером для пакетов, при переходе от одной цепочке к другой.

Каждый раз после прохождения одной из цепочек, пакет попадает в соответствующий ему стек, откуда перенаправляется в другую цепочку.

Перечень основных стеков протоколов ядра, с которыми взаимодействует iptables:

- TCP/IP;
- IPX/SPX;
- NetBIOS/SMB;
- DECnet;
- SNA;
- OSI.

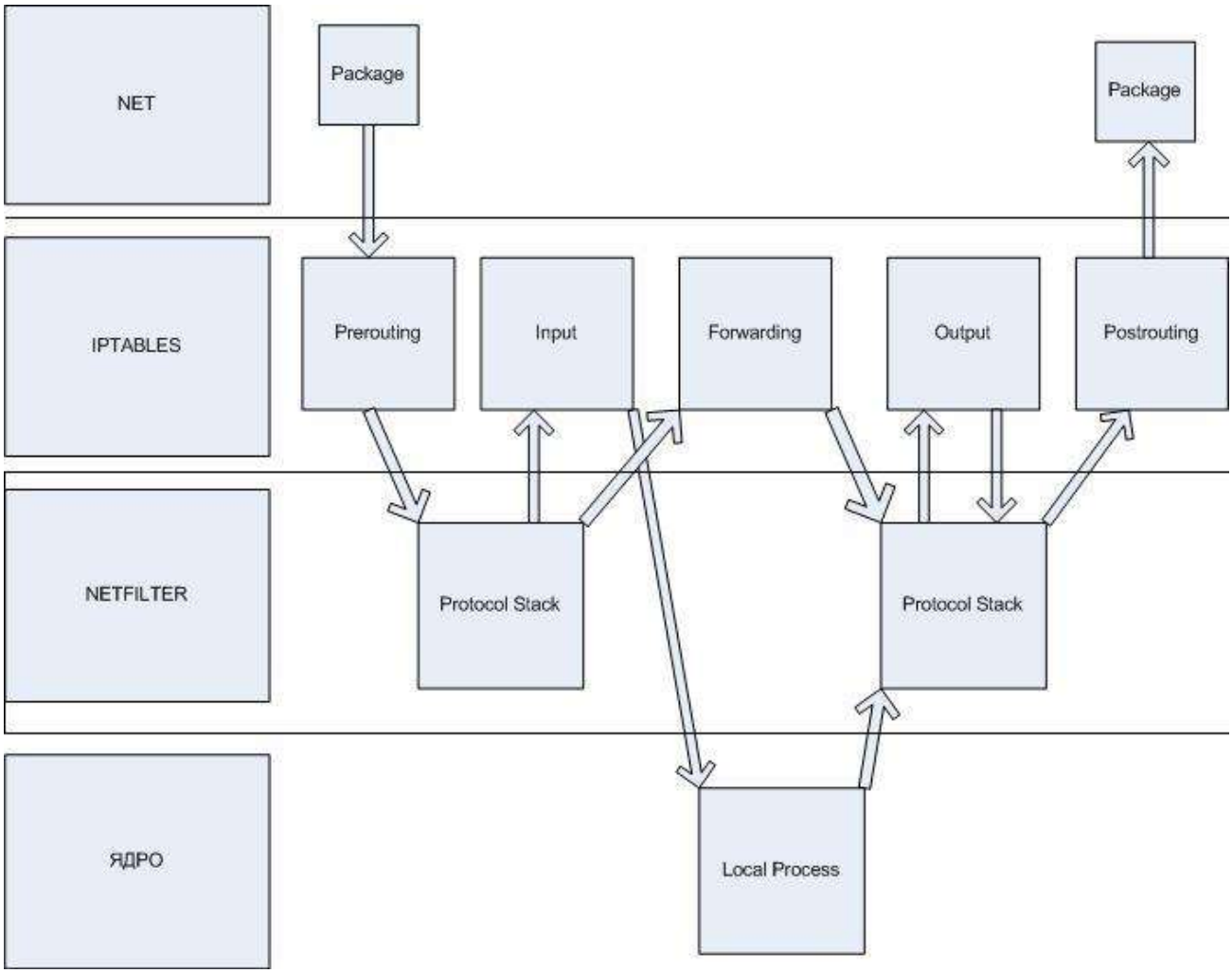


Рис. 23 – Схема обработки пакета в разбивке, на участвующие в обработке процессы

7.2. Действия iptables

В каждом правиле МЭ определяются критерии для пакета и цели. Если пакет не попадает под действие правила, проверяется следующее правило в цепочке; если попадает – проверяется правило, указанное в цели, которая может быть именем новой цепочки или одной из специальных целей:

- ACCEPT – означает принять (пропустить далее) пакет. Пакет прекращает движение по цепочке (и всем вызвавшим цепочкам, если текущая цепочка была вложенной) и считается принятым, тем не менее, пакет продолжит движение по цепочкам в других таблицах и может быть отвергнут там;
- DROP – означает проигнорировать (отбросить) пакет. Отброшенные пакеты прекращают свое движение полностью;
- QUEUE – передать пакет в адресное пространство пользователя;
- RETURN – остановить применение правил этой цепочки и передать пакет следующему правилу предыдущей (вызывающей) цепочки. Если достигается конец встроенной (предопределенной) цепочки или в ней к пакету применяется правило с целью RETURN, то окончательное действие над пакетом (цель) определяется стратегией (политикой) цепочки. Если текущая цепочка лежит на самом верхнем уровне (например, INPUT), то к пакету будет применена политика по умолчанию.

По умолчанию выполняется действие: ACCEPT.

7.3. Расширения действий/целей

iptables может использовать расширенные действия/цели, приведенные в этом подразделе.

7.3.1. REJECT

REJECT – отклонить с уведомлением, используется, как правило, в тех же самых ситуациях, что и DROP, но в отличие от него, выдает сообщение об ошибке на хост, передавший пакет. Действие REJECT на сегодняшний день может использоваться только в цепочках INPUT, FORWARD и OUTPUT (и во вложенных в них цепочках). Пока существует только единственный ключ `--reject-with`,

управляющий поведением REJECT – отбросить входной пакет и отправить обратно соответствующее сообщение об ошибке ICMP (по умолчанию используется `port-unreachable`).

Синтаксис:

- `--reject-with тип` – возможные значения типа, возвращающего соответствующее сообщение об ошибке ICMP приведены в таблице 6.

Таблица 6

Тип, псевдоним	Описание
`icmp-net-unreachable`, `net-unreach`	Сеть недоступна
`icmp-host-unreachable`, `host-unreach`	Хост недоступен
`icmp-proto-unreachable`, `proto-unreach`	Неподдерживаемый протокол
`icmp-port-unreachable`, `port-unreach`	Порт недоступен (по умолчанию)
`icmp-net-prohibited`, `net-prohib`	Сеть запрещена
`icmp-host-prohibited`, `host-prohib`	Хост запрещен
`tcp-reset`, `tcp-rst`	Можно использовать для правил, которые соответствуют только протоколу TCP, вызывает отправку пакета TCP RST обратно
`icmp-admin-prohibited`, `admin-prohib`	Запрещен в административном порядке

7.3.2. LOG

LOG – действие, которое служит для журналирования отдельных пакетов и событий. В журнал могут заноситься заголовки IP-пакетов и другая интересующая информация. Информация из журнала может быть затем прочитана с помощью dmesg или syslogd либо с помощью других программ. Может применяться для отладки созданных правил, например, на период отладки правил вместо действия DROP (см. п. 7.2) использовать действие LOG, чтобы до конца убедиться, что настройки МЭ работают верно. Обратите внимание так же на действие ULOG

(см. п. 7.3.4), которое позволяет выполнять запись журналируемой информации не в системный журнал, а в базу данных.

Действие LOG имеет ключи/опции, приведенные в таблице 7.

Т а б л и ц а 7 – LOG синтаксис ключей/опций

Ключи	Описание
`--log-level level`	Используется для задания уровня журналирования. `level` – число или тип уровня, полный список уровней см. в руководстве (man) по `syslog.conf`. Например, можно задать следующие уровни (см. также таблицу 8): `debug`, `info`, `notice`, `warning`, `warn`, `err`, `error`, `crit`, `alert`, `emerg` и `panic`. **Пример**: `iptables -A FORWARD -p tcp -j LOG --log-level debug`

`--log-prefix prefix`	Ключ задает текст (префикс), которым будут выделяться сообщения iptables. Сообщения со специфичным префиксом затем можно найти, к примеру, с помощью команды grep. Префикс может содержать до 29 символов, включая и пробелы.
	Примеры:
	`iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"`
	`iptables -A INPUT -j LOG --log-prefix "INPUT_T:"`

`--log-tcp-sequence`	Этот ключ позволяет заносить в системный журнал номер TCP Sequence пакета. Номер TCP Sequence идентифицирует каждый пакет в потоке и определяет порядок "сборки" потока. Этот ключ потенциально опасен для безопасности системы, системный журнал не должен иметь доступ на чтение для всех пользователей. Как и любой другой журнал, содержащий сообщения от iptables. **Пример**: `iptables -A INPUT -p tcp -j LOG --log-tcp-sequence`
`--log-tcp-options`	Этот ключ позволяет заносить в системный журнал различные сведения из заголовка TCP-пакета. Такая возможность может быть полезна при отладке. Этот ключ не имеет дополнительных параметров. **Пример**: `iptables -A FORWARD -p tcp -j LOG --log-tcp-options`
`--log-ip-options`	Этот ключ позволяет заносить в системный журнал данные из заголовка IP-пакета. **Пример**: `iptables -A FORWARD -p tcp -j LOG --log-ip-options`
`--log-uid`	Этот ключ позволяет заносить в системный журнал идентификатор пользователя, которому принадлежит процесс, сгенерировавший пакет.
`--log-macdecode`	Этот ключ позволяет заносить в системный журнал расшифрованные MAC-адреса и протокол.

7.3.3. LOGMARK

LOGMARK – специальная модификация действия LOG (см. п. 7.3.2), реализованная в комплекте xtables-addons. Отличается тем, что заносит в лог информацию, специфичную для системы conntrack, в частности, маркировку соединения (connmark aka ctmark), состояния соединения (ctstate и ctstatus) и т. п.

Синтаксис ключей/опций LOGMARK:

- `--log-level level` – уровень журналирования (номер, см. таблицу 8);
- `--log-prefix prefix` – префикс для выделения записей среди других сообщений.

Применение аналогично ключам/опциям LOG (см. п. 7.3.2).

Пример:

```
iptables -A INPUT -j LOGMARK --log-prefix "BAD_INPUT: " --log-level 3
```

Таблица 8 – `log-level` перечень значений

Номер	Имя	Описание
7	debug	отладочная информация
6	info	информационное сообщение
5	notice	уведомление
4	warning	предупреждение
3	error	ошибка
2	crit	критично
1	alert	тревога
0	emerg	отказ системы

7.3.4. ULOG

ULOG предоставляет возможность журналирования пакетов в пользовательское пространство, заменяет действие LOG (см. п. 7.3.2). При использовании этого действия, пакет через сокеты netlink, передается специальному демону который может выполнять детальное журналирование в различных форматах и к тому же поддерживает возможность добавления надстроек (плагинов) для формирования различных выходных форматов и обработки сетевых протоколов.

Синтаксис ключей/опций ULOG приведен в таблице 9.

Т а б л и ц а 9 – ULOG синтаксис ключей/опций

Ключи	Описание
`--ulog-nlgroup`	Сообщает ULOG в какую группу netlink должен быть передан пакет. Всего существует 32 группы (от 1 до 32). По умолчанию используется 1-я группа. **Пример:** `iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-nlgroup 2`
`--ulog-prefix`	Имеет тот же смысл, что и аналогичная опция в действии LOG (см. п. 7.3.2). Длина строки префикса не должна превышать 32 символа. **Пример:** `iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-prefix "SSH connection attempt: "`
`--ulog-cprange size`	Определяет, какую долю пакета, в байтах, надо передавать демону ULOG. Если указать вместо size число 100, как показано в примере, то демону будет передано только 100 байт из пакета, это означает, что демону будет передан заголовок пакета и некоторая часть области данных пакета. Если указать 0, то будет передан весь пакет, независимо от его размера. Значение по умолчанию равно 0. **Пример:** `iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-cprange 100`
`--ulog-qthreshold size`	Устанавливает величину буфера в области ядра. Например, если задать величину буфера равной 10, как в примере, то ядро будет накапливать журналируемые пакеты во внутреннем буфере и передавать в пользовательское пространство группами по 10 пакетов. По умолчанию размер буфера равен 1 (из-за сохранения обратной совместимости). **Пример:** `iptables -A INPUT -p TCP --dport 22 -j ULOG --ulog-qthreshold 10`

7.3.5. NFLOG

NFLOG обеспечивает возможность журналирования пакетов. Когда эта цель установлена для правила, ядро Linux передаст загруженный пакет для регистрации в журнале. Это обычно используется в сочетании с nfnetlink_log для многоадресной рассылки. Один или несколько процессов пользовательского пространства могут подписаться на определенную группу для получения пакетов. Как и LOG (см. п. 7.3.2), это не завершающаяся цель, то есть обход продолжится на следующем правиле.

Синтаксис ключей/опций NFLOG:

- `--nflog-group NUM` – сообщает какой номер группы `NUM` netlink, к которой относится пакет (применимо только для nfnetlink_log). Значение по умолчанию равно 0;
- `--nflog-prefix STRING` – строка префикса, включаемая в сообщение журнала, длиной до 64 символов, полезна для различения сообщений в журналах;
- `--nflog-range NUM` – сообщает количество байтов (`NUM`), которые будут скопированы в пользовательское пространство (применимо только для nfnetlink_log). Экземпляры nfnetlink_log могут указывать свой собственный диапазон, этот параметр переопределяет его;
- `--nflog-threshold NUM` – сообщает количество пакетов (`NUM`), стоящих в очереди внутри ядра перед отправкой их в пользовательское пространство (применимо только для nfnetlink_log). Значение по умолчанию равно 1.

7.3.6. NFQUEUE

NFQUEUE является расширением QUEUE (см. п. 7.2). В отличие от QUEUE, позволяет поместить пакет в любую конкретную очередь, идентифицируемую 16-битным номером.

Синтаксис ключей/опций NFQUEUE:

- `--queue-num [значение]` – сообщает номер (`[значение]`) используемой очереди (QUEUE). Допустимые номера очередей от 0 до 65535. Значение по умолчанию равно 0;
- `--queue-balance значение1:значение2` – сообщает диапазон используемых очередей. Затем пакеты балансируются по заданным очередям. Это полезно для многоядерных систем: запустите несколько экземпляров программы userspace в очередях x, x+1, .. x+n и используйте "`--queue-balance x:x+n`". Пакеты, принадлежащие одному и тому же соединению, помещаются в один и тот же nfqueue;

- `--queue-bypass` – сообщает что нужно выполнить обход очереди, если очередь не существует;
- `--queue-cpu-fanout` – сообщает что нужно использовать текущее CPU (не хэширование).

7.3.7. SNAT

SNAT (Source Network Address Translation) используется для преобразования сетевых адресов, т. е. изменение исходящего IP-адреса в IP-заголовке пакета.

Синтаксис ключей/опций SNAT:

- `--to-source ipaddr[-ipaddr][:port[-port]]` – IP-адрес или диапазон адресов (включительно) источника, а также диапазон портов;
- `[--random] [--persistent]` – произвольные значения для адресов, портов, например, 194.236.50.155-194.236.50.160:1024-32000.

7.3.8. DNAT

DNAT (Destination Network Address Translation) используется для преобразования адреса места назначения в IP-заголовке пакета.

Синтаксис ключей/опций DNAT:

- `--to-destination [ipaddr][-ipaddr][:port[-port]]` – определяет новый целевой IP-адрес, либо диапазон (включительно) IP-адресов и диапазон портов;
- `[--random] [--persistent]` – произвольные значения для адресов, портов.

7.3.9. MASQUERADE

MASQUERADE означает скрыть (маскировать) IP.

В основе своей представляет то же самое, что и SNAT только не имеет ключа `--to-source`. Причиной тому то, что MASQUERADE может работать, например, с dialup подключением или DHCP, т. е. в тех случаях, когда IP-адрес присваивается устройству динамически. Если используется динамическое подключение, то нужно использовать MASQUERADE, если же используется статическое IP-подключение, то лучшим выходом будет использование действия SNAT (см. п. 7.3.7).

Синтаксис ключей/опций MASQUERADE:

- `--to-ports port[-port]` – определяет диапазон используемых исходных портов, отменяя выбора исходного порта SNAT (см. п. 7.3.7). Допустима только в том случае, если также указаны `-p tcp`, `-p udp` (см. п. 7.9.2);
- `--random` – случайно выбранные порты источника.

7.3.10. NETMAP

NETMAP позволяет статически сопоставить сеть адресов с другой сетью адресов. Можно использовать только для правил в таблице `nat` (см. п. 7.4.2).

Синтаксис ключей/опций NETMAP:

- `--to address [/ mask]` – сообщает сетевой адрес (или диапазон адресов). Результирующий адрес будет построен следующим образом: все биты «единицы» в маске заполняются с нового «адреса». Все биты, которые равны нулю в маске, заполняются из исходного адреса.

7.3.11. REDIRECT

Выполняет перенаправление пакетов и потоков на другой порт той же самой машины. К примеру, можно пакеты, поступающие на HTTP-порт перенаправить на порт HTTP proxy. Действие REDIRECT очень удобно для выполнения «прозрачного» проксирования (transparent proxy), когда компьютеры в локальной сети даже не подозревают о существовании прокси.

Синтаксис ключей/опций REDIRECT:

- `--to-ports port[-port]` – диапазон портов, на которые следует перенаправлять пакет. Опция допустима только в том случае, если также указаны `-p tcp`, `-p udp` (см. п. 7.9.2);
- `[--random]` – при использовании отображение портов будет случайным.

7.3.12. TTL

Действие TTL используется для установки значения поля TTL (Time To Live) пакета. Можно присваивать определенное значение этому полю, чтобы скрыть МЭ от «чересчур любопытных» провайдеров (Internet Service Providers). Дело в том, что отдельные провайдеры не приветствуют, когда одно подключение разделяется

несколькими компьютерами, и тогда они начинают проверять значение TTL приходящих пакетов и используют его как один из критериев определения того, один ли компьютер на подключении или несколько.

Поле TTL определяет, сколько переходов (маршрутизаторов) может пройти пакет, пока не будет превышено время его жизни.

Установка или увеличение поля TTL потенциально могут быть очень опасными, поэтому этого следует избегать.

Никогда не устанавливайте и не увеличивайте значение для пакетов, которые покидают вашу локальную сеть в таблице `mangle` (см. п. 7.4.3).

Синтаксис ключей/опций TTL:

- `--ttl-set value` – установить TTL в выбранное значение (от 0 до 255);
- `--ttl-dec value` – уменьшить TTL в 1 – 255 (`value`) раз;
- `--ttl-inc value` – увеличить TTL в 1 – 255 (`value`) раз.

7.3.13. NETFLOW

NETFLOW – быстрый контролер и обработчик трафика, предназначен для применения в сетях с высокой пропускной способностью, состоит из модуля ядра ipt_NETFLOW и iptables, поддерживает NetFlow v5, v9, v10(IPFIX).

7.3.14. TARPIT

TARPIT – цель, позволяющая создать порт-ловушку, который используется для замедления входящих соединений, имитирует открытые порты ловушку для атакующих TCP-соединений. При срабатывании правила TARPIT отправитель получает ответ SYN/ACK с принудительным установлением размера TCP-окна, равным нулю. Отправитель при этом не может отправлять никаких данных, а попытки разорвать соединение игнорируются. То есть соединение подвисает до истечения таймаута, при этом компьютер, который инициировал соединение будет тратить свои системные ресурсы, а МЭ просто будет останавливать попытки передачи данных.

Например, открываем порт 443 для пустых соединений:

```
iptables -A INPUT -i eth0 -p tcp -m tcp --destination-port 443 -j TARPIT
```

7.3.15. DELUDE

DELUDE – создает видимость открытого TCP-порта, на SYN-пакеты отвечает пакетами SYN/ACK, на все прочие пакеты отвечает RST. Таким образом, может применяться для введения в заблуждение злоумышленника, сканирующего порты, например, будет показывать сканеру атакующей машины (nmap) при SYN-сканировании, что целевой порт открыт, в то время как на самом деле порт закрыт.

7.3.16. CHAOS

CHAOS – для каждого нового TCP-соединения случайно выбирает одно из двух действий:

- `--delude` – включить обработку DELUDE (см. п. 7.3.15) для TCP-соединений;
- `--tarpit` – включить обработку TARPIT (см. п. 7.3.14) для TCP-соединений.

В частности, при использовании действия CHAOS и опции `-delude` для всех неиспользуемых портов, сканирующий порты злоумышленник получит совершенно неверную информацию о реальном состоянии портов. В случае с CHAOS и опции `--tarpit` ситуация усугубится еще и «подвисающими» соединениями.

7.3.17. TOS

TOS устанавливает поле Type of Service в заголовке IPv4 (включая биты «приоритета») или поле Priority в заголовке IPv6. Это поле используется для назначения сетевой политики обслуживания пакета, т. е. задает желаемый вариант маршрутизации. Другими словами, не следует изменять состояние этого поля для пакетов, уходящих в Интернет, потому что на роутерах, которые обслуживают это поле, может быть принято неправильное решение при выборе маршрута. Обратите внимание, что TOS использует те же биты, что и DSCP (см. п. 7.3.18) и ECN (см. п. 7.3.23). Цель TOS действительна только в таблице `mangle` (см. п. 7.4.3).

Синтаксис ключей/опций TOS:

- `--set-tos value[/mask]` – установить значение в поле Type of Service/Priority (обнуление битов в маске и значении XOR в TOS);
- `--set-tos символ` – установить поле TOS (только для IPv4) по символу (доступные значения 16, 8, 4, 2, 0) (зануляет 4 бита из приоритетной части!).

Принятые символические имена для значений:

а) (0x10) 16 Minimize-Delay;

б) (0x08) 8 Maximize-Throughput;

в) (0x04) 4 Maximize-Reliability;

г) (0x02) 2 Minimize-Cost;

д) (0x00) 0 Normal-Service.

Доступны также следующие мнемоники:

- `--and-tos биты` – выполнить операцию И (AND) над значением TOS с битами;
- `--or-tos биты` – выполнить операцию ИЛИ (OR) над значением TOS с битами;
- `--xor-tos биты` – выполнить операцию XOR над значением TOS с битами.

7.3.18. DSCP

Позволяет изменять значение битов DSCP в заголовке TOS пакета IPv4. Поскольку DSCP манипулирует пакетом, его можно использовать только в таблице `mangle` (см. п. 7.4.3).

Синтаксис ключей/опций DSCP:

- `--set-dscp value` – устанавливает в поле DSCP числовое значение (может быть десятичным или шестнадцатеричным);
- `--set-dscp-class class` – устанавливает поле DSCP в класс DiffServ.

7.3.19. MARK

Действие MARK устанавливает специальную метку (netfilter) на пакет, которая затем может быть проверена другими правилами в iptables или другими программами, например, iproute2. С помощью меток можно управлять маршрутизацией пакетов, ограничивать траффик и т. п.

MARK может быть использован только в таблице `mangle` (см. п. 7.4.3).

Синтаксис ключей/опций MARK:

- `--set-xmark value[/mask]` – обнуляет биты, заданные значением маски и XOR, в метке пакета («nfmark»). Если маска пропущена, то предполагается 0xFFFFFFFF.

Доступны следующие мнемоники:

- `--and-mark bits` – двоичный код И (AND) nfmark с битами (мнемоника для `--set-xmark 0/invbits`, где `invbits` – двоичное отрицание битов);
- `--or-mark bits` – двоичный код ИЛИ (OR) nfmark с битами (мнемоника для `--set-xmark bits/bits`);
- `--xor-mark bits` – двоичный код XOR nfmark с битами (мнемоника для `--set-xmark bits/0`).

7.3.20. CLASSIFY

Этот модуль позволяет установить skb → значение приоритета (и таким образом классифицировать пакет в определенный класс CBQ).

Синтаксис ключей/опций CLASSIFY:

- `--set-class major:minor` – устанавливает major и minor значения класса. Значения всегда интерпретируются как шестнадцатеричные, даже если префикс 0x не задан.

7.3.21. CONNMARK

Этот модуль устанавливает значение метки Netfilter, связанное с соединением. Метка имеет ширину 32 бита.

Синтаксис ключей/опций CONNMARK:

- `--set-xmark value[/mask]` – обнуляет биты, заданные маской и значением XOR в ctmark;
- `--save-mark [--nfmask nfmask] [--actmask actmask]` – копирует метку пакета (метку nf) в метку соединения (метку ct), используя заданные маски. Новое значение nfmark определяется следующим образом:

  `ctmark = (ctmark & ~ctmask) ^ (nfmark & nfmask)`

  т. е. ctmask определяет, какие биты очистить и nfmask, какие биты nfmark XOR в ctmark. ctmask и unmask по умолчанию имеют значение 0xFFFFFFFF;
- `--restore-mark [--nfmask nfmask] [--actmask actmask]` – копирует метку соединения (ctmark) в метку пакета (nfmark), используя заданные маски. Новое значение марки КТ определяется следующим образом:

  `nfmark = (nfmark & ~nfmask) ^ (ctmark & ctmask)`;

  т. е. netmask определяет, какие биты очистить, а ctmax – какие биты ctmark XOR в nfmark. ctmax и nfmask по умолчанию имеют значение 0xFFFFFFFF. Опция действительна только в таблице `mangle` (см. п. 7.4.3).

Для `--set-xmark` доступны следующие мнемоники:

- `--and-mark bits` – двоичный код И (AND) ctmark с битами (мнемоника для `--set-xmark 0/invbits`, где invbits – двоичное отрицание битов);
- `--or-mark bits` – двоичный ИЛИ (OR) ctmark с битами (мнемоника для `--set-xmark bits/bits`);
- `--xor-mark bits` – двоичный XOR ctmark с битами (мнемоника для `--set-xmark bits/0`).

7.3.22. TCPMSS

Этот модуль позволяет изменять значение MSS в TCP/SYN пакетах, для контроля максимального размера пакетов в этом соединении (обычно ограничивая его MTU исходящего интерфейса минус 40 байт для IPv4 или минус 60 для IPv6). TCPMSS можно использовать только в сочетании с `-p tcp` (см. п. 7.9.2). Действителен только в таблице `mangle` (см. п. 7.4.3).

Этот модуль используется для преодоления блокировки пакетов «ICMP Fragmentation Needed» или «ICMPv6 Packet Too Big». Симптомы этой проблемы заключаются в том, что, например, на МЭ все работает, а компьютеры, стоящие за ним не могут обмениваться большими пакетами:

- веб-браузеры подключаются, а затем зависают без получения каких-либо данных;
- маленькие электронные письма приходят нормально, но большие письма висят;
- ssh работает, но scp зависает после подтверждения связи.

Решением данной проблемы является добавление правила в конфигурацию МЭ, например:

```
iptables -t mangle -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu
```

Синтаксис ключей/опций TCPMSS:

- `--set-mss value` – явная установка параметра MSS в заданное значение. Если MSS пакета уже ниже значения, он не будет увеличен, применяется чтобы избежать дополнительных проблем с хостами, полагающимися на надлежащий MSS;
- `--clamp-mss-to-pmtu` – автоматическая установка значения MSS в (path_MTU – 40 для IPv4; и 60 для IPv6). Может не сработать должным образом там, где существуют асимметричные маршруты с различным MTU пути – ядро использует MTU пути, которые оно использовало бы для отправки пакетов от себя к IP-адресам источника и назначения.

Опции являются взаимоисключающими.

7.3.23. ECN

Этот модуль позволяет избирательно обходить известные ECN дыры. Его можно использовать только в таблице `mangle` (см. п. 7.4.3).

Синтаксис ключей/опций ECN:

- `--ecn-tcp-remove` – удаляет все биты ECN из заголовка TCP. Возможно использование только в сочетании с `-p tcp` (см. п. 7.9.2).

7.3.24. TCPOPTSTRIP

Этот модуль удаляет TCP-параметры TCP-пакета (на самом деле он заменит их на NO-OPs). Поэтому необходимо использовать в сочетании с `-p tcp` (см. п. 7.9.2).

Синтаксис ключей/опций TCPOPTSTRIP:

`--strip-options option[,option...]`

Удаляет указанные параметры, которые можно задавать номером опции TCP или символическим именем (таблица 10). Список распознанных опций можно также получить, вызвав iptables с помощью `-j TCPOPTSTRIP -h`.

Т а б л и ц а 10

Имя	TCP-опция (см. таблицу 17)
Wscale	Window Scale
Mss	Maximum Segment Size
sack-permitted	SACK-Permitted
sack	SACK
timestamp	Timestamps
md5	MD5 Signature Option

7.3.25. TPROXY

Этот модуль действителен только в таблице `mangle` (см. п. 7.4.3), в маршрутизации цепочки PREROUTING и пользовательских цепочках, которые вызываются только из этой цепочки. Он перенаправляет пакет в локальный сокет без какого-либо изменения заголовка пакета, также может изменить значение метки, которое затем может быть использовано в расширенных правилах маршрутизации.

Синтаксис ключей/опций TPROXY:

- `--on-port port` – указывает порт назначения для использования. Это обязательный параметр. 0 означает, что новый порт назначения совпадает с исходным. Использование `--on-port` допустимо только в том случае, если в правиле также указаны `-p tcp` или `-p udp` (см. п. 7.9.2);
- `--on-ip address` – указывает адрес назначения для использования. По умолчанию этот адрес является IP-адресом входящего интерфейса.

Использование `--on-ip` допустимо только в том случае, если в правиле также указаны `-p tcp` или `-p udp` (см. п. 7.9.2);

- `--tproxy-mark value[/mask]` – помечает пакеты с заданным значением/маской. Значение fwmark, установленное здесь, может быть использовано расширенной маршрутизацией (требуется для работы прозрачного проксирования: в противном случае эти пакеты будут перенаправлены).

7.3.26. NOTRACK

Эта цель отключает отслеживание соединений для всех пакетов, соответствующих этому правилу, можно использовать только в таблице `raw` (см. п. 7.4.4). Дополнительных опций/ключей не имеет.

7.3.27. CT

CT позволяет задать параметры для пакета или связанного с ним соединения, добавляет к пакету запись отслеживания соединения «шаблон», которая затем используется ядром conntrack при инициализации новой записи ct, можно использовать только в таблице `raw` (см. п. 7.4.4).

Синтаксис ключей/опций CT:

- `--notrack` – отключает отслеживание соединения для этого пакета;
- `--helper name` – использует помощника, идентифицированного по имени для подключения. Этот способ более гибкий, чем загрузка вспомогательных модулей conntrack с предустановленными портами;
- `--ctevents event[,...]` – генерирует только указанные события conntrack для этого соединения. Возможные типы событий: new, related, destroy, reply, assured, protoinfo, helper, mark (относится к ctmark, а не к nfmark), natseqinfo, secmark (ctsecmark);
- `--expevents event[,...]` – генерирует только указанные события ожидания для этого соединения. Возможные типы событий: new;
- `--zone-orig {id|mark}` – для трафика, идущего с направления ORIGINAL, назначает этот пакет идентификатору зоны и выполнит поиск только в этой

зоне. Если вместо id используется mark, то используется зона из пакета nfmark;

- `--zone-reply {id|mark}` – для трафика, поступающего с направления REPLY, назначает этот пакет идентификатору зоны и выполняет поиск только в этой зоне. Если вместо id используется mark, то используется зона из пакета nfmark;
- `--zone {id|mark}` – назначает этот пакет идентификатору зоны и выполняет поиск только в этой зоне. Если вместо id используется mark, то используется зона из пакета nfmark. По умолчанию пакеты имеют зону 0. Эта опция применима к обоим направлениям;
- `--timeout name` – использует политику тайм-аута, определенную по имени для соединения. Обеспечивает более гибкое определение политики таймаута, чем глобальные значения таймаута, доступные в `/proc/sys/net/netfilter/nf_conntrack_*_timeout_*`.

7.4. Таблицы

Опция `-t` (см. п. 7.1) в правиле указывает на используемую таблицу (например, `-t mangle`). С ключом `-t` можно указывать следующие основные таблицы:

- `filter` (см. п. 7.4.1);
- `nat` (см. п. 7.4.2);
- `mangle` (см. п. 7.4.3).

При желании можно добавлять свои таблицы.

7.4.1. filter

`filter` таблица по умолчанию (если не передана опция `-t`), используется главным образом для фильтрации пакетов.

Содержит три встроенных цепочки:

- INPUT – цепочка предназначена для обработки входящих пакетов, направляемых локальным приложениям. Здесь производится фильтрация входящего трафика. Помните, что все входящие пакеты, адресованные

непосредственно МЭ, проходят через эту цепочку, независимо от того с какого интерфейса они поступили. (действия/цели, применяемые к пакетам ACCEPT, DROP (см. п. 7.2));

- FORWARD – цепочка используется для фильтрации пакетов, которые идут транзитом на другой хост. Вся фильтрация транзитного трафика должна выполняться здесь. Не забывайте, что через эту цепочку проходит траффик в обоих направлениях, обязательно учитывайте это обстоятельство при написании правил фильтрации;
- OUTPUT – цепочка используется для фильтрации исходящих пакетов, сгенерированных локальными приложениями, процессами МЭ.

Действие может быть именем цепочки, определенной пользователем, или одной из специальных целей: ACCEPT, DROP, QUEUE, RETURN (см. п. 7.2), также здесь можно выполнить (см. п. 7.3):

1) LOG (см. п. 7.3.2);
2) LOGMARK (см. п. 7.3.3);
3) ULOG (см. п. 7.3.4);
4) NFLOG (см. п. 7.3.5);
5) NETFLOW (см. п. 7.3.13);
6) TARPIT (см. п. 7.3.14);
7) DELUDE (см. п. 7.3.15);
8) CHAOS (см. п. 7.3.16).

7.4.2. nat

`nat` таблица преобразования сетевых адресов NAT (Network Address Translation). Как уже упоминалось ранее, только первый пакет из потока проходит через цепочки этой таблицы, трансляция адресов или маскировка применяются ко всем последующим пакетам в потоке автоматически. Для этой таблицы характерны действия:

- DNAT (Destination Network Address Translation) – производит преобразование адресов назначения в заголовках пакетов. Другими словами,

этим действием производится перенаправление пакетов на другие адреса, отличные от указанных в заголовках пакетов (см. п. 7.3.8);

- SNAT (Source Network Address Translation) – используется для изменения исходных адресов пакетов. С помощью этого действия можно скрыть структуру локальной сети, а заодно и разделить единственный внешний IP-адрес между компьютерами локальной сети для выхода в Интернет. В этом случае МЭ, с помощью SNAT, автоматически производит прямое и обратное преобразование адресов, тем самым давая возможность выполнять подключение к серверам в Интернете с компьютеров в локальной сети (см. п. 7.3.7);
- MASQUERADE (см. п. 7.3.9) – (маскировка) применяется в тех же целях, что и SNAT, но в отличие от последней, MASQUERADE дает более сильную нагрузку на систему. Происходит это потому, что каждый раз, когда требуется выполнение этого действия – производится запрос IP-адреса для указанного в действии сетевого интерфейса, в то время как для SNAT IP-адрес указывается непосредственно. Однако, благодаря такому отличию, MASQUERADE может работать в случаях с динамическим IP-адресом, т. е. когда подключение к Интернету выполняется, например, через PPP, SLIP или DHCP.

Доступные действия по умолчанию для каждой из встроенных цепочек – ACCEPT, DROP, QUEUE, RETURN (см. п. 7.2), и некоторые специальные действия и цели для каждой (см. п. 7.3).

Встроенные цепочки:

- PREROUTING – используется для трансляции (для транзитных пакетов), преобразования (для пакетов для локальных приложений) сетевых адресов DNAT. SNAT выполняется позднее, в другой цепочке. Любого рода фильтрация в этой цепочке может производиться только в исключительных случаях (доступные действия/цели DNAT, REDIRECT, NETMAP);

- INPUT – здесь осуществляется преобразование сетевого адреса пакета перед тем как он будет передан локальному приложению (доступные действия/цели DNAT, SNAT, REDIRECT);
- OUTPUT – используется для трансляции сетевых адресов (NAT) в пакетах, исходящих от локальных процессов МЭ (доступные действия/цели DNAT, SNAT, REDIRECT);
- POSTROUTING – предназначена в первую очередь для SNAT. Не используйте ее для фильтрации без особой на то необходимости. Однако и здесь можно останавливать пакеты, применяя политику по умолчанию DROP (для пакетов от локальных процессов), здесь же выполняется и маскарадинг (MASQUERADE) (для транзитных пакетов) (доступные действия/цели SNAT, MASQUERADE, REDIRECT, NETMAP).

7.4.3. mangle

Таблица предназначена, главным образом для внесения изменений в заголовки пакетов, т. е. в этой таблице можно устанавливать биты TOS (Type Of Service) и т. д.

ВНИМАНИЕ!
В этой таблице не следует производить любого рода фильтрацию, маскировку или преобразование адресов (DNAT, SNAT, MASQUERADE).

Доступные действия по умолчанию для каждой из встроенных цепочек – ACCEPT, DROP, QUEUE, RETURN (см. п. 7.2), и некоторые специальные действия и цели для каждой (см. п. 7.3).

Встроенные цепочки:

- PREROUTING – обычно эта цепочка используется для внесения изменений в заголовок пакета (транзитного или для локального приложения). Например: для изменения битов TOS у транзитных пакетов, для установки битов TOS для пакетов локальных приложений (доступные действия/цели TOS, DSCP, TTL, TCPMSS, MARK, CONNMARK, CLASSIFY, TCPMSS, TCPOPTSTRIP, TPROXY, ECN);

- INPUT – здесь вносятся изменения в заголовок пакета перед тем как он будет передан локальному приложению (доступные действия/цели TOS, DSCP, TTL, TCPMSS, MARK, CONNMARK, CLASSIFY, TCPMSS, TCPOPTSTRIP, TPROXY, ECN);
- FORWARD – после PREROUTING обработки транзитный пакет попадает в цепочку FORWARD таблицы `mangle`, которая должна использоваться только в исключительных случаях, когда необходимо внести некоторые изменения в заголовок пакета между двумя точками принятия решения о маршрутизации (доступные действия/цели TOS, DSCP, TTL, TCPMSS, MARK, CONNMARK, CLASSIFY, TCPMSS, TCPOPTSTRIP, TPROXY, ECN);
- OUTPUT – здесь производится внесение изменений в заголовок пакетов, исходящих от локальных процессов. Выполнение фильтрации в этой цепочке может иметь негативные последствия (доступные действия/цели TOS, DSCP, TTL, TCPMSS, MARK, CONNMARK, CLASSIFY, TCPMSS, TCPOPTSTRIP, TPROXY, ECN);
- POSTROUTING – предназначена для внесения изменений в заголовок пакета (как транзитного, так и созданного локальными процессами) уже после того как принято последнее решение о маршрутизации (доступные действия/цели TOS, DSCP, TTL, TCPMSS, MARK, CONNMARK, CLASSIFY, TCPMSS, TCPOPTSTRIP, TPROXY, ECN).

7.4.4. raw

Эта таблица используется в основном для настройки исключений из отслеживания соединений в сочетании с целью NOTRACK (см. п. 7.3.26). Он регистрируется на перехватчиках netfilter с более высоким приоритетом и поэтому вызывается перед ip_conntrack или любыми другими IP-таблицами.

Доступные действия по умолчанию для каждой из встроенных цепочек – ACCEPT, DROP, QUEUE, RETURN (см. п. 7.2), и некоторые специальные действия и цели для каждой (см. п. 7.3).

Встроенные цепочки:

- PREROUTING – для пакетов, поступающих через любой сетевой интерфейс (доступные действия/цели ACCEPT, DROP, NOTRACK (см. п. 7.3.26), CT (см. п. 7.3.27));
- OUTPUT – для пакетов, сгенерированных локальными процессами (доступные действия/цели ACCEPT, DROP, NOTRACK (см. п. 7.3.26), CT (см. п. 7.3.27)).

7.5. Опции

Опции iptables могут быть разделены на несколько групп.

7.5.1. Команды

В таблице 11 приводится список команд и правила их использования. Обычно предполагается одно из двух действий – добавление нового правила в цепочку или удаление существующего правила из той или иной таблицы.

В правиле может быть указана только одна из этих опций, если противоположное явно не указано в описании. Длинные команды можно сокращать до первых букв.

Команда должна быть указана всегда. Список доступных команд можно просмотреть в консоли с помощью команды `iptables -h`. Некоторые команды могут использоваться совместно с дополнительными ключами. В п. 7.5.2 приводится список дополнительных ключей и описывается результат их действия. При этом стоит отметить, что здесь не приводятся дополнительные ключи, которые используются при построении дополнительных критериев (matches см. п. 7.6.3) или действий/целей (targets п. 7.3).

Таблица 11 – Команды

Ключ	Описание
-A, --append	-A, --append цепочка критерии_сравнения Добавить одно или несколько правил в конец указанной цепочки. Если имя источника и/или назначения соответствует нескольким адресам, правило будет добавлено для всех возможных комбинаций адресов. **Пример:** iptables -A INPUT -p icmp --icmp-type timestamp-request -j DROP
-D, --delete	-D, --delete цепочка критерии_сравнения -D, --delete цепочка порядковый_номер_правила Удаление правила из цепочки. Если задается критерий сравнения, то удаляется правило, которое имеет в себе этот критерий, если задается номер правила, то будет удалено правило с заданным номером. Счет правил в цепочках начинается с 1. **Пример:** iptables -D INPUT --dport 80 -j DROP, iptables -D INPUT 1
-E, --rename-chain	-E, --rename-chain старое_имя новое_имя Команда выполняет переименование пользовательской цепочки. В примере цепочка allowed будет переименована в цепочку disallowed. Эти переименования не изменяют порядок работы, а носят только косметический характер. **Пример:** iptables -E allowed disallowed
-F, --flush	-F, --flush цепочка Сброс (удаление) всех правил из заданной цепочки (таблицы). Если имя цепочки и таблицы не указывается, то удаляются все правила, во всех цепочках. Если не указана таблица ключом -t (--table), то очистка цепочек производится только в таблице filter. **Пример:** iptables -F INPUT Удаление всех существующих правил фильтрации: iptables -F
-h	Помощь, выдает короткое описание синтаксиса команд iptables.
-I, --insert	-I, --insert цепочка номер_правила Вставляет новое правило в цепочку. Число, следующее за именем цепочки указывает номер правила, перед которым нужно вставить новое правило, т. е. задает номер для вставляемого правила. В примере указывается, что данное правило должно быть 1-м в цепочке INPUT. **Пример:** iptables -I INPUT 1 --dport 80 -j ACCEPT
-L, --list	-L, --list цепочка Вывод списка правил в заданной цепочке. Если имя цепочки не указывается, то выводится список правил для всех цепочек. Формат вывода зависит от наличия дополнительных ключей в команде, например -n, -v, и пр. Если не указана конкретная таблица, то по умолчанию используется filter.

Окончание таблицы 11

Ключ	Описание

	Примеры:
	`iptables -L INPUT`
	Вывод списка правил NAT:
	`iptables -t nat -n –L`
	Просмотр списка созданных правил:
	`iptables -L –n -v`

`-N, --new-chain`	`-N, --new-chain цепочка`
	Создается новая цепочка с заданным именем. Имя цепочки должно быть уникальным и не должно совпадать с зарезервированными именами цепочек и действий (такими как DROP, REJECT и т. п.).
	Пример:
	`iptables -N allowed`

`-P, --policy`	`-P, --policy цепочка действие`
	Задает политику по умолчанию для заданной цепочки. Политика по умолчанию определяет действие, применяемое к пакетам не попавшим под действие ни одного из правил в цепочке. В качестве политики по умолчанию допускается использовать DROP и ACCEPT.
	Пример:
	`iptables -P INPUT DROP`

`-R, --replace`	`-R, --replace цепочка номер_правила критерии_сравнения`
	Эта команда заменяет одно правило другим. В основном она используется во время отладки новых правил.
	Пример:
	`iptables -R INPUT 1 -s 192.168.0.1 -j DROP`

`-S, --list-rules`	`-S, --list-rules цепочка`
	Выводит все правила в выбранной цепочке. Если ни одна цепочка не выбрана, то выводятся все сохраненные. Если не указана конкретная таблица, то по умолчанию используется filter.
	Пример:
	Вывод на экран таблицы фильтрации:
	`iptables –S`

`-X, --delete-chain`	`-X, --delete-chain цепочка`
	Удаление заданной цепочки из заданной таблицы. Удаляемая цепочка не должна иметь правил и не должно быть ссылок из других цепочек на удаляемую цепочку. Если имя цепочки не указано, то будут удалены все цепочки в заданной таблице кроме встроенных.
	Пример:
	`iptables -X allowed`

`-Z, --zero`	`-Z, --zero цепочка`
	Обнуление всех счетчиков в заданной цепочке. Если имя цепочки не указывается, то подразумеваются все цепочки. При использовании ключа -v совместно с командой `–L`, на вывод будут поданы и состояния счетчиков пакетов, попавших под действие каждого правила. Допускается совместное использование команд `–L` и `–Z`. В этом случае будет выдан сначала список правил со счетчиками, а затем произойдет обнуление счетчиков.
	Пример:
	`iptables -Z INPUT`

7.5.2. Дополнительные ключи

У команд из п. 7.5.1 имеются дополнительные ключи (таблица 12), используемые вместе с ними в правилах.

Т а б л и ц а 12 – Дополнительные ключи

Ключ	Описание
`-v, --verbose`	Используется для повышения информативности вывода и, как правило, используется совместно с командой `--list`. В случае использования с командой `--list`, в вывод этой команды включаются так же имя интерфейса, имя интерфейса, параметры правил, маски TOS, счетчики пакетов и байт для каждого правила. Формат вывода счетчиков предполагает вывод кроме цифр числа еще и символьные множители K (x1000), M (x1,000,000) и G (x1,000,000,000). Для того, чтобы заставить команду `--list` выводить полное число (без употребления множителей) требуется применять ключ `-x`, который описан ниже. Если ключ используется с командами `--append`, `--insert`, `--delete` или `--replace`, то будет выведен подробный отчёт о произведенной операции. Команды, с которыми используется (см. таблицу 11): `--list`, `--append`, `--insert`, `--delete`, `--replace`.
`-n, --numeric`	Выводит IP-адреса и номера портов в числовом виде предотвращая попытки преобразовать их в символические имена. Команды, с которыми используется (см. таблицу 11): `-L/--list`
`-x, --exact`	Для всех чисел в выходных данных выводит их точные значения без округления и без использования множителей K, M, G. Команды, с которыми используется (см. таблицу 11): `-L/--list` (не применим с другими командами).

`--line-numbers`	При перечислении правил включает режим добавления номеров строк, соответствующие позиции этого правила в цепочке.
	Команды, с которыми используется (см. таблицу 11): `-L/--list`
	Пример:
	Вывода таблицы правил фильтрации цепочки INPUT:
	`iptables -L INPUT -n --line-numbers`
	или вывод статистики правил iptables и счетчиков обработанных пакетов в цепочке INPUT:
	`iptables -nvL INPUT --line-numbers`
	Удаление правила фильтрации из таблицы правил, например, в строке 2:
	`iptables -D INPUT 2`

`--modprobe=команда`	Определяет команду загрузки модуля ядра (при добавлении правил в цепочку). Может использоваться в случае, когда модули ядра находятся вне стандартного пути поиска. Этот ключ может использоваться с любой командой.

7.6. Критерии пакетов

В этом подразделе рассматриваются критерии выделения пакетов, используемые в фильтре iptables.

7.6.1. Общие критерии

В таблице 13 приведены общие критерии, которые могут использоваться в любых правилах, используются с командами delete, insert, replace и append (см. п. 7.5.1).

Т а б л и ц а 13 – Общие критерии

Критерий	Пояснения

`-p, --protocol`	`-p, --protocol [!] протокол`
	Используется для указания типа протокола. Основные типы протоколов: TCP, UDP и ICMP. Список остальных протоколов можно посмотреть в п. 7.9.2 и файле `/etc/protocols`. Прежде всего, в качестве имени протокола в данный критерий можно передавать три вышеупомянутых протокола, а также ключевое слово `ALL` (эквивалент число 0). В качестве протокола допускается передавать число – номер протокола. Соответствия между номерами протоколов и их именами можно посмотреть в файле `/etc/protocols`. Для логической инверсии критерия, перед именем протокола (списком протоколов) используется символ !, например, `--protocol ! tcp` подразумевает пакеты протоколов, UDP и ICMP.
	Пример:
	`iptables -A INPUT -p tcp`
	Создание правил фильтрации сетевого трафика по сетевому адресу хоста отправителя для протокола icmp:
	`iptables -I INPUT -p icmp -s IP_отправителя -j ACCEPT`
	`iptables -A INPUT -p icmp -j DROP`
	`iptables -I OUTPUT -p icmp -j ACCEPT`

`-s, --src, --source`	`-s, --src, --source [!] адрес[/маска]`
	IP-адрес(а) источника пакета. Адрес источника может указываться без маски или префикса (например, 192.168.1.1), тогда подразумевается единственный IP-адрес, может быть сетевое имя, имя хоста, а можно указать диапазон адресов в виде address/mask, например, как 192.168.0.0/255.255.255.0, или определяя диапазон адресов 192.168.0.0/24.
	Символ `!`, установленный перед адресом, также означает логическое отрицание, т. е.
	`--source ! 192.168.0.0/24` означает любой адрес кроме адресов 192.168.0.x.
	Пример:
	`iptables -A INPUT -s 192.168.1.1`

Продолжение таблицы 13

Критерий	Пояснения
	Добавление правила перенаправления сетевого трафика по сетевому адресу: `iptables -A FORWARD -s <IP-адрес АРМ> -d <IP-адрес сервера> -j ACCEPT`

`-d, --dst, --destination`	`-d, --dst, --destination [!] адрес[/маска]`
	IP-адрес(а) получателя. Имеет синтаксис схож с критерием `--source`, за исключением того, что подразумевает адрес места назначения. Правила указания адресов аналогичные.
	Символ `!` используется для логической инверсии критерия.
	Примеры:
	`iptables -A INPUT -d 192.168.1.1`
	Запретить все исходящие пакеты на хост 192.168.1.95:
	`iptables -A OUTPUT -d 192.168.156.156 -j DROP`
	Запретить доступ к ресурсу vk.com:
	`iptables -A OUTPUT -d vk.com -j REJECT`

`-i, --in-interface`	`-i, --in-interface [!] имя`
	Указывает интерфейс, с которого был получен пакет.
	Использование этого критерия допускается только в цепочках INPUT, FORWARD и PREROUTING, в любых других случаях будет вызывать сообщение об ошибке.
	При отсутствии этого критерия предполагается любой интерфейс, что равносильно использованию критерия `-i +`. Как и прежде, символ `!` инвертирует результат совпадения. Если имя интерфейса завершается символом +, то критерий задает все интерфейсы, начинающиеся с заданной строки, например `-i PPP+` обозначает любой PPP интерфейс, а запись `-i ! eth+` – любой интерфейс, кроме любого eth.
	Примеры:
	`iptables -A INPUT -i eth0`
	`iptables -A INPUT -i enp0s3 -j DROP`

`-o, --out-interface`	`-o, --out-interface [!] имя`
	Задает имя выходного интерфейса, через который отправляется пакет. Этот критерий допускается использовать только в цепочках OUTPUT, FORWARD и POSTROUTING, в противном случае будет генерироваться сообщение об ошибке. При отсутствии этого критерия предполагается любой интерфейс, что равносильно использованию критерия `-o +`. Как и прежде, символ `!` инвертирует результат совпадения. Если имя интерфейса завершается символом +, то критерий задает все интерфейсы, начинающиеся с заданной строки, например `-o eth+` обозначает любой eth интерфейс, а запись `-o ! eth+` - любой интерфейс, кроме любого eth.
	Примеры: `iptables -A FORWARD -o eth0`
	Добавление правила перенаправления сетевого трафика:
	по интерфейсу объекта (на уровне сетевого адреса):
	`iptables -A FORWARD -i <имя входящего сетевого интерфейса> -o <имя исходящего сетевого интерфейса> -j ACCEPT`

Окончание таблицы 13

Критерий	Пояснения

`-f`, `--fragment`	`[!] -f, --fragment`
	Означает, что это правило будет применяться ко второму и последующим фрагментам фрагментированного пакета. Так как у фрагмента невозможно определить номер порта источника или цели, а также тип ICMP, такие пакеты не обрабатываются правилами, содержащими номера портов или тип ICMP. Если перед флагом `-f` указан "!", то правило будет применяться только к первым фрагментам фрагментированных пакетов и/или к нефрагментированным пакетам.
	В ГИ МЭ ИВК КОЛЬЧУГА-К для `--fragments`:
	-  – первые фрагменты фрагментированных;
	-  – нефрагментированные пакеты.
	Пример:
	`iptables -A INPUT -f`
	Установить правила запрета отправки фрагментированных пакетов:
	`iptables -A INPUT -f -j DROP`
	`iptables -A OUTPUT -f -j DROP`
	`iptables -A FORWARD -f -j DROP`

7.6.2. Неявные критерии

В таблице 14 приведены специфичные неявные критерии, которые применяются только к типу пакетов, указанных сначала в общем критерии (в таблице см. предварительный критерий), например, `-p tcp` (см. таблицу 13) применяется к TCP-пакетам, также в таблице приведены неявные критерии, которые применяются только к UDP, ICMP, SCTP и DCCP пакетам.

Т а б л и ц а 14 – Неявные критерии

Критерий	Пояснения
`-sport`, `--source-port`	`[!] --source-port,--sport port[:port]` Исходный порт, с которого был отправлен пакет. В качестве параметра может указываться номер порта или название сетевой службы. Соответствие имен сервисов и номеров портов можно найти в файле `/etc/services` при указании номеров портов правила отрабатывают несколько быстрее. Номера портов могут задаваться в виде интервала из минимального и максимального номеров, например, `--source-port 22:80`. Если опускается минимальный порт, т.е. когда критерий записывается как `--source-port :80`, то в качестве начала диапазона принимается число 0. Если опускается максимальный порт, т.е. когда критерий записывается как `--source-port 22:`, то в качестве конца диапазона принимается число 65535.

Продолжение таблицы 14

Критерий	Пояснения

	Аналогично символ ! используется для инверсии. Так критерий `--source-port ! 22` подразумевает любой порт, кроме 22. Инверсия может применяться и к диапазону портов, например `--source-port ! 22:80`. За дополнительной информацией обращайтесь к описанию критерия multiport (см. таблицу 15).
	Предварительный критерий (см. таблицу 13 и см. п. 7.9.2):
	`-p [tcp, udp, sctp, dccp]`
	Пример:
	`iptables -A INPUT -p tcp --sport 22`
	Заблокировать все входящие запросы порта 80:
	`iptables -A INPUT -p tcp --dport 80 -j DROP`

`--dport`,
`--destination-port`
	`[!] --destination-port,--dport port[:port]`
	Порт, на который адресован пакет. Аргументы задаются в том же формате, что и для `--source-port`.
	Пример:
	`iptables -A INPUT -p tcp --dport 22`
	Установка правил запрета отправки на все порты, кроме определенного порта:
	`iptables -A OUTPUT -p tcp --dport 21 -j ACCEPT`
	`iptables -A OUTPUT -p all -j DROP`
	`iptables -A INPUT -p tcp --sport 21 -j ACCEPT`
	`iptables -A INPUT -p all -j DROP`
	Разрешить подключения по HTTP:
	`iptables -A INPUT -p tcp --dport 80 -j ACCEPT`
	Разрешить подключения по SSH:
	`iptables -A INPUT -p tcp -i eth0 --dport 22 -j ACCEPT`
	Разрешить получать данные от DHCP-сервера:
	`iptables -A INPUT -p UDP --dport 68 --sport 67 -j ACCEPT`
	Разрешить исходящие HTTP, FTP, DNS, SSH, SMTP:
	`iptables -A OUTPUT -p TCP -o eth0 --dport 443 -j ACCEPT`
	`iptables -A OUTPUT -p TCP -o eth0 --dport 80 -j ACCEPT`
	`iptables -A OUTPUT -p TCP -o eth0 --dport 53 -j ACCEPT`
	`iptables -A OUTPUT -p UDP -o eth0 --dport 53 -j ACCEPT`
	`iptables -A OUTPUT -p TCP -o eth0 --dport 25 -j ACCEPT`
	`iptables -A OUTPUT -p TCP -o eth0 --dport 22 -j ACCEPT`
	`iptables -A OUTPUT -p TCP -o eth0 --dport 21 -j ACCEPT`
	Разрешить mysql для локальных пользователей:
	`iptables -I INPUT -p tcp --dport 3306 -j ACCEPT`

`--tcp-flags`	`[!] --tcp-flags mask comp` Выделяет пакеты, только с установленными TCP-флагами. Пакет считается удовлетворяющим критерию, если из перечисленных флагов в первом списке `mask` (проверяемые) в единичное состояние установлены флаги из второго списка `comp` (установленные). В качестве аргументов критерия могут выступать флаги (см. описание в п. 7.9.5): SYN, ACK, FIN, RST, URG, PSH, а также зарезервированные идентификаторы ALL и NONE. ALL означает ВСЕ флаги, а NONE – НИ ОДИН флаг.

Продолжение таблицы 14

Критерий	Пояснения
	Так, критерий `--tcp-flags ALL NONE` означает, что все флаги в пакете должны быть сброшены. Как и ранее, символ ! означает инверсию критерия. Имена флагов в каждом списке должны разделяться запятыми, пробелы служат для разделения списков. Предварительный критерий (см. таблицу 13): `-p tcp` **Пример:** `iptables -p tcp --tcp-flags SYN,FIN,ACK SYN`

`--syn`	`[!] --syn`
	Выделяет пакеты, только с установленным флагом SYN и сброшенными битами ACK,RST и FIN. Этот критерий аналогичен критерию: `--tcp-flags SYN,RST,ACK,FIN SYN`.
	Такие пакеты используются для открытия соединения TCP. Блокировка таких пакетов предотвратит входящие TCP-соединения, но исходящие TCP-соединения не будут затронуты. Как и ранее, допускается инвертирование критерия символом !. Так критерий `! --syn` означает – «все пакеты, не являющиеся запросом на соединение».
	Предварительный критерий (см. таблицу 13): `-p tcp`
	Пример:
	`iptables -p tcp --syn`

`--tcp-option`	`[!] --tcp-option number` Удовлетворяющим условию данного критерия будет считаться пакет, TCP параметр которого равен заданному числу (number – см. таблицу 17). TCP Option – это часть заголовка пакета. Как и ранее, допускается использование флага инверсии условия `!`. Предварительный критерий (см. таблицу 13): `-p tcp` **Пример:** `iptables -p tcp --tcp-option 16`
`--chunk-types`	`[!] --chunk-types {all|any|only} chunktype[:flags] [...]` В верхнем регистре указывает на то, что удовлетворяющим условию данного критерия будет считаться пакет, если флаг установлен, в нижнем регистре указывает на соответствие тех пакетов, в которых флаг не установлен. Возможные типы блоков (chunktype): DATA INIT INIT_ACK SACK HEARTBEAT HEARTBEAT_ACK ABORT SHUTDOWN SHUTDOWN_ACK ERROR COOKIE_ECHO COOKIE_ACK ECN_ECNE ECN_CWR SHUTDOWN_COMPLETE ASCONF ASCONF_ACK Доступные флаги (flags): DATA U B E u b e ABORT T t SHUTDOWN_COMPLETE T t (нижний регистр означает, что флаг должен быть «выключен», верхний – «включен»). Предварительный критерий (см. таблицу 13 и п. 7.9.2): `-p stcp`

Окончание таблицы 14

Критерий	Пояснения

	Примеры:
	`iptables -A INPUT -p sctp --dport 80 -j DROP`
	`iptables -A INPUT -p sctp --chunk-types any DATA,INIT -j DROP`
	`iptables -A INPUT -p sctp --chunk-types any DATA:Be -j ACCEPT`

`--dccp-types`	`[!] --dccp-types mask` Удовлетворяющим условию данного критерия будет считаться пакет, если тип пакета DCCP является одним из mask. mask – это список типов пакетов, разделенных запятыми. Типы пакетов: REQUEST RESPONSE DATA ACK DATAACK CLOSEREQ CLOSE RESET SYNCACK SYNCACK INVALID. Предварительный критерий (см. таблицу 13 и п. 7.9.2): `-p dccp`

`--icmp-type`	`[!] --icmp-type {type[/code]|typename}`
	Тип сообщения ICMP определяется номером или именем. Числовые значения определяются в RFC 792. Чтобы получить список имен ICMP значений выполните команду
	`iptables -p icmp –help`, также см. п. 7.9.3.
	Символ ! инвертирует критерий, например, все типы сообщения кроме типа 9: `--icmp-type ! 9`.
	Предварительный критерий (см. таблицу 13 и п. 7.9.2): `-p icmp`
	Примеры:
	`iptables -A INPUT -p icmp --icmp-type 8`
	Разрешить входящие эхо-запросы:
	`iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT`

7.6.3. Дополнительные критерии (matches)

Перед использованием этих дополнительных критериев (таблица 15), они должны быть загружены явно, с помощью ключа `-m` или `--match`. Так, например, если использовать критерии `state`, то необходимо явно указать это в строке правила: `-m state` левее используемого критерия.

Символ ! инвертирует указанный критерий.

Т а б л и ц а 15 – Дополнительные критерии (matches)

Критерий	Описание ключей/опций
State	[!] --state состояние_соединения Проверяется признак состояния соединения. Можно указывать состояния через запятую (состояние_соединения): INVALID, ESTABLISHED, NEW, RELATED, UNTRACKED (см. п. 7.9.1). **Примеры**: iptables -A INPUT -m state --state RELATED,ESTABLISHED Добавление правил логирования (в /var/log/messages) и фильтрации пакетов со статусом INVALID и отбрасывания таких пакетов: iptables -A INPUT -m state --state INVALID -j LOG --log-prefix "INVALLID:" iptables -A INPUT -m state --state INVALID -j DROP Разрешить rsync с определенной сети: iptables -A INPUT -i eth0 -p tcp -s 192.168.1.0/24 --dport 873 -m state --state NEW,ESTABLISHED -j ACCEPT iptables -A OUTPUT -o eth0 -p tcp --sport 873 -m state --state ESTABLISHED -j ACCEPT Разрешить IMAP/IMAP2 трафик: iptables -A INPUT -i eth0 -p tcp --dport 143 -m state --state NEW,ESTABLISHED -j ACCEPT iptables -A OUTPUT -o eth0 -p tcp --sport 143 -m state --state ESTABLISHED -j ACCEPT
Conntrack	В сочетании с отслеживанием соединения позволяет получить доступ к состоянию отслеживания соединения для пакета/соединения. Расширение подразумевает наличие одного из приведенных далее ключей.
	[!] --ctstate список_состояний Список состояний подключения, разделенных запятыми. Возможные состояния перечислены в п. 7.9.1. Пример: iptables -I INPUT -m conntrack --ctstate INVALID -j DROP
	[!] --ctproto proto Проверяет принадлежность пакета к протоколу транспортного уровня, указанному номером или символьным именем. Возможные протоколы перечислены в п. 7.9.2.
	[!] --ctorigsrc address[/mask] [!] --ctorigdst address[/mask] [!] --ctreplsrc address[/mask] [!] --ctrepldst address[/mask] Проверяет соответствие исходного (нетранслированного) адреса отправителя/получателя указанному адресу или диапазону адресов.

Продолжение таблицы 15

Критерий	Описание ключей/опций

	`[!] --ctorigsrcport port`
	`[!] --ctorigdstport port`
	`[!] --ctreplsrcport port`
	`[!] --ctrepldstport port`
	Проверяет соответствие исходного (нетранслированного) порта TCP/UDP/SCTP или ключа GRE указанному отправителю/получателю.

	`[!] --ctstatus список_состояний` Проверяет соответствие пакета одному или группе внутренних состояний conntrack. Возможные значения: - NONE – ни один из перечисленных; - EXPECTED – ожидается соединение; - SEEN_REPLY – видны пакеты, которые идут в обоих направлениях; - ASSURED – не должно быть преждевременного истечения срока на вход; - CONFIRMED – соединение подтверждено.
	`[!] --ctexpire time[:time]` Проверяет соответствие оставшегося для пакета времени жизни заданному значению или диапазону.
	`--ctdir {ORIGINAL|REPLY}` Проверяет соответствие направления пакета указанному. Если флаг не указан, подразумеваются пакеты в обоих направлениях.

`Iprange`	Возможные опции:
	`[!] --src-range from[-to]`
	Задает диапазон IP-адресов отправителя.
	`[!] --dst-range from[-to]`
	Задает диапазон IP-адресов получателя.

`String`	Позволяет выполнять фильтрацию пакетов, основываясь на анализе содержимого области данных пакета. Расширение подразумевает наличие одного из приведенных далее ключей.

	`--algo {bm|kmp}`
	Алгоритм сравнения/поиска по содержимому пакета bm = Boyer-Moore, kmp = Knuth-Pratt-Morris.
	Пример:
	`iptables -A INPUT -m string --algo bm --string "<script" -j DROP`

	`--from позиция` Установка позиции, с которой начинается поиск любого совпадения. Если не задано, то устанавливается по умолчанию – 0.

Продолжение таблицы 15

Критерий	Описание ключей/опций
	`--to позиция` Установка позиции, при достижении которой следует прекращать поиск. Если не задано, то устанавливается размер пакета.
	`--icase` Указание игнорировать регистр (по умолчанию: 0).
	`[!] --string string` Установка определенной последовательности символов в пакете.

	`[!] --hex-string string`
	Установка определенной последовательности символов в пакете (в шестнадцатеричном представлении).
	Пример:
	Добавление правил для логирования пакетов с уровнями конфиденциальности 1, 2, 3 соответственно:
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 02|' -j LOG --log-prefix "Security label level 1"`
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 04|' -j LOG --log-prefix "Security label level 2"`
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 06|' -j LOG --log-prefix "Security label level 3"`
	Добавление правил для блокирования пакетов с уровнями безопасности с уровнями конфиденциальности 1, 2, 3 соответственно:
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 02|' -j DROP`
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 04|' -j DROP`
	`iptables -A FORWARD -m string --algo kmp --hex-string '|82 04 ab 06|' -j DROP`

`multiport`	Позволяет указывать в тексте правила несколько портов и диапазонов портов. Расширение подразумевает наличие одного из приведенных далее ключей.

	`[!] --source-ports,--sports port[,port|,port:port]...`
	Служит для указания списка исходящих портов. С помощью данного критерия можно указать до 15 различных портов. Названия портов в списке должны отделяться друг от друга запятыми, пробелы в списке не допустимы. Диапазон задается через двоеточие. Данное расширение может использоваться только совместно с критериями `-p tcp` или `-p udp`. Главным образом используется как расширенная версия обычного критерия `--source-port`.
	Пример:
	`iptables -A INPUT -p tcp -m multiport --source-port 22,53,80,110`

Продолжение таблицы 15

Критерий	Описание ключей/опций
	`[!] --destination-ports,--dports port[,port|,port:port]...` Служит для указания списка входных портов. Формат задания аргументов аналогичен `-m multiport --source-port`.
	`[!] --ports port[,port|,port:port]...` Данный критерий проверяет как исходящий, так и входящий порт пакета. Формат аргументов аналогичен критериям `--source-port` и `--destination-port`. Обратите внимание на то, что данный критерий проверяет порты в обоих направлений, т. е. если указано `-m multiport --port 80`, то под данный критерий попадают пакеты, идущие с порта 80 и на порт 80.

`mark`	`[!] --mark value[/mask]`
	Критерий производит проверку пакетов, которые были предварительно «помечены». Метки устанавливаются действием MARK (см. п. 7.4.3). Все пакеты, проходящие через netfilter имеют специальное поле mark, состояние этого поля вместе с пакетом в сеть не передается. Поле mark является целым беззнаковым, таким образом можно создать не более 4294967296 различных меток. Допускается использовать маску с метками. В данном случае критерий будет выглядеть подобным образом: `--mark 1/1`. Если указывается маска, то выполняется логическое И (AND) метки и маски.
	Пример:
	`iptables -t mangle -A INPUT -m mark --mark 1`

`connmark`	`[!] --mark value[/mask]` Критерий производит проверку пакетов, которые соответствует полю метки netfilter, связанному с соединением (может быть установлено с помощью цели CONNMARK (см. п. 7.4.3)). Сопоставляет пакеты в соединениях с заданным значением метки (если маска указана, перед сравнением то она объединяется оператором логическое И (AND) с меткой).
`connbytes`	`[!] --connbytes from[:to]` Критерий производит сопоставление по количеству байтов или пакетов, переданных на данный момент соединением (или одним из двух потоков, составляющих соединение), или по среднему количеству байтов на пакет). Основное использование обнаружение и пометка долгоживущих загрузок, для планирования и управления трафиком. Переданные байты на каждое соединение также можно просмотреть с помощью `conntrack -L` и получить доступ через ctnetlink. Расширение подразумевает наличие одного из приведенных далее ключей.

Продолжение таблицы 15

Критерий	Описание ключей/опций
	`--connbytes-dir {original|reply|both}` – определяет тип рассматриваемых пакетов. `--connbytes-mode {packets|bytes|avgpkt}` – определяет проверять ли количество пакетов, количество переданных байтов или средний размер (в байтах) всех полученных на данный момент пакетов. Обратите внимание, что когда «both» используются вместе с «avgpkt» и данные идут (в основном) только в одном направлении (например, HTTP), средний размер пакета будет примерно половина фактических пакетов данных. **Пример:** Добавление правила отбрасывания входящих и исходящих пакетов больше определенного размера: `iptables -A INPUT -i <сетевой_интерфейс> -m connbytes --connbytes 10000:20000 --connbytes-dir both --connbytes-mode bytes -j DROP`
`limit`	Добавляя этот критерий тем самым устанавливается предельное число пакетов в единицу времени, которое способно пропустить правило. Правило, использующее это расширение, будет осуществлять проверку, пока не будет достигнут этот предел (если не используется флаг «!» – в этом случае подразумевается, что пакеты будут проходить правило только после превышения ограничения). Его можно использовать в сочетании с целью LOG (см. п. 7.3.2), например, для ограниченного ведения журнала. Расширение `-m limit` подразумевает наличие ключей `--limit` и `--limit-burst`. Если ключи не указаны, то они принимают значение по умолчанию.

	`--limit rate[/second|/minute|/hour|/day]`
	Устанавливается средняя скорость «освобождения» счетчика за единицу времени. В качестве аргумента указывается число пакетов и время. Допустимыми считаются следующие единицы измерения времени: `/second /minute /hour /day`. По умолчанию принято значение 3 пакета в час, или `3/hour`.
	Пример:
	`iptables -A INPUT -m limit --limit 3/hour`

	`--limit-burst number` Устанавливает максимальное значение числа `burst limit` для критерия `limit`. Это число увеличивается на единицу если получен пакет, подпадающий под действие данного правила, и при этом средняя скорость (задаваемая ключом `--limit`) поступления пакетов уже достигнута. Так происходит до тех пор, пока число `burst limit` не достигнет максимального значения, устанавливаемого ключом `--limit-burst`.

Продолжение таблицы 15

Критерий	Описание ключей/опций
	После этого правило начинает пропускать пакеты со скоростью, задаваемой ключом `--limit`. Значение по умолчанию принимается равным 5. **Пример:** `iptables -A INPUT -m limit --limit-burst 5`

`mac`	`[!] --mac-source address`
	Задает MAC-адрес сетевого хоста, передавшего пакет. MAC-адрес должен указываться в форме XX:XX:XX:XX:XX:XX. Как и ранее, символ ! используется для инверсии критерия, например, `--mac-source ! 00:00:00:00:00:01` означает – "пакет с любого хоста, кроме хоста, который имеет MAC-адрес 00:00:00:00:00:01" . Этот критерий доступен цепочках PREROUTING, FORWARD и INPUT и нигде более.
	Пример:
	`iptables -A INPUT -m mac --mac-source 00:00:00:00:00:01`
	Добавление правила перенаправления сетевого трафика по сетевому адресу по MAC-адресу:
	`iptables -A FORWARD -m mac --mac-source <mac-адрес АРМ1> -j ACCEPT`

`time`	`--datestart time`
	`--datestop time`
	Пропускать и останавливать пакеты при указанной дате (ISO 8601). Формат даты: `YYYY[-MM[-DD[Thh[:mm[:ss]]]]]`.

	`--timestart time`
	`--timestop time`
	Пропускать и останавливать пакеты при указанном времени (`hh:mm[:ss]`) (между 00:00:00 и 23:59:59).

	`[!] --monthdays value` Список дней, которые соответствуют месяцу, через запятую. Возможные дни: 1 до 31; по умолчанию все.
	`[!] --weekdays value` Список дней, которые соответствуют дню недели, через запятую. Возможные дни: Mon,Tue,Wed,Thu,Fri,Sat,Sun или с 1 по 7, по умолчанию все дни недели.
	`--kerneltz` Работа в зоне ядра timezone вместо UTC
`ndpi`	`--error` Критерий производит обнаружение пакетов, которые имеют ошибки в процессе.
	`--have-master` Критерий срабатывает для пакетов при обнаружении мастер протокола.

Продолжение таблицы 15

Критерий	Описание ключей/опций
	`--match-master` Критерий срабатывает для пакетов только для пакетов мастер протокола.
	`--match-proto` Критерий срабатывает для пакетов только для протокола.
	`--host str` Критерий срабатывает для пакетов только при совпадении с указанным именем хоста сервера.
	`--cert str` Критерий срабатывает для пакетов только при совпадении с указанным именем сертификата сервера.
	`--host-or-cert str` Критерий срабатывает для пакетов только при совпадении с указанным именем сертификата SSL сервера.
	`--protoname` или `--proto protoname[,protoname...]` Возможные варианты поддерживаемых протоколов приведены в п. 7.9.7, также возможно указание следующих значений: `--all` – соответствует выбору всех протоколов; `--unknown` – пакеты неизвестного протокола. Примеры: `iptables -A INPUT -m ndpi --SSH -j DROP` Добавление правила перенаправления сетевого трафика для HTTP-трафика: `iptables -A FORWARD -m ndpi --HTTP -j ACCEPT`

`set`	Этот модуль соответствует наборам IP, которые можно определить с помощью ipset (8) см. также п. 7.8.3 и п. 7.10.
	`[!] --match-set SETNAME flag [, flag] ...`
	где флаги – это разделенный запятыми список спецификаций источника (src) и/или получателя (dst), и их не может быть больше шести. Например, команда:
	`iptables -A FORWARD -m set --match-set test src, dst`
	будет соответствовать пакетам, для которых пара адресов источника и порта назначения может быть найдена в указанном наборе (SETNAME). Если тип набора указанного набора является одномерным (например, ipmap), то команда будет сопоставлять пакеты, для которых адрес источника может быть найден в указанном наборе.
	Виды флагов и их опций:
	`[!] --match-set SETNAME flags [--return-nomatch]`
	`[! --update-counters] [! --update-subcounters]`
	`[[!] --packets-eq value | --packets-lt value | --packets-gt value`
	`[[!] --bytes-eq value | --bytes-lt value | --bytes-gt value`
	Опцию `--match-set` можно заменить на `--set`, если она не конфликтует с опцией других расширений.

Окончание таблицы 15

Критерий	Описание ключей/опций
`ttl`	TTL используется для изменения содержимого поля Time To Live в IP-заголовке. Для данного действия предусмотрены ключи: - `[!] --ttl-eq value` – значение TTL должно быть равно указанному; - `--ttl-lt value` – значение TTL должно быть меньше указанного; - `--ttl-gt value` – значение TTL должно быть больше указанного.

`geoip`	`[!] --src-cc, --source-country country[,country...]`
	Критерий производит проверку входящих пакетов, которые относятся одной или нескольким указанным странам источникам.
	`[!] --dst-cc, --destination-country country[,country...]`
	Критерий производит проверку отправляемых пакетов, которые относятся одной или нескольким указанным странам получателям.
	Страна вводится в виде ISO3166 кода.
	Также см. список поддерживаемых стран в п. 7.9.6.

`helper`	`[!] --helper string` Критерий производит проверку пакетов, которые относятся к конкретному conntrack-helper. Например, строка "ftp" для пакетов, относящихся к ftp-сеансу на порту по умолчанию. Для других портов добавьте к значению -portnr, т. е. «ftp-2121».

7.7. Действия и переходы

7.7.1. Действие или переход к цепочке (`-j, --jump`)

Синтаксис в правиле:

```
-j, --jump цель
```

Действие или переход к цепочке (`-j`) – позволяет указать действие или цель (см. пп. 7.2 – 7.3) для пакетов, которые попадут под действие правила.

Целью может быть:

- определенная пользователем цепочка (отличная от цепочки правила);

- одна из встроенных целей, которая определит судьбу пакета.

Если опция не задана в правиле (и ключ `-g` не использован), то сработает только счетчик количества правил.

7.7.2. Переход к цепочке (`-g`, `--goto`)

Синтаксис в правиле:

`-g, --goto цепочка`

Переход к цепочке (`-g`) – указывает, что обработка должна продолжаться в указанной пользователем цепочке.

В отличие от опции `-jump` (см. п. 7.7.1), после действия RETURN из вызванной цепочки, применение правил будет продолжено не в текущей цепочке, а в той цепочке, которая вызвала текущую через `--jump`.

7.8. Графический интерфейс межсетевого экрана

Для настройки правил необходимо от администратора зайти в ГИ МЭ ИВК КОЛЬЧУГА-К раздел «Сеть» (см. п. 5.3) подраздел «Межсетевой экран» (рис. 24).

Описание основных графических элементов приведено в п. 5.4.

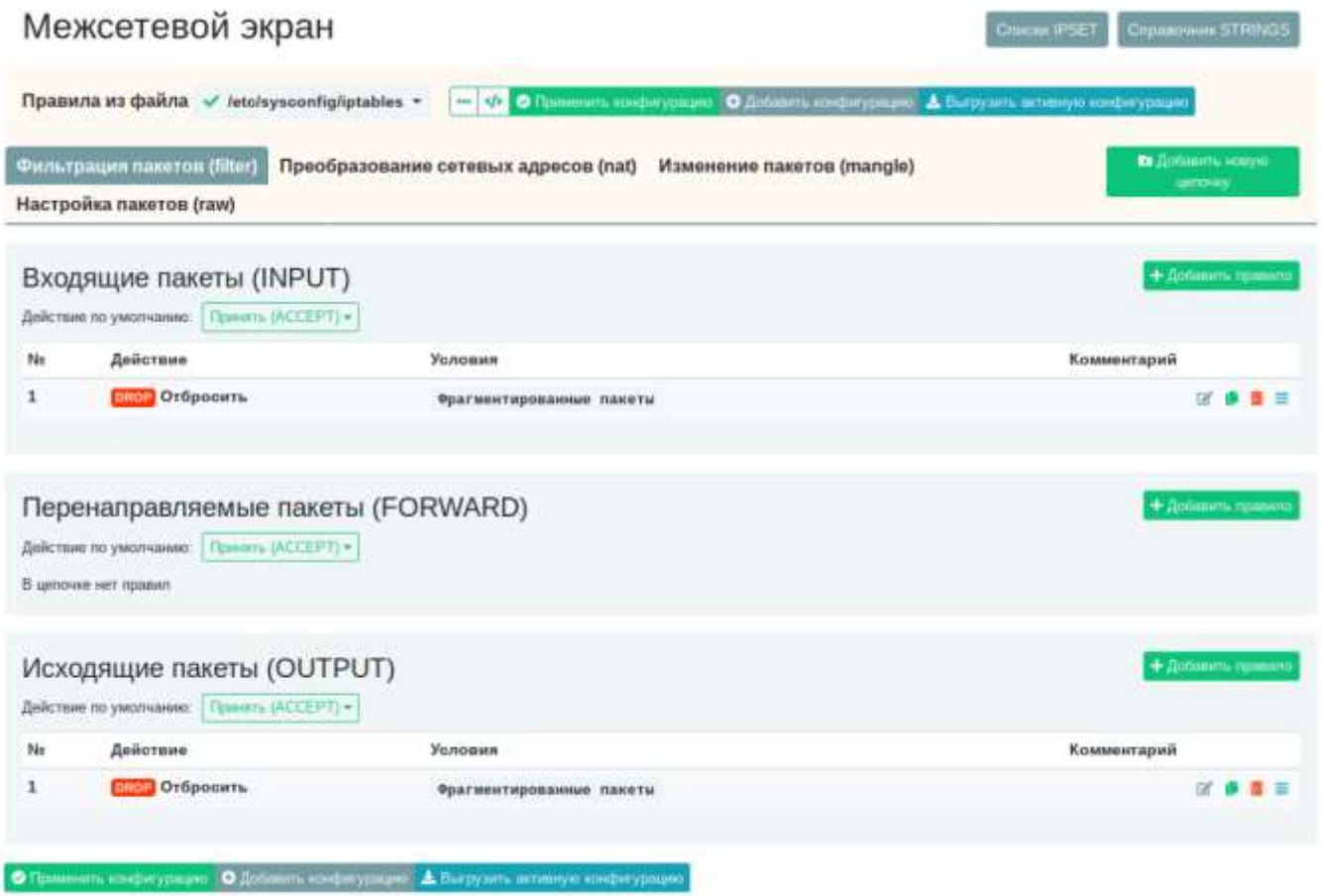


Рис. 24 – Межсетевой экран

Прежде, чем приступить к созданию набора правил, необходимо определиться с политиками (`-P`, `--policy` – см. таблицу 11) цепочек по умолчанию, задать стратегию.

`действие_по_умолчанию` (политика по умолчанию) (см. п. 7.2) применяется к пакету, не попавшему под действие ни одного из правил в цепочке.

В ГИ МЭ ИВК КОЛЬЧУГА-К разделе «Сеть» (см. п. 5.3) подраздел «Межсетевой экран» выберите таблицу (см. п. 7.4):

- `filter` – таблица, используемая по умолчанию (рис. 24) (см. п. 7.4.1);
- `nat` (см. п. 7.4.2);
- `mangle` (см. п. 7.4.3);
- `raw` (см. п. 7.4.4).

В таблице выберите встроенную цепочку и с помощью выпадающего списка задайте *`действие_по_умолчанию`* (рис. 25) (см. п. 7.2).

В консоли политика по умолчанию устанавливается командой:

```
iptables -P цепочка действие_по_умолчанию
```

ВНИМАНИЕ!

Команда применима только к встроенным цепочкам, т. е. INPUT, FORWARD, OUTPUT и т. п., и не применима к пользовательским цепочкам. Будьте предельно осторожны с установкой политик по умолчанию для цепочек из таблиц, не предназначенных для фильтрации.

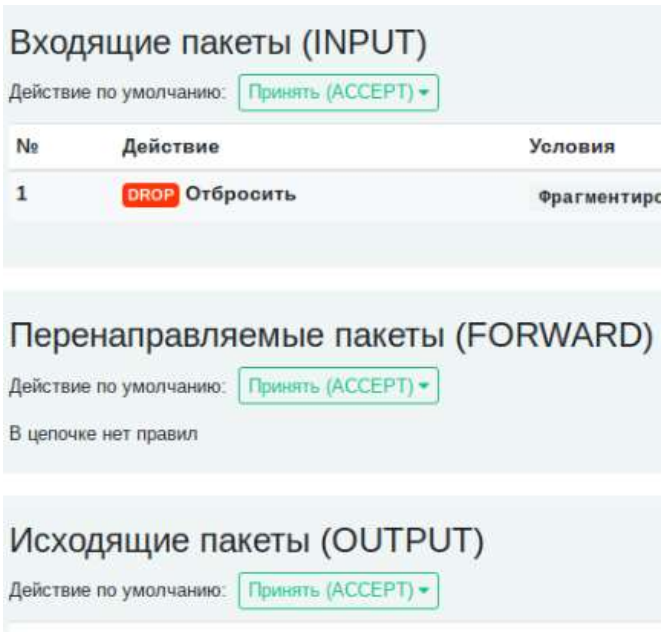


Рис. 25 – Цепочки пакетов

При необходимости можно добавить пользовательские цепочки (рис. 26), нажав на соответствующую кнопку 

. Создание дополнительных цепочек помогает в группировке правил между собой по выбранным признакам.

Порядок создания правил для всех видов цепочек аналогичен и подробнее описан в п. 7.8.2.

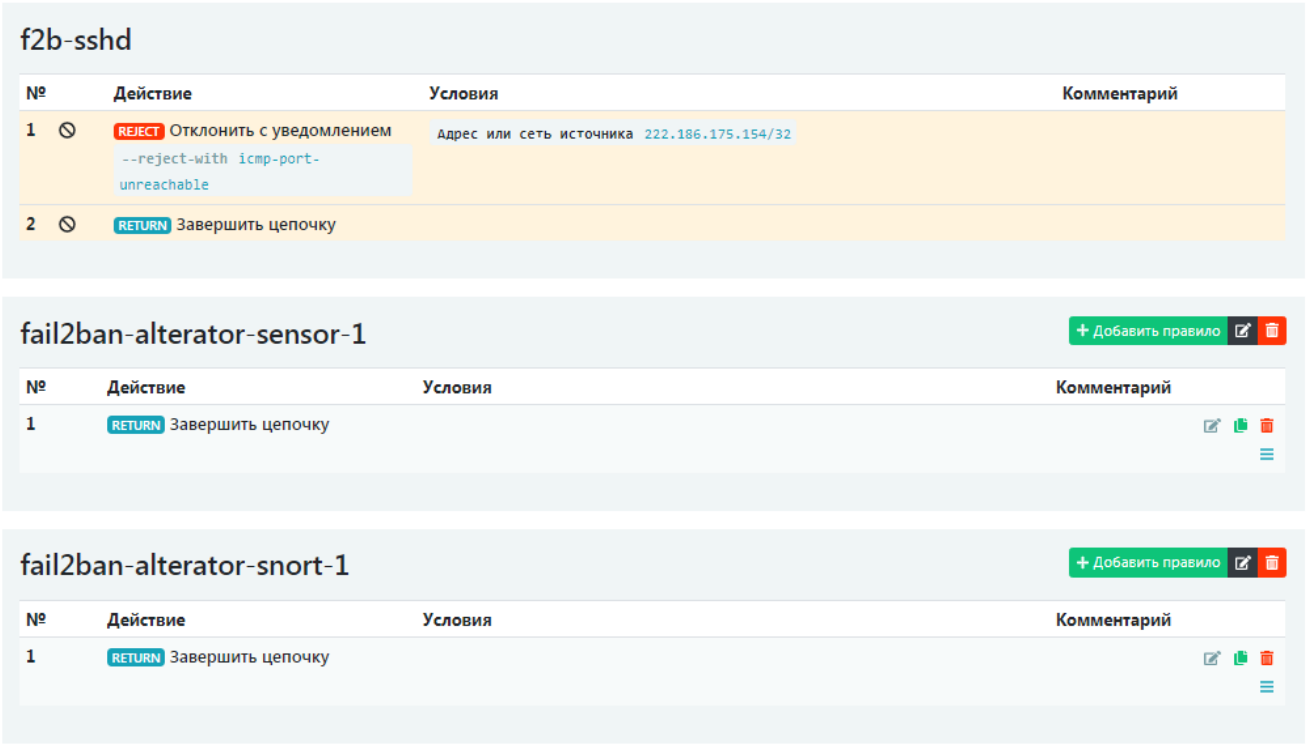


Рис. 26 – Цепочки

7.8.1. Конфигурация МЭ

Для быстрого доступа к настроенным файлам конфигурации правил МЭ или файлу конфигурации, заданному по умолчанию, перейдите на панель работы с конфигурацией и выберите поле «Правила из файла» (рис. 27) и задайте файл из выпадающего списка.

Файлы конфигурации МЭ расположены на файловой системе в папке `/usr/share/web-kolchuga/iptables/configs` (кроме файла конфигурации по умолчанию).

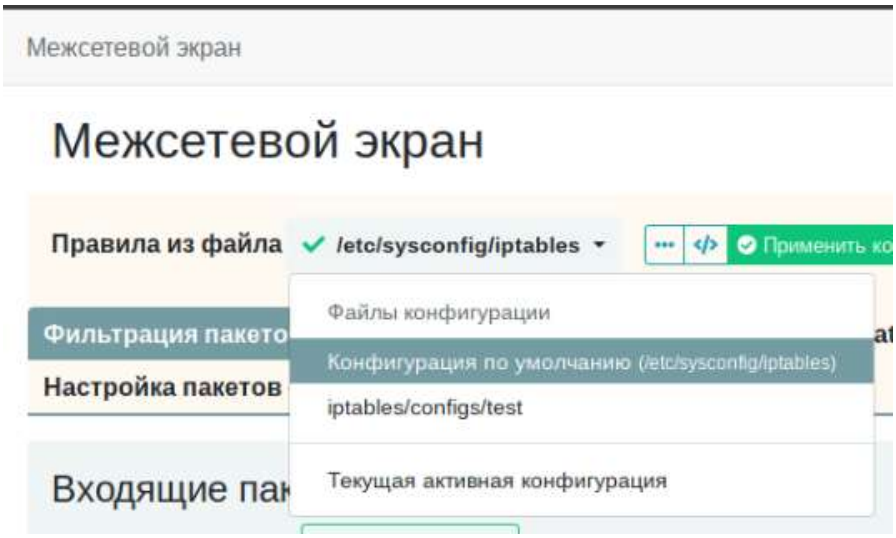


Рис. 27 – Вкладка «Правила из файла»

После нажатия на выбранный файл конфигурации откроется окно просмотра с параметрами текущей конфигурации (рис. 28). При необходимости изменения, добавления или удаления правил можно отредактировать файл конфигурации в этом окне и нажать на кнопку «Сохранить» для подтверждения настроек.

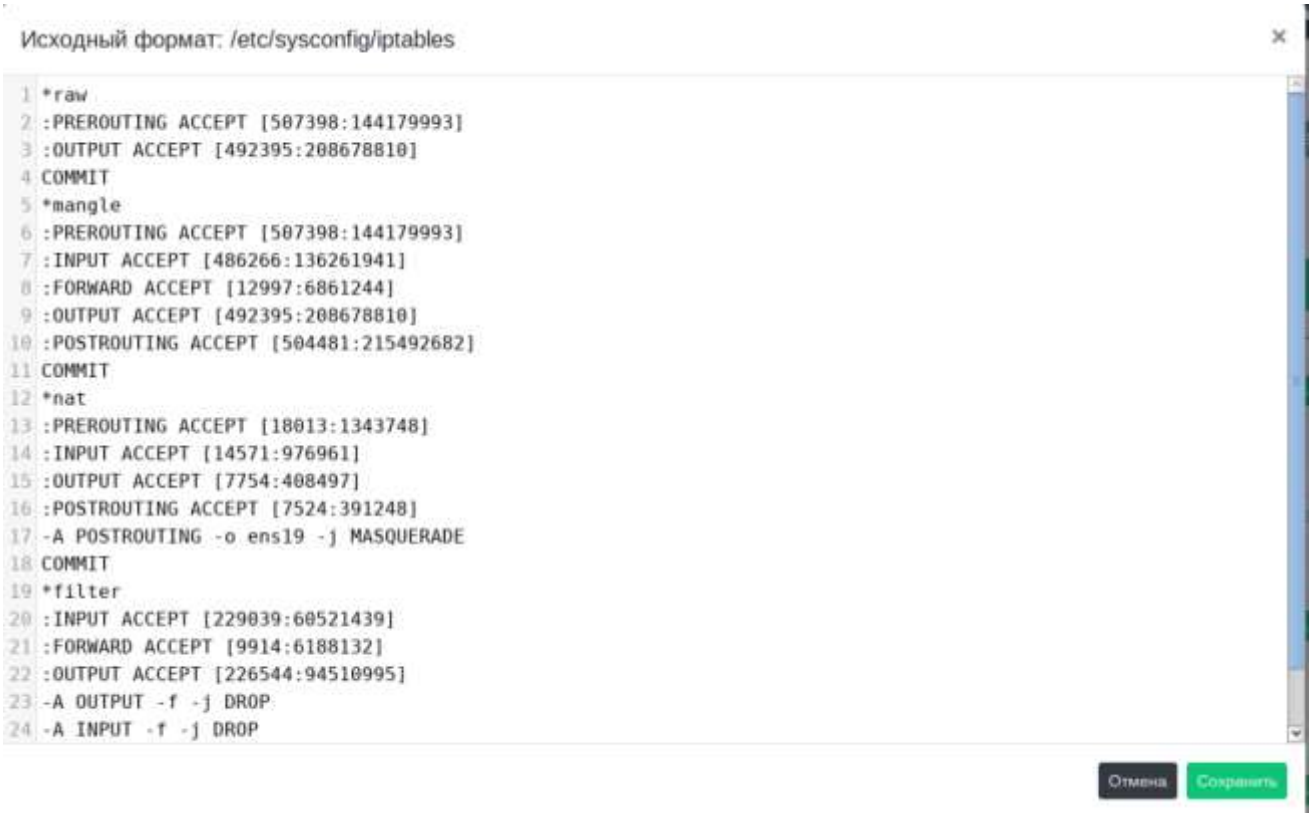


Рис. 28 – Просмотр выбранной конфигурации

Описание основных элементов панели конфигурации приведено в таблице 5 (п. 5.4).

Кнопка  (рис. 29):

- создать новый файл конфигурации правил;
- очистить конфигурацию;
- удалить файл конфигурации;
- выбрать просмотр истории внесенных изменений (рис. 30).

Операция удаления не доступна для файла конфигурации по умолчанию.

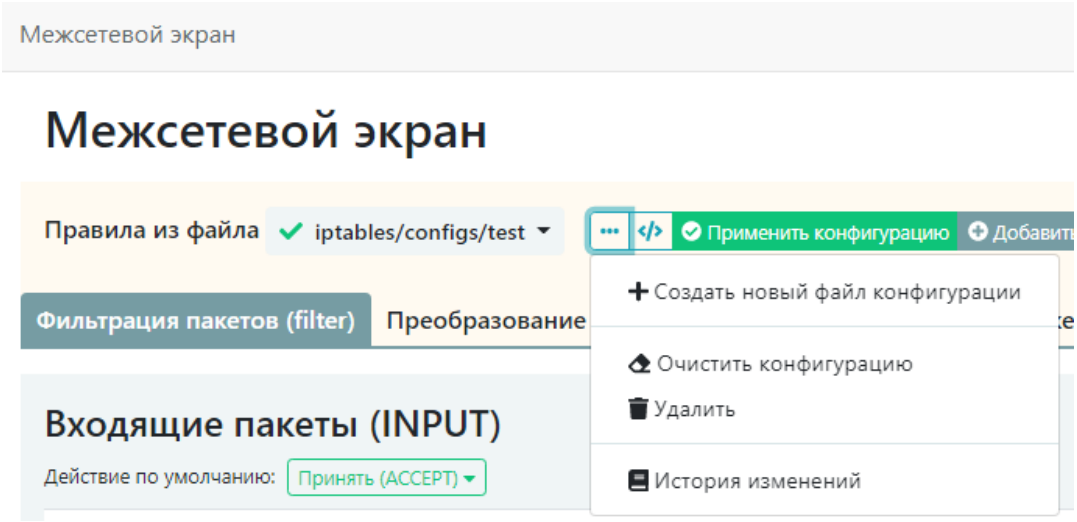


Рис. 29 – Кнопка выбора действий с файлом конфигурации

Создание нового файла конфигурации происходит на основе шаблона – файла конфигурации, расположенного на файловой системе по пути:
`/usr/share/web-kolchuga/config/templates/iptables.config.default.template`

По умолчанию шаблон файла конфигурации не содержит никаких правил.

Пример окна истории изменений файла конфигурации приведен на рис. 30. Файлы отображаются списком. Файлы истории сохраняются на файловой системе в папке `/usr/share/web-kolchuga/iptables/history`. Для более детального просмотра нажмите на кнопку «Открыть» (рис. 30). Если необходимо «Очистить историю», нажмите на соответствующую кнопку.

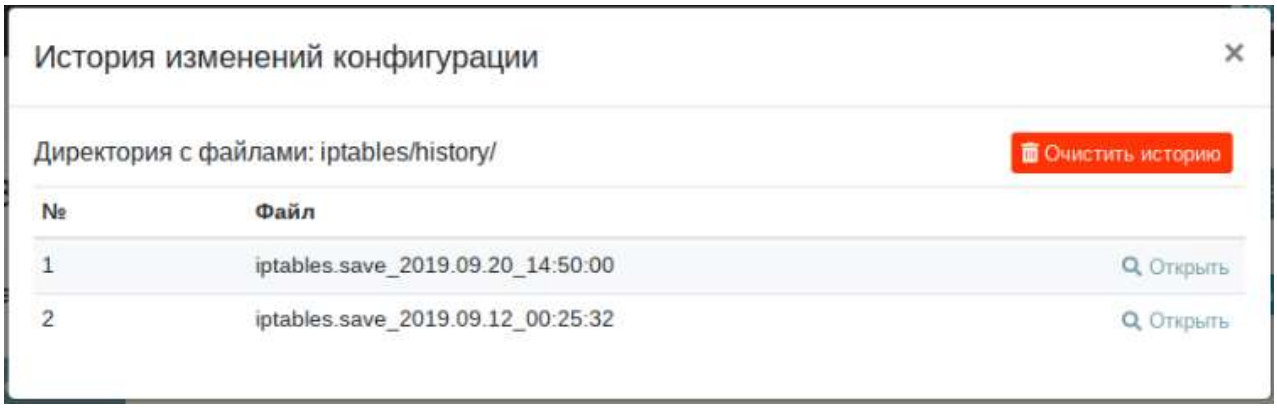


Рис. 30 – Окно история изменений файла конфигурации

Далее следуют кнопки, название которых поясняет их возможности:

- «Применить конфигурацию» (после редактирования или замены файла конфигурации);
- «Добавить конфигурацию»;
- «Выгрузить активную конфигурацию».

Для сохранения конфигурации МЭ в файл в консоли выполните добавление правил, например:

```
iptables -A INPUT -s IP-адрес -j ACCEPT
iptables -A INPUT -m ndpi --SSH -j ACCEPT
iptables -A INPUT -m ndpi --HTTP -j ACCEPT
```

и сохраните затем в файл конфигурации, выполнив команду:

```
iptables-save > /etc/sysconfig/iptables
```

Перезапустите iptables и проверьте установку правил фильтрации:

```
service iptables restart
iptables -S
```

7.8.2. Добавление правила

Для добавления нового правила фильтрации выберите таблицу фильтрации (п. 7.4) и интересующую в ней цепочку, нажмите на кнопку + Добавить правило, соответствующую цепочке.

На рис. 31 представлена форма добавления правила для входящих пакетов (цепочка INPUT).

В процессе создания правила с помощью выпадающего списка также можно изменить выбранную цепочку пакетов, для которой добавляется правило (рис. 32).

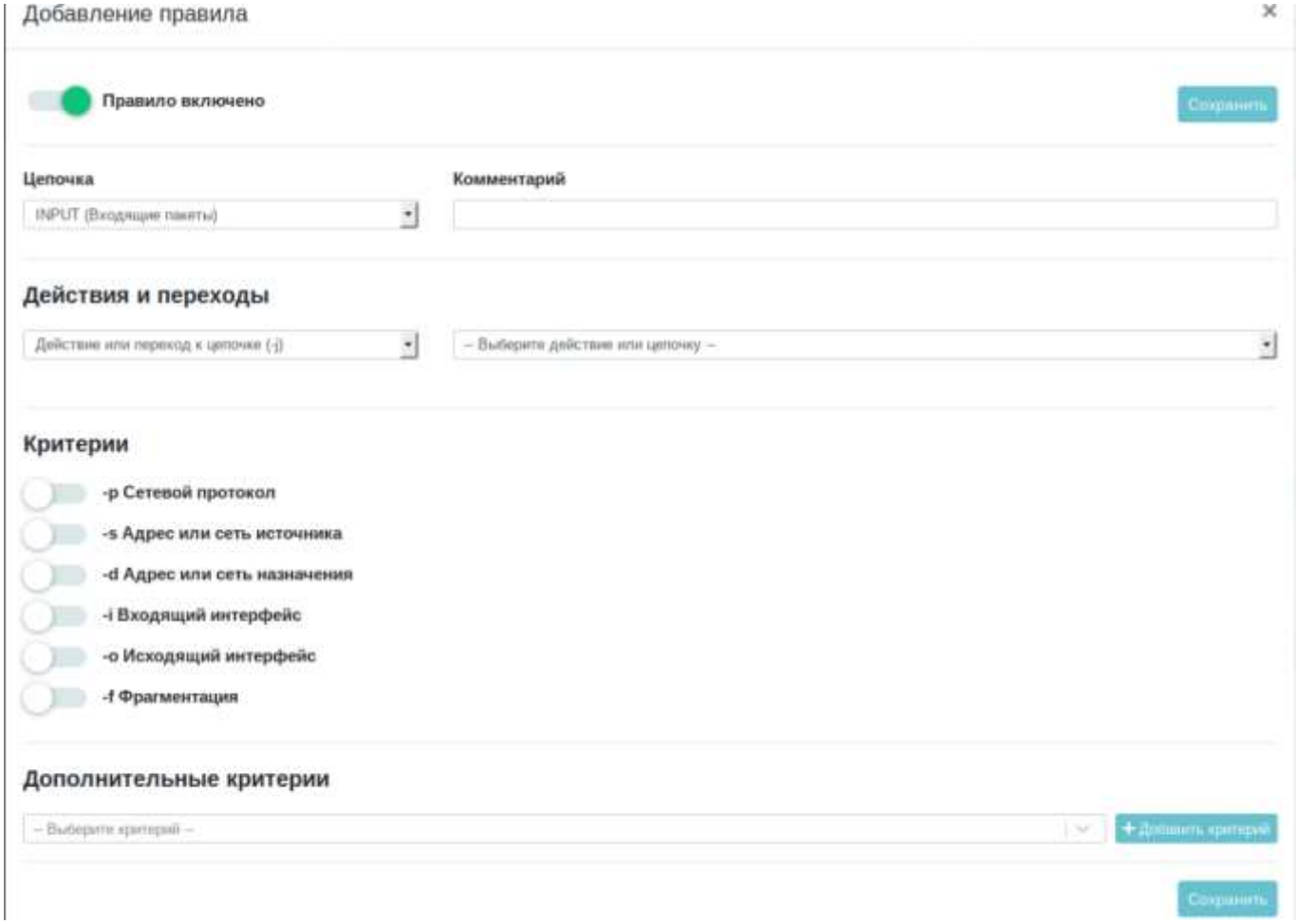


Рис. 31 – Форма добавления правила

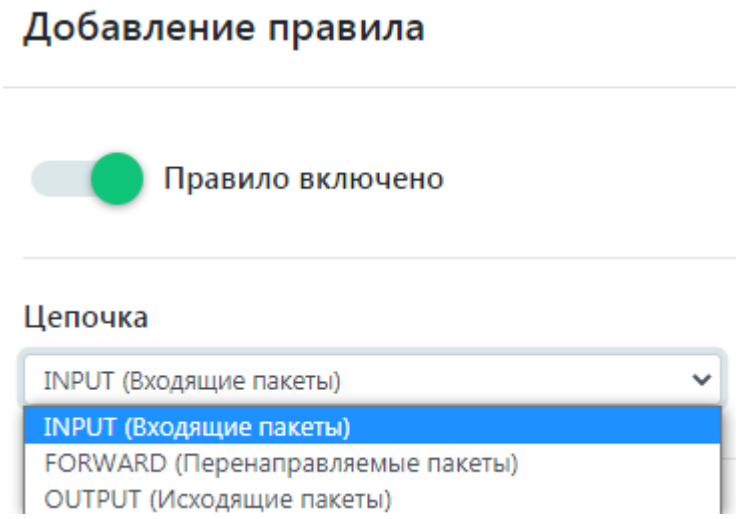


Рис. 32 – Добавление правила, выбор цепочки

Выбор необходимого значения действия из выпадающего списка действий и переходов (см. п. 7.7) является обязательным полем при составлении правила (рис. 33).

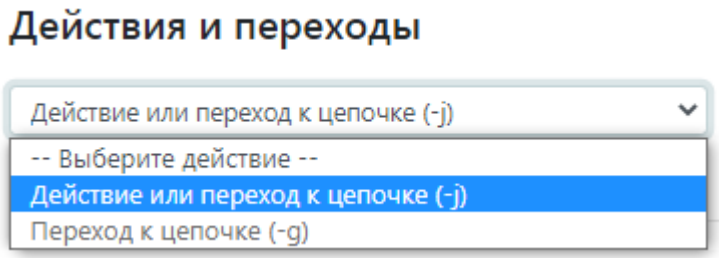


Рис. 33 – Добавление правила, выбор действия или перехода

Далее выбирается конкретное действие/цель (см. пп. 7.2 – 7.3) или цепочка для пакетов, которые попадут под действие, доступное для выбранной таблицы (рис. 34) (см. п. 7.4).

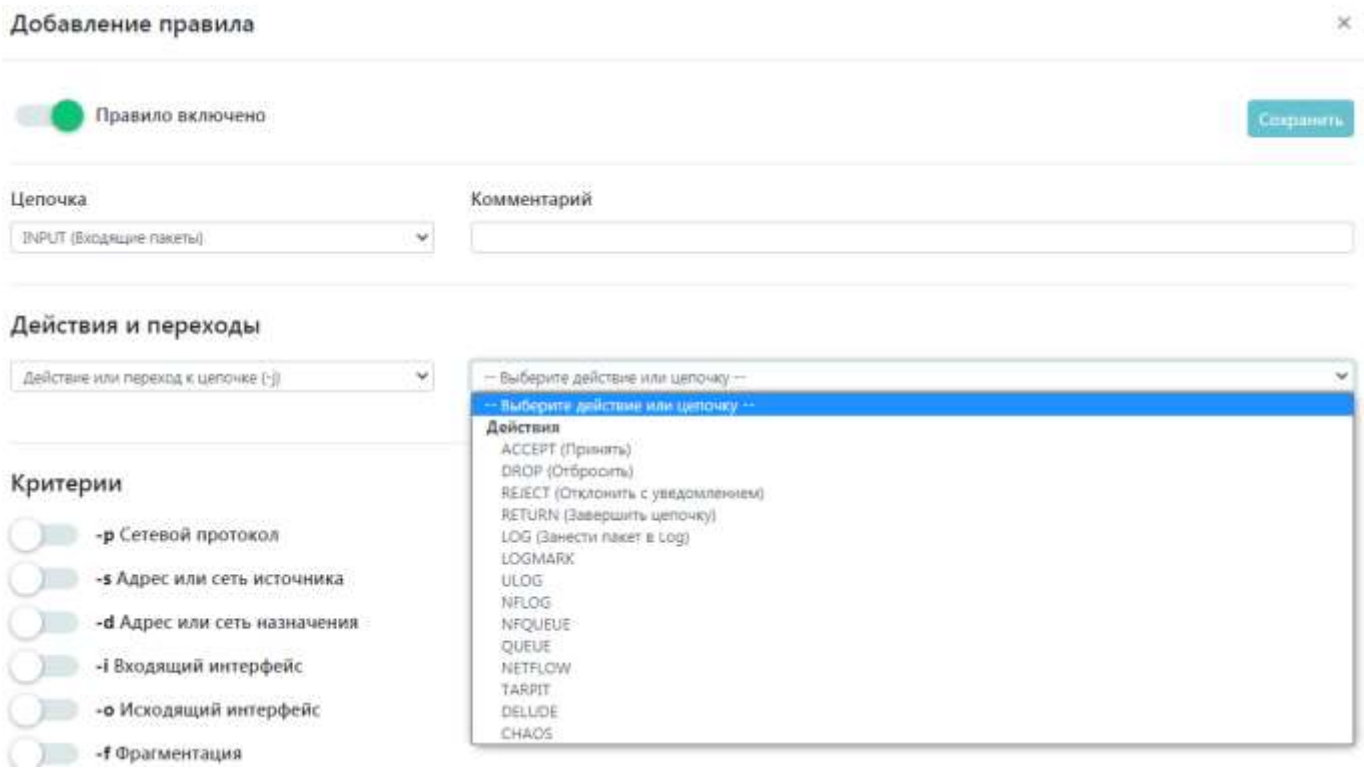


Рис. 34 – Выбор действия или цепочки

Существует несколько основных настраиваемых критериев для фильтрации пакетов, описания которых приведены в п. 7.6.1.

Включение нужного критерия происходит перемещением переключателя вправо ( ).

После этого, напротив включенного критерия, появится переключатель соответствия критерию:

- = – равен;
- ≠ – не равен (аналогичен символу ! инвертации критерия в правиле).

Если для выбранного общего критерия (см. п. 7.6.1) существует неявный критерий (см. п. 7.6.2), то в выпадающем списке следует выбрать его значение (рис. 35).

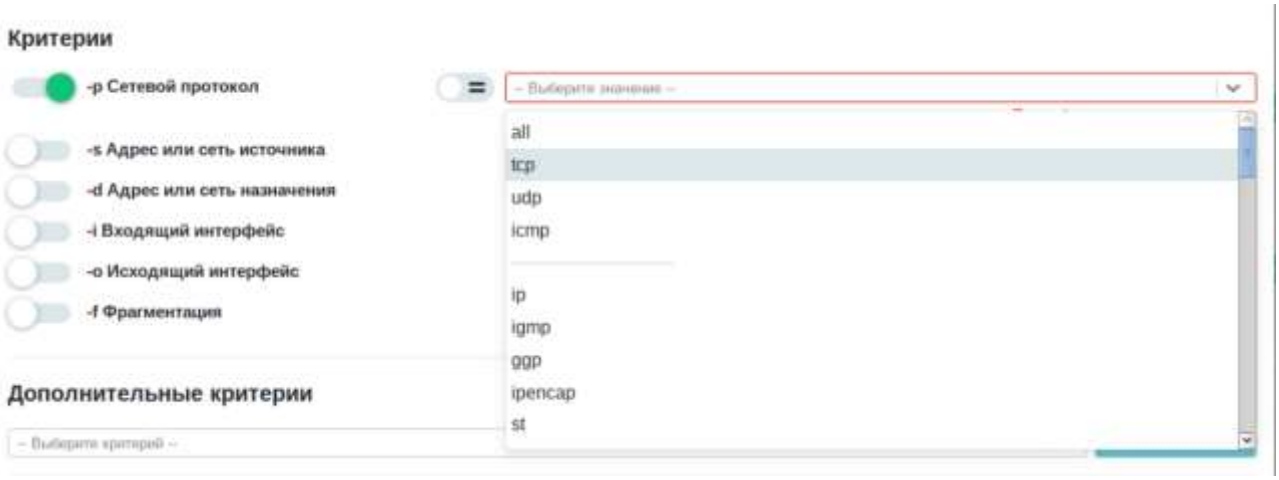


Рис. 35 – Уточнение значения критерия

Также в нижней строке окна «Добавление правила» при назначении критериев правила можно добавить дополнительные критерии настройки (см. п. 7.6.3), для этого из выпадающего списка следует выбрать нужный критерий и нажать на кнопку справа + Добавить критерий. Выбрать одну из опций критерия и заполнить необходимыми для настройки значениями при запросе (см. таблицу 15).

Для подтверждения внесенных настроек правила фильтрации нажмите кнопку «Сохранить» (см. рис. 31).

Если добавляемое правило содержит ошибки, появится предупреждающее сообщение и подсказки о возможности просмотра дополнительной информации о возникших ошибках (рис. 36).

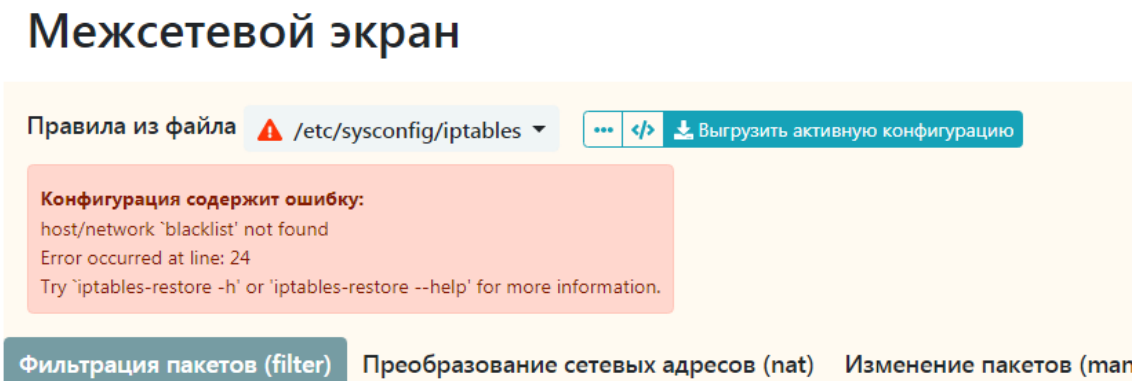


Рис. 36 – Ошибки в конфигурации правил

Для активации правила в окне настроек правила (см. рис. 33) переключатель правила должен быть выставлен в положение «Правило включено».

Для отключения правила установить переключатель влево – «Правило отключено». В списке правил отключенное станет отмечено знаком # после сохранения конфигурации.

При любом обновлении конфигурации нажимайте на кнопку «Сохранить» для подтверждения.

Пример настроенных правил фильтрации в ГИ МЭ ИВК КОЛЬЧУГА-К приведен на рис. 37.

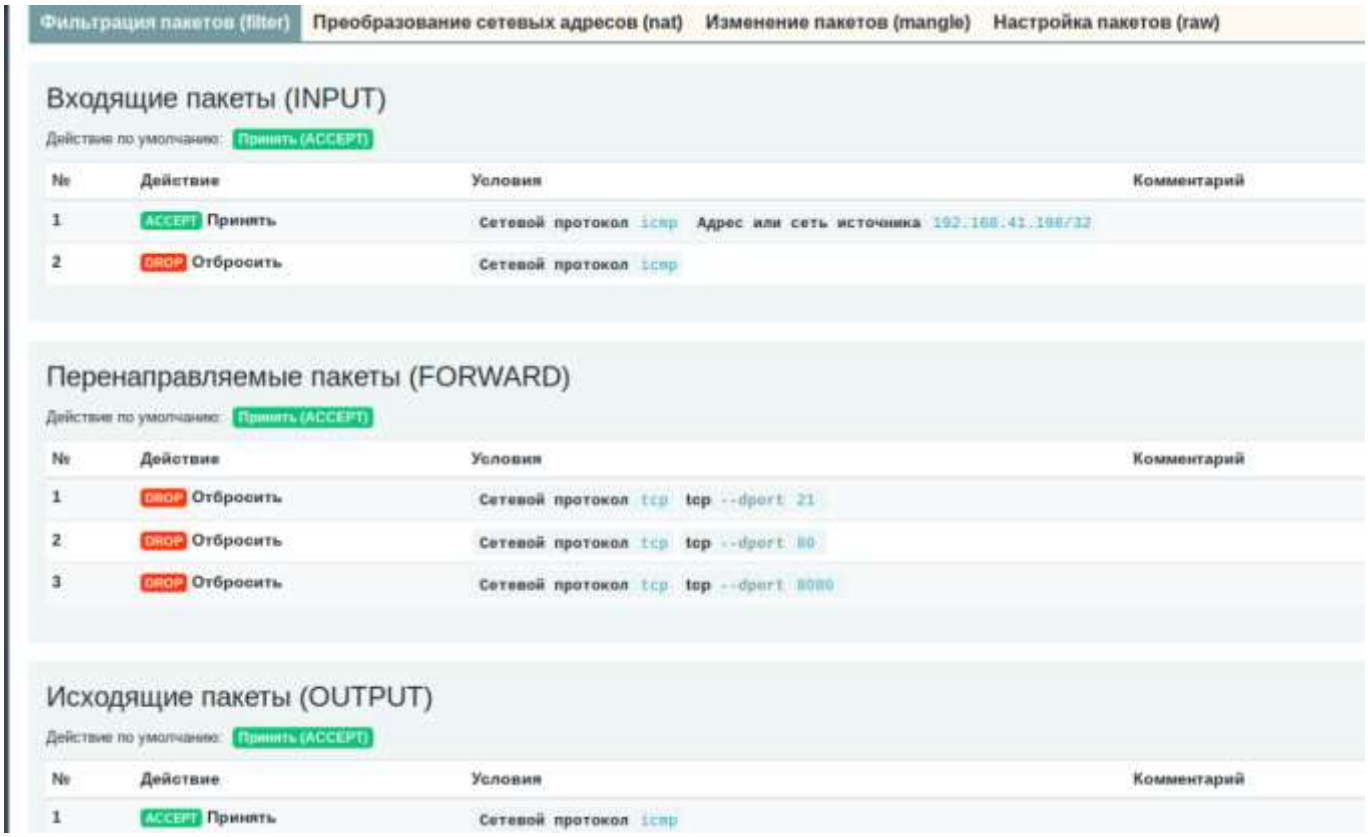


Рис. 37 – Пример настроенных правил фильтрации ГИ МЭ ИВК КОЛЬЧУГА-К

7.8.3. Списки IPSET

В ГИ МЭ ИВК КОЛЬЧУГА-К раздел «Сеть» (см. п. 5.3), подраздел «Межсетевой экран», в верхнем правом углу располагается кнопка 

для перехода в раздел «Списки IPSET», в котором можно создать наборы, используемые утилитой ipset, в правилах фильтрации при добавлении дополнительного критерия `set` (см. таблицу 15). Подробнее о возможностях ipset приведено в п. 7.10.

Списки IPSET используются для:

- хранения большого количества IP-адресов или номеров портов (TCP/UDP), MAC-адреса, имена интерфейсов или их комбинации, и сопоставления их с коллекцией iptables;
- динамического обновление списка правил iptables для IP-адресов или портов без снижения производительности;
- повышают скорость применения правил iptables.

Для добавления нового списка нажать на кнопку «Добавить список» 

и заполните поля (рис. 38):

- имя списка;
- тип списка – выбрать из выпадающего списка (см. п. 7.10.5);
- опции (при необходимости).

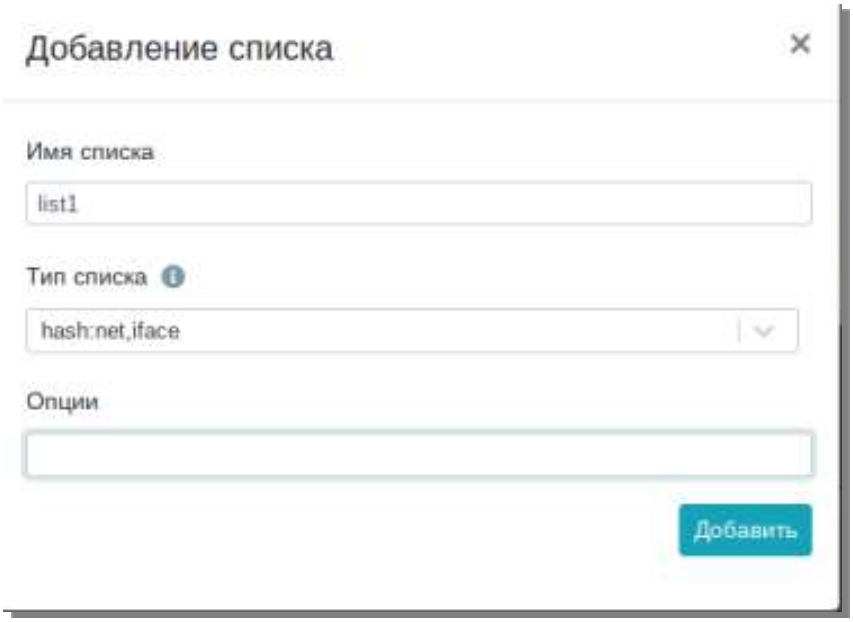


Рис. 38

При выборе типа списка с помощью кнопки можно просмотреть подробную информацию о выбранном типе списка (рис. 39).

Нажать на кнопку «Добавить», появится сообщение об успешном добавлении и новый набор (список) появится в таблице (рис. 40).

Справка по типу: hash:net,iface

```
hash:net,iface type specific options:

create SETNAME hash:net,iface
               [family inet|inet6]|[-4|-6]
               [hashsize VALUE]
               [maxelem VALUE]
               [timeout VALUE]
               [counters]
               [comment]
               [forceadd]
               [skbinfo]
add    SETNAME hash:net,iface IP[/CIDR]|FROM-TO,[physdev:]IFACE
               [timeout VALUE]
               [nomatch]
               [packets VALUE]
               [bytes VALUE]
               [comment "string"]
               [skbmark VALUE]
               [skbprio VALUE]
               [skbqueue VALUE]
del    SETNAME hash:net,iface IP[/CIDR]|FROM-TO,[physdev:]IFACE
test   SETNAME hash:net,iface IP[/CIDR],[physdev:]IFACE
               [nomatch]
```

Рис. 39

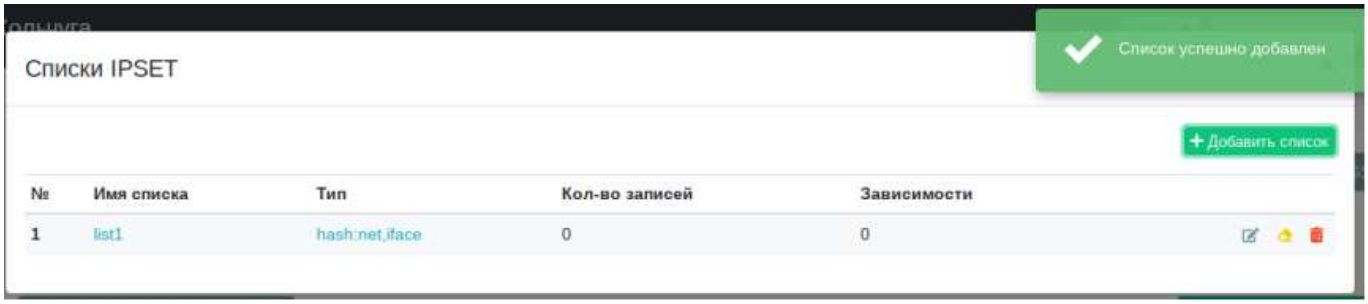


Рис. 40

В созданном списке можно отредактировать имя списка – .

С помощью кнопки – очистить все записи списка.

Или удалить список с помощью кнопки .

Для заполнения списка IPSET объектами нажмите на поле «Имя списка».

Окно редактирования объектов списка IPSET состоит из трех вкладок:

- «Поиск»;

- «Добавление»;

- «Удаление».

При добавлении объектов в списке на соответствующей вкладке, ввод значений осуществляется в порядке, в зависимости от типа выбранного списка (подробнее см. п. 7.10). При успешном добавление появится новая запись списка (рис. 41).

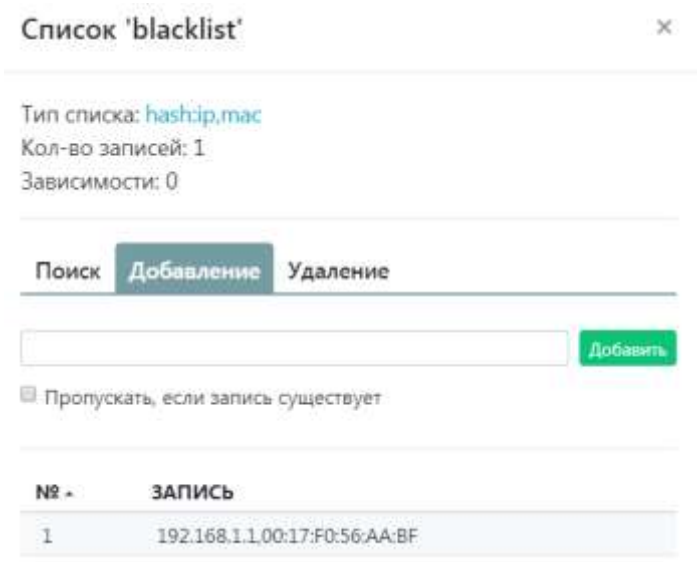


Рис. 41

При некорректном вводе значений или повторном вводе существующего объекта, появится предупреждающие сообщение и объект добавлен не будет.

ВНИМАНИЕ!

При добавлении, например, сначала IP-адреса, а потом всей подсети, в которую входит этот адрес, необходимо ставить флаг «Пропускать, если запись существует», чтобы корректно добавить в список все адреса подсети.

Аналогично, при групповом удалении IP-адресов из списка путем задания подсети, флаг «Пропускать, если запись не существует» позволит учесть IP-адреса подсети, которые уже исключены из него.

Созданный список IPSET применяется вместе с дополнительным критерием `set` (см. таблицу 15) и выставленной опцией `--match-set` для пакетов, проходящих через какую-либо цепочку фильтра iptables.

Пример добавления правила (см. п. 7.8.1):

1) выбрать цепочку;
2) выбрать значения «Действия и переходы»;
3) выбрать все необходимые критерии (дополнительные критерии);
4) добавить дополнительный критерий `set` (см. таблицу 15);
5) выставить опцию `--match-set`;
6) выбрать значение переключателя: равен/не равен;
7) выбрать название списка IPSET (SETNAME);
8) выставить флаг, который показывает какие IP-адреса сравнивать со списком:
   - `src` – источник;
   - `dst` – назначение;
9) нажать кнопку «Сохранить»;
10) переместить правило в перечне правил цепочки с помощью кнопки ☰, для применения в нужной очередности.

7.8.4. Справочник STRINGS

В ГИ МЭ ИВК КОЛЬЧУГА-К раздел «Сеть» (см. п. 5.3), подраздел «Межсетевой экран», в верхнем правом углу располагается кнопка Справочник STRINGS для перехода в раздел создания справочника по запрещенному/разрешенному содержимому пакетов; информация используется в правилах фильтрации при добавлении дополнительного критерия `string` (см. таблицу 15).

Записями справочника могут быть (рис. 42):

- команды;
- мобильный код;
- прикладное ПО (приложения).

Рис. 42

Для добавления строки записи в справочник нажать на кнопку «Добавить список» и заполните поля:

- название;
- значение;
- комментарий.

Для подтверждения добавления нажать на кнопку «Сохранить». В результате строка будет добавлена в справочник и появится всплывающее уведомление об успешном событии.

Для применения элемента справочника при фильтрации его необходимо добавлять при создании правила (см. п. 7.8.1).

Пример добавления правила в ГИ МЭ ИВК КОЛЬЧУГА-К:

1) выбрать цепочку;
2) выбрать значения «Действия и переходы»;
3) выбрать все необходимые критерии (дополнительные критерии);
4) добавить дополнительный критерий `string` (см. таблицу 15);
5) выставить опцию `--string` (см. таблицу 15) – последовательность символов, которую следует искать в пакете;
6) выбрать значение переключателя: равен/не равен;
7) выбрать элемент справочника STRINGS;
8) выставить также опцию `--algo` (см. таблицу 15), выбрать стратегию сравнения `bm`;
9) нажать кнопку «Сохранить».

Пример добавления аналогичных правил в консоли:

```
iptables -A INPUT -m string --algo bm --string "<script" -j DROP
iptables -A INPUT -m string --algo bm --string ".js" -j DROP
```

7.9. Перечни возможных значений

7.9.1. Состояний соединения

Вместе с опциями `--state` и `--ctstate` задаются следующие виды состояний соединения (разделяются запятыми):

- NEW (Новое) – пакет запрашивает новое соединение, как, например, HTTP-запрос;
- ESTABLISHED (Установлено) – пакет относится к существующему соединению;
- RELATED (Связанный) – означает, что пакет принадлежит уже существующему соединению, но при этом он открывает новое соединение, как в случае с активными FTP-соединениями, когда подключение идет к порту 20, а передача выполняется через любой неиспользуемый порт, например, 1023;
- INVALID (Неверный) – пакет не является частью ни одного из соединений в таблице отслеживания соединений, его не удалось идентифицировать по какой-либо причине, включая нехватку памяти и ошибки ICMP, которые не соответствуют какому-либо известному соединению;
- UNTRACKED – отслеживание состояния отключено.

Дополнительно для `--ctstate` также доступны:

- SNAT – источник NATd. Соответствующее виртуальное состояние, если исходный адрес источника отличается от адреса назначения ответа;
- DNAT – назначение NATd. Соответствующее виртуальное состояние, если исходное место назначения отличается от источника ответа.

7.9.2. Список протоколов TCP/IP

Перечень протоколов для критерия `-p`, `--protocol` (см. таблицу 13) или при предварительном указании критерия «-p Сетевой протокол» в ГИ МЭ ИВК КОЛЬЧУГА-К, и для опции `--ctproto` дополнительного критерия `conntrack` (см. таблицу 15), при предварительном указании этого критерия в правиле:

- all – все протоколы;
- tcp;
- udp;
- icmp;
- и следующие: ip, igmp, ggp, ipencap, st, cbt, egp, igp, bbn-rcc, nvp, pup, argus, emcon, xnet, chaos, mux, dcn, hmp, prm, xns-idp, trunk-1, trunk-2, leaf-1, leaf-2, rdp, irtp, iso-tp4, netblt, mfe-nsp, merit-inp, sep, 3pc, idpr, xtp, ddp, idpr-cmtp, tp++, il, ipv6, sdrp, ipv6-route, ipv6-frag, idrp, rsvp, gre, mhrp, bna, ipv6-crypt, ipv6-auth, i-nlsp, swipe, narp, mobile, tlsp, skip, ipv6-icmp, ipv6-nonxt, ipv6-opts, cftp, sat-expak, kryptolan, rvd, ippc, sat-mon, visa, ipcv, cpnx, cphb, wsn, pvp, br-sat-mon, sun-nd, wb-mon, wb-expak, iso-ip, vmtp, secure-vmtp, vines, ttp, nsfnet-igp, dgp, tcf, eigrp, ospf, sprite-rpc, larp, mtp, ax.25, ipip, micp, scc-sp, etherip, encap, gmtp, ifmp, pnni, pim, aris, scps, qnx, a/n, ipcomp, snp, compaq-peer, ipx-in-ip, vrrp, pgm, l2tp, ddx, iatp, st, srp, uti, smp, sm, ptp, isis, fire, crtp, crdup, sscopmce, iplt, sps, pipe, sctp, fc.

7.9.3. Перечень типов сообщений ICMP (icmp-type)

В таблице 16 приведен перечень возможных типов сообщений для неявного критерия – опции `--icmp-type` (см. таблицу 14), доступна при предварительном указании критерия `-p icmp` (см. п. 7.9.2) или при предварительном указании критерия «-p Сетевой протокол» → выбор протокола «icmp» в правиле фильтрации через ГИ МЭ ИВК КОЛЬЧУГА-К.

Т а б л и ц а 16 – Тип сообщения ICMP

Номер	Имя	Примечание	Запрос	Ошибка
0	echo-reply	echo-ответ, пинг	x	
3	destination-unreachable	адресат недосягаем. Выбор всех сообщений типа		x
3/0	network-unreachable	сеть недостижима		x
3/1	host-unreachable	хост назначения недоступен		x
3/2	protocol-unreachable	протокол назначения недоступен		x
3/3	port-unreachable	порт назначения недоступен		x
3/4	fragmentation-needed	уведомляет о необходимости фрагментации сообщения		x
3/5	source-route-failed	исходный маршрут не прошел		x
3/6	network-unknown	сеть неизвестна		x
3/7	host-unknown	хост неизвестен		x
3/9	network-prohibited	сеть запрещена в административном порядке		x
3/10	host-prohibited	хост запрещен в административном порядке		x
3/11	TOS-network-unreachable	сеть недоступна для ToS		x
3/12	TOS-host-unreachable	хост недоступен для ToS		x
3/13	communication-prohibited	связь запрещена в административном порядке		x
3/14	host-precedence-violation	нарушение приоритета хоста		x
3/15	precedence-cutoff	действует ограничение приоритета		x
4	source-quench	подавление источника, замедление отправки пакетов		
5	redirect	выбор всех сообщений типа переадресовать (изменить маршрут)		
5/0	network-redirect	переадресовать в другую сеть		
5/1	host-redirect	переадресовать на другой хост		
5/2	TOS-network-redirect	Перенаправление для TOS и сети		
5/3	TOS-host-redirect	Перенаправление для TOS и хоста		
8	echo-request	echo-запрос, пинг	x	
9	router-advertisement	объявление маршрутизатора		x
10	router-solicitation	запрос маршрутизатора		x
11	time-exceeded	выбор всех сообщений с истечением времени жизни (ttl=0)		x
11/0	ttl-zero-during-transit	время жизни истекло при передаче		x
11/1	ttl-zero-during-reassembly	время жизни истекло при при сборке (случай фрагментации)		x
12	parameter-problem	выбор всех сообщений с ошибками в параметрах		x
12/0	ip-header-bad	ошибка в IP-заголовке		x
12/1	required-option-missing	отсутствует необходимая опция		x
13	timestamp-request	запрос временной метки	x	
14	timestamp-reply	отклик на запрос временная метка	x	
17	address-mask-request	запрос адресной маски	x	
18	address-mask-reply	отклик на запрос адресной маски	x	

7.9.4. Перечень TCP-опций

В таблице 17 приведен перечень возможных TCP-опций для неявного критерия – опции `--tcp-option` (см. таблицу 14), доступна при предварительном указании критерия: `-p tcp` (см. п. 7.9.2) или при предварительном указании критерия «-p Сетевой протокол» → выбор протокола «tcp» в правиле фильтрации через ГИ МЭ ИВК КОЛЬЧУГА-К.

Т а б л и ц а 17 – TCP-опции

Значение	Описание	Значение	Описание
0	End of Option List	18	Trailer Checksum Option
1	No-Operation	19	MD5 Signature Option (obsoleted by option 29)
2	Maximum Segment Size	20	SCPS Capabilities
3	Window Scale	21	Selective Negative Acknowledgements
4	SACK Permitted	22	Record Boundaries
5	SACK	23	Corruption experienced
6	Echo (obsoleted by option 8)	24	SNAP
7	Echo Reply (obsoleted by option 8)	25	Unassigned
8	Timestamps	26	TCP Compression Filter
9	Partial Order Connection Permitted (obsolete)	27	Quick-Start Response
10	Partial Order Service Profile (obsolete)	28	User Timeout Option (also, other known unauthorized use)
11	CC (obsolete)	29	TCP Authentication Option (TCP-AO)
12	CC.NEW (obsolete)	30	Multipath TCP (MPTCP)
13	CC.ECHO (obsolete)	34	TCP Fast Open Cookie
14	TCP Alternate Checksum Request (obsolete)	69	Encryption Negotiation (TCP-ENO)
15	TCP Alternate Checksum Data (obsolete)	253	RFC3692-style Experiment 1 (also improperly used for shipping products)
16	Skeeter	254	RFC3692-style Experiment 2 (also improperly used for shipping products)
17	Bubba		

7.9.5. TCP-флаги

В данном пункте приведено описание TCP-флагов для неявного критерия – опции `--tcp-flags` (см. таблицу 14), доступной при предварительном указании критерия: `-p tcp` (см. п. 7.9.2) или при предварительном указании критерия «-p Сетевой протокол» → выбор протокола «tcp» в правиле фильтрации через ГИ МЭ ИВК КОЛЬЧУГА-К.

SYN (Synchronize Sequence Number – синхронизация порядковых номеров) – устанавливается для пакетов SYN/SYN-ACK на этапе организации соединения.

ACK (Acknowledgement) – подтверждение, во всех случаях, кроме стартового пакета SYN.

FIN (End of Data: FINished) – завершение передачи данных, показывает отсутствие данных для передачи (маловероятно использование этого флага с какими-либо флагами, кроме ACK).

RST (Reset Connection – сброс соединения) – устанавливается для сброса соединения (маловероятно использование этого флага с какими-либо флагами.

URG (Urgent Flag – флаг срочности) – указывает на срочность данных (маловероятно использование этого флага с какими-либо флагами, кроме ACK).

PSH (Push Function Field – выталкивание данных). В общем случае произвольно меняется между 0 и 1. Однако одно из значений может встречаться чаще другого в основном определяется используемым стеком протоколов).

7.9.6. Страны (countries)

В таблице 18 приведен перечень возможных стран для опций `--src-cc` и `--dst-cc`, доступные при предварительном указании критерия дополнительного критерия: `-m geoip` (см. таблицу 15) или при предварительном выборе этого дополнительного критерия при добавлении правила фильтрации через ГИ МЭ ИВК КОЛЬЧУГА-К (см. рис. 47).

Т а б л и ц а 18 – Страны (countries)

Код	Название страны	Код	Название страны	Код	Название страны	Код	Название страны
AD	Андорра	EE	Эстония	LA	Лаос	RE	Реюньон
AE	ОАЭ	EG	Египет	LB	Ливан	RO	Румыния
AF	Афганистан	EH	САДР	LC	Сент-Люсия	RS	Сербия
AG	Антигуа и Барбуда	ER	Эритрея	LI	Лихтенштейн	RU	Россия
AI	Ангилья	ES	Испания	LK	Шри-Ланка	RW	Руанда
AL	Албания	ET	Эфиопия	LR	Либерия	SA	Саудовская Аравия
AM	Армения	EU	Флаг ЕС Европейский союз	LS	Лесото	SB	Соломоновы Острова
AO	Ангола	FI	Финляндия	LT	Литва	SC	Сейшельские Острова
AQ	Антарктида	FJ	Фиджи	LU	Люксембург	SD	Судан
AR	Аргентина	FK	Фолклендские острова	LV	Латвия	SE	Швеция
AS	Американское Самоа	FM	Микронезия	LY	Ливия	SG	Сингапур
AT	Австрия	FO	Фареры	MA	Марокко	SH	Острова Святой Елены, Вознесения и Тристан-да-Кунья
AU	Австралия	FR	Франция	MC	Монако	SI	Словения
AW	Аруба	GA	Габон	MD	Молдавия	SJ	Флаг Шпицбергена и Ян-Майена
AX	Аландские острова	GB	Великобритания	ME	Черногория	SK	Словакия
AZ	Азербайджан	GD	Гренада	MF	Сен-Мартен	SL	Сьерра-Леоне
BA	Босния и Герцеговина	GE	Грузия	MG	Мадагаскар	SM	Сан-Марино
BB	Барбадос	GF	Гвиана	MH	Маршалловы Острова	SN	Сенегал
BD	Бангладеш	GG	Гернси	MK	Северная Македония	SO	Сомали
BE	Бельгия	GH	Гана	ML	Мали	SR	Суринам
BF	Буркина-Фасо	GI	Гибралтар	MM	Мьянма	SS	Южный Судан
BG	Болгария	GL	Гренландия	MN	Монголия	ST	Сан-Томеи Принсипи
BH	Бахрейн	GM	Гамбия	MO	Макао	SV	Сальвадор
BI	Бурунди	GN	Гвинея	MP	Северные Марианские Острова	SX	Синт-Мартен
BJ	Бенин	GP	Гваделупа	MQ	Мартиника	SY	Сирия
BL	Сен-Бартелеми	GQ	Экваториальная Гвинея	MR	Мавритания	SZ	Эсватини
BM	Бермуды	GR	Греция	MS	Монтсеррат	TC	Теркси Кайкос
BN	Бруней	GS	Южная Георгия и Южные Сандвичевы Острова	MT	Мальта	TD	Чад

Продолжение таблицы 18

Код	Название страны	Код	Название страны	Код	Название страны	Код	Название страны
BO	Боливия	GT	Гватемала	MU	Маврикий	TF	Французские Южные и Антарктические Территории
BQ	Бонэйр, Синт-Эстатиуси Саба	GU	Гуам	MV	Мальдивы	TG	Того
BR	Бразилия	GW	Гвинея-Бисау	MW	Малави	TH	Таиланд
BS	Багамские Острова	GY	Гайана	MX	Мексика	TJ	Таджикистан
BT	Бутан	HK	Гонконг	MY	Малайзия	TK	Токелау
BV	Остров Буве	HM	Херди Макдональд	MZ	Мозамбик	TL	Восточный Тимор
BW	Ботсвана	HN	Гондурас	NA	Намибия	TM	Туркмения
BY	Белоруссия	HR	Хорватия	NC	Новая Каледония	TN	Тунис
BZ	Белиз	HT	Гаити	NE	Нигер	TO	Тонга
CA	Канада	HU	Венгрия	NF	Остров Норфолк	TR	Турция
CC	Кокосовые острова	ID	Индонезия	NG	Нигерия	TT	Тринидади Тобаго
CD	ДР Конго	IE	Флаг Ирландии Ирландия	NI	Никарагуа	TV	Тувалу
CF	ЦАР	IL	Израиль	NL	Нидерланды	TW	Китайская Республика
CG	Республика Конго	IM	Остров Мэн	NO	Норвегия	TZ	Танзания
CH	Швейцария	IN	Индия	NP	Непал	UA	Украина
CI	Кот-д’Ивуар	IO	Британская территория в Индийском океане	NR	Науру	UG	Уганда
CK	Острова Кука	IQ	Ирак	NU	Ниуэ	UM	Внешние малые острова(США)
CL	Чили	IR	Иран	NZ	Новая Зеландия	US	США
CM	Камерун	IS	Исландия	OM	Оман	UY	Уругвай
CN	Китай Китайская Народная Республика	IT	Италия	PA	Панама	UZ	Узбекистан
CO	Колумбия	JE	Джерси	PE	Перу	VA	Ватикан
CR	Коста-Рика	JM	Ямайка	PF	Французская Полинезия	VC	Сент-Винсент и Гренадины
CU	Куба	JO	Иордания	PG	Папуа-Новая Гвинея	VE	Венесуэла
CV	Кабо-Верде	JP	Япония	PH	Филиппины	VG	Виргинские Острова (Великобрита-ния)
CW	Кюрасао	KE	Кения	PK	Пакистан	VI	Виргинские Острова (США)

Окончание таблицы 18

Код	Название страны	Код	Название страны	Код	Название страны	Код	Название страны
CX	Остров Рождества	KG	Киргизия	PL	Польша	VN	Вьетнам
CY	Кипр	KH	Камбоджа	PM	Сен-Пьери Микелон	VU	Вануату
CZ	Чехия	KI	Кирибати	PN	Острова Питкэрн	WF	Уоллиси Футуна
DE	Германия	KM	Коморы	PR	Пуэрто-Рико	WS	Самоа
DJ	Джибути	KN	Сент-Китси Невис	PS	Государство Палестина	YE	Йемен
DK	Дания	KP	КНДР (Корейская Народно-Демократическая Республика)	PT	Португалия	YT	Майотта
DM	Доминика	KR	Республика Корея	PW	Палау	ZA	ЮАР
DO	Доминиканская Республика	KW	Кувейт	PY	Парагвай	ZM	Замбия
DZ	Алжир	KY	Острова Кайман	QA	Катар	ZW	Зимбабве
EC	Эквадор	KZ	Казахстан				

7.9.7. Перечень значений для опции `--proto`

Перечень возможных значений для опции `--proto` дополнительного критерия `-m ndpi` (см. таблицу 15), при предварительном указании этого критерия в правиле фильтрации:

AFP, AJP, AMQP, Aimini, Amazon, AmazonVideo, Apple, AppleJuice, ApplePush, AppleStore, AppleiCloud, AppleiTunes, Armagetron, Ayiya, BGP, BJNP, BattleField, BitTorrent, CHECKMK, CNN, COAP, CSGO, CiscoSkinny, CiscoVPN, Citrix, Cloudflare, Collectd, Corba, Crossfire, DCE_RPC, DHCP, DHCPV6, DNS, DNScrypt, DRDA, Deezer, Diameter, DirectConnect, Direct_Download_Link, Dofus, Dropbox, EAQ, EGP, FIX, FTP_CONTROL, FTP_DATA, Facebook, FacebookZero, FastTrack, Fiesta, Florensia, Free, Free, Free, Free, Free, Free_49, GMail, GRE, GTP, GenericProtocol, Git, Github, Gnutella, Google, GoogleDocs, GoogleDrive, GoogleHangout, GoogleMaps, GooglePlus, GoogleServices, Guildwars, H323, HEP, HTTP, HTTP_ActiveSync, HTTP_Connect, HTTP_Download, HTTP_Proxy, HalfLife2, Hotmail, HotspotShield, IAX, ICMP, ICMPV6, IFLIX, IGMP, IMAP, IMAPS, IPP, IP_in_IP, IPsec, IRC, IceCast, Instagram, KakaoTalk, KakaoTalk_Voice, Kerberos, Kontiki, LDAP, LISP, LLMNR, LastFM, LinkedIn, LotusNotes, MDNS, MGCP,

MPEG_TS, MQTT, MSN, MS_OneDrive, MapleStory, Megaco, Memcached, Messenger, Microsoft, Mining, MsSQL-TDS, MySQL, NFS, NOE, NTP, NestLogSink, NetBIOS, NetFlix, NetFlow, Nintendo, OCS, OSPF, Office365, Ookla, OpenDNS, OpenFT, OpenVPN, Oracle, Oscar, POP3, POPS, PPLive, PPStream, PPStream, PPTP, Pando_Media_Booster, Pandora, Pastebin, PcAnywhere, PlayStore, Playstation, PostgreSQL, QQ, QQLive, QUIC, RDP, RSYNC, RTCP, RTMP, RTP, RTSP, RX, Radius, Redis, RemoteScan, SAP, SCTP, SIP, SMBv1, SMBv23, SMPP, SMTP, SMTPS, SNMP, SOCKS, SOMEIP, SSDP, SSH, SSL, SSL_No_Cert, STUN, ShoutCast, Signal, Sina(Weibo), Skype, SkypeCall, Slack, Snapchat, Sopcast, Soulseek, SoundCloud, Spotify, Starcraft, Stealthnet, Steam, Syslog, TFTP, TINC, TVUplayer, TeamSpeak, TeamViewer, Telegram, Telnet, Teredo, Thunder, Tor, TruPhone, Tuenti, Tvants, Twitch, Twitter, UBNTAC2, UPnP, UbuntuONE, Unencrypted_Jabber, Usenet, VHUA, VMware, VNC, VRRP, Vevo, Viber, Warcraft3, Waze, WeChat, Webex, WhatsApp, WhatsAppFiles, WhatsAppVoice, Whois-DAS, Wikipedia, WindowsUpdate, WorldOfKungFu, WorldOfWarcraft, XDMCP, Xbox, Yahoo, YouTube, YouTubeUpload, Zattoo, ZeroMQ, eBay, eDonkey, ntop, sFlow.

7.10. IPSET

ipset – инструмент администрирования наборов IP-адресов.

В этом подразделе описана общая логика работы с данным инструментом.

Описанные далее команды можно напрямую использовать в консольном интерфейсе или использовать параметры в графическом (см. п. 7.8.3).

7.10.1. Синтаксис

```
ipset [ OPTIONS ] COMMAND [ COMMAND-OPTIONS ]
```

```
COMMANDS (команды) :=
{ create | add | del | test | destroy | list | save | restore | flush
| rename | swap | help | version | - }
```

```
OPTIONS (параметры):= { -exist | -output { plain | save | xml }
| -quiet | -resolve | -sorted | -name | -terse | -file filename }
```

```
ipset create SETNAME TYPENAME [ CREATE-OPTIONS ]
```

```
ipset add SETNAME ADD-ENTRY [ ADD-OPTIONS ]
```

```
ipset del SETNAME DEL-ENTRY [ DEL-OPTIONS ]
ipset test SETNAME TEST-ENTRY [ TEST-OPTIONS ]
ipset destroy [ SETNAME ]
ipset list [ SETNAME ]
ipset save [ SETNAME ]
ipset restore
ipset flush [ SETNAME ]
ipset rename SETNAME-FROM SETNAME-TO
ipset swap SETNAME-FROM SETNAME-TO
ipset help [ TYPENAME ]
ipset version
ipset -
```

7.10.2. Описание

ipset используется для установки, обслуживания и проверки так называемых наборов IP-адресов в ядре Linux. В зависимости от типа набора там могут храниться IP-адреса (IPv4/IPv6), номера портов (TCP/UDP), пары IP- и MAC-адресов, пары номеров портов и т. д. Определения типов наборов представлены в п. 7.10.5.

Совпадения (match) iptables и цели (target), относящиеся к наборам, создают ссылки, которые защищают данные наборы в ядре. Набор не может быть уничтожен, пока существует единственная ссылка, указывающая на него.

7.10.3. Параметры (options)

Параметры, распознаваемые ipset, можно разделить на несколько различных групп.

7.10.3.1. Команды (command)

Эти параметры определяют желаемое действие для выполнения. Можно указать только один из них, если иное не указано ниже. Для всех длинных версий имен команд. Синтаксический анализатор ipset следует указанному здесь порядку при поиске кратчайшего совпадения в длинных именах команд, поэтому следует

использовать ровно столько букв, чтобы ipset мог отличить их от всех других альтернативных команд.

Команды:

- n, create SETNAME TYPENAME [CREATE-OPTIONS]

Создает набор, идентифицированный с помощью имени набора и указанного типа. Для типа могут потребоваться параметры, специфичные для типа. Если указан параметр -exist, ipset игнорирует ошибку, которая в противном случае возникает, когда такой же набор (имя набора и параметры создания идентичны) уже существует;

- add SETNAME ADD-ENTRY [ADD-OPTIONS]

Добавляет заданную запись в набор. Если указан параметр -exist, ipset игнорирует его, если запись уже добавлена в набор;

- del SETNAME DEL-ENTRY [DEL-OPTIONS]

Удаляет запись из набора. Если указан параметр -exist и запись отсутствует в наборе (возможно, уже просрочена), то команда игнорируется;

- test SETNAME TEST-ENTRY [TEST-OPTIONS]

Проверяет, находится запись в наборе или нет. Номер состояния выхода равен нулю, если проверяемая запись есть в наборе, и отличному от нуля, если она отсутствует в наборе;

- x, destroy [*SETNAME*]

Уничтожает указанный набор или все наборы, если ни один из них не указан. Если у набора есть ссылки, ничего не делается и набор не уничтожается.

- list [*SETNAME*] [*OPTIONS*]

Перечисляет данные заголовка и записи для указанного набора, или для всех наборов, если они не указаны. Параметр -resolve (см. п. 7.10.3.2) можно использовать для принудительного поиска по имени, что может замедлять поиск. Когда указан параметр -sorted (см. п. 7.10.3.2), записи отображаются/сохраняются отсортированными, что также может замедлять. Параметр -output (см. п. 7.10.3.2) можно использовать для управления форматом списка: обычный (plain), сохранить (save) или xml,

по умолчанию используется обычный. Если указан параметр `-name` (см. п. 7.10.3.2), то будут перечислены только имена существующих наборов. Если указан параметр `-terse` (см. п. 7.10.3.2), то перечисляются только имена наборов и заголовки. Выходные данные выводятся на стандартный вывод, параметр `-file` (см. п. 7.10.3.2) можно использовать для указания имени файла вместо стандартного вывода;

- `save [ SETNAME ]`

Сохраняет заданный набор или все наборы, если ни один из них не передан в стандартный вывод (stdout), в формате, который может быть прочитан `restore`. Параметр `-file` (см. п. 7.10.3.2) можно использовать для указания имени файла вместо стандартного вывода;

- `restore`

Возобновляет сохраненный сеанс, сгенерированный с помощью `save`. Сохраненный сеанс можно загрузить из ввода (stdin) или использовать параметр `-file` (см. п. 7.10.3.2) для указания имени файла. Обратите внимание, что существующие наборы и элементы не удаляются при восстановлении, если это не указано в файле восстановления. В режиме восстановления разрешены все команды, кроме списка, справки, версии, интерактивного режима и самого восстановления;

- `flush [ SETNAME ]`

Очищает все записи из указанного набора или очищает все наборы, если ни один из них не задан;

- `e, rename SETNAME-FROM SETNAME-TO`

Переименовывает набор. Набор, идентифицируемый по `SETNAME-TO` не будет существовать;

- `w, swap SETNAME-FROM SETNAME-TO`

Позволяет менять местами содержимое двух наборов. Указанные наборы должны существовать, и можно менять местами только совместимые типы наборов;

- help [*TYPENAME*]

Выводит справку, если указано TYPENAME, то выводится справка по этому типу;

- version

Выводит версию ipset;

- –

Если в качестве команды указано тире, то ipset переходит в простой интерактивный режим и команды считываются со стандартного ввода. Интерактивный режим можно завершить, введя псевдокоманду quit.

7.10.3.2. Другие параметры

Можно указать следующие дополнительные параметры (таблица 19). Длинные имена параметров нельзя сокращать.

Таблица 19

Параметр	Описание
-!, -exist	Игнорирует ошибки, когда требуется создать точно такой же набор, или добавляется уже созданная запись, или удаляется отсутствующая запись.
-o, -output { plain|save|xml }	Выбирает формат вывода для команды list.
-q, -quiet	Предотвращает любой вывод в stdout и stderr ipset завершится с ошибкой, если не сможет продолжить работу.
-r, -resolve	При перечислении наборов использует принудительный поиск по имени. ipset попытается отобразить записи IP, разрешенные для имен хостов, для которых требуется поиск по slow DNS.
-s, -sorted	Отсортированный вывод. При перечислении или сохранении наборов записи перечисляются отсортированными.
-n, -name	Перечисляет только имена существующих наборов, т. е. не перечисляет заголовки и элементы набора.
-t, -terse	Перечисляет имена и заголовки наборов, элементы набора не выводит.
-f, -file *filename*	Выводит имя файла вместо стандартного вывода (совместно с командой list или save) или вместо стандартного ввода (совместно с командой restore).

7.10.4. Общие параметры для `create`, `add`

Тип набора состоит из метода хранения, с помощью которого хранятся данные, и типа(ов) данных, которые хранятся в наборе. Поэтому, параметр `TYPENAME` команды `create` соответствует синтаксису:

`TYPENAME := method:datatype[,datatype[,datatype]]`

где стандартным списком методов являются `bitmap`, `hash` и `list`, а возможными типами данных являются `ip`, `net`, `mac`, `port` и `iface`. Размерность набора равна количеству типов данных в имени его типа.

При добавлении, удалении или тестировании записей в наборе для параметра ввода команд должен использоваться тот же синтаксис данных, разделенных запятыми, т.е.

```
ipset add foo ipaddr,portnum,ipaddr
```

Если вместо IP-адресов или номеров служб используются имена хостов или сервисов, в имени которых есть тире, то имя хоста или сервисное имя должно быть заключено в квадратные скобки. Пример:

```
ipset add foo [test-hostname],[ftp-data]
```

В случае имен хостов распознаватель DNS вызывается внутри ipset, но, если он возвращает несколько IP-адресов, используется только первый из них.

Типы `bitman` и `list` используют хранилище фиксированного размера. Тип hash использует хэш для хранения элементов. Во избежание совпадений в хэше ограничено количество цепочек, а если оно исчерпано, то удваивается размер хэша при добавлении записей командой ipset. Когда добавляются записи iptables/ip6tables с помощью SET целей (target), размер хэша фиксируется, и набор не будет дублироваться, даже если новая запись не может быть добавлена в набор.

7.10.4.1. timeout

Все типы наборов поддерживают необязательный параметр `timeout` (время ожидания) при создании набора и добавлении записей. Значение параметра `timeout` для команды `create` означает значение времени ожидания по умолчанию (в секундах) для новых записей. Если набор создан с поддержкой `timeout`, то тот же параметр можно использовать для указания значений, отличных от значений по

умолчанию, при добавлении записей. Нулевое значение времени ожидания означает, что запись добавляется в набор на постоянной основе. Значение времени ожидания уже добавленных элементов можно изменить, повторно добавив элемент с помощью параметра `-exist`. Максимально возможное значение времени ожидания составляет 2147483 секунд.

Пример:

```
ipset create test hash:ip timeout 300
ipset add test 192.168.0.1 timeout 60
ipset -exist add test 192.168.0.1 timeout 600
```

Внаборе, количество записей, может быть больше, чем указанное количество записей для наборов с расширениями тайм-аута: количество записей в наборе обновляется при добавлении/удалении элементов в набор и, периодически, записи с истекшим временем ожидания удаляются.

7.10.4.2. counter, packets, bytes

Все типы наборов поддерживают необязательную опцию `counter` (счетчик) при создании набора. Если опция указана, то набор создается с помощью счетчиков, пакетов и байтов для элементов. Счетчики, пакеты и байты обнуляются, когда элементы повторно добавляются в набор, если только их значения не указаны явно параметрами `packets` и `bytes`. Пример добавления элемента в набор с ненулевыми значениями счетчика:

```
ipset create foo hash:ip counters
ipset add foo 192.168.1.1 packets 42 bytes 1024
```

7.10.4.3. comment

Все типы наборов поддерживают необязательное расширение `comment`. Включение этого расширения в ipset позволяет комментировать запись ipset произвольной строкой. Эта строка полностью игнорируется как ядром, так и самим ipset и предназначена исключительно для обеспечения удобного средства документирования причины существования записи. Комментарии не должны содержать кавычек, при этом символ \ не имеет значения. Например, следующая команда недопустима:

```
ipset add foo 1.1.1.1 comment "this comment is \"bad\""
```

В приведенном выше примере ipset увидит кавычки в поле для комментария, что приведет к ошибке синтаксического анализа. Следует избегать создания комментариев, содержащих кавычки, если не хотите нарушать «ipset save» и «ipset restore», тем не менее, вполне допустима следующая запись:

```
ipset create foo hash:ip comment
ipset add foo 192.168.1.1/24 comment "allow access to SMB share
on \\\\fileserv\\"
the above would appear as: "allow access to SMB share on
\\fileserv\"
```

7.10.4.4. skbinfo, skbmark, skbprio, skbqueue

Все типы наборов поддерживают дополнительное расширение `skbinfo`. Это расширение позволяет хранить метаинформацию (метка межсетевого экрана, класс tc и аппаратная очередь (hardware queue)) для каждой записи и сопоставлять ее с пакетами с помощью SET netfilter target с опцией `--map-set`. Формат параметра `skbmark`: MARK или MARK/MASK, где MARK и MASK – 32-битные шестнадцатеричные числа с префиксом 0x. Если используется только метка mark, то указывается маска 0xffffffff. Параметр `skbprio` имеет формат класса tc: MAJOR:MINOR, где MAJOR и MINOR являются шестнадцатеричными числами без префикса 0x. Параметр `skbqueue` это просто десятичное число.

```
ipset create foo hash:ip skbinfo
ipset add foo skbmark 0x1111/0xff00ffff skbprio 1:10 skbqueue 10
```

7.10.4.5. hashsize

Этот параметр действителен для команды `create` для всех наборов типа hash. Он определяет начальный размер хэша для набора, по умолчанию 1024. Размер хэша должен быть степенью двойки, ядро автоматически округляет размер хэша, не равный степени двойки, до первого правильного значения. Пример:

```
ipset create test hash:ip hashsize 1536
```

7.10.4.6. maxelem

Этот параметр действителен для команды `create` для всех наборов типа hash. Он определяет максимальное количество элементов, которые могут храниться в наборе, по умолчанию 65536. Пример:

```
ipset create test hash:ip maxelem 2048.
```

7.10.4.7. family { inet | inet6 }

Этот параметр действителен для команды `create` для всех наборов типа hash, кроме hash:mac. Он определяет протоколы IP-адресов, которые будут храниться в наборе. По умолчанию используется inet, т. е. IPv4. Для inet family можно добавить или удалить несколько записей, указав диапазон или сеть IPv4-адресов в части записи IP-адреса:

```
ipaddr := { ip | fromaddr-toaddr | ip/cidr }
netaddr := { fromaddr-toaddr | ip/cidr }
```

Пример:

```
ipset create test hash:ip family inet6
```

7.10.4.8. nomatch

Типы наборов hash, которые могут хранить данные сетевого типа (например, hash:*net*), поддерживают необязательную опцию `nomatch` при добавлении записей. При сопоставлении элементов в наборе элементы, помеченные как `nomatch`, пропускаются, как если бы они не были добавлены в набор, что позволяет формировать наборы с исключениями. Пример хэш-типа hash:net рассмотрен в п. 7.10.5.7.

Когда элементы проверяются ipset, учитываются флаги nomatch. Если кто-то хочет проверить наличие элемента, помеченного как nomatch в наборе, то флаг также должен быть указан.

7.10.4.9. forceadd

Все типы наборов хэшей поддерживают необязательный параметр forceadd при создании набора. Когда наборы, созданные с помощью этой опции, заполнятся, следующее добавление к этому набору может завершиться успешно и исключить случайную запись набора.

Пример:

```
ipset create foo hash:ip forceadd
```

7.10.5. Типы наборов

7.10.5.1. bitmap:ip

Тип набора `bitmap:ip` использует диапазон памяти для хранения IPv4 хостов (по умолчанию) или сетевых адресов IPv4. Набор типа `bitmap:ip` может хранить до 65536 записей.

Синтаксис:

```
CREATE-OPTIONS := range fromip-toip|ip/cidr [ netmask cidr ]
[ timeout value ] [ counters ] [ comment ] [ skbinfo ]

ADD-ENTRY := { ip | fromip-toip | ip/cidr }

ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ]
[ comment string ]          [ skbmark value ]          [ skbprio value ]
[ skbqueue value ]

DEL-ENTRY := { ip | fromip-toip | ip/cidr }

TEST-ENTRY := ip
```

Обязательные параметры для `create`:

`range fromip-toip|ip/cidr`

Создает набор из указанного диапазона адресов, выраженного диапазоном адресов IPv4 или сетью. Размер диапазона (в записях) не может превышать ограничение в 65536 элементов.

Дополнительные параметры для `create`:

`netmask cidr`

Если указан необязательный параметр `netmask`, сетевые адреса будут храниться в наборе вместо IP-адресов узлов.

Значение префикса `cidr` должно находиться в диапазоне от 1 до 32. IP-адрес будет указан в наборе, если его сетевой адрес, будет соответствовать результату маскировки адреса для указанной маски сети.

Тип `bitmap:ip` поддерживает добавление или удаление нескольких записей в одной команде.

Примеры:

```
ipset create foo bitmap:ip range 192.168.0.0/16
ipset add foo 192.168.1/24
ipset test foo 192.168.1.1
```

7.10.5.2. bitmap:ip,mac

Тип набора `bitmap:ip,mac` использует диапазон памяти для хранения пар IPv4 и MAC-адресов. Набор типа `bitmap:ip,mac` может хранить до 65536 записей.

Синтаксис:

```
CREATE-OPTIONS := range fromip-toip|ip/cidr [ timeout value ]
[ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := ip[,macaddr]
ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ]
[ comment string ]          [ skbmark value ]          [ skbprio value ]
[ skbqueue value ]
DEL-ENTRY := ip[,macaddr]
TEST-ENTRY := ip[,macaddr]
```

Обязательные параметры для `create` набора типа `bitmap:ip,mac`:

`range fromip-toip|ip/cidr`

Создает набор из включенного указанного диапазона адресов, выраженного в диапазоне IPv4 или сети. Размер диапазона не может превышать ограничение в 65536 записей.

Тип `bitmap:ip,mac` является исключительным в том смысле, что часть MAC может быть опущена при добавлении/удалении/тестировании записей в наборе. Если добавить запись без указанного MAC-адреса, то при первом совпадении записи ядро автоматически заполнит недостающий MAC-адрес MAC-адресом из пакета. MAC-адрес источника используется, если запись соответствует параметру `src` набора соответствий, а MAC-адрес назначения используется, если он доступен, и запись соответствует параметру `dst`. Если запись была указана со значением тайм-аута, таймер запустится, после завершения сопоставления пар IP- и MAC-адресов.

Для наборов типа `bitmap:ip,mac` требуются два параметра src/dst для модулей ядра set match и SET target netfilter. Информацию о совпадениях по MAC-адресам назначения смотрите в п. 7.10.5.17.

Примеры:

```
ipset create foo bitmap:ip,mac range 192.168.0.0/16
ipset add foo 192.168.1.1,12:34:56:78:9A:BC
ipset test foo 192.168.1.1
```

7.10.5.3. bitmap:port

Тип набора `bitmap:port` использует диапазон памяти для хранения до 65536 номеров портов.

Синтаксис:

```
CREATE-OPTIONS := range fromport-toport [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := { [proto:]port | [proto:]fromport-toport }
ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := { [proto:]port | [proto:]fromport-toport }
TEST-ENTRY := [proto:]port
```

Обязательные параметры, используемые для `create`:

`range [proto:]fromport-toport`

Создает набор в указанном диапазоне.

Модули ядра set match и SET target netfilter интерпретируют сохраненные номера как номера портов TCP или UDP.

`proto` необходимо указывать только в том случае, если используется имя службы, и это имя не существует в качестве службы TCP.

Пример:

```
ipset create foo bitmap:port range 0-1024
ipset add foo 80
ipset test foo 80
ipset del foo udp:[macon-udp]-[tn-tl-w2]
```

7.10.5.4. hash:ip

Тип набора `hash:ip` использует хэш для хранения IP-адресов узлов (по умолчанию) или сетевых адресов. IP-адрес с нулевым значением не может храниться в наборе типа `hash:ip`.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ netmask cidr ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := ipaddr
```

ADD-OPTIONS := [timeout *value*] [packets *value*] [bytes *value*] [comment *string*] [skbmark *value*] [skbprio *value*] [skbqueue *value*]

DEL-ENTRY := *ipaddr*

TEST-ENTRY := *ipaddr*

Дополнительные параметры для `create`:

`netmask cidr`

Если указан необязательный параметр `netmask`, сетевые адреса будут храниться в наборе вместо IP-адресов узлов. Значение префикса `cidr` должно находиться в диапазоне от 1 до 32 для IPv4 и от 1 до 128 для IPv6. IP-адрес будет указан в наборе, если его сетевой адрес, будет соответствовать результату маскировки адреса для указанной маски сети.

Примеры:

```
ipset create foo hash:ip netmask 30
ipset add foo 192.168.1.0/24
ipset test foo 192.168.1.2
```

7.10.5.5. hash:mac

Тип набора `hash:mac` использует хэш для хранения MAC-адресов. MAC-адреса с нулевым значением не могут храниться в наборе типа `hash:mac`. Информацию о совпадениях по MAC-адресам назначения смотрите в п. 7.10.5.17.

Синтаксис:

CREATE-OPTIONS := [hashsize *value*] [maxelem *value*] [timeout *value*] [counters] [comment] [skbinfo]

ADD-ENTRY := *macaddr*

ADD-OPTIONS := [timeout *value*] [packets *value*] [bytes *value*] [comment *string*] [skbmark *value*] [skbprio *value*] [skbqueue *value*]

DEL-ENTRY := *macaddr*

TEST-ENTRY := *macaddr*

Примеры:

```
ipset create foo hash:mac
ipset add foo 01:02:03:04:05:06
ipset test foo 01:02:03:04:05:06
```

7.10.5.6. hash:ip,mac

Тип набора `hash:ip,mac` использует хэш для хранения пар IP- и MAC-адресов. MAC-адреса с нулевым значением не могут храниться в наборе типа `hash:ip,mac`. Информацию о совпадениях по MAC-адресам назначения смотрите в п. 7.10.5.17.

Синтаксис:

```
    CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ]
[ maxelem value ]    [ timeout value ]    [ counters ]    [ comment ]
[ skbinfo ]
    ADD-ENTRY := ipaddr,macaddr
    ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ]
[ comment string ]         [ skbmark value ]         [ skbprio value ]
[ skbqueue value ]
    DEL-ENTRY := ipaddr,macaddr
    TEST-ENTRY := ipaddr,macaddr
```

Примеры:

```
    ipset create foo hash:ip,mac
    ipset add foo 1.1.1.1,01:02:03:04:05:06
    ipset test foo 1.1.1.1,01:02:03:04:05:06
```

7.10.5.7. hash:net

Тип набора `hash:net` использует хэш для хранения сетевых IP-адресов разного размера. Сетевой адрес с нулевым размером префикса не может храниться в наборах этого типа.

Синтаксис:

```
    CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ]
[ maxelem value ]    [ timeout value ]    [ counters ]    [ comment ]
[ skbinfo ]
    ADD-ENTRY := netaddr
    ADD-OPTIONS :=  [ timeout value ]  [ nomatch ]  [ packets value ]
[ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ]
[ skbqueue value ]
    DEL-ENTRY := netaddr
    TEST-ENTRY := netaddr
    where netaddr := ip[/cidr]
```

При добавлении/удалении/тестировании записей, если параметр префикса `cidr` не указан, предполагается значение префикса хоста. При добавлении/удалении записей добавляется/удаляется точный элемент, а перекрывающиеся элементы не проверяются ядром. При тестировании записей, если проверяется адрес хоста, ядро пытается сопоставить адрес в сетях, добавленных в набор, и сообщает результат соответственно.

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный префикс) до самого большого добавленного в набор (наименее специфичный префикс). При добавлении/удалении IP-адресов в набор с помощью SET netfilter target он будет добавлен/удален по наиболее специфичному префиксу, который можно найти в наборе, или по значению префикса хоста, если набор пуст.

Время поиска растет линейно с количеством различных значений префикса, добавляемых к набору.

Пример:

```
ipset create foo hash:net
ipset add foo 192.168.0.0/24
ipset add foo 10.1.0.0/16
ipset add foo 192.168.0/24
ipset add foo 192.168.0/30 nomatch
```

При сопоставлении элементов в указанном выше наборе будут соответствовать все IP-адреса из сетей 192.168.0.0/24, 10.1.0.0/16 и 192.168.0/24, кроме адресов из 192.168.0/30.

7.10.5.8. hash:net,net

Тип набора `hash:net,net` использует хэш для хранения пар сетевых IP-адресов разного размера. Важно знать, что первый параметр имеет приоритет над вторым, поэтому запись о несоответствии может быть потенциально неэффективной, если существует более специфичный первый параметр с подходящим вторым параметром. Сетевой адрес с нулевым размером префикса не может быть сохранен в наборе этого типа.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := netaddr,netaddr
ADD-OPTIONS := [ timeout value ] [ nomatch ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := netaddr,netaddr
TEST-ENTRY := netaddr,netaddr
where netaddr := ip[/cidr]
```

При добавлении/удалении/тестировании записей, если параметр префикса `cidr` не указан, предполагается значение префикса хоста. При добавлении/удалении записей добавляется/удаляется точный элемент, а перекрывающиеся элементы не проверяются ядром. При тестировании записей, если проверяется адрес хоста, ядро пытается сопоставить адрес хоста в сетях, добавленных в набор, и сообщает результат соответственно.

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный префикс) до наибольшего (наименее специфичный префикс) с первым параметром, имеющим приоритет. При добавлении/удалении IP-адресов в набор с SET netfilter target он будет добавлен/удален по наиболее специфичному префиксу, который можно найти в наборе, или по значению префикса хоста, если набор пуст.

Время поиска растет линейно с количеством различных значений префикса, добавляемых к первому параметру набора. Количество вторичных префиксов еще больше увеличивается, поскольку список вторичных префиксов просматривается для каждого первичного префикса.

Пример:

```
ipset create foo hash:net,net
ipset add foo 192.168.0.0/24,10.0.1.0/24
ipset add foo 10.1.0.0/16,10.255.0.0/24
ipset add foo 192.168.0/24,192.168.54.0-192.168.54.255
ipset add foo 192.168.0/30,192.168.64/30 nomatch
```

При сопоставлении элементов в указанном выше наборе будут соответствовать все IP-адреса из сетей 192.168.0.0/24<->10.0.1.0/24, 10.1.0.0/16<->10.255.0.0/24 и 192.168.0/24. <->192.168.54.0/24 кроме адресов 192.168.0/30<->192.168.64/30.

7.10.5.9. hash:ip,port

Тип набора `hash:ip,port` использует хэш для хранения пар IP-адреса и номера порта. Номер порта интерпретируется вместе с протоколом (по умолчанию TCP), и нулевой номер протокола не может использоваться.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := ipaddr,[proto:]port
ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := ipaddr,[proto:]port
TEST-ENTRY := ipaddr,[proto:]port
```

[proto:]port элемент, где допустимы варианты диапазона при добавлении или удалении записей, может быть выражен в следующих формах:

- *portname*[*-portname*]

  Порт TCP или диапазон портов, выраженный в идентификаторах имен портов TCP из `/etc/services`;

- *portnumber*[*-portnumber*]

  Порт TCP или диапазон портов, выраженный в номерах портов TCP;

- `tcp|sctp|udp|udplite:`*portname*|*portnumber*[*-portname*|*portnumber*]

  TCP, SCTP, UDP или UDPLITE порт или диапазон портов, выраженный в именах или номерах портов;

- `icmp:`*codename*|*type/code*

  Кодовое имя ICMP или тип/код. Поддерживаемые идентификаторы кодовых имен ICMP всегда можно просмотреть с помощью команды help;

- `icmpv6:`*`codename|type/code`*

  Кодовое имя ICMPv6 или тип/код. Поддерживаемые идентификаторы кодовых имен ICMPv6 всегда можно просмотреть с помощью команды help;

- `proto:0`

  Все остальные протоколы, как идентификатор из `/etc/protocols` или номер. Номер псевдопорта должен быть равен нулю.

Наборы типа `hash:ip,port` требуют сопоставления двух параметров `src`/`dst` для модулей ядра set match и SET target.

Примеры:

```
ipset create foo hash:ip,port
ipset add foo 192.168.1.0/24,80-82
ipset add foo 192.168.1.1,udp:53
ipset add foo 192.168.1.1,vrrp:0
ipset test foo 192.168.1.1,80
```

7.10.5.10. hash:net,port

Тип набора `hash:net,port` использует хэш для хранения разных размеров пары сетевого IP-адреса и порта. Номер порта интерпретируется вместе с протоколом (по умолчанию TCP), нулевой номер протокола не может использоваться. Сетевой адрес с нулевым размером префикса также не принимается.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := netaddr,[proto:]port
ADD-OPTIONS := [ timeout value ] [ nomatch ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := netaddr,[proto:]port
TEST-ENTRY := netaddr,[proto:]port
where netaddr := ip[/cidr]
```

Описание элемента `netaddr` приведено в hash:net (п. 7.10.5.7). Для элемента [*proto:*]*port* описание приведено в hash:ip,port (п. 7.10.5.9).

При добавлении/удалении/тестировании записей, если параметр префикса `cidr` не указан, предполагается значение префикса хоста. При добавлении/удалении записей добавляется/удаляется точный элемент, а перекрывающиеся элементы не проверяются ядром. При тестировании записей, если проверяется адрес хоста, ядро пытается сопоставить адрес хоста в сетях, добавленных в набор, и сообщает результат соответственно.

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный префикс) до самого большого добавленного в набор (наименее специфичный префикс). При добавлении/удалении IP-адресов в набор с помощью SET netfilter target он будет добавлен/удален по наиболее специфичному префиксу, который можно найти в наборе, или по значению префикса хоста, если набор пуст.

Время поиска растет линейно с количеством различных значений префикса, добавляемых к набору.

Примеры:

```
ipset create foo hash:net,port
ipset add foo 192.168.0/24,25
ipset add foo 10.1.0.0/16,80
ipset test foo 192.168.0/24,25
```

7.10.5.11. hash:ip,port,ip

Тип набора `hash:ip,port,ip` использует следующий набор из трех параметров (триплет): хэш для хранения IP-адреса, номер порта и второй IP-адрес. Номер порта интерпретируется вместе с протоколом (по умолчанию TCP), нулевой номер протокола не может использоваться.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := ipaddr,[proto:]port,ip
```

ADD-OPTIONS := [timeout *value*] [packets *value*] [bytes *value*] [comment *string*] [skbmark *value*] [skbprio *value*] [skbqueue *value*]

DEL-ENTRY := *ipaddr*,[*proto:*]*port*,*ip*

TEST-ENTRY := *ipaddr*,[*proto:*]*port*,*ip*

Описание элементов ipaddr и [proto:]port parts представлены в описании hash:ip,port (п. 7.10.5.9).

Для наборов типа hash:ip,port,ip требуются три параметра src/dst для модулей ядра set match и SET target.

Примеры:

```
ipset create foo hash:ip,port,ip
ipset add foo 192.168.1.1,80,10.0.0.1
ipset test foo 192.168.1.1,udp:53,10.0.0.1
```

7.10.5.12. hash:ip,port,net

Тип набора hash:ip,port,net использует следующий набор из трех параметров (триплет): хэш для хранения IP-адреса, номер порта и сетевой IP-адрес. Номер порта интерпретируется вместе с протоколом (по умолчанию TCP), нулевой номер протокола не может использоваться. Сетевой адрес с нулевым размером префикса также не может быть сохранен.

Синтаксис:

CREATE-OPTIONS := [family { inet | inet6 }] | [hashsize *value*] [maxelem *value*] [timeout *value*] [counters] [comment] [skbinfo]

ADD-ENTRY := *ipaddr*,[*proto:*]*port*,*netaddr*

ADD-OPTIONS := [timeout *value*] [nomatch] [packets *value*] [bytes *value*] [comment *string*] [skbmark *value*] [skbprio *value*] [skbqueue *value*]

DEL-ENTRY := *ipaddr*,[*proto:*]*port*,*netaddr*

TEST-ENTRY := *ipaddr*,[*proto:*]*port*,*netaddr*

где *netaddr* := *ip*[*/cidr*]

Описание элементов ipaddr и [proto:]port размещено в hash:ip,port п.7.10.5.9. Для netaddr описание приведено в hash:net п. 7.10.5.7.

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный cidr) до самого большого добавленного в набор (наименее специфичный cidr). При добавлении/удалении триплетов в набор SET netfilter target, будет добавлен/удален наиболее специфичный префикс, который может быть найден в наборе, или значение префикса хоста, если набор пуст.

Время поиска растет линейно с количеством различных значений `cidr`, добавляемых в набор.

Наборы типа `hash:ip,port,net` требуют трех параметров src/dst для модулей ядра set match и SET target.

Примеры:

```
ipset create foo hash:ip,port,net
ipset add foo 192.168.1,80,10.0.0/24
ipset add foo 192.168.2,25,10.1.0.0/16
ipset test foo 192.168.1,80,10.0.0/24
```

7.10.5.13. hash:ip,mark

Тип набора `hash:ip,mark` использует хэш для хранения пар IP-адреса и метки пакета.

Синтаксис:

```
    CREATE-OPTIONS := [ family { inet | inet6 } ] | [ markmask value ]
[ hashsize value ]  [ maxelem value ]  [ timeout value ]  [ counters ]
[ comment ] [ skbinfo ]
    ADD-ENTRY := ipaddr,mark
    ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ]
[ comment string ]          [ skbmark value ]          [ skbprio value ]
[ skbqueue value ]
    DEL-ENTRY := ipaddr,mark
    TEST-ENTRY := ipaddr,mark
```

Дополнительные параметры для `create`:

`markmask value`

Позволяет установить биты в метке пакета. Эти значения используются для выполнения побитовой операции AND для каждой добавленной метки. `markmask` может быть любым значением от 1 до 4294967295, по умолчанию – 32 бита.

`mark` может быть любым значением от 0 до 4294967295.

Наборы типа `hash:ip,mark` требуют двух параметров src/dst для модулей ядра set match и SET target.

Примеры:

```
ipset create foo hash:ip,mark
ipset add foo 192.168.1.0/24,555
ipset add foo 192.168.1.1,0x63
ipset add foo 192.168.1.1,111236
```

7.10.5.14. hash:net,port,net

Тип набора `hash:net,port,net` ведет себя аналогично `hash:ip,port,net`, но принимает значение `cidr` как для первого, так и для последнего параметра. Любая подсеть может быть /0, если нужно сопоставить порт между всеми пунктами назначения.

Синтаксис:

```
CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize value ] [ maxelem value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := netaddr,[proto:]port,netaddr
ADD-OPTIONS := [ timeout value ] [ nomatch ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := netaddr,[proto:]port,netaddr
TEST-ENTRY := netaddr,[proto:]port,netaddr
where netaddr := ip[/cidr]
```

Описание элементов `[proto:]port` представлено в hash:ip,port (п. 7.10.5.9), для `netaddr` описание в hash:net (п. 7.10.5.7).

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный cidr) до самого большого (наименее специфичный `cidr`), добавленного в набор. При добавлении/удалении триплетов в набор SET netfilter target, он будет добавлен/удален по наиболее специфичному префиксу `cidr`, который будет найден в наборе, или по `cidr` хоста, если набор пуст. Первая подсеть имеет приоритет при выполнении наиболее конкретного поиска, как и для hash:net,net.

Время поиска увеличивается линейно с количеством различных значений `cidr`, добавляемых в набор, и с количеством вторичных значений `cidr` для каждого первичного.

Наборы типа `hash:net,port,net` требуют трех параметров src/dst для модулей ядра set match и SET target.

Примеры:

```
ipset create foo hash:net,port,net
ipset add foo 192.168.1.0/24,0,10.0.0/24
ipset add foo 192.168.2.0/24,25,10.1.0.0/16
ipset test foo 192.168.1.1,80,10.0.0.1
```

7.10.5.15. hash:net,iface

Тип набора `hash:net,iface` использует хэш для хранения разного размеров пар сетевых IP-адресов и имен интерфейсов.

Синтаксис:

`CREATE-OPTIONS := [ family { inet | inet6 } ] | [ hashsize` *value* `] [ maxelem` *value* `] [ timeout` *value* `] [ counters ] [ comment ] [ skbinfo ]`

`ADD-ENTRY :=` *netaddr*`,[physdev:]`*iface*

`ADD-OPTIONS := [ timeout` *value* `] [ nomatch ] [ packets` *value* `] [ bytes` *value* `] [ comment` *string* `] [ skbmark` *value* `] [ skbprio` *value* `] [ skbqueue` *value* `]`

`DEL-ENTRY :=` *netaddr*`,[physdev:]`*iface*

`TEST-ENTRY :=` *netaddr*`,[physdev:]`*iface*

где *netaddr* := *ip*[*/cidr*]

Для `netaddr` описание представлено в hash:net (п. 7.10.5.7).

При добавлении/удалении/тестировании записей, если параметр префикса `cidr` не указан, предполагается значение префикса хоста. При добавлении/удалении записей добавляется/удаляется точный элемент, а перекрывающиеся элементы не проверяются ядром. При тестировании записей, если проверяется адрес хоста, ядро пытается сопоставить адрес хоста с сетями, добавленными в набор, и сообщает результат.

С точки зрения совпадений set netfilter поиск совпадений всегда начинается с наименьшего размера сетевого блока (наиболее специфичный префикс) до самого большого добавленного в набор (наименее специфичный префикс). При добавлении/удалении IP-адресов в набор с помощью SET netfilter target он будет добавлен/удален по наиболее специфичному префиксу, который можно найти в наборе, или по значению префикса хоста, если набор пуст.

Второй параметр направления set match и SET target соответствует входящему/исходящему интерфейсу: src – входящему (аналогично флагу `-i` в iptables), a dst – исходящему (аналогично флагу `-o` в iptables). Когда интерфейс помечен с помощью `physdev:`, интерфейс интерпретируется как входящий/исходящий порт моста.

Время поиска растет линейно с количеством различных значений префикса, добавляемых к набору.

Внутреннее ограничение типа набора `hash:net,iface` заключается в том, что один и тот же сетевой префикс не может быть сохранен более чем с 64 различными интерфейсами в одном наборе.

Примеры:

```
ipset create foo hash:net,iface
ipset add foo 192.168.0/24,eth0
ipset add foo 10.1.0.0/16,eth1
ipset test foo 192.168.0/24,eth0
```

7.10.5.16. list:set

Тип `list:set` использует простой список, в котором можно хранить имена наборов.

Синтаксис:

```
CREATE-OPTIONS := [ size value ] [ timeout value ] [ counters ] [ comment ] [ skbinfo ]
ADD-ENTRY := setname [ { before | after } setname ]
ADD-OPTIONS := [ timeout value ] [ packets value ] [ bytes value ] [ comment string ] [ skbmark value ] [ skbprio value ] [ skbqueue value ]
DEL-ENTRY := setname [ { before | after } setname ]
TEST-ENTRY := setname [ { before | after } setname ]
```

С помощью команды ipset можно добавлять, удалять и тестировать имена наборов в типе набора `list:set`.

В set match икb SET target netfilter можно тестировать, добавлять или удалять записи в наборах, добавленных в тип `list:set`. match пытается найти соответствующую запись в наборах, а target пытается добавить запись в первый набор, к которому она может быть добавлена. Важно количество вариантов опций направления match и target: наборы, которые требуют больше параметров, чем указано, пропускаются, а элемменты, проверяемых наборов с равными или меньшими параметрами, добавляются/удаляются. Например, если *a* и *b* являются наборами типа `list:set`, то в команде

```
iptables -m set --match-set a src,dst -j SET --add-set b src,dst
```

match и target пропустят любой набор *a* и *b*, который хранит триплеты данных, но будет сопоставлять все наборы с одинарным или двойным хранением данных и прекратит сопоставление при первом успешном наборе, и добавит src к первому одиночному, или src, dst к первому двойному набору хранения данных в *b*, к которому можно добавить запись. Можно представить набор типа `list:set` как упорядоченное объединение элементов набора.

Обратите внимание: с помощью команды ipset можно добавлять, удалять и проверять имена наборов в типах list:set, а не наличие члена набора (например, IP-адреса).

7.10.5.17. Комментарии

При хранении подсети одинакового размера из данной сети (скажем, блоки /24 из сети /8), используйте тип набора `bitmap:ip`. Если хранить случайные сети одинакового размера (скажем, случайные блоки /24), следует использовать тип набора hash:ip. При случайном размере сетевых блоков, используйте `hash:net`.

Сопоставление MAC-адресов назначения с использованием параметра dst модуля ядра set netfilter будет работать только в том случае, если MAC-адрес назначения доступен в пакете на данном этапе обработки, то есть это применимо только для входящих пакетов в PREROUTING, INPUT и цепочки FORWARD против MAC-адреса, изначально найденного в полученном пакете

(обычно это один из MAC-адресов локального хоста). Это не MAC-адрес назначения, IP-адрес назначения назначается после маршрутизации. Если MAC-адрес недоступен (например, в цепочке OUTPUT), пакет просто не будет соответствовать.

Обратная совместимость сохраняется, и старый синтаксис ipset по-прежнему поддерживается.

Типы наборов `iptree` и `iptreemap` будут автоматически заменяться наборами типа `hash:ip`.

7.11. Примеры использования

7.11.1. Блокировка диапазонов IP-адресов

Сначала необходимо добавить список IPSET и диапазоны IP-адресов в нем для блокировки (см. п. 7.8.3)(рис. 43, рис. 44).

Рис. 43

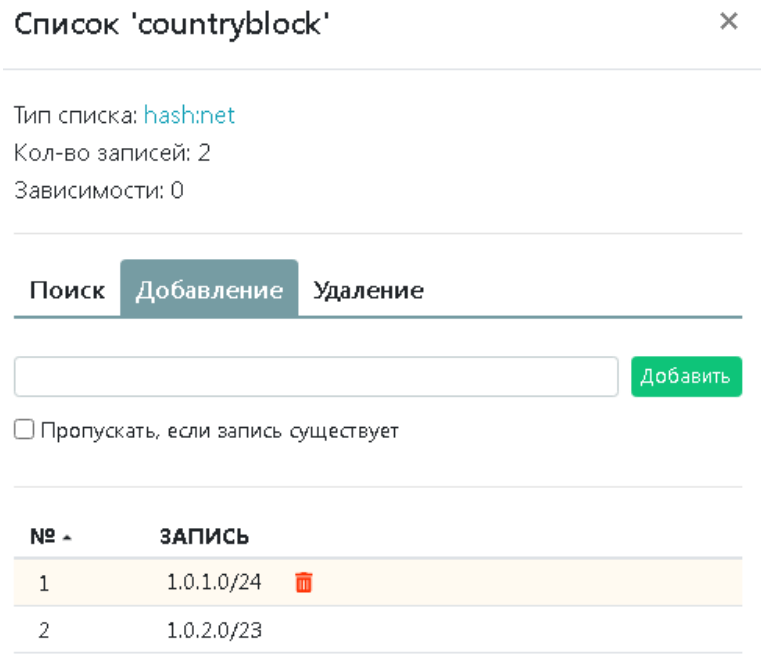


Рис. 44

Для блокировки трафика, исходящего из диапазонов IP-адресов, определенных подсетей в вышеописанном сгенерированном наборе IPSET countryblock, необходимо добавить следующее правило (см. п. 7.8.2):

```
# iptables -A INPUT -m set --match-set countryblock src -j DROP
```

Для добавления правила выберите таблицу фильтрации пакетов (filter) (п. 7.4) и интересующую в ней цепочку, нажмите на кнопку + Добавить правило , соответствующую цепочке INPUT.

Далее выбрать «Действия и переходы»: Действие и переход к цепочке (-j) и DROP (отбросить). В поле «Дополнительные параметры» выбрать `set` и нажать на кнопку «Добавить критерий», активировать с помощью переключателя опцию `--match-set` выбрать из списка наименование IPSET набора `countryblock` и указать для каких пакетов использовать – src источник. Далее нажать кнопку «Сохранить».

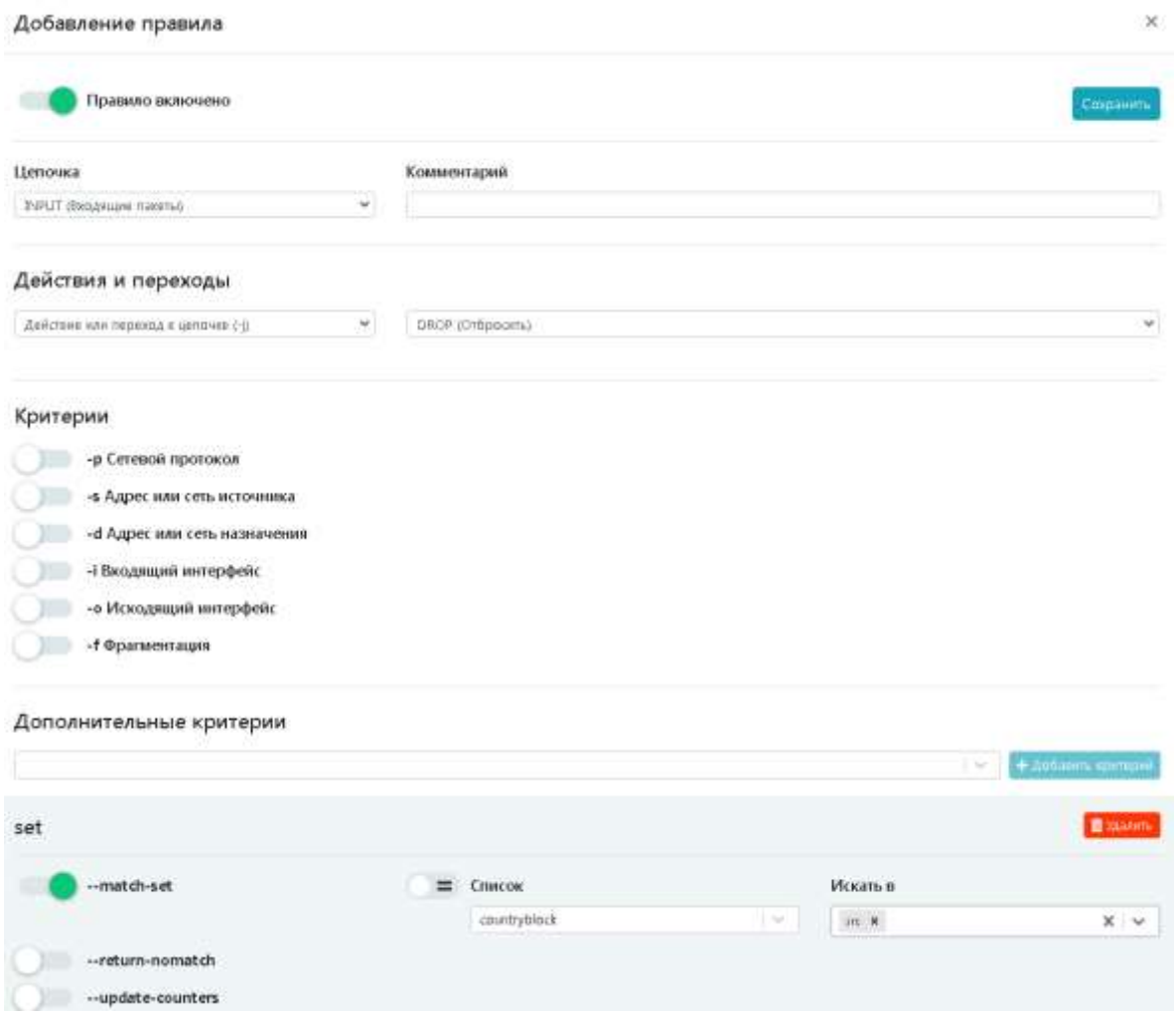


Рис. 45

Добавленное правило (рис. 46) можно увидеть при просмотре  файла конфигурации (см. рис. 27).

Исходный формат: /etc/sysconfig/iptables

```
1 *filter
2 :INPUT ACCEPT [0:0]
3 :FORWARD ACCEPT [0:0]
4 :OUTPUT ACCEPT [0:0]
5 :TEST - [0:0]
6 -A OUTPUT -j NFQUEUE --queue-num 0 --queue-bypass
7 -A FORWARD -j NFQUEUE --queue-num 0 --queue-bypass
8 -A INPUT -j NFQUEUE --queue-num 0 --queue-bypass
9 -A INPUT -m set --match-set countryblock src -j DROP
```

Рис. 46

7.11.2. Блокировка трафика других стран

Для обновления базы данных GeoIP в консольном интерфейсе МЭ ИВК КОЛЬЧУГА-К выполнить команду:

```
sudo /usr/libexec/xtables-addons/geoip-update.sh
```

Для блокировки входящего трафика, например, из ОАЭ (AE) и Соединенных Штатов (США, US)(коды стран см. в п. 7.9.6), необходимо использовать следующую команду iptables:

```
# iptables -I INPUT -m geoip --src-cc AE, US -j DROP
```

Чтобы заблокировать весь входящий, например, некитайский трафик:

```
# iptables -I INPUT -m geoip ! --src-cc CN -j DROP
```

Блокировка исходящего трафика, предназначенного для Индии (IN):

```
# iptables -A OUTPUT -m geoip --dst-cc IN -j DROP
```

Введем правило для исходящего трафика в графическом интерфейсе, для этого, перейдем на вкладку раздела «Сеть» (см. п. 5.3) подраздел «Межсетевой экран» (см. рис. 24), выберем цепочку OUTPUT, нажать кнопку «Добавить правило». Далее заполним «Действия и переходы»: Действие и переход к цепочке (-j) и DROP (отбросить). В поле «Дополнительные параметры» выбрать `geoip` и нажать на кнопку «Добавить критерий». Активировать переключателем параметр

назначения `--dst-cc` и в выпадающем списке выбрать нужную страну, нажать «Сохранить» (рис. 47).

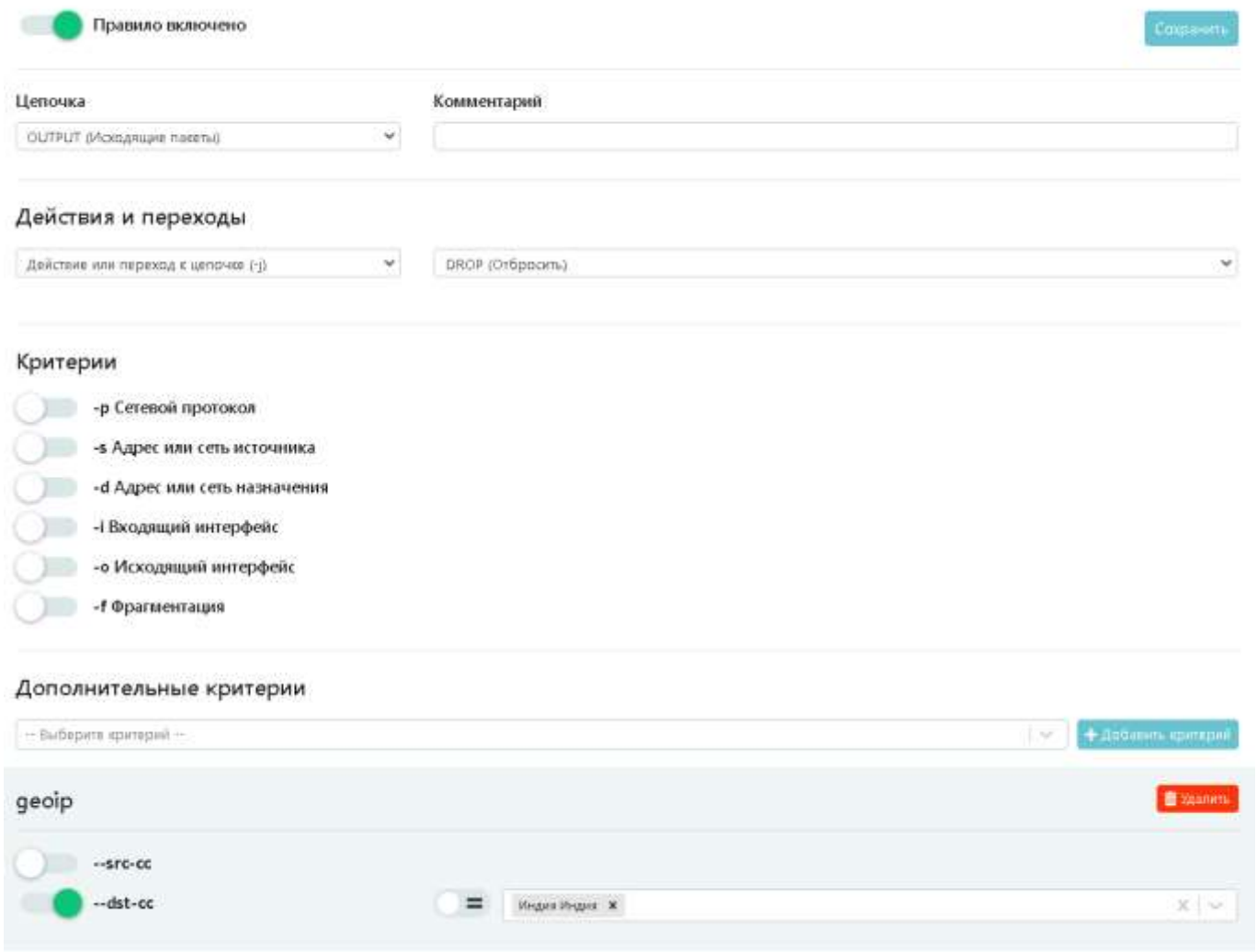


Рис. 47

Добавленное правило (рис. 48) можно увидеть при просмотре  файла конфигурации (см. рис. 27).

Исходный формат: /etc/sysconfig/iptables

```
1 *filter
2 :INPUT ACCEPT [0:0]
3 :FORWARD ACCEPT [0:0]
4 :OUTPUT ACCEPT [0:0]
5 :TEST - [0:0]
6 -A OUTPUT -m geoip --dst-cc IN -j DROP
```

Рис. 48

8. ИНТЕРАКТИВНАЯ ПАНЕЛЬ МОНИТОРИНГА

Интерактивная панель мониторинга (рис. 49) отображает состояние производительности и работоспособности элементов системы МЭ в реальном времени, представляет графически множество метрик, разбитых на группы и подгруппы.

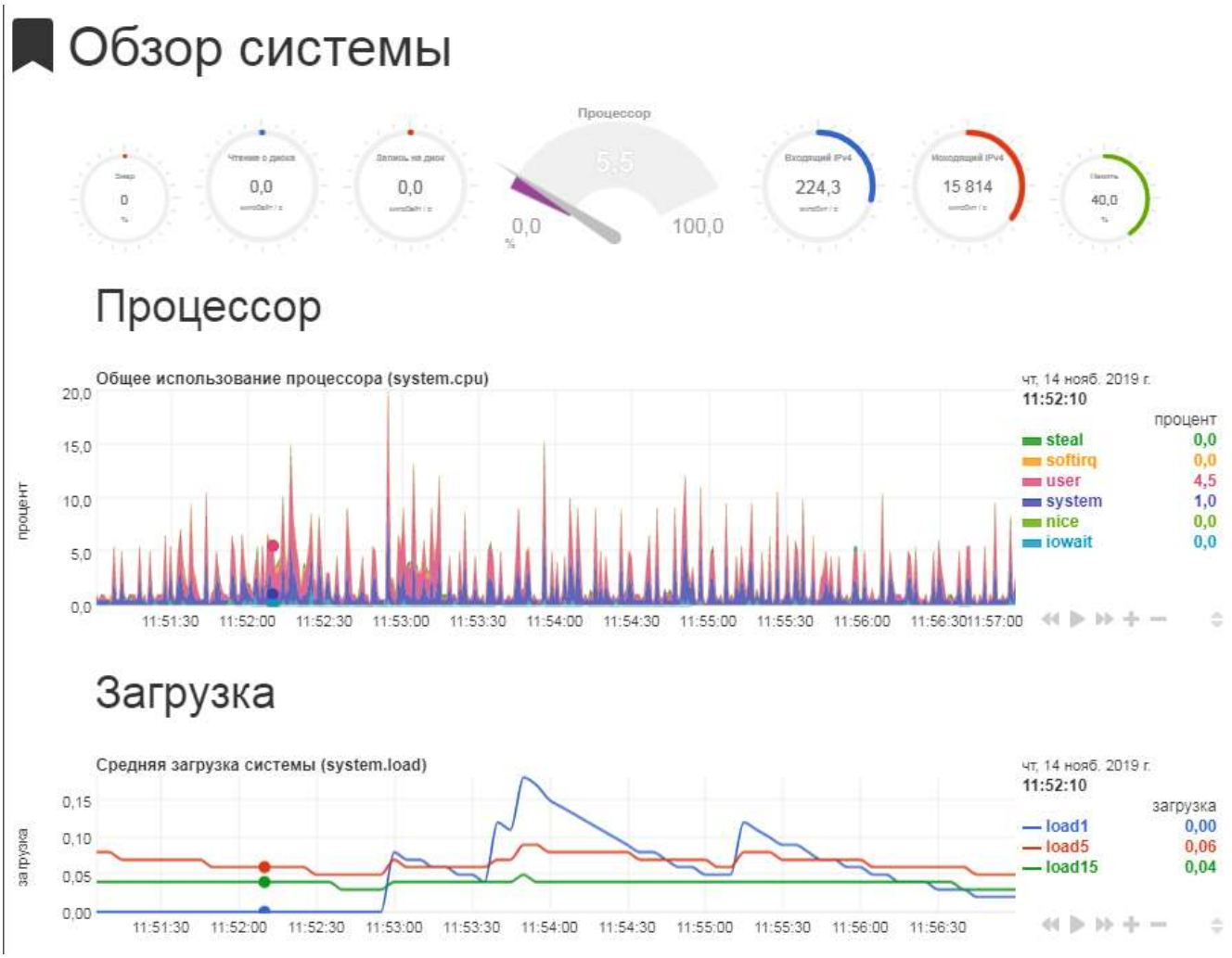


Рис. 49 – Интерактивная панель мониторинга

У каждого графика на интерактивной панели мониторинга есть легенда. Для просмотра можно выбрать одну из метрик или несколько – нажать клавишу «Ctrl» и указателем мыши выделить интересующие показатели.

Для масштабирования и изменения отображения информации на графике есть кнопки управления (рис. 50). Описание кнопок приведено в таблице 20.

Рис. 50 – Кнопки управления

Т а б л и ц а 20

Кнопки управления	Описание
	Переход по времени отображения информации показателя
	Возвращение на просмотр текущего состояния графика
	Увеличивают/уменьшают временные интервалы на графике
	Управляют размерами отображения графика по вертикали

Справа от информационной интерактивной панели мониторинга представлена панель элементов – список доступных для мониторинга показателей состояния системы, объединенных в группы (рис. 51):

- обзор системы;
- оперативная память;
- процессоры;
- диски;
- сеть IPv4;
- сеть IPv6;
- брандмауэр (netfilter);
- качество сервиса;
- сетевые интерфейсы;
- приложения;
- пользователи;
- мониторинг сетевых данных.

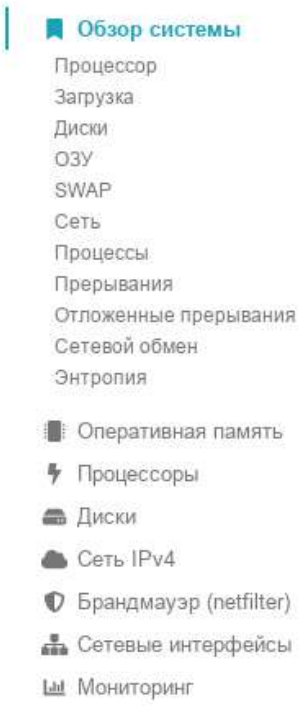


Рис. 51 – Список метрик

8.1. Дополнительные настройки

Справа вверху рабочей области (см. рис. 12) расположены дополнительные настройки (возможности) панели мониторинга (рис. 52):

-  «Настройки» – открытие диалогового окна с настройками работы и отображение панели мониторинга;
-  «Печать» – вывод данных панели мониторинга на печать.

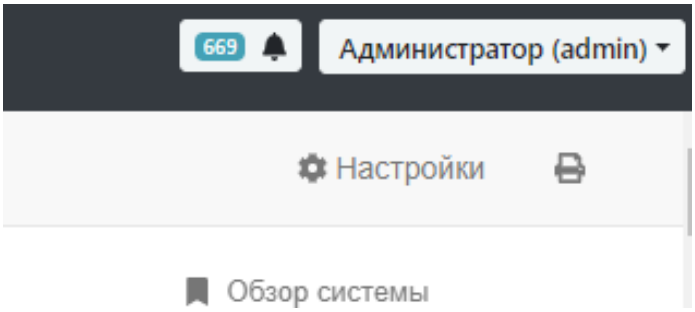


Рис. 52

8.1.1. Настройки отображения

Данные настройки определяют только локальное поведение и отображение панели мониторинга и сохраняются в локальном хранилище веб-браузера. Применение параметров происходит сразу после изменения.

В настройках интерактивной панели мониторинга расположено три вкладки с настройками (рис. 53):

- «Производительность» – представлены параметры, влияющие на скорость работы панели мониторинга;
- «Синхронизация» – представлены параметры, определяющие синхронное обновление графиков;
- «Отображение» – представлены параметры отображения интерактивной панели мониторинга.

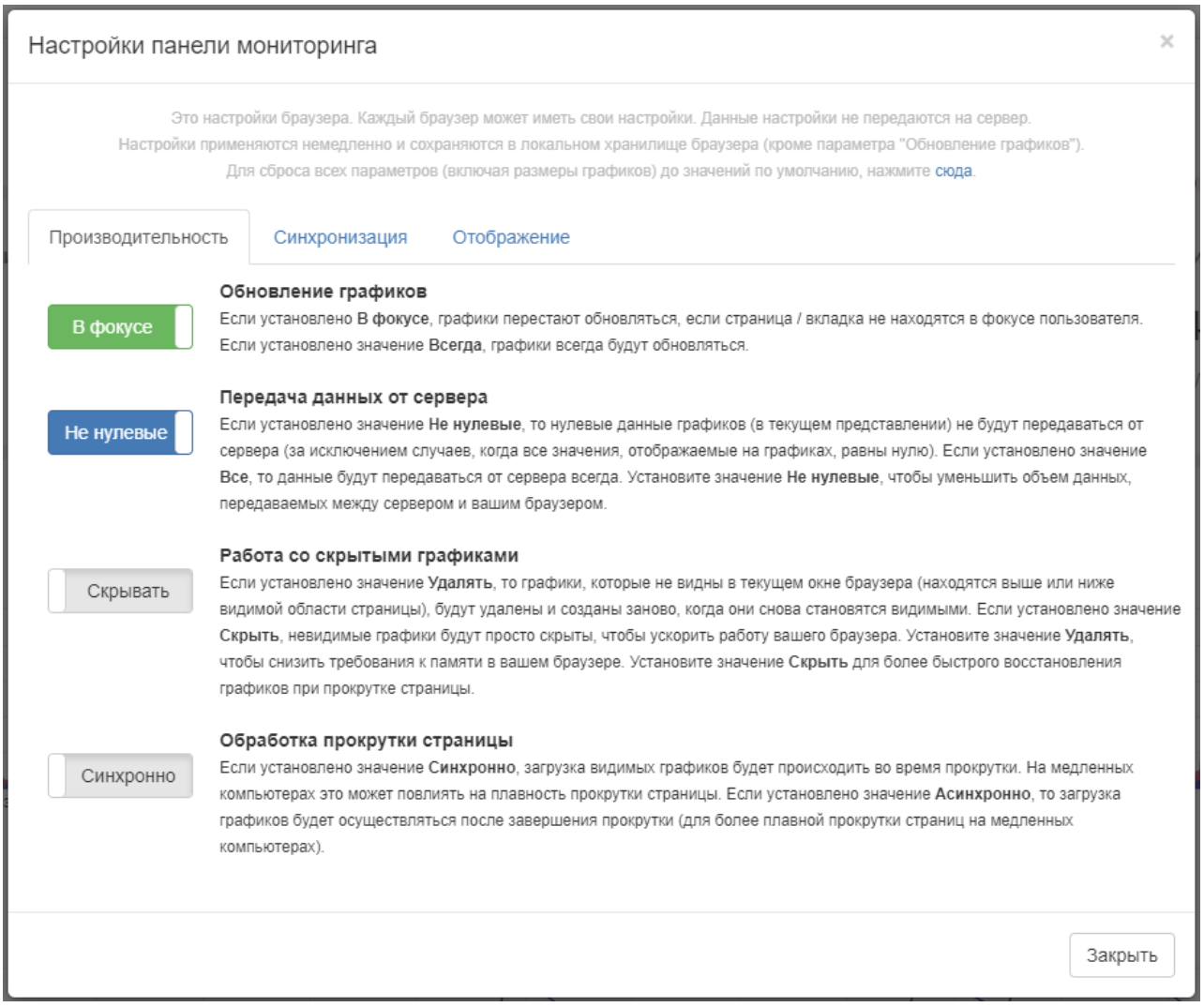


Рис. 53 – Параметры вкладки «Производительность»

8.1.1.1. Вкладка «Производительность»

Доступны следующие параметры вкладки «Производительность» (см. рис. 53):

- обновление графиков – может принимать значения «В фокусе» или «Всегда»;
- передача данных от сервера – может принимать значения «Не нулевые» или «Все»;
- работа со скрытыми графиками – может принимать значения «Скрывать» или «Удалять»;
- обработка прокрутки страницы – может принимать значения «Синхронно» или «Асинхронно».

8.1.1.2. Вкладка «Синхронизация»

Доступны следующие параметры вкладки «Синхронизация» (рис. 54):

- политика обновления графиков – может принимать значения «Параллельно» или «Последовательно»;
- повторная синхронизация при обновлении графиков – может принимать значения «Синхронизировать» или «Не синхронизировать»;
- синхронизация при наведении – может принимать значения «Синхронизировать» или «Не синхронизировать»;
- синхронизация панорамирования и масштабирования – может принимать значения «Синхронизировать» или «Не синхронизировать».

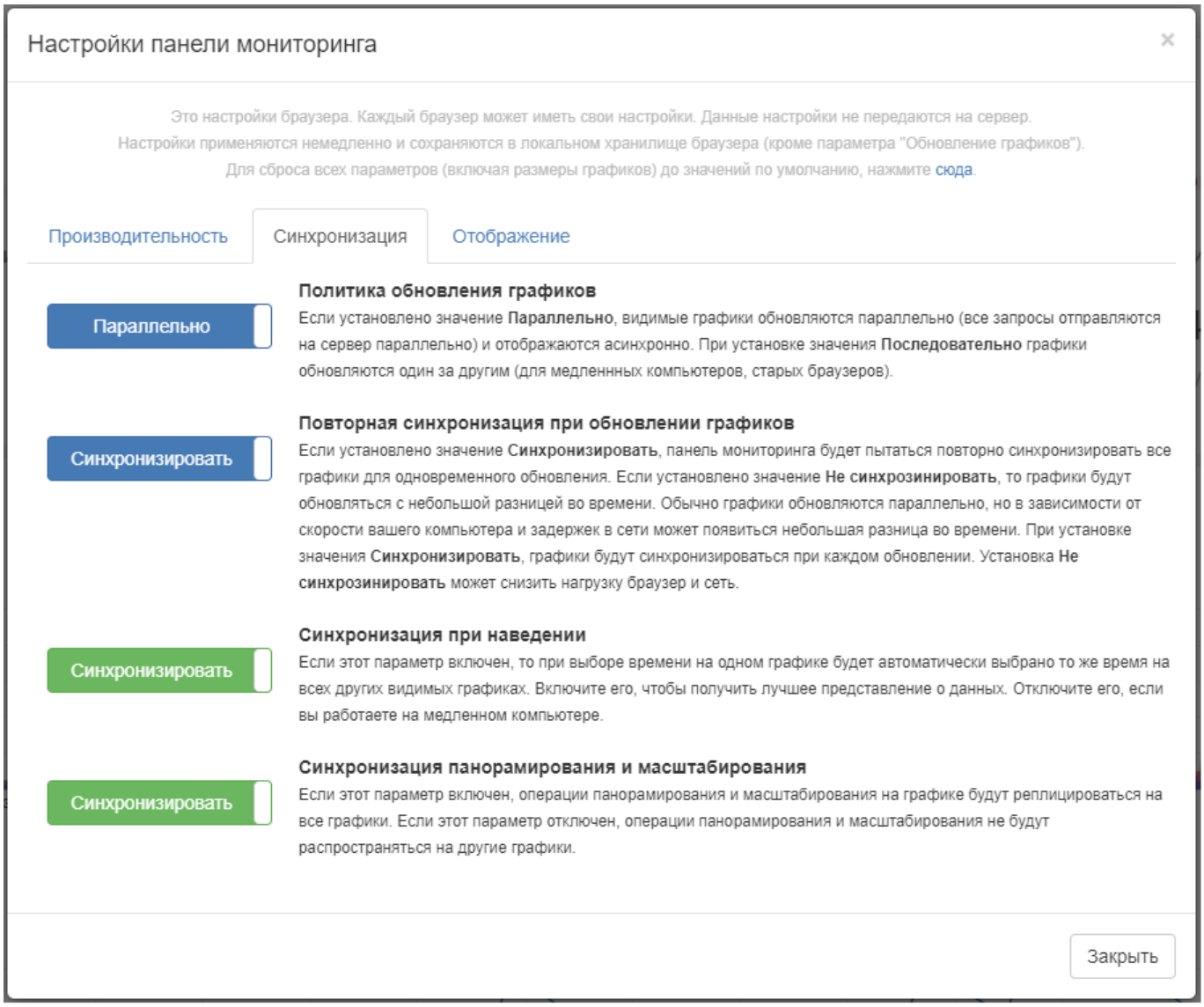


Рис. 54 – Параметры вкладки «Синхронизация»

8.1.1.3. Вкладка «Отображение»

Доступны следующие параметры вкладки «Отображение» (рис. 55):

- тема – может принимать значения «Светлая» или «Темная»;
- отображение подсказок – может принимать значения «Включено» или «Выключено»;
- загрузка данных при панорамировании и масштабировании – может принимать значения «Загружать» или «Не загружать»;
- сглаживание линий графиков – может принимать значения «Плавные» или «Прямые».

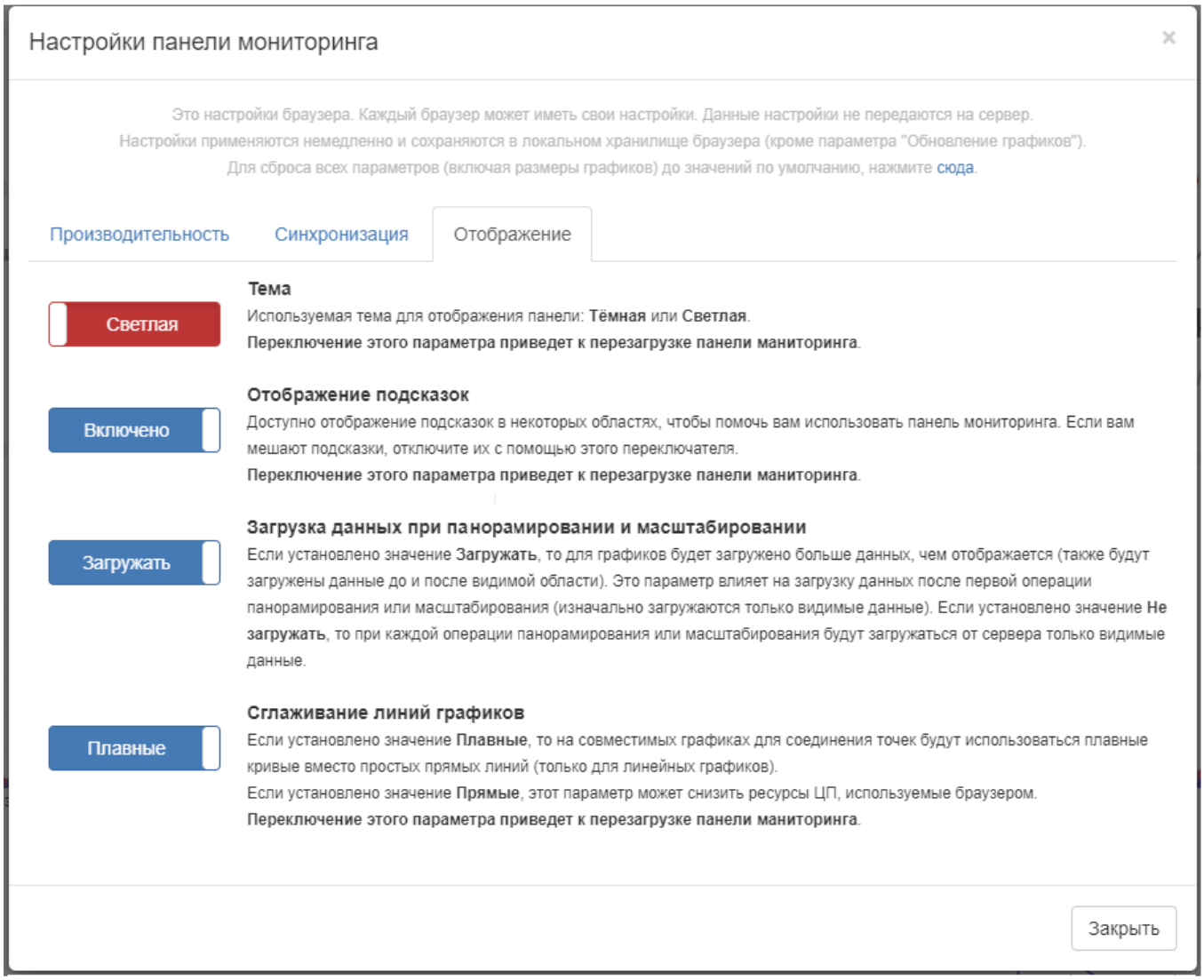


Рис. 55 – Параметры вкладки «Отображение»

8.1.2. Печать

При нажатии на кнопку «Печать» (см. рис. 52) появится всплывающее окно с подтверждением печати – нажать кнопку «Печать» (рис. 56).

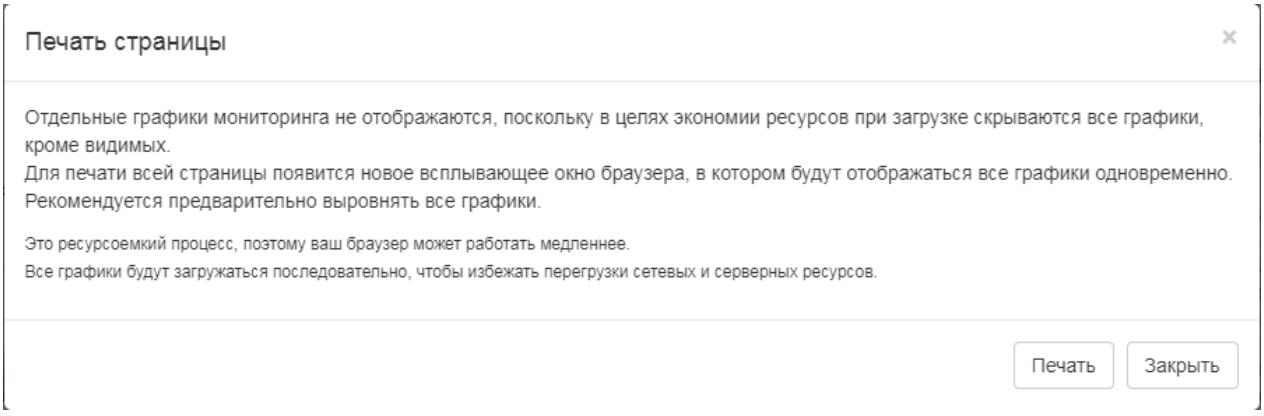


Рис. 56

Далее будет произведена подготовка данных графиков к печати (рис. 57).

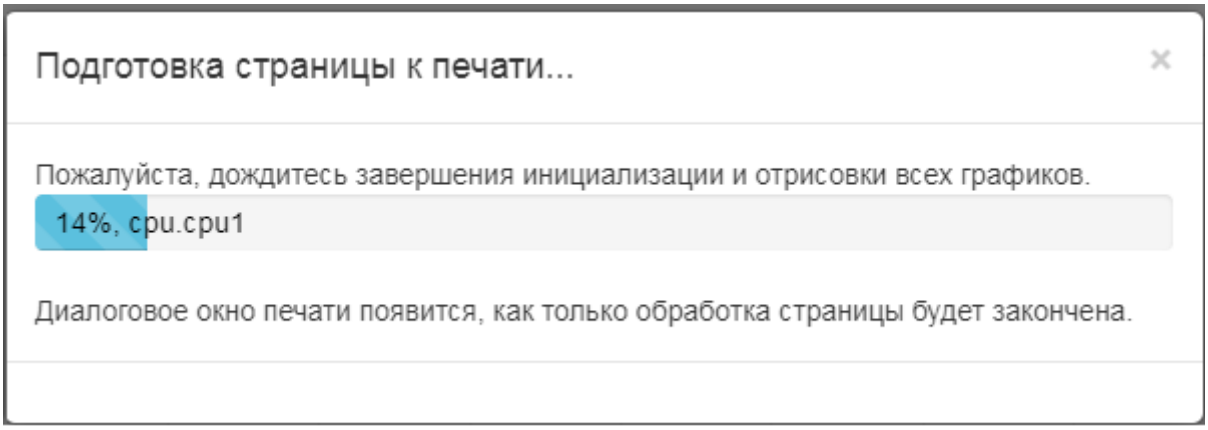


Рис. 57 – Подготовка к печати

После завершения процесса подготовки страницы откроется окно или страница с предварительным просмотром данных в веб-браузере (рис. 58). Установите настройки и для отправки задания на принтер нажмите кнопку «Печать».

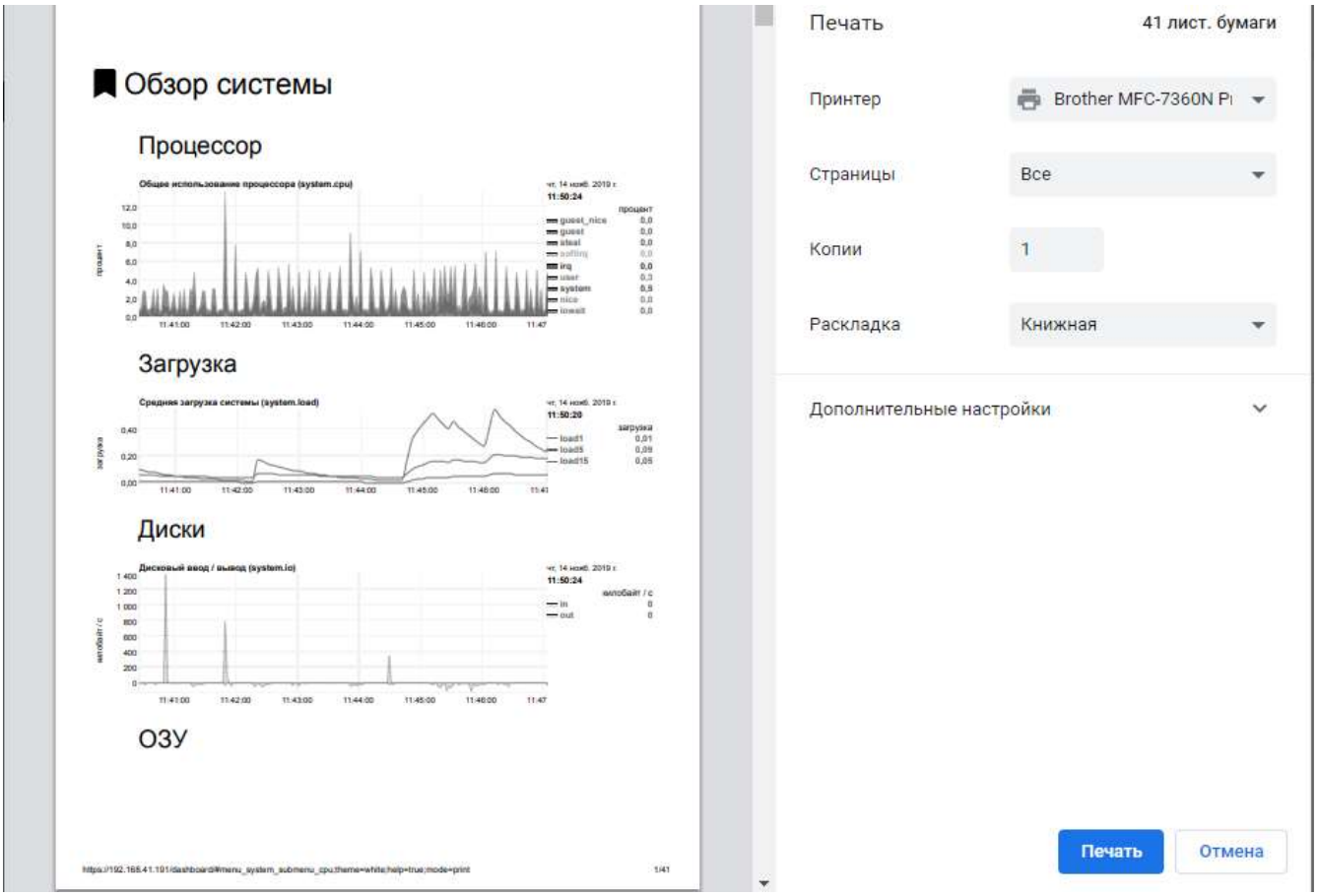


Рис. 58 – Предварительный просмотр

8.2. Обзор системы

Мониторинг по основным характеристикам системы выполняется путем анализа панели индикаторов в определенный момент времени (рис. 59).

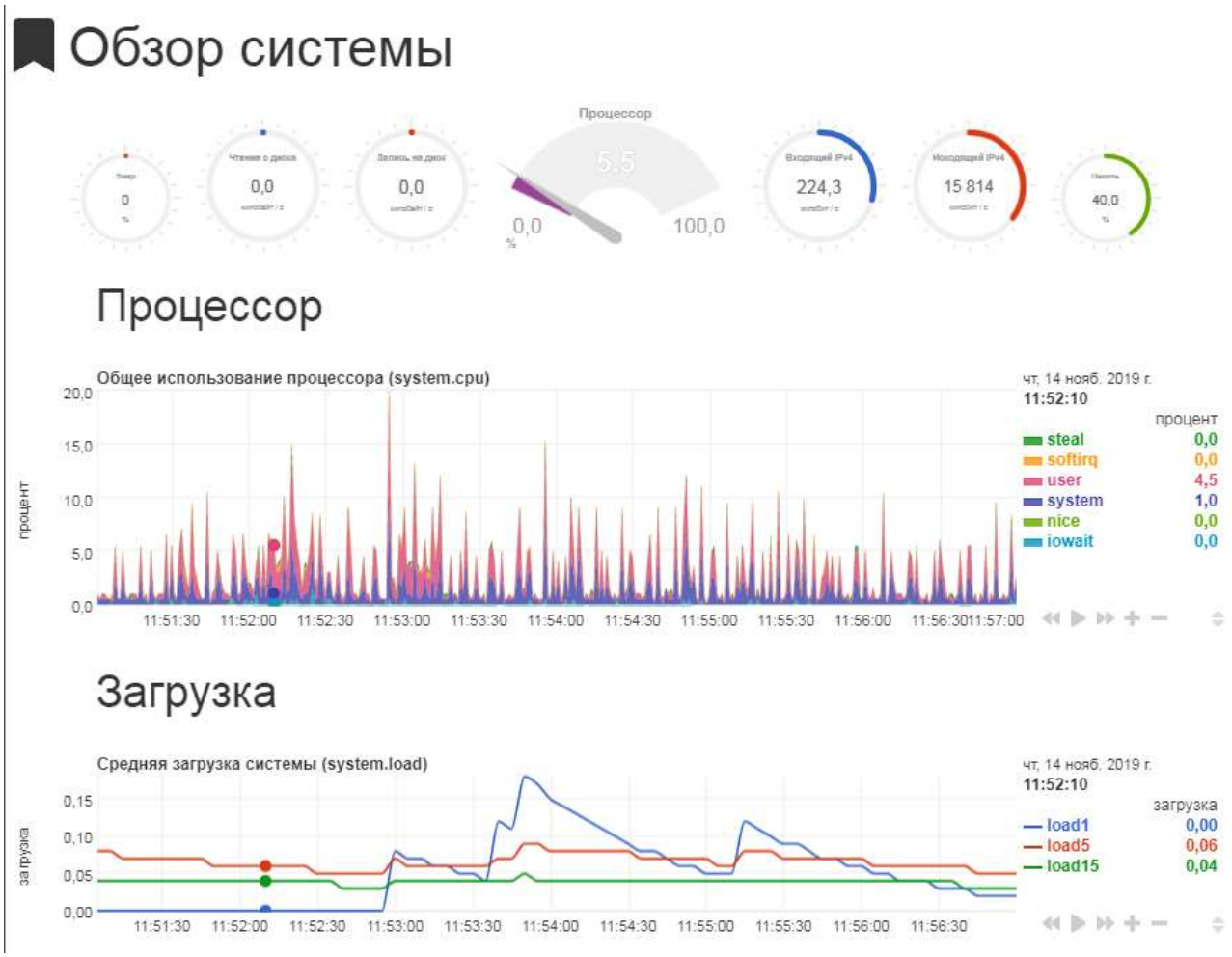


Рис. 59 – Панель индикаторов

Значения метрик панели индикаторов изменяются каждый раз, при наведении указателя мыши на график, отображая показатели системы на выбранный момент времени (рис. 60).

Рис. 60 – Индикаторы состояния системы

Состав учитываемых характеристик системы:

- процессор (п. 8.2.1);
- загрузка (п. 8.2.2);
- диски (п. 8.2.3);
- оперативное запоминающие устройство (п. 8.2.4);
- swap (п. 8.2.5);
- сеть (п. 8.2.6);
- процессы (п. 8.2.7);
- прерывания (п. 8.2.8);
- отложенные прерывания (п. 8.2.9);
- сетевой обмен (softnet) (п. 8.2.10);
- энтропия (п. 8.2.11).

Далее располагаются примеры графиков состояний каждой из основных характеристик состояния системы. Набор графиков и отображаемых характеристик, может меняться в зависимости от текущего состояния системы.

8.2.1. Процессор

Центральный процессор (CPU, от англ. Central Processing Unit) – это основной рабочий компонент компьютера, который выполняет арифметические и логические операции, заданные программой, управляет вычислительным процессом и координирует работу всех устройств компьютера.

На графике состояния общего использования процессора отображаются текущие параметры (таблица 21) в процентах.

Т а б л и ц а 21

Метрика	Описание метрики
steal time	процессорное время, в течение которого, виртуальная машина не имела доступа к ресурсам центрального процессора
softirq	процессорное время, занятое в работе механизма отложенных прерываний
user	процессорное время, занятое в процессах пользовательского пространства
system	процессорное время, занятое в процессах пространства ядра
nice	время, затраченное процессором на процессы с низким приоритетом
iowait	время процессора в режиме ожидания ввода-вывода

8.2.2. Загрузка

Средняя загрузка (англ. load average) – среднее значение загрузки системы за период времени; как правило, отображается в виде трех значений, которые представляют собой усредненные величины за последние 1, 5 и 15 минут (рис. 61).

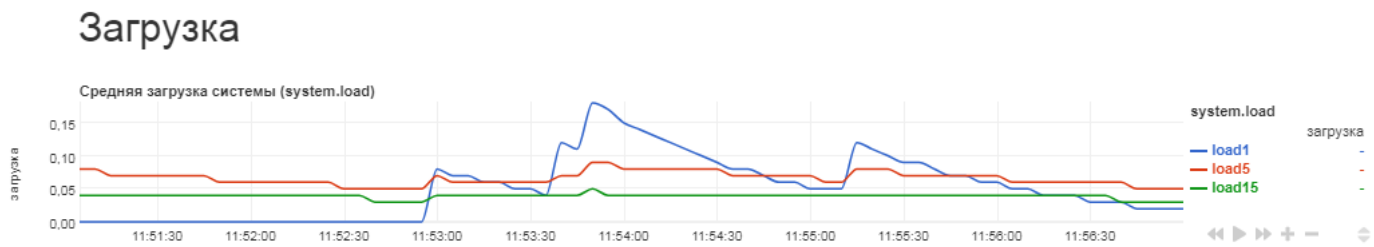


Рис. 61 – Состояние средней загрузки системы

8.2.3. Диски

Дисковый ввод/вывод – количество операций ввода/вывода в секунду на диск.

in – характеристика рабочей нагрузки, измеряется в количеством байт, которые требует приложение в секунду.

out – характеристика рабочей нагрузки, измеряется в количеством байт, которые вырабатывает приложение в секунду (рис. 62).

Рис. 62

8.2.4. Оперативное запоминающее устройство (ОЗУ)

ОЗУ – энергозависимая часть системы компьютерной памяти, в которой во время работы компьютера хранится выполняемый машинный код (программы), а также входные, выходные и промежуточные данные, обрабатываемые процессором.

На графике (рис. 63) отображается загрузка ОЗУ отображает размер памяти в зависимости от типа:

- free – свободно;
- used – используется;
- cached – содержится в кэше;
- buffers – хранится в буфере.

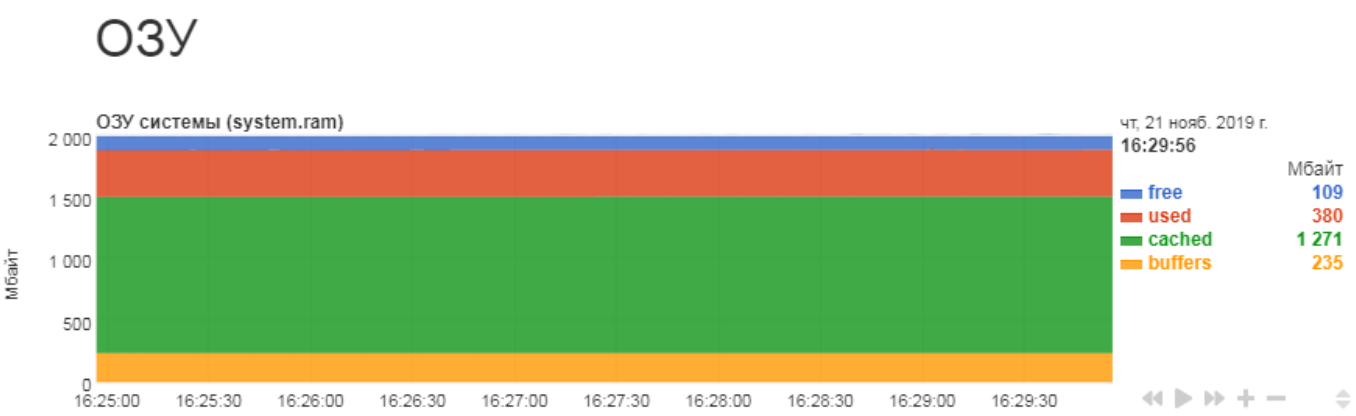


Рис. 63 – График состояния системы ram

8.2.5. Swap

Swap – подкачка страниц – механизм выделения виртуальной памяти, при котором часть данных из оперативной памяти (обычно не активной) перемещается на жесткий диск или иное вторичное хранилище в случае переполнения и исчерпания гарантированного объема оперативной памяти для освобождения фрагментов памяти для других активных загрузок.

На графиках характеристик Swap отображаются метрики, приведенные в таблице 22.

Таблица 22

График	Описание	Метрика	Единица
Системный swap	подкачка страниц системы	free – свободная память used – используется при подкачке страниц	количество байт
Swap I/O	отображает изменения дискового ввода/вывода (in/out) при использовании механизма подкачки страниц	in – ввод out – вывод	количество байт в секунду

8.2.6. Сеть

На графиках (рис. 64) отображается состояние полосы пропускания для IPv4, показано какой объем данных получен (reseived) и какой отправлен (sent) в единицу времени. Измеряется количеством передаваемых данных в секунду.

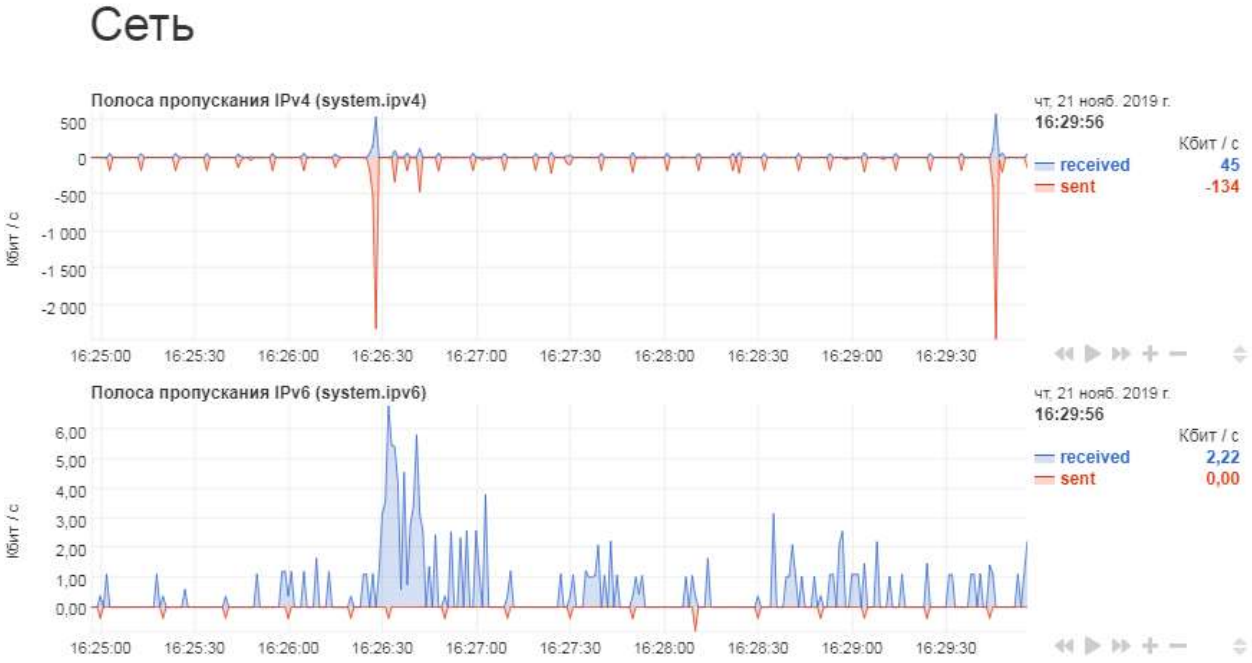


Рис. 64 – Полоса пропускания

8.2.7. Процессы

Метрики процессов, выполняемых процессором в пространстве ядра операционной системы, зависят от состояния других выполняемых процессов и количества процессоров или ядер, имеющихся в компьютере.

На графиках характеристик процессора могут отображаться метрики, приведенные в таблице 23.

Т а б л и ц а 23

График	Метрики	Единица
Процессы	running – запущенные	количество процессов
	blocked – заблокированные	
Запущенные процессы	–	количество процессов в секунду
Активные процессы	–	количество процессов
Переключение контекста процессора	–	количество переключений в секунду
Время простоя процессора	–	количество потерянных микросекунд в секунду

8.2.8. Прерывания

На графике прерывания процессора (рис. 65) отображается работа механизма прерываний, измеряется в количестве прерываний в секунду.

В момент времени t_1 процессор получает сигнал прерывания. Он приостанавливает выполнение одной Программы 1, запоминает адрес ее последней команды и активизирует находящуюся в памяти программу обработки прерываний. После того как эта программа выполнит свою задачу, в момент времени t_2 управление возвращается Программе 1, и ее выполнение продолжается с того адреса, на котором выполнение было прервано в момент времени t_1.

Рис. 65 – Прерывания процессора

8.2.9. Отложенные прерывания

Отложенные прерывания (softirqs) – это механизм исполнения кода вне контекста обработчика прерываний, реализованного драйвером.

На графике (рис. 66) отображается процесс отложенных прерываний в системе с учетом метрик, описанных в таблице 24. Измеряется количеством отложенных прерываний в секунду.

Таблица 24

Метрика	Описание
TIMER	таймер
NET_RX	сетевые пакеты
BLOCK	блокирование прерываний
TASKLET	тасклет
SCHED	плановое прерывание
RCU	Read-Copy-Update (чтение – копирование – обновление) – механизм синхронизации

Рис. 66 – Отложенные прерывания

8.2.10. Сетевой обмен (softnet)

На графике (рис. 67) отображена статистика работы обмена между устройствами сети, в качестве метрики использует количество обработанных событий процессором в секунду.

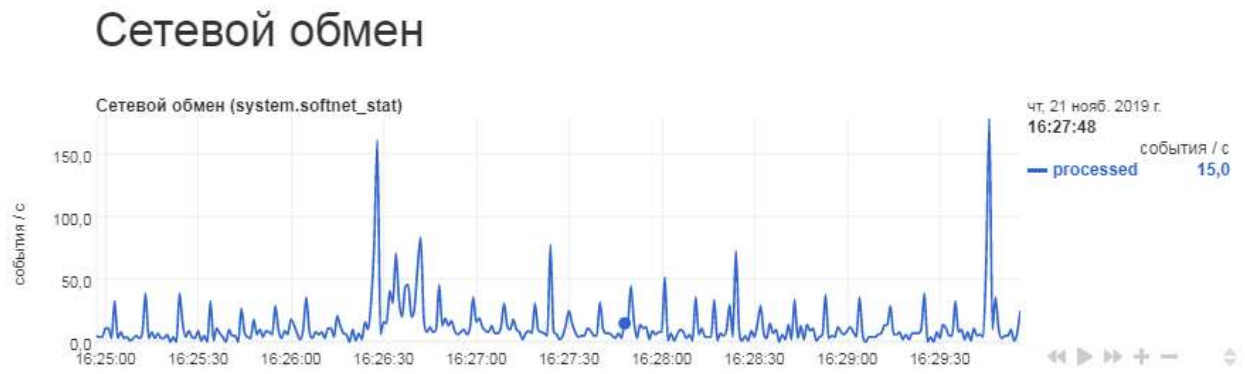


Рис. 67 – Сетевой обмен

8.2.11. Энтропия

Энтропия (entropy) – это мера беспорядочности, собираемая операционной системой или приложением для использования в целях, требующих случайных данных (рис 68).

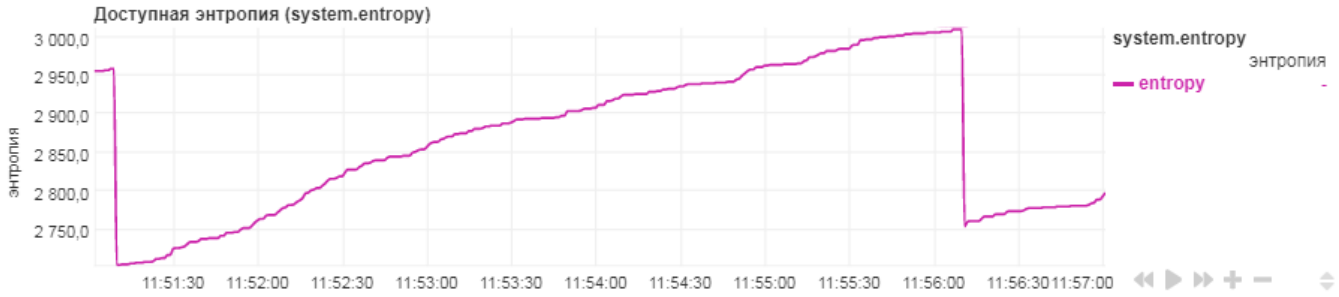


Рис. 68 – График энтропии

8.3. Оперативная память

В данном подразделе рассматриваются метрики потребления оперативной памяти, разбитые по группам:

- система;
- ядро;
- использование памяти ядра.

На графиках характеристик потребления оперативной памяти отображаются метрики, приведенные в таблице 25.

Таблица 25

График	Метрика	Доп. метрики	Единицы
Система	Ошибки страниц памяти	minor – незначительные ошибки	Количество ошибок страниц в секунду
		major – значительные ошибки	
	Выделенная память	Committed_AS – выделенная память	Мбайт
Ядро	Кэш-память	Dirty – измененные страницы в памяти, еще не перемещенные на диск	Мбайт
		Writeback – страницы памяти, которые в настоящий момент сбрасываются на диск	
	Память ядра	Slab – распределение памяти	Мбайт
		KernelStack – стек ядра	
		PageTables – память под таблицу страниц	
Использование памяти ядра	Восстанавливаемая память	reclaimable – распределенная память unreclaimable – неиспользованная память	Мбайт

8.4. Процессоры

В данном разделе рассматриваются следующие графики метрик процессоров:

- использование;
- отложенные прерывания;
- сетевой обмен.

Для каждого из процессоров системы выводятся свои графики по метрикам, описанным в таблице 26.

Таблица 26

График	Метрика	Описание	Единица
Использование	steal	время, потраченное на использование виртуализации	%
	softirq	время, потраченное на механизм отложенных прерываний	
	user	время на обычные процессы пользователя, приоритет которых не менялся	
	system	использование системными процессами	
	iowait	время ожидания завершения ввода-вывода	
Отложенные прерывания	TIMER	таймер	количество отложенных прерываний в секунду
	NET_RX	сетевые пакеты	
	BLOCK	блокирование прерываний	
	TASKLET	тасклет	
	SCHED	плановое прерывание	
Сетевой обмен	для каждого из процессоров	processed – обрабатываемые	количество событий в секунду

8.5. Диски

В Linux все отображается в файловом виде, в том числе и устройства.

В данном разделе отображены графики состояний дисков, разделенные по группам имеющихся внешних и внутренних устройств, корневого каталога `/` и каталога `/var` – каталога в дисковом пространстве, предназначенного для хранений данных большого и часто меняющегося размера.

Все подключенные к операционной системе Linux устройства размещаются в каталоге `/dev/` (микрофоны, камеры, жесткие диски, USB-флеш, то есть все внешние и внутренние устройства).

Жесткие диски имеют особенные названия. В зависимости от интерфейса, через который подключен жесткий диск, название может начинаться на:

- sr0 – это первое устройство, подключенное по SCSI, CD-ROM в системе;
- sd – устройство, подключенное по SCSI;
- hd – устройство ATA;
- vd – виртуальное устройство;
- mmcblk – обозначаются флешки, подключенные через картридер.

Третья буква в имени диска означает его порядковый номер в системе. Используется алфавитная система. Например, sda – первый диск, sdb – второй диск, sdc – третий и так далее. Самый простой способ увидеть все подключенные диски – это посмотреть содержимое каталога `/dev/` и отфильтровать устройства sd.

На графиках отображается текущее состояние дисков и каталогов с учетом метрик, приведенных в таблицах 27 и 28.

Таблица 27

Диски	Графики метрик	Единица
sda, sr0 – внешние и внутренние устройства Примечание. Наименования зависят от количества и состава устройств системы.	Пропускная способность дискового ввода/вывода	количество записанных байт в секунду
	Завершенные операции ввода/вывода	количество операций записи в секунду
	Очередь операций ввода/вывода	количество операций
	Ожидаемая продолжительность операций ввода/вывода в очереди	отставание (мс)
	Использование диска	% рабочего времени
	Среднее время выполнения операций ввода/вывода	мс/на операцию записи
	Средняя пропускная способность операции ввода/вывода	количество записанных байт за операцию
	Среднее время обслуживания операции ввода/вывода	мс/ на операцию
	Объединенные операции	операций объединения в секунду
	Общее время ввода/вывода	мс/с

Таблица 28

Каталоги	Графики метрик	Дополнительные метрики	Единица
`/` – корневой каталог. `/var`.	Использование дискового пространства	avail – доступное	Гбайт
		used – используемое	
		reserved for root – зарезервированное за суперпользователем	
	Использование файлов	avail – доступное	количество файлов
		used – используемое	

Для внутренних жестких дисков отображается также панель общих индикаторов состояния: чтение (Кбайт/с), запись (Кбайт/с), использование (% от рабочего времени).

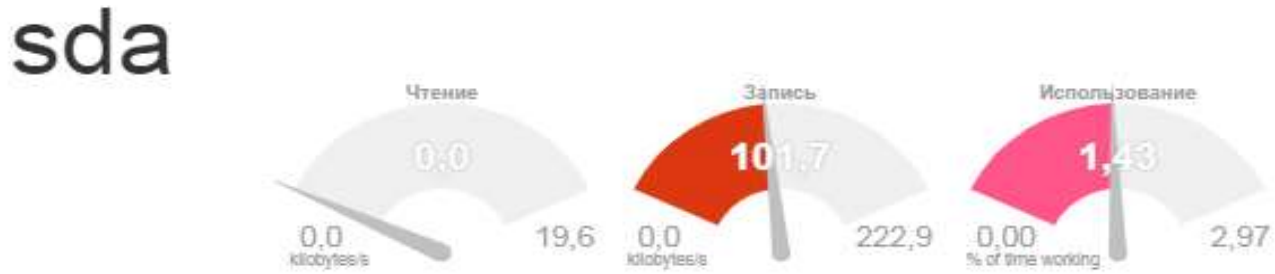


Рис. 69

8.6. Сеть IPv4

IPv4 – четвертая версия интернет протокола, кратко обозначается как IPv4, спецификация RFC 791.

Традиционная форма записи адреса – четыре десятичных числа, разделенные точками. Каждая число может иметь значение от 0 до 255. Например, 192.0.2.235.

8.6.1. Протокол TCP

TCP (Transmission Control Protocol) – протокол управления передачей данных.

На графиках характеристик TCP IPv4 (рис. 70) отображаются метрики, приведенные в таблице 29.

Таблица 29

График	Метрика
TCP-соединения IPv4	connection – количество активных соединений
TCP-пакеты IPv4	Пакетов в секунду: recevied – полученных
	sent – отправленных
Ошибки TCP IPv4	RetransS – ретранслятор домена, измеряется количеством пакетов в секунду
Проблемы установки соединения TCP IPv4	Количество событий в секунду: EstabResets – сброс сокета
	OutRsts – сброшенные отправленные пакеты
	ActiveOpen – активные запросы на открытие
	PassiveOpen – пассивные запросы на открытие
Прерывания TCP-соединений	Прерывания, плохие данные (baddata) – количество соединений в секунду
Переупорядоченные TCP-пакеты	Количество пакетов в секунду: TCP timestamp – с возвратом метки времени отправленного пакета
	sack – с опцией выборочного подтверждения
	fack – с прямым подтверждением
	reno – с повторной пересылкой и восстановлением
TCP Out-Of-Order очередь	Количество пакетов в секунду: inqueue – с переполнением очереди запросов
	dropped – с отбрасыванием конца очереди
	merged – с объединением
	pruned – с удалением запросов

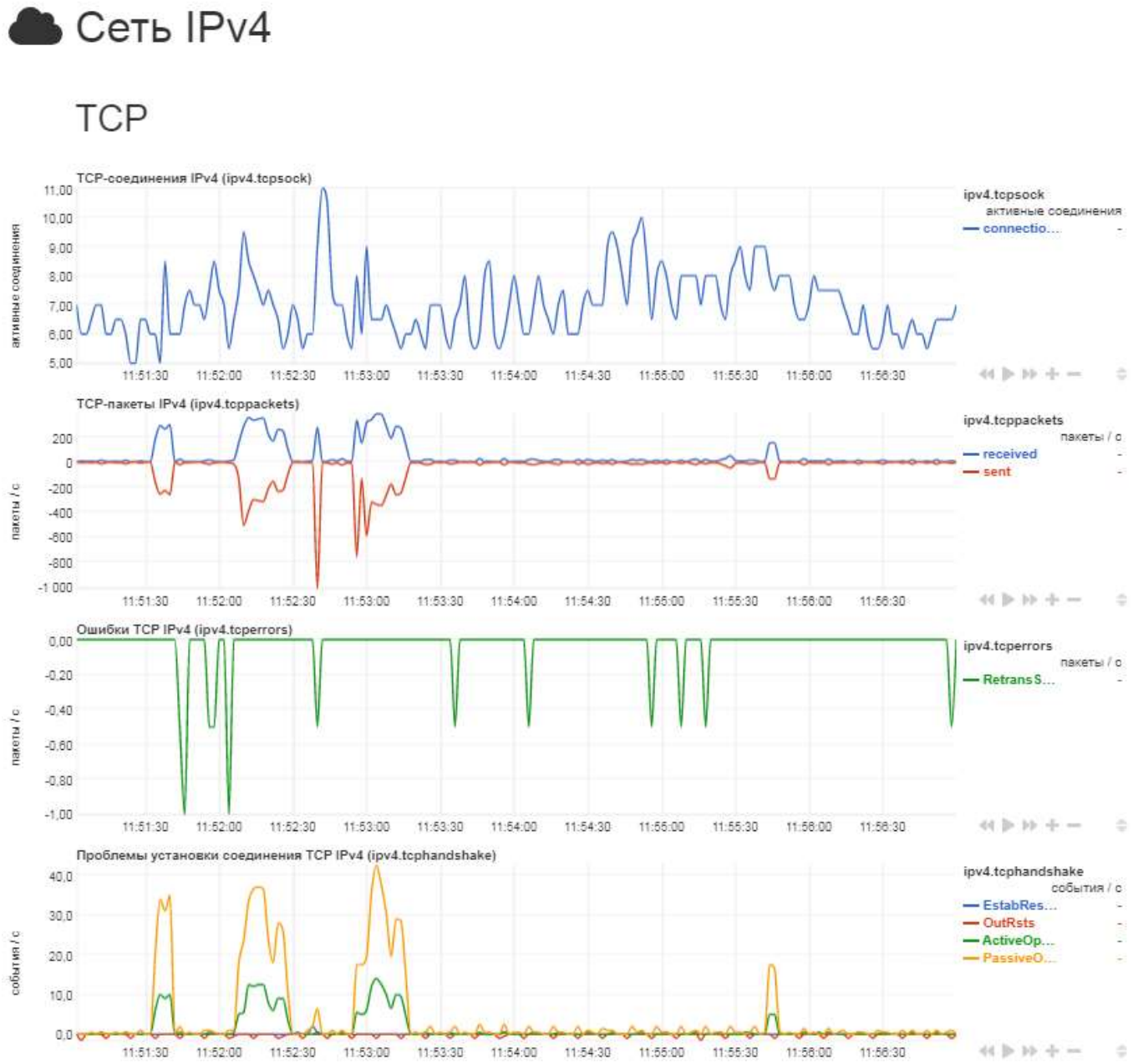


Рис. 70 – TCP IPv4

8.6.2. UDP

UDP (User Datagram Protocol) – протокол пользовательских датаграмм.

С UDP приложения могут посылать сообщения (датаграммы) другим хостам по IP-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных.

На графиках характеристик IPv4 UDP отображаются метрики, приведенные в таблице 30.

Т а б л и ц а 30

График	Метрика
IPv4 UDP пакеты	Количество пакетов в секунду: - recevied – полученных; - sent – отправленных.
IPv4 UDP ошибки	Количество событий в секунду: IgnoredMulti – игнорирование множественных повторов.

8.6.3. ICMP

ICMP (Internet Control Message Protocol) – протокол межсетевых управляющих сообщений. ICMP используется для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных.

На графиках характеристик ICMP IPv4 приведены метрики, указанные в таблице 31, измеряются в количестве пакетов в секунду.

Т а б л и ц а 31

График	Метрики
IPv4 ICMP пакеты	recevied – полученные пакеты
	sent – отправленные пакеты
IPv4 ICMP ошибки	InErrors – ошибки на входе
	OutErrors – ошибки на выходе
	InCsumErrors – ошибки инкрементального контрольного суммирования
IPv4 ICMP сообщения	InEchoReps – входящий ответный пакет
	OutEchoReps – исходящий ответный пакет
	InDestUnrechable – с недоступным направлением для входящего
	OutDestUnrechable – с недоступным направлением для исходящего
	InRedirects – с входящей информацией о маршрутизации
	OutRedirects – с выходящей информацией о маршрутизации
	InEchos – входящий эхо-запрос
	OutEchos – исходящий эхо-ответ

8.6.4. UDPLite

UDPLite – облегченный протокол пользовательских дейтаграмм, является протоколом без установления соединения, что позволяет потенциально поврежденные данные полезной нагрузки доставлять приложению, а не отбрасывать принимающей станцией.

На графиках характеристик UDP-Lite отображаются метрики, приведенные в таблице 32, измеряются в количестве пакетов в секунду.

Т а б л и ц а 32

График	Метрики
IPv4 UDPLite пакеты	recevied – полученные пакеты
	sent – отправленные пакеты
IPv4 UDPLite ошибки	RcvbufErrors – ошибки, связанные с размером буфера, предназначенного для получения UDP-пакета
	SndbufErrors – ошибки, связанные с размером буфера, предназначенного для отправки UDP-пакета
	NoPorts – нет портов источника или назначения
	IgnoredMulti – игнорирование множественных повторов
	InErrors – ошибки на выходе
	InCsumErrors – ошибки инкрементального контрольного суммирования

8.6.5. Пакеты

На графике (рис. 71) приведены характеристики передачи пакетов IPv4, измеряются количеством пакетов в секунду:

- received – полученные;
- sent – отправленные;
- forwarded – перенаправленные;
- delivered – доставленные.

Рис. 71 – IPv4 пакеты

8.6.6. Ошибки

На графике IPv4 ошибки – отображаются ошибки, возникшие при передаче данных по протоколу IPv4, измеряются количеством пакетов в секунду:

- InDiscards – отклоненные на входе в результате сбоя выделения памяти, или в результате сбоя контрольной суммы при обрезании пакета;
- OutDiscards – отклоненные на выходе в результате сбоя выделения памяти, или в результате сбоя контрольной суммы при обрезании пакета;
- InHdrErrors – с поврежденными заголовками;
- InAddrErrors – с проблемами доступа адреса хоста;
- InUnknow – неизвестные.

На графике IPv4 входящие ошибки – отображаются ошибки в пакетах, измеряемые количеством пакетов в секунду:

- norouters – нет маршрута;
- truncated – отбрасывания;
- checksum – ошибки контрольных сумм.

8.6.7. Фрагменты (fragments)

Фрагментация IP – это процесс Интернет-протокола (IP), который разбивает пакеты на более мелкие фрагменты, так что полученные фрагменты могут проходить через канал с меньшим максимальным блоком передачи, чем исходный размер пакета.

На графике Отправленные фрагменты IPv4 представлены данные по отправленным фрагментам в количестве пакетов в секунду:

- ok – прошедшие;
- falled – не прошедшие;
- created – созданные пакеты.

На графике Повторная сборка фрагментов IPv4 представлены данные по повторно собранным фрагментам в количестве пакетов в секунду:

- ok – прошедшие;
- falled – не прошедшие;
- all – все пакеты.

8.6.8. Broadcast

На графике (рис. 72) отображается состояние пропускной способности IPv4 Broadcast – получения данных (received) в количестве данных в секунду.

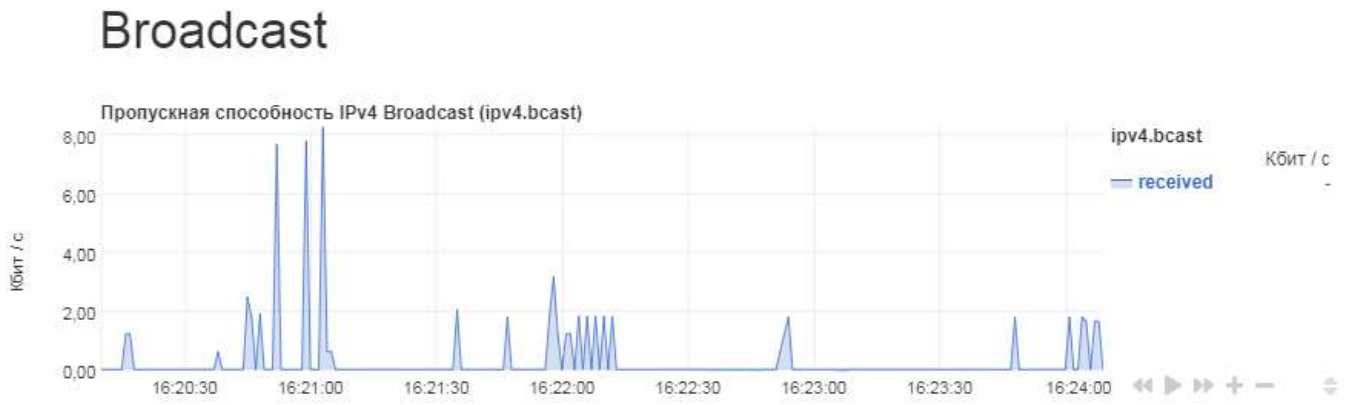


Рис. 72 – Пропускная способность IPv4

На графике (рис. 73) отображается состояние IPv4 Broadcast пакетов – received в количестве полученных пакетов в секунду.

Рис. 73 – IPv4 Broadcast пакеты

8.6.9. Multicast IPv4

Состояние IPv4 Multicast пакеты измеряется количеством пакетов в секунду и разделяется на:

- received – полученные;

- sent – отправленные.

Пропускная способность IPv4 Multicast измеряется количеством данных в секунду и разделяется на:

- received – получение;

- sent – отправление.

8.6.10. ECN

Explicit Congestion Notification (ECN) с англ. – «явное уведомление о перегруженности» – расширение протокола IP, описанное в RFC 3168. Позволяет обеим сторонам в сети узнавать о возникновении затора на маршруте к заданному хосту или сети без отбрасывания пакетов. Это дополнительная функция, которая используется только в том случае, когда обе конечные точки обмена информацией сообщают, что они хотят ее использовать.

На графике (рис. 74) представлено состояние показателя NotECTP – (Not-ECN-Capable Transport) – поток, не поддерживающий ECN, измеряется количеством пакетов в секунду.

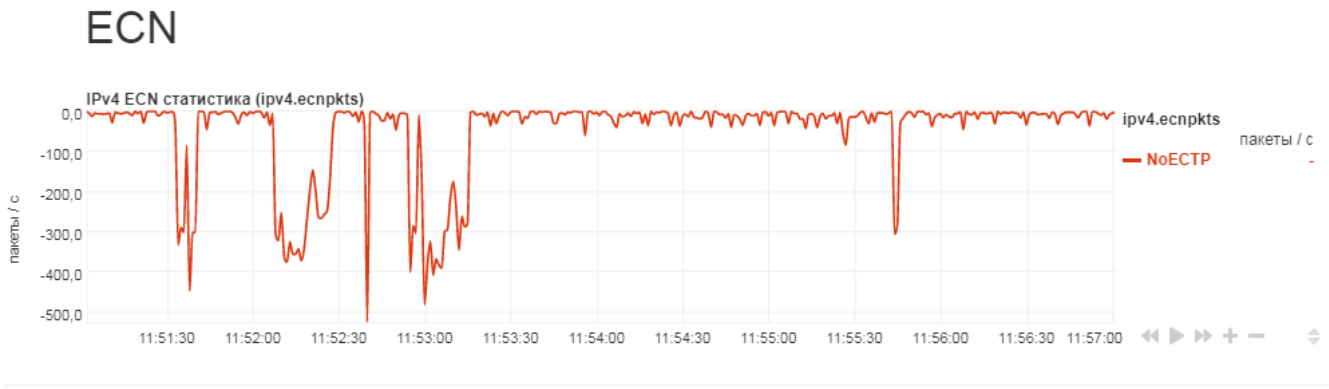


Рис. 74 – ECN

8.7. Брандмауэр (netfilter)

МЭ ИВК КОЛЬЧУГА-К осуществляет контроль и фильтрацию проходящего через него сетевого трафика в соответствии с заданными правилами пакетного фильтра.

8.7.1. Отслеживание соединений

На рис. 75 приведены примеры графиков характеристик отслеживания соединений (conntrack) для метрик, приведенных в таблице 33.

Таблица 33

График	Метрика
Соединения	connection – отображает количество активных соединений
Новые соединения	ignore – проигнорированные (количество соединений/сек)
	invalid – поврежденные (количество соединений/сек)
Изменения	Количество повторений в секунду: inserted – вложенные
	deleted – удаленные
	delete_list – удаление списка
Ожидания	Количество ожиданий в секунду: created – созданные
	deleted – удаленные
	new – новые
Ошибки	Количество ошибок в секунду: icmp_error – ошибки межсетевого протокола управляющих сообщений
	insert_falled – ошибки добавления
	drop – отброшенные
	early_drop – рано отброшенные
Поиски	Количество поисковых запросов/сек: searched – поиск
	restarted – перезапущенные
	found – найденные

Рис. 75 – Отслеживание соединений

8.8. Качество сервиса

Отображение качества сервиса включается при использовании функций ограничения трафика.

В разделе «Качество сервиса» на графиках отображаются характеристики состояния трафика, проходящего через физический Ethernet-интерфейс, например, ens18 (рис. 77), и состояние трафика, перенаправляемое на ens18-ifb (рис. 76) интерфейс, исходящий от сервера, входящий для клиентов:

- «Использование классов», измеряется в количестве данных в секунду;
- «Пакеты классов», измеряется количеством пакетов в секунду.

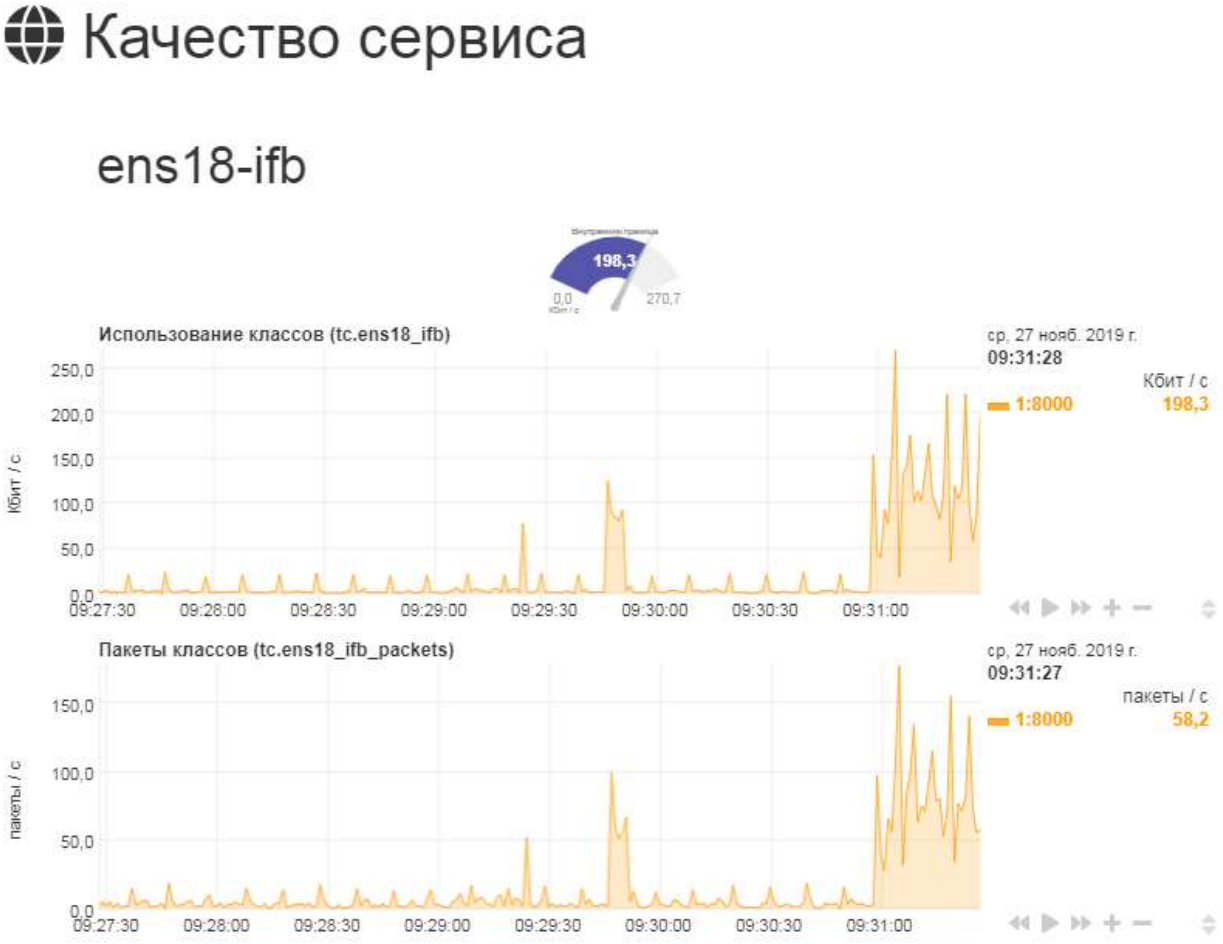


Рис. 76

Рис. 77

Скорость на внутренней и внешней границе сети в текущий момент времени также отображается на панелях индикаторов состояния в количестве данных в секунду (рис. 78).

Рис. 78

8.9. Сетевые интерфейсы

В данном разделе отображается состояние характеристик имеющихся сетевых интерфейсов (рис. 79, на примере интерфейса ens18):

1) пропускная способность, измеряется количеством данных в секунду и разделяется на:
   - received – получение;
   - sent – отправление;
2) сетевые пакеты, измеряются количеством пакетов в секунду:
   - received – полученных;
   - sent – отправленных;
3) отброшенные пакеты, измеряются количеством отброшенных пакетов в секунду.

На общих индикаторах отображается состояние принятых и отправленных данных в текущий момент времени. Измеряются количеством данных в секунду.

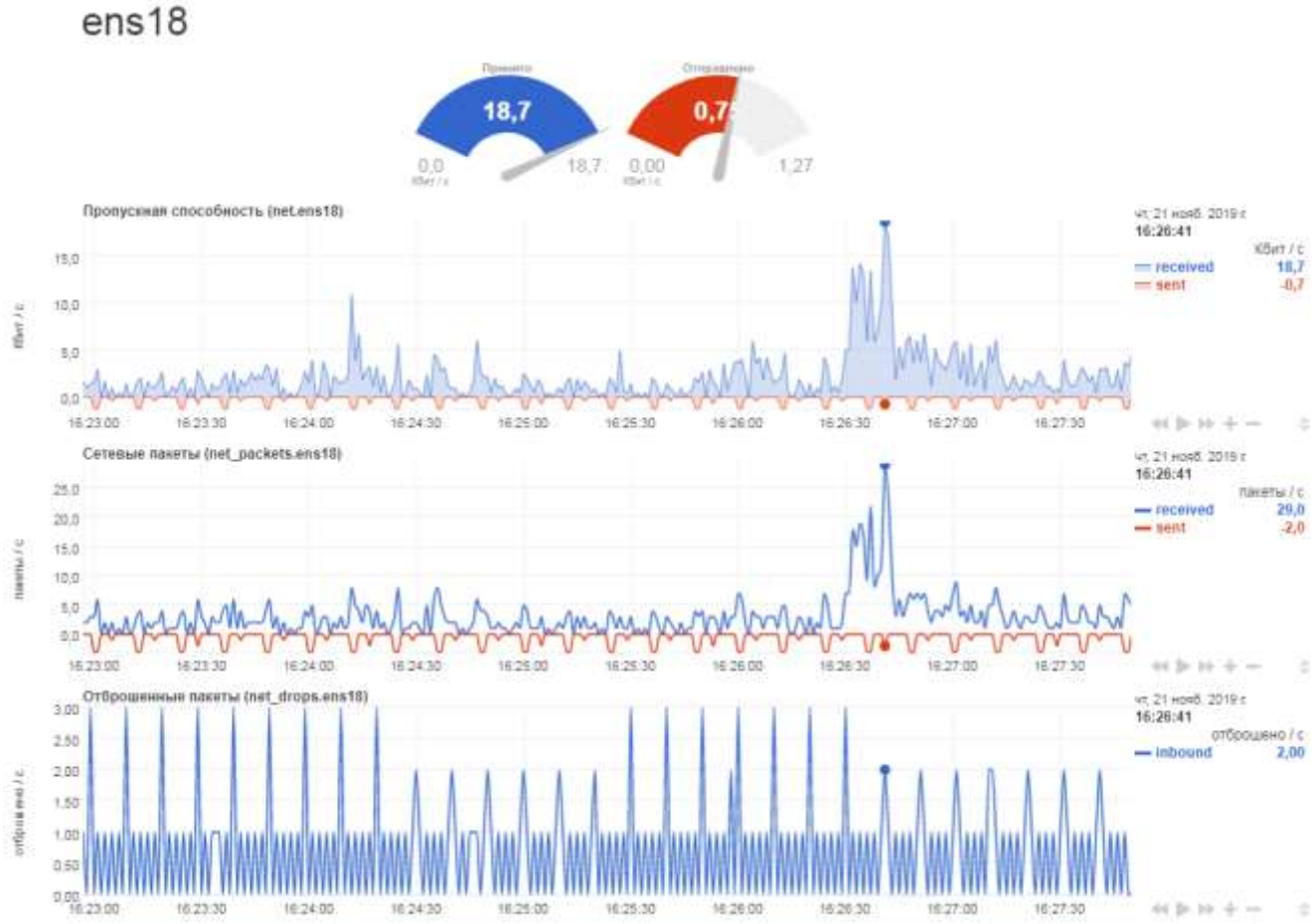


Рис. 79 – Сетевые интерфейсы

8.10. Мониторинг

В данном разделе приведены графики характеристик мониторинга (рис. 70) с отображением метрик, приведенных в таблице 34.

Таблица 34

График	Метрика	Доп. метрика	Единица измерения
Служба мониторинга	Севой трафик	in – входящие	Количество передаваемых данных в секунду
		out – исходящие	
	Использование процессора	user – пользователем	мс/с
		system – системой	
	Веб-клиенты		Количество подключенных клиентов
	Веб-запросы		Количество запросов в секунду
	Время ответа API	average – среднее	мс/запрос
		max – максимальное	
	Сжатие ответов API		процент
Внутренние процессы	Плагин CGroups: использование процессора	user – пользователем	мс/с
		system – системой	
	Плагин Proc: использование процессора	user – пользователем	мс/с
		system – системой	
Плагин ТС	Плагин ТС: использование процессора	user – пользователем	мс/с
		system – системой	
	Исполнение ТС скриптов		мс/запуск
Плагин приложений	Плагин приложений: использование процессора	user – пользователем	мс/с
		system – системой	
	Плагин приложений: файлы	files	файлов/с
		pids	
		fds	
		targets	
	Плагин приложений: коэффициент нормализации и Плагин приложений: коэффициент нормализации дочерних процессов	utime	процент %
		stime	
		gtime	
		minfit	
		majfit	

9. СИСТЕМА

Доступные функции раздела  «Система» ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 80) описаны в таблице 35.

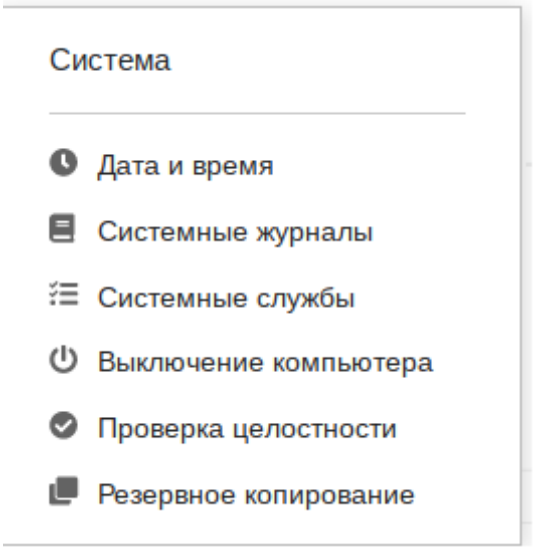


Рис. 80 – Раздел «Система»

Т а б л и ц а 35 – Функции раздела Система

Обозначение	Описание
Дата и время	Управление установкой даты и времени (п. 9.1)
Системные журналы	Журналирование информации о работе системы и ее ошибках (п. 9.2)
Системные службы	Управление системными службами (п. 9.4)
Выключение компьютера	Настройки управления выключением компьютера (п. 9.5)
Проверка целостности	Настройки процедуры проверки целостности (п. 9.6.2.1)
Резервное копирование	Настройки резервного копирования (п. 9.8.1.2)

9.1. Дата и время

Изменение и даты задается с помощью выпадающего календаря (рис. 81).

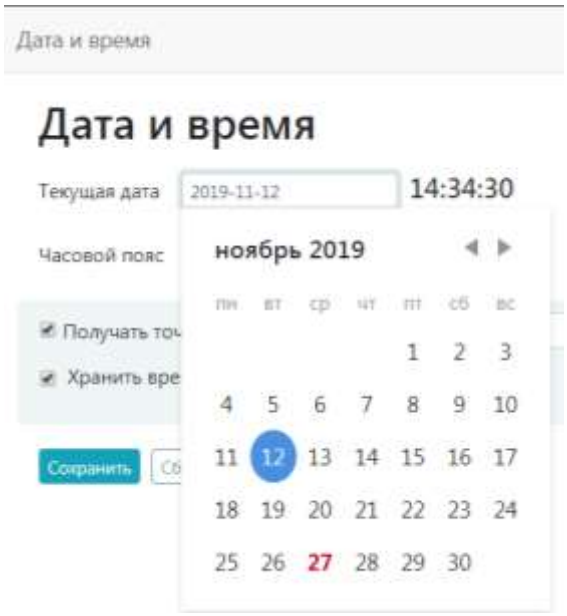


Рис. 81 – Выбор даты

Для установки актуального системного времени выберите напротив часового пояса пункт «Изменить» (рис. 82), в выпадающих списках выбрать страну и часовой пояс. Для подтверждения выбора нажать на кнопку «Сохранить».

Рис. 82 – Настройка часового пояса

Для настройки автоматической синхронизации времени с NTP-сервера установите флаг «Получать точное время с NTP-сервера» и укажите один из доступных серверов точного времени (рис. 83).

Если необходимо, чтобы происходил автоматический переход на летнее время и обратно следует выбрать флаг «Хранить время в BIOS по Гринвичу».

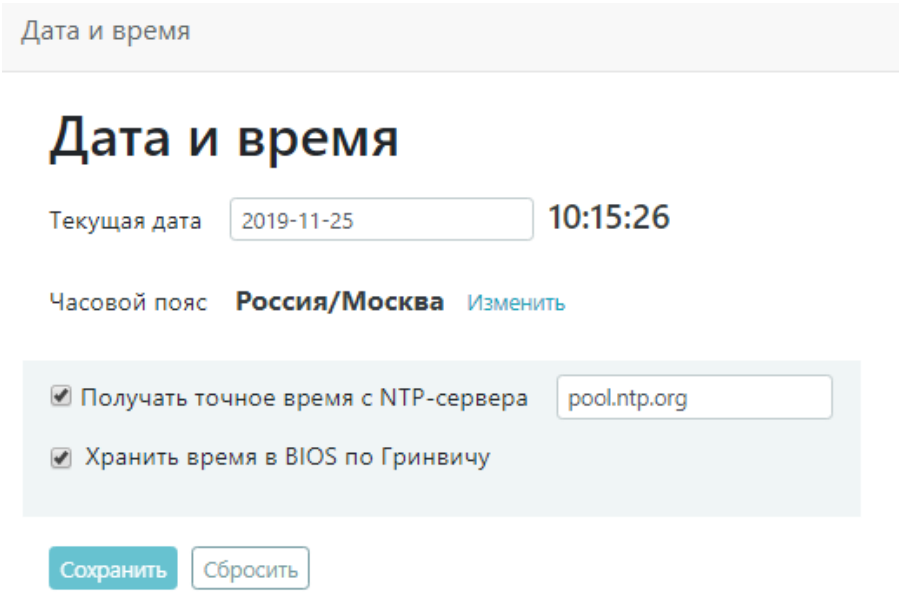


Рис. 83 – Настройка текущей даты и времени

Для сброса всех настроек нажать на кнопку «Сбросить» (рис. 83).

9.2. Аудит и системные журналы

Журналирование является основным источником информации о работе системы и ее ошибках.

Системные журналы (см. п. 9.2.11) – это централизованное место хранения и анализа сгенерированных событий от различных программных и аппаратных источников системы.

9.2.1. Журнал событий

Журналирование МЭ ИВК КОЛЬЧУГА-К имеет две основные части: одна работает всегда (базовое журналирование в `audit.c`), а другой можно управлять во время загрузки или работы сервера (аудит системных вызовов в `auditsc.c`).

Журналирование использует сетевой сокет для связи с пользовательским пространством. Если служба запущена – все сообщения журналируются через этот сетевой сокет. Иначе, сообщения журналируются через printk посредством службы syslog (по умолчанию).

Сообщения могут не журналироваться (настраивается отдельно) в зависимости от скорости появления сообщений или загруженности памяти.

Когда какая-то часть ядра генерирует часть сообщения аудита, эта часть будет немедленно послана в пользовательское пространство, и автоматически выставится флаг, указывающий, что этот системный вызов находится под аудитом. Таким образом, при выходе из системного вызова будет сформирована дополнительная информация (если включен аудит системных вызовов).

9.2.2. AUDITD – служба аудита Linux

AUDITD – это прикладной компонент системы аудита Linux. Он ведет протокол аудита на диске. Для просмотра протоколов предназначены команды `ausearch` и `aureport`. Команда `auditctl` позволяет настраивать правила аудита администратору МЭ ИВК КОЛЬЧУГА-К. Кроме того, при загрузке загружаются правила из файла `/etc/audit/rules.d/10-base-config.rules`. Некоторые параметры самой службы можно изменить в файле `auditd.conf`.

Файлы:

`/etc/audit/auditd.conf` – файл конфигурации службы аудита.

Синтаксис:

```
auditd [-f] [-l] [-n]
```

Опции:

- `-f` – не переходить в фоновый режим (для отладки). Сообщения программы будут направляться в стандартный вывод для ошибок (`stderr`), а не в файл;
- `-l` – включить следование по символическим ссылкам при поиске конфигурационных файлов;
- `-n` – не создавать дочерний процесс. Для запуска из `inittab`.

Сигналы:

- `SIGHUP` – перезагрузить конфигурацию – загрузить файл конфигурации с диска. Если в файле не окажется синтаксических ошибок, внесенные в него изменения вступят в силу. При этом в протокол будет добавлена запись о событии `DAEMON_CONFIG`. В противном случае действия службы будут зависеть от параметров `space_left_action`, `admin_space_left_action`, `disk_full_action`, `disk_error_action` файла `auditd.conf`;

- SIGTERM – прекратить обработку событий аудита и завершить работу, о чем предварительно занести запись в протокол;
- SIGUSR1 – создать новый файл для протокола, перенумеровав старые файлы или удалив часть из них, в зависимости от параметра max_log_size_action.

Файлы:

- /etc/audit/auditd.conf – файл конфигурации службы аудита;
- /etc/audit/audit.rules – файл правил аудита (загружается при запуске службы);
- /etc/audit/rules.d/ – каталог, содержащий отдельные наборы правил, которые будут скомпилированы в один файл утилитой augenrules.

9.2.3. Процесс аудита

Аудит в МЭ ИВК КОЛЬЧУГА-К производится по следующим правилам:

1) во время создания процесса, формируется контекст аудита и привязывается к структуре, описывающей процесс;
2) во время входа в системный вызов, заполняется следующая информация в контексте аудита, если он есть: номер системного вызова, дата и время, но не аргументы;
3) в ходе работы системного вызова перехватываются обращения к getname() и path_lookup(). Эти процедуры вызываются, когда ядро действительно собирается искать информацию, для принятия решения, будет ли системный вызов успешно выполнен или нет. Перехватывать вызовы нужно для того, чтобы не допустить копирование информации, которую генерирует getname, поскольку getname уже сделал приватную (для ядра) копию этой информации;
4) следует заметить, что сохранение копий всех аргументов системного вызова усложняет реализацию и требует увеличения расхода ресурсов. С этим патчем, к примеру, если пользователь непривилегированный, то chroot("foo") будет завершен с ошибкой – «foo» не будет отражено в

записи аудита, потому что ядро определяет еще перед поиском «foo», что работа системного вызова не может быть продолжена. Этот подход предотвращает сохранение пользовательской информации, которая может быть обманчивой или ненадежной (например, из-за атаки на совместно используемую память) в отличие от отчетной информации, фактически используемой ядром;

5) во время выхода из системного вызова генерируется та часть сообщения аудита, которая ответственна за информацию о системном вызове, включая имена файлов и номера inode (если доступны). Сообщение о системном вызове генерируется, если выставлен флаг, указывающий, что системный вызов находится под аудитом (он выставляется, например, часть ядра определяет, что должно формироваться сообщение для аудита). Следует заметить, что полное сообщение аудита приходит в пользовательское пространство по частям, это позволяет не хранить сообщения неопределенный срок внутри ядра;
6) во время завершения процесса контекст аудита уничтожается.

Во время шагов 1), 2) и 4) может быть выполнена простая фильтрация (например, для увеличения производительности – отключение аудита системных вызовов, выполняемых от имени пользователя, работающего с базой данных). Фильтрация может быть, как простой, так и сложной. Фильтрация реализована на столько полно на сколько возможно без существенного увеличения потребления ресурсов (например, `d_path()`).

В файле `/etc/audit/auditd.conf` определяются параметры службы аудита. На одной строке может быть не больше одной директивы. Директива состоит из ключевого слова (названия параметра), знака равенства и соответствующих ему данных (значения параметра).

Допустимые ключевые слова описаны в таблице 36.

Т а б л и ц а 36 – Директивы

Ключевое слово	Описание
log_file	Полное имя файла, в который следует записывать протокол.
log_format	Оформление данных в протоколе. Допустимы два значения: raw и nolog. При указании raw, данные будут записываться в том виде, в котором они получаются от ядра. Значение nolog отключает запись данных об аудите. Этот параметр не влияет на обработку данных диспетчером событий системы аудита.
priority_boost	Неотрицательное число, определяющее повышение приоритета выполнения службы аудита. Значение по умолчанию: 3. Для того чтобы не изменять приоритет, укажите 0.
Flush	Стратегия работы с дисковым буфером. Допустимые значения: none, incremental, data и sync. Вариант none, отключает какие-либо дополнительные действия со стороны службы по синхронизации буфера с диском. При значении incremental, запросы на перенос данных из буфера на диск выполняются с частотой задаваемой параметром freq. При значении data данные файла синхронизируются немедленно. Значение sync указывает на необходимость немедленной синхронизации, как данных, так и метаданных файла при записи на диск.
Freq	Максимальное число записей протокола, которые могут храниться в буфере. При достижении этого числа производится запись буферизованных данных на диск. Данный параметр допустим только в том случае, когда flush имеет значение incremental.
num_logs	Максимальное число файлов с протоколами. Используется в том случае, если параметр max_log_file_action имеет значение rotate. Если указано число меньше 2, при достижении ограничения на размер файла он обнуляется. Значение параметра не должно превышать 99. Значение по умолчанию: 0. При указании большого числа может потребоваться увеличить ограничение на количество ожидающих запросов. Это можно сделать в файле /etc/audit/audit.rules.
dispatcher	Диспетчер – программа, которой (на стандартный ввод) будут передаваться копии сообщений о событиях аудита. Она запускается (с правами администратора) службой аудита при загрузке последней.

Продолжение таблицы 36

Ключевое слово	Описание
`disp_qos`	Разрешение блокирования при взаимодействии с диспетчером. Для передачи информации диспетчеру используется буфер размером 128 Кбайт. Это значение является оптимальным для большинства случаев. Если блокирование запрещено (`lossy`), то все сообщения, поступающие при полном буфере, не будут доходить до диспетчера (записи о них по-прежнему будут вноситься в файл на диске, если только `log_format` не равно `nolog`). В случае, если блокирование разрешено (`lossless`), служба аудита будет ожидать появления свободного места в очереди, передавать сообщение диспетчеру и только потом записывать его на диск. Допустимые значения: `lossy` и `lossless`. Значение по умолчанию – `lossy`.
`max_log_file`	Ограничение на размер файла протокола в мегабайтах. Действие, выполняемое при достижении размера файла указанного значения, можно настроить с помощью следующего параметра.
`max_log_file_action`	Действие, предпринимаемое при достижении размером файла протокола максимального значения. Допустимые значения: `ignore`, `syslog`, `suspend`, `rotate` и `keep_logs`. Вариант ignore, отключает контроль над размером файла. При значении `syslog` в системный протокол будет внесено соответствующее сообщение. При значении `suspend` дальнейшее ведение протокола будет прекращено. Служба по-прежнему будет работать. При значении `rotate` текущий файл будет переименован и для протокола будет создан новый файл. Имя предыдущего протокола будет дополнено числом `1`, а номера других протоколов (если они имеются) будут увеличены на единицу. Таким образом, чем больше номер у протокола, тем он старше. Максимальное число файлов определяется параметром `num_logs` (естественно, соответствие ему достигается за счет удаления самых старых протоколов). Такое поведение аналогично поведению утилиты `logrotate`. Вариант `keep_logs` аналогичен предыдущему, но число файлов не ограничено.
`action_mail_acct`	Адрес электронной почты. Значение по умолчанию: `root`. Если адрес не локальный по отношению к данной системе, необходимо чтобы в ней был настроен механизм отправки почты. В частности, требуется наличие программы `/usr/lib/sendmail`.

Продолжение таблицы 36

Ключевое слово	Описание
space_left	Минимум свободного пространства в мегабайтах, при достижении которого должно выполняться действие, определяемое следующим параметром.
space_left_action	Действие, предпринимаемое при достижении объемом свободного пространства на диске указанного минимума. Допустимые значения – ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в action_mail_acct, отправляется уведомление. При значении exec <путь к программе> запускается программа по указанному пути. Передача параметров не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведет к переводу компьютера в однопользовательский режим. Указание halt приведет к выключению компьютера.
admin_space_left	Критический минимум свободного пространства в мегабайтах, при достижении которого должно выполняться действие, определяемое следующим параметром. Данное действие следует рассматривать как последнюю меру, предпринимаемую перед тем, как закончится место на диске. Значение настоящего параметра должно быть меньше значения space_left.
admin_space_left_action	Действие, предпринимаемое при достижении объемом свободного пространства на диске указанного критического минимума. Допустимые значения – ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в action_mail_acct отправляется уведомление. При значении exec <путь к программе> запускается программа по указанному пути. Передача параметров не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведет к переводу компьютера в однопользовательский режим. Указание halt приведет к выключению компьютера.

Окончание таблицы 36

Ключевое слово	Описание
disk_full_action	Действие, предпринимаемое при обнаружении отсутствия свободного пространства на диске. Допустимые значения – ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в action_mail_acct отправляется уведомление. При значении exec /некоторый-путь запускается сценарий по указанному пути. Передача параметров сценарию не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведет к переводу компьютера в однопользовательский режим. Указание halt приведет к выключению компьютера.
disk_error_action	Действие, предпринимаемое при возникновении ошибки в работе с диском. Допустимые значения – ignore, syslog, email, exec, suspend, single и halt. При значении ignore, никаких действий не производится. При значении syslog в системный протокол добавляется соответствующая запись. При значении email по адресу, указанному в action_mail_acct отправляется уведомление. При значении exec /некоторый-путь, запускается сценарий по указанному пути. Передача параметров сценарию не поддерживается. При значении suspend служба аудита прекратит вести протокол событий на диске, но будет продолжать работать. Указание single приведет к переводу компьютера в однопользовательский режим. Указание halt приведет к выключению компьютера.

Примечания:

1. В среде CAPP (Controlled Access Protection Profile – контролируемый профиль защиты доступа) ведение протоколов настолько важно, что невозможность его продолжения может служить основанием отказа в доступе к ресурсам. Поэтому рекомендуется выделять для файла /var/log/audit специальный раздел. Кроме того, параметру flush следует присвоить значение sync или data.

2. Для обеспечения полного использования раздела параметрам max_log_file и num_logs следует присвоить соответствующие значения. Учитывайте, что чем больше файлов создается на диске (и соответственно переименовывается), тем

больше времени будет уходить на обработку событий при достижении размером очередного файла максимума. Параметру `max_log_file_action` рекомендуется присвоить значение `keep_logs`.

3. Значение `space_left` должно быть таким, которое позволит администратору вовремя среагировать на предупреждение. Обычно в число действий, выполняемых администратором, входит запуск `aureport -t` и архивирование самых старых протоколов. Значение `space_left` зависит от системы, в частности от частоты поступления сообщений о событиях. Значение `space_left_action` рекомендуется установить в `email`. Если требуется отправка сообщения `snmp trap`, укажите вариант `exec`.

4. Установите значение `admin_space_left` таким образом, чтобы хватило свободного места для сохранения записей о последующих действиях администратора. Значение параметра `admin_space_left_action` следует установить в `single`, ограничив таким образом способы взаимодействия с системой консолью.

5. Действие, указанное в `disk_full_action`, выполняется, когда в разделе уже не осталось свободного места. Доступ к ресурсам машины должен быть полностью прекращен, т. к. нет возможности контролировать работу системы. Это можно сделать, указав значение `single` или `halt`.

6. Значение `disk_error_action` следует установить в `syslog`, `single`, либо `halt` в зависимости от соглашения относительно обращения со сбойным аппаратным обеспечением.

9.2.4. Утилита AUDITCTL

AUDITCTL используется для контроля поведения, получения состояния и добавления/удаления правил аудита.

Опции приведены в таблице 37.

Т а б л и ц а 37 – Опции

Опции	Описание
`-b backlog`	Установить максимальное количество доступных для аудита буферов, ожидающих обработки (значение в ядре по умолчанию – 64). Если все буферы заняты, то флаг сбоя будет выставлен ядром для его дальнейшей обработки.
`-i`	Игнорировать ошибки при чтении правил из файла.
`-l`	Вывести список всех правил по одному правилу в строке.
`-e [0..2]`	Установить флаг блокировки. 0 позволит на время отключить аудит, включить его обратно можно, передав 1 как параметр. Если установить значение опции 2, то это защитит конфигурацию аудита от изменений. Каждый, кто захочет воспользоваться этой возможностью, может поставить эту

Продолжение таблицы 37

Опции	Описание
	команду последней в файле `audit.rules`. После этой команды все попытки изменить конфигурацию будут отвергнуты с уведомлением в журналах аудита. В этом случае, чтобы задействовать новую конфигурацию аудита, необходимо перезагрузить систему аудита.

`-f [0..2]`	Установить способ обработки для флага сбоя:
	`0=silent`
	`1=printk`
	`2=panic`
	Эта опция позволяет определить, каким образом ядро будет обрабатывать критические ошибки. Например, флаг сбоя выставляется при следующих условиях: ошибки передачи в пространство службы аудита, превышение лимита буферов, ожидающих обработки, выход за пределы памяти ядра, превышение лимита скорости выдачи сообщений. Значение по умолчанию: 1. Для систем с повышенными требованиями к безопасности, значение 2 может быть более предпочтительно.

`-h`	Краткая помощь по аргументам командной строки.
`-k ключ`	Установить на правило ключ фильтрации. Ключ фильтрации – это произвольная текстовая строка длиной не больше 31 символа. Ключ помогает уникально идентифицировать записи, генерируемые в ходе аудита за точкой наблюдения.
`-m текст`	Послать в систему аудита пользовательское сообщение. Это может быть сделано только из-под учетной записи root.
`-p [r|w|x|a]`	Установить фильтр прав доступа для точки наблюдения. `r=чтение`, `w=запись`, `x=исполнение`, `a=изменение атрибута`. Не путайте эти права доступа с обычными правами доступа к файлу – они определяют типы системных вызовов, которые выполняют данные действия. Заметьте, системные вызовы `read` и `write` не включены в этот набор, поскольку логи аудита были бы перегружены информацией о работе этих вызовов.
`-r частота`	Установить ограничение скорости выдачи сообщений в секунду (0 – нет ограничения). Если эта частота не нулевая и она превышается в ходе аудита, флаг сбоя выставляется ядром для выполнения соответствующего действия. Значение по умолчанию: 0.
`-s`	Получить статус аудита.

Продолжение таблицы 37

Опции	Описание
`-a список,действие`	Добавить правило с указанным действием к концу списка. Заметьте, что запятая разделяет эти два значения. Отсутствие запятой вызовет ошибку. Ниже описаны имена доступных списков: - `task` – добавить правило к списку, отвечающему за процессы. Этот список правил используется только во время создания процесса – когда родительский процесс вызывает `fork()` или `clone()`. При использовании этого списка возможно использовать только те поля, которые известны во время создания процесса: `uid`, `gid` и т. д.; - `entry` – добавить правило к списку, отвечающему за точки входа системных вызовов. Этот список применяется, когда необходимо создать событие для аудита, привязанное к точкам входа системных вызовов; - `exit` – добавить правило к списку, отвечающему за точки выхода из системных вызовов. Этот список применяется, когда необходимо создать событие для аудита, привязанное к точкам выхода из системных вызовов; - `user` – добавить правило, отвечающее за список фильтрации пользовательских сообщений. Этот список используется ядром, чтобы отфильтровать события, приходящие из пользовательского пространства, перед тем как они будут переданы службе аудита. Необходимо отметить, что только следующие поля могут быть использованы: `uid`, `auid`, `gid` и `pid`. Все остальные поля будут обработаны, как если бы они не совпали; - `exclude` – добавить правило к списку, отвечающего за фильтрацию событий определенного типа. Этот список используется, чтобы отфильтровывать ненужные события. Например, чтобы не видеть `avc` сообщения, используйте этот список. Тип сообщения задается в поле `msgtype`. Ниже описаны доступные действия для правил: - `never` – аудит не будет генерировать никаких записей. Это может быть использовано для подавления генерации событий. Обычно необходимо подавлять генерацию в верху списка, а не внизу, т. к. событие инициируется на первом совпавшем правиле; - `always` – установить контекст аудита. Всегда заполнять его во время входа в системный вызов, и всегда генерировать запись во время выхода из системного вызова.

Продолжение таблицы 37

Опции	Описание
`-R файл`	Читать правила из файла. Правила должны быть расположены по одному в строке и в том порядке, в каком они должны исполняться. Следующие ограничения накладываются на файл: владельцем должен быть root и доступ на чтение должен быть только у него. Файл может содержать комментарии, начинающиеся с символа #. Правила, расположенные в файле, идентичны тем, что набираются в командной строке, без указания `auditctl`.
`-S [Имя или номер системного вызова|all]`	Любой номер или имя системного вызова может быть использовано. Также возможно использование ключевого слова `all`. Если какой-либо процесс выполняет указанный системный вызов, то аудит генерирует соответствующую запись. Если значения полей сравнения заданы, а системный вызов не указан, правило будет применяться ко всем системным вызовам. В одном правиле может быть задано несколько системных вызовов – это положительно сказывается на производительности, поскольку заменяет обработку нескольких правил.
`-F [n=v | n!=v | n<v | n>v | n<=v | n>=v | n&v | n&=v]`	Задать поле сравнения для правила. Атрибуты поля следующие: объект, операция, значение. Возможно задать до 64 полей сравнения в одной команде. Каждое новое поле должно начинаться с `-F`. Аудит будет генерировать запись, если произошло совпадение по всем полями сравнения. Допустимо использование одного из следующих 8 операторов: равно, не равно, меньше, больше, меньше либо равно, больше либо равно, битовая маска (`n&v`) и битовая проверка (`n&=v`). Битовая проверка выполняет операцию `and` над значениями и проверяет, равны ли они. Битовая маска просто выполняет операцию `and`. Поля, оперирующие с идентификатором пользователя, могут также работать с именем пользователя – программа автоматически получит идентификатор пользователя из его имени. То же самое можно сказать и про имя группы. Поля сравнения могут быть заданы для следующих объектов: `a0, a1, a2, a3` Четыре первых аргумента, переданных системному вызову. Строковые аргументы не поддерживаются. Это связано с тем, что ядро должно получать указатель на строку, а проверка поля по значению адреса указателя не желательна. Таким образом, необходимо использовать только цифровые значения.

Продолжение таблицы 37

Опции	Описание
`-A список,действие`	Добавить правило с указанным действием в начало списка.
`-d список,действие`	Удалить правило с указанным действием из списка. Правило удаляется только в том случае, если полностью совпали и имя системного вызова и поля сравнения.
`-D`	Удалить все правила и точки наблюдения.
`arch`	Архитектура процессора, на котором выполняется системный вызов. Используйте `uname -m`, чтобы определить архитектуру. Если архитектура неизвестна и необходимо использовать таблицу 32-х битных системных вызовов, при этом компьютер поддерживает 32 бита, можно использовать b32. Подобно этому b64 может быть использовано для использования таблицы 64-х битных системных вызовов.
`auid`	Это аббревиатура: audit uid – идентификатор пользователя, использованный для входа в систему.
`devmajor`	Главный номер устройства (DeviceMajorNumber).
`devminor`	Вспомогательный номер устройства (Device Minor Number).
`egid`	Действительный идентификатор группы.
`euid`	Действительный идентификатор пользователя.
`exitPORT`	Значение, возвращаемое системным вызовом при выходе.
`fsgid`	Идентификатор группы, применяемый к файловой системе.
`fsuid`	Идентификатор пользователя, применяемый к файловой системе.
`gid`	Идентификатор группы.
`inode`	Номер inode.
`key`	Альтернативный способ установить ключ фильтрации. См. выше описание опции `-k`.
`msgtype`	Используется для проверки совпадения с числом, описывающим тип сообщения. Может быть использован только в списке exclude.
`path`	Полный путь к файлу для точки наблюдения. См.ниже описание опции `-w`. Может быть использован только в списке exit.
`perm`	Фильтр прав доступа для файловых операций. См. выше описание опции `-p`. Может быть использован только в списке exit.
`pers`	Персональный номер операционной системы.
`pid`	Идентификатор процесса.
`ppid`	Идентификатор родительского процесса.
`sgid`	Установленный идентификатор группы.

Окончание таблицы 37

Опции	Описание
`success`	Если значение, возвращаемое системным вызовом, больше либо равно 0, данный объект будет равен «true/yes», иначе «false/no». При создании правила используйте 1 вместо «true/yes» и 0 вместо «false/no».
`suid`	Установленный идентификатор пользователя.
`uid`	Идентификатор пользователя.
`-w путь`	Добавить точку наблюдения за файловым объектом, находящемуся по указанному пути. Добавление точки наблюдения к каталогу верхнего уровня запрещено ядром. Групповые символы (wildcards) также не могут быть использованы, попытки их использования будут генерировать предупреждающее сообщение. Внутренне точки наблюдения реализованы как слежение за inode. Таким образом, если установить точку наблюдения за каталогом, можно увидеть файловые события, которые в действительности будут означать обновления метаданных этой inode, и можно не увидеть событий, непосредственно связанных с файлами. Если необходимо следить за всеми файлами в каталоге, рекомендуется создавать индивидуальную точку наблюдения для каждого файла. В противоположность к правилам аудита системных вызовов, точки наблюдения не оказывают влияния на производительность, связанную с количеством правил, посылаемых в ядро.
`-W путь`	Удалить точку наблюдения за файловым объектом, находящемуся по указанному пути.

Примеры:

1) чтобы увидеть все системные вызовы, используемые определенным процессом:

```
auditctl -a entry,always -S all -F pid=1005
```

2) чтобы увидеть все файлы, открытые определенным пользователем:

```
auditctl -a exit,always -S open -F auid=510
```

3) чтобы увидеть неудачные попытки вызова системной функции open:

```
auditctl -a exit,always -S open -F success!=0
```

Файлы правил хранятся в каталоге `/etc/audit/rules.d/` и обязательно должны иметь расширение `.rules`.

9.2.5. Предопределенные правила аудита

В файле правил аудита `/etc/audit/rules.d/10-base-config.rules`, запускаемом при старте службы аудита определен перечень правил.

Подробное описание предопределенных объектов аудита, типов системных вызовов и ключевых слов приведено в таблице 38.

Т а б л и ц а 38

Объект аудита	Типы системных вызовов для объекта наблюдения	Ключевое слово
/etc/passwd /etc/tcb /etc/shadow /etc/audit /etc/pam.d	w=запись, a=изменение атрибута	secur
/bin/mount	w=запись, x=исполнение, a=изменение атрибута	mount
/etc/osec	w=запись, a=изменение атрибута	integ
/sbin/xtables-multi	w=запись, x=исполнение, a=изменение атрибута	iptabl
/etc/init.d/iptables	w=запись, x=исполнение, a=изменение атрибута	afirewall
/etc/net	w=запись, a=изменение атрибута	s_net
/etc/init.d/network	w=запись, x=исполнение, a=изменение атрибута	s_net
/etc/openssh	w=запись, a=изменение атрибута	s_net
Системное время	Все изменения внутреннего представления времени.	clock_time
Аудитор	Все системные вызовы, выполняемые пользователем аудитор.	uau
Администратор	Все системные вызовы, выполняемые пользователем администратор.	uad
/etc/rsnapshot	w=запись, x=исполнение, a=изменение атрибута	Backup
sbin/service	x=исполнение	statservice
sbin/chkconfig	x=исполнение	statchkonfig

Поиск событий в лог-файле аудита осуществляется командой:

```
ausearch -k [ключевое слово]
```

Для добавления новых правил аудита необходимо в `/etc/audit/rules.d/` создать новый файл вида `число-название.rules`, куда необходимо внести соответствующие правила аудита.

Правила аудита создаются в соответствии порядком работы с AUDITCLT (таблица 37).

9.2.6. AUREPORT

aureport – это инструмент, который генерирует итоговые отчеты на основе логов службы аудита. aureport может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов. В шапке каждого отчета для каждого столбца есть заголовок – это облегчает понимание данных. Все отчеты, кроме основного итогового отчета, содержат номера событий аудита. Используя их, можно найти полные данные о событии с помощью `ausearch -a номер события`. Если в отчете слишком много данных, можно задать время начала и время окончания для уточнения временного промежутка. Отчеты, генерируемые AUREPORT, могут быть использованы как исходный материал для получения более развернутых отчетов.

Опции приведены в таблице 39.

Т а б л и ц а 39 – Опции

Опции	Описание
`-au, --auth`	Отчет о всех попытках аутентификации.
`-a, --avc`	Отчет о всех avc сообщениях.
`-c, --config`	Отчет о изменениях конфигурации.
`-cr, --crypto`	Отчет о событиях, связанных с кодированием.
`-e, --event`	Отчет о событиях.
`-f, --filуде`	Отчет о файлах.
`--failed`	Для обработки в отчетах выбирать только неудачные события. По умолчанию показываются и удачные, и неудачные события.
`-h, --host`	Отчет о хостах.
`-i, --interpret`	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет оттранслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен aureport. т. е. если

Продолжение таблицы 39

Опции	Описание
	учетные записи пользователей переименованы или на компьютере нет таких же учетных записей, то можно получить результаты, вводящие в заблуждение.
`-if, --input файл`	Использовать указанный файл вместо логов аудита. Это может быть полезно при анализе логов с другой машины или при анализе частично сохраненных логов.
`-l, --login`	Отчет о попытках входа в систему.
`-m, --mods`	Отчет об изменениях пользовательских учетных записей.
`-ma, --mac`	Отчет о событиях в системе, обеспечивающей мандатное управление доступом – Mandatory Access Control (MAC).
`-p, --pid`	Отчет о процессах.
`-r, --response`	Отчет о реакциях на аномальные события.
`-s, --syscall`	Отчеты о системных вызовах.
`--success`	Для обработки в отчетах выбирать только удачные события. По умолчанию показываются и удачные, и неудачные события.
`--summary`	Генерировать итоговый отчет, который дает информацию только о количестве элементов в том или ином отчете. Такой режим есть не у всех отчетов.
`-t, --log`	Этот параметр генерирует отчет о временных рамках каждого отчета.
`-te, --end [дата] [время]`	Искать события, которые произошли раньше (или вовремя) указанной временной точки. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время – как 18:00:00. Можно также использовать ключевые слова: now (сейчас), recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent – 10 минут назад. yesterday – первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из региональных настроек (localtime). this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.

Окончание таблицы 39

Опции	Описание
`-tm`, `--terminal`	Отчет о терминалах.
`-ts`, `--start [дата] [время]`	Искать события, которые произошли после (или вовремя) указанной временной точки. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время – как 18:00:00. Можно также использовать ключевые слова: now (сейчас), recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent – 10 минут назад. yesterday – первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из региональных настроек (localtime). this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-u`, `--user`	Отчет о пользователях.
`-v`, `--version`	Вывести версию программы и выйти.
`-x`, `--executable`	Отчет об исполняемых объектах.

9.2.7. AUSEARCH

Программа AUSEARCH является инструментом поиска по журналу аудита. AUSEARCH может также принимать данные со стандартного ввода (stdin) до тех пор, пока на входе будут необработанные данные логов. Все условия, указанные в параметрах, объединяются логическим «И». К примеру, при указании `-m` и `-ui` в качестве параметров будут показаны события, соответствующие заданному типу и идентификатору пользователя.

Стоит отметить, что каждый системный вызов ядра из пользовательского пространства и возвращение данных в пользовательское пространство имеет один уникальный (для каждого системного вызова) идентификатор события.

Различные части ядра могут добавлять дополнительные записи. Например, в событие аудита для системного вызова open добавляется запись PATH с именем файла. AUSEARCH показывает все записи события вместе. Это означает, что при запросе определенных записей результат может содержать записи SYSCALL.

Также помните, что не все типы записей содержат указанную информацию. Например, запись PATH не содержит имя хоста или loginuid.

Опции приведены в таблице 40.

Т а б л и ц а 40 – Опции

Опции	Описание
-a, --event audit-event-id	Искать события с заданным идентификатором события. Сообщения обычно начинаются примерно так: msg=audit(1116360555.329:2401771). Идентификатор события – это число после ':'. Все события аудита, связанные с одним системным вызовом, имеют одинаковый идентификатор.
-c, --comm comm-name	Искать события с заданным comm-name. comm-name – имя исполняемого файла задачи.
-f, --file file-name	Искать события с заданным именем файла.
-ga, --gid-all all-group-id	Искать события с заданным эффективным или обычным идентификатором группы.
-ge, --gid-effective effective-group-id	Искать события с заданным эффективным идентификатором группы или именем группы.
-gi, --gidgroup-id	Искать события с заданным идентификатором группы или именем группы.
-h, --help	Справка.
-hn, --hosthost-name	Искать события с заданным именем хоста. Имя хоста может быть именем хоста, полным доменным именем или цифровым сетевым адресом.
-i, --interpret	Транслировать числовые значения в текстовые. Например, идентификатор пользователя будет оттранслирован в имя пользователя. Трансляция выполняется с использованием данных с той машины, где запущен ausearch. т. е. если учетные записи пользователей переименованы или таких же учетных записей нет на компьютере, то можно получить результаты, вводящие в заблуждение.

Продолжение таблицы 40

Опции	Описание
`-if`, `--input file-name`	Использовать указанный файл вместо логов аудита. Это может быть полезно при анализе логов с другой машины или при анализе частично сохраненных логов.
`-k`, `--key key-string`	Искать события с заданным ключевым словом.
`-m`, `--message message-type | comma-sep-message-type-list`	Искать события с заданным типом. Возможно указать список значений, разделенных запятыми. Можно указать несуществующий в событиях тип ALL, который позволяет получить все сообщения системы аудита. Список допустимых типов большой и будет показан, если указать эту опцию без значения. Тип сообщения может быть строкой или числом. В списке значений этого параметра в качестве разделителя используются запятые и пробелы недопустимы.
`-p`, `--pid process-id`	Искать события с заданным идентификатором процесса.
`-pp`, `--ppid parent-process-id`	Искать события с заданным идентификатором родительского процесса.
`-r`, `--raw`	Необработанный вывод. Используется для извлечения записей для дальнейшего анализа.
`-sc, --success syscall-name-or-value`	Искать события с заданным системным вызовом. Можно указать его номер или имя. Если указали имя, оно будет проверено на компьютере, где запущен ausearch.
`-sv`, `--success success-value`	Искать события с заданным флагом успешного выполнения. Допустимые значения: `yes` (успешно) и `no` (неудачно).
`-te`, `--end [end-date] [end-time]`	Искать события, которые произошли раньше (или вовремя) указанной временной точки. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается текущий момент (now). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время – как 18:00:00. Можно также использовать ключевые слова: now (сейчас), recent, today, yesterday, this-week, this-month, this-year. Today означает первую секунду

Продолжение таблицы 40

Опции	Описание
	после полуночи текущего дня. recent – 10 минут назад. yesterday – первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из региональных настроек (localtime). this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-ts`, `--start [start-date] [start-time]`	Искать события, которые произошли после (или во время) указанной временной точки. Формат даты и времени зависит от региональных настроек. Если дата не указана, то подразумевается текущий день (today). Если не указано время, то подразумевается полночь (midnight). Используйте 24-часовую нотацию времени, а не AM/PM. Например, дата может быть задана как 10/24/2005, а время – как 18:00:00. Можно также использовать ключевые слова: now (сейчас), recent, today, yesterday, this-week, this-month, this-year. today означает первую секунду после полуночи текущего дня. recent – 10 минут назад. yesterday – первую секунду после полуночи предыдущего дня. this-week означает первую секунду после полуночи первого дня текущей недели, первый день недели определяется из региональных настроек (localtime). this-month означает первую секунду после полуночи первого числа текущего месяца. this-year означает первую секунду после полуночи первого числа первого месяца текущего года.
`-tm`, `--terminal terminal`	Искать события с заданным терминалом. Некоторые службы (такие как `cron` и `atd`) используют имя службы как имя терминала.
`-ua`, `--uid-all all-user-id`	Искать события, у которых любой из идентификатора пользователя, эффективного идентификатора пользователя или `loginuid` (`auid`) совпадают с заданным идентификатором пользователя.

Окончание таблицы 40

Опции	Описание
`-ue`, `--uid-effective effective-user-id`	Искать события с заданным эффективным идентификатором пользователя.
`-ui`, `--uid user-id`	Искать события с заданным идентификатором пользователя.
`-ul`, `--loginuid login-id`	Искать события с заданным идентификатором пользователя. Все программы, которые его используют, должны использовать `pam_loginuid`.
`-v`, `--verbose`	Показать версию и выйти.
`-w`, `--word`	Совпадение с полным словом. Поддерживается для имени файла, имени хоста, терминала.
`-x`, `--executable executable`	Искать события с заданным именем исполняемой программы.

9.2.8. AUTRACE

autrace – это программа, которая добавляет правила аудита для того, чтобы следить за использованием системных вызовов в указанном процессе подобно тому, как это делает strace. После добавления правил она запускает процесс с указанными аргументами. Результаты аудита будут либо в логах аудита (если служба аудита запущена), либо в системных логах. Внутри autrace устроена так, что удаляет все предыдущие правила аудита, перед тем как запустить указанный процесс и после его завершения. Поэтому, в качестве дополнительной меры предосторожности, программа не запустится, если перед ее использованием правила не будут удалены с помощью audtictl – предупреждающее сообщение известит об этом.

Опции: `-r`

Ограничить сбор информации о системных вызовах только теми, которые необходимы для анализа использования ресурсов. Это может быть полезно при моделировании внештатных ситуаций, к тому же позволяет уменьшить нагрузку на файлы логирования.

Пример обычного использования программы:

```
autrace /bin/ls /tmp
ausearch --start recent -p 2442 -i
```

Еще один пример для режима ограниченного сбора информации:

```
-r /bin/ls
ausearch --start recent -p 2450 --raw | aureport --file --summary
ausearch --start recent -p 2450 --raw | aureport --host -summary
```

9.2.9. Панель уведомлений

Уведомления об основных событиях безопасности отображаются на панели пользователя в ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 84). Количество поступивших уведомлений отображается в счетчике (см. рис. 13) на панели уведомлений.

Для просмотра списка уведомлений и детализированной информации, нужно навести курсор и щелкнуть левой кнопкой мыши на счетчик уведомлений 

.

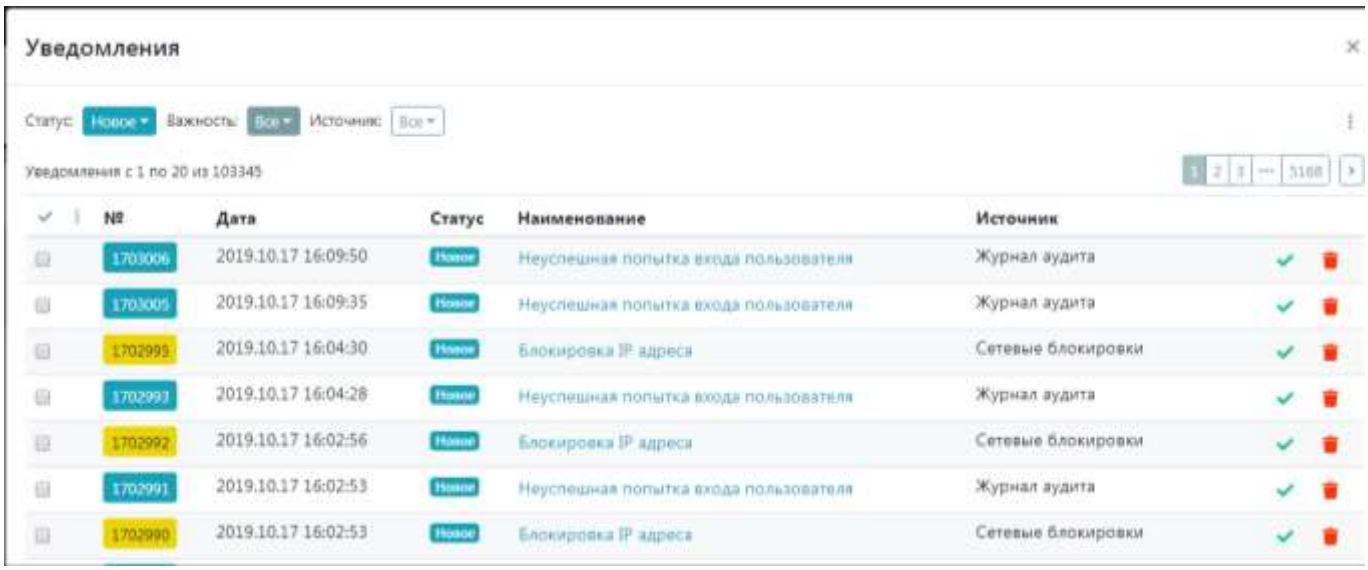


Рис. 84 – Окно просмотра списка уведомлений

Фильтр управления отображением списка уведомлений расположен в верхней части окна (см. рис. 84). Настройка просмотра формируется по значениям видов уведомлений:

1) статус (рис. 85):
   - новое;
   - подтверждено;
   - все;
2) важность:
   - отказ системы;

- тревога;
- критично;
- ошибка;
- предупреждение;
- уведомление;
- информационное сообщение;
- отладочная информация;
- все;

3) источник:

- вход в веб-интерфейс;
- журнал аудита;
- проверка целостности;
- сетевые блокировки;
- системный журнал;
- межсетевой экран;
- резервное копирование;
- события сервера;
- все.

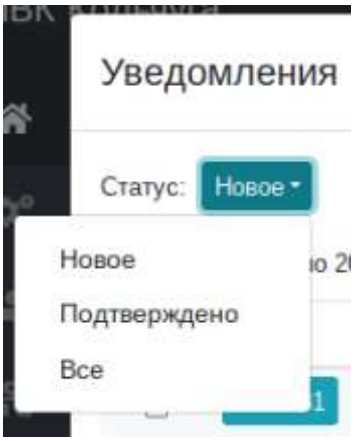


Рис. 85 – Статус уведомления

После выставления параметров фильтра управлением отображением списка, ко всем отображаемым уведомлениям, можно нажать на кнопку (в конце строки фильтра) и применить одно из действий (рис. 86):

- подтвердить отфильтрованные;
- удалить отфильтрованные;
- подтвердить все;
- удалить все.

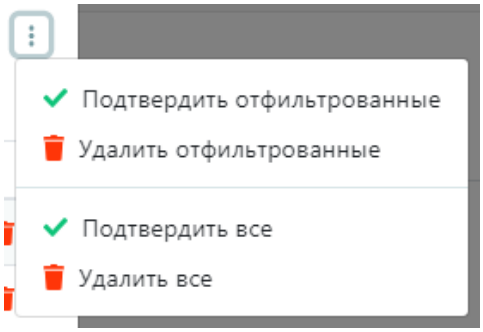


Рис. 86

Результат примененных действий отображается в виде всплывающего сообщения.

Все уведомления нумеруются, указывается дата и время, наименование произошедшего события.

В каждой строке уведомления есть две кнопки:

- «Подтвердить» – позволяет отметить статус ознакомления администратора с содержанием уведомления;
- «Удалить» – позволяет удалить информацию об уведомлении.

Несколько уведомлений можно вручную отметить флагом или выбрать все отображаемые на странице с помощью кнопки , далее применить к ним действия «Подтвердить» или «Удалить» с помощью контекстного меню (рис. 87) по мере ознакомления и появления.

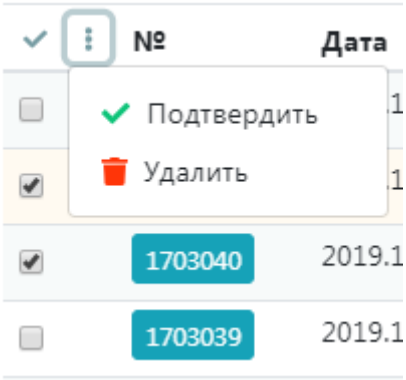


Рис. 87 – Кнопка контекстного меню

Для просмотра подробной информации об уведомлении нажмите на его наименование в списке (рис. 88).

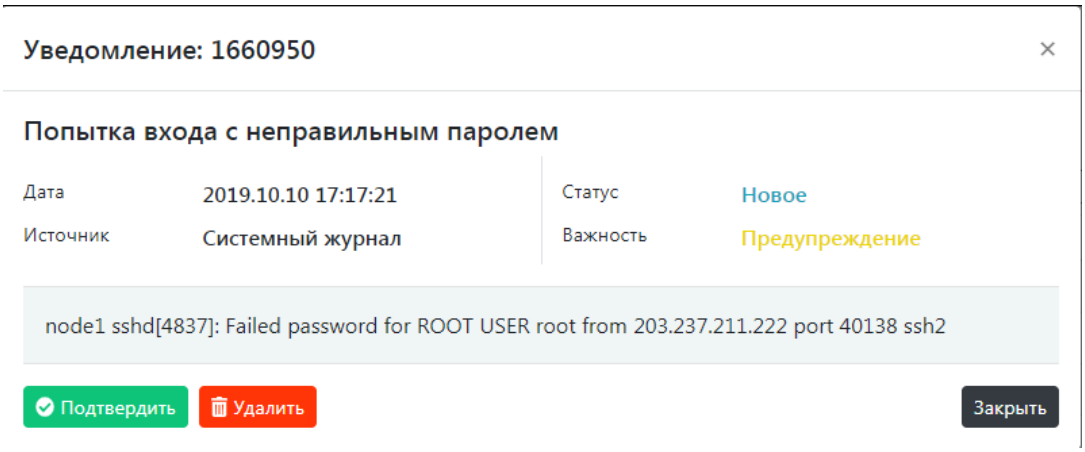


Рис. 88

9.2.10. Настройка уведомлений

Настройка формирования и отправки уведомлений МЭ ИВК КОЛЬЧУГА-К осуществляется через конфигурационный файл:

`/usr/share/web-kolchuga/conf/notifications.conf`

Параметры задаются в формате JSON. При изменении конфигурационного файла требуется перезапуск сервера ГИ МЭ ИВК КОЛЬЧУГА-К – служба node.

В системе предусмотрены несколько механизмов формирования уведомлений, а именно:

- уведомления на основе анализа данных из файлов журналов;

- уведомления на основе анализа изменений файлов;

- уведомления на основе внутренних событий сервера ГИ МЭ ИВК КОЛЬЧУГА-К.

Предусмотрены следующие уровни важности уведомлений:

- `emergency = 0` – отказ системы;
- `alert = 1` – тревога;
- `critical = 2` – критично;
- `error = 3` – ошибка;
- `warning = 4` – предупреждение;
- `notice = 5` – уведомление;
- `informational = 6` – информационное сообщение;
- `debug = 7` – отладочная информация.

Важность формируемых уведомлений определяется в настройках индивидуально для каждого уведомления.

Все сформированные уведомления сохраняются в локальную базу данных сервера. Также сформированные уведомления могут быть отправлены на электронную почту (п. 9.2.10.5).

9.2.10.1. Анализ файлов журнала для формирования уведомлений

Анализ файлов журнала построен на основе регулярных выражений и происходит при каждом изменении файлов журналов, заданных в настройках формирования уведомлений (п. 9.2.10.2).

Настройка анализатора файлов журналов осуществляется через конфигурационный файл `/usr/share/web-kolchuga/conf/log.parsers.conf`. Параметры задаются в формате JSON. При изменении конфигурационного файла требуется перезапуск сервера ГИ МЭ ИВК КОЛЬЧУГА-К – служба node.

В параметре `parsers` определен список анализаторов файлов журналов. Для каждого формата анализируемых файлов журнала необходимо задать свой анализатор. Один анализатор может быть использован для нескольких файлов журналов, имеющих идентичный формат записей.

Настройки анализатора определяют, каким образом и какие данные необходимо получать из каждой записи журнала.

Доступны следующие параметры анализатора файлов:

1) `name` – уникальное (в рамках файла конфигурации) имя анализатора, которое используется при формировании уведомлений;
2) `format` – настройка правил извлечения данных из записей файла журнала; имеет вложенные параметры:
   - `pattern` – шаблон записи файла журнала. Шаблон используется в качестве регулярного выражения, с помощью которого происходит извлечение данных из записей журналов. Использование скобочных групп регулярных выражений в шаблоне не допускается;
   - `variables` – переменные, используемые в шаблоне записи журнала. Список переменных, используемых для извлечения данных. Переменные `date` (дата записи) и `text` (текст записи) являются обязательными для использования в шаблоне записи журнала. Значение переменной – это регулярное выражение для поиска значения переменной. Использование скобочных групп регулярных выражений в значении переменной не допускается. Данные переменные можно использовать (для проверки или отображения) при формировании уведомлений;
3) `dateFormat` – формат даты, извлеченной в параметр `date` шаблона записи (необходим для преобразования полученной строки в дату);
4) `dateAsLong` – необходимо установить true, если в записи журнала дата представлена в миллисекундах;
5) `calculateYear` – флаг автоматического вычисления года для даты на основе даты изменения файла журнала (необходимо установить true, в случае, если дата записи журнала не имеет года).

9.2.10.2. Формирование уведомлений на основе анализа файлов журналов

Настройки формирования уведомлений на основе анализа файлов журналов определяются параметром `logsNotifications` конфигурационного файла.

С помощью вложенного параметра enable происходит включение/выключение формирования данных уведомлений. В параметре `watchConfigs` задаются правила формирования уведомлений. Каждое правило (источник уведомлений) определяется следующими параметрами:

- `name` – имя источника уведомлений;
- `file` – абсолютный путь к анализируемому файлу журнала;
- `parser` – имя используемого анализатора файла журнала;
- `events` – перечень формируемых уведомлений для данного источника.

Каждое формируемое уведомление определяется следующими параметрами:

- `enable` – включение/выключение формирования уведомления;
- `severity` – важность уведомления (указывается имя важности латинскими буквами);
- `name` – заголовок уведомления;
- `text` – шаблон текста уведомления;
- `conditions` – условия формирования уведомления.

В значении параметра `text` можно использовать переменные, сформированные указанным анализатором при извлечении данных из записи журнала. При использовании в шаблоне переменных их необходимо указывать в фигурных скобках {}. В качестве текста записи журнала (переменная анализатора `text`) необходимо использовать переменную `logText`. По умолчанию, для параметра `text` используется значение: `{logText}`.

Пример шаблона уведомления с использованием переменных:

```
"text": "Событие журнала аудита: {type}\n{logText}"
```

Параметром `conditions` определяются условия, при которых для записи журнала будет сформировано уведомление.

Доступны следующие параметры, определяющие условия формирования уведомлений:

- `regex` – регулярное выражение для проверки текста (извлеченного анализатором в переменную `text`) записи журнала;
- `string` – проверка вхождения строки в текст (извлеченный анализатором в переменную `text`) записи журнала;
- `variables` – проверка соответствия или несоответствия переменной (извлеченной анализатором) заданному значению.

Для проверки соответствия переменной заданному значению необходимо указать в параметре `variables` имя проверяемой переменной и ее значение.

Например: `"variables": {"type": "USER_AUTH"}`, что соответствует условию: переменная `type` равна значению `USER_AUTH`.

Для проверки несоответствия переменной заданному значению необходимо указать в параметре `variables` имя проверяемой переменной и ее значение в формате JSON полями `equal` и `value`. Поле `equal` должно принимать значение `false`, а в поле `value` необходимо указать имя проверяемой переменной.

Например: `"variables": {"type": {"equal": false, "value": "CONFIG_CHANGE"}}`, что соответствует условию: переменная `type` не равна значению `CONFIG_CHANGE`.

9.2.10.3. Формирование уведомлений при изменении файлов

Настройки формирования уведомлений на основе анализа файлов журналов определяются параметром `filesChangesNotifications` конфигурационного файла.

С помощью вложенного параметра `enable` происходит включение/выключение формирования данных уведомлений. В параметре `watchConfigs` задаются правила формирования уведомлений. Каждое правило (источник уведомлений) определяется следующими параметрами:

- `name` – имя источника уведомлений;
- `file` – абсолютный путь файлу;
- `events` – перечень формируемых уведомлений для данного источника.

Каждое формируемое уведомление определяется следующими параметрами:

- `enable` – включение/выключение формирования уведомления;
- `severity` – важность уведомления (указывается имя важности латинскими буквами);
- `name` – заголовок уведомления;
- `text` – шаблон текста уведомления;
- `conditions` – условия формирования уведомления.

В значении параметра `text` можно использовать переменные, их необходимо указывать в фигурных скобках {}:

- `date` – дата изменения файла;
- `file` – путь к файлу;
- `fileData` – содержимое файла.

Пример использования переменных в шаблоне уведомления:

```
"text": "Изменение контролируемых файлов.\nОтчет {file} от {date}.\n-----\n{fileData}\n-----\n"
```

Параметром `conditions` определяются условия, при которых будет сформировано уведомление при изменении отслеживаемого файла. Доступны следующие параметры, определяющие условия формирования уведомлений:

- `regex` – регулярное выражение для проверки содержимого отслеживаемого файла;
- `string` – проверка вхождения строки в содержимое отслеживаемого файла.

9.2.10.4. Формирование уведомлений сервера графического интерфейса

Настройки формирования уведомлений сервера определяются параметром `serverNotifications` конфигурационного файла. С помощью вложенного параметра `enable` происходит включение/выключение формирования данных уведомлений. В параметре `name` задается имя источника уведомлений, используемое для уведомлений сервера. В параметре `events` задается перечень формируемых уведомлений для данного источника.

Каждое формируемое уведомление определяется следующими параметрами:

- `enable` – включение/выключение формирования уведомления;

- severity – важность уведомления (указывается имя важности латинскими буквами);
- name – заголовок уведомления;
- type – имя внутреннего события сервера, для которого необходимо сформировать уведомление;
- text – шаблон текста уведомления.

В шаблоне текста уведомления можно использовать параметры события сервера, состав и назначение которых может изменяться в зависимости от типа события сервера. Доступные для использования параметры событий сервера:

- user – имя пользователя, инициировавшего событие сервера;
- login – логин пользователя;
- password – пароль пользователя;
- ip – IP-адрес;
- journal – имя журнала;
- file – имя файла при операциях с файлами;
- message – сообщение для события;
- accountName – имя учетной записи;
- newName – новое имя;
- iface – интерфейс.

Доступны следующие события сервера (имя события необходимо указать в параметре type для формирования уведомления):

- journal.open – открытие журнала;
- iptables.restore – применение конфигурации МЭ;
- iptables.save – выгрузка конфигурации МЭ в файл;
- iptables.write – сохранение конфигурации МЭ;
- iptables.clear – очистка конфигурации МЭ;
- iptables.clearhistory – очистка истории изменения конфигурации МЭ;
- fs.change – изменение файла конфигурации;
- fs.upload – загрузка файла конфигурации на сервер;

- fs.download – скачивание файла конфигурации;
- user.rename – изменение имени пользователя;
- user.data.change – изменение данных пользователя;
- user.password.change – изменение пароля пользователя;
- user.root.sshkey.add – добавление SSH-ключа;
- user.root.sshkey.delete – удаление SSH-ключа;
- power.reboot – перезагрузка МЭ;
- power.shutdown – выключение МЭ;
- power.config.change – изменение параметров выключения МЭ;
- datetime.change – изменение параметров даты и времени;
- datetime.zone.change – изменение часового пояса;
- config.change – изменение настроек МЭ;
- network.settings.change – изменение настроек сети;
- network.iface.add – добавление интерфейса;
- network.iface.update – изменение данных интерфейса;
- network.iface.remove – удаление интерфейса;
- network.ip.add – добавление IP-адреса;
- network.ip.remove – удаление IP-адреса;
- network.route.add – добавление маршрута;
- network.route.update – изменение маршрута;
- network.route.remove – удаление маршрута;
- network.connection.add – создание сетевого соединения;
- network.connection.update – изменение настроек сетевого соединения;
- network.connection.remove – удаление сетевого соединения;
- network.connection.status.change – изменение статуса соединения.

9.2.10.5. Настройка отправки уведомлений на электронную почту

Настройки отправки уведомлений определяются параметром transports конфигурационного файла. Параметры отправки уведомлений на электронную почту расположены в блоке, у которого вложенный параметр name равен email.

Доступны следующие параметры отправки уведомлений на электронную почту:

1) `format` – шаблон текста сообщения электронной почты;

2) `messageOptions` – параметры почтового сообщения, а именно:

- `type` – определяет формат сообщения: `html` или простой текст (значения `html` и `plain` соответственно);
- `subject` – тема сообщения;
- `from` – отправитель сообщения;
- `to` – получатель сообщения.

В параметрах `format` и `subject` можно использовать данные уведомления путем вставки в шаблон имен переменных в фигурных скобках {}, а именно:

- `name` – заголовок уведомления;
- `date` – дата уведомления;
- `severity` – важность уведомления;
- `text` – текст уведомления.

Если задать параметры `from` и `to`, тогда будут переопределены значения по умолчанию из настроек отправки электронной почты. Если параметры не задавать, то будут использоваться значения по умолчанию (п. 12.3).

9.2.11. Системные журналы в графическом интерфейсе

В разделе меню «Системы», подразделе «Системные журналы» (рис. 89), предусмотрены различные типы журналов системы. Описание журналов, приведено в таблице 41.

Таблица 41 – Системные журналы

Системный журнал	Описание
Веб-сервер	Журнал включает все события о веб-сервере МЭ ИВК КОЛЬЧУГА-К.
Системные сообщения (syslog)	Журнал системных сообщений, собранных с помощью syslog, включает общую информация обо всех событиях системы.
Аудит	Журнал включает все события, возникающие при работе службы аудита Linux – auditd.

Окончание таблицы 41

Системный журнал	Описание
Резервное копирование	Журнал включает все события, возникающие при работе утилиты резервного копирования rsync.
Безопасность	В журнале отображаются записи всех событий, связанных с безопасностью системы.
Настройка межсетевого экрана через веб-интерфейс	Журнал включает все события, связанные с операциями изменения правил МЭ через ГИ МЭ ИВК КОЛЬЧУГА-К.
Электронная почта	В этом журнале записываются время и статус принятых и отправленных сообщений.
Ядро	В журнале отображается информация о событиях, связанных с взаимодействием с ресурсами компьютера и системными сервисами. Сообщения этого журнала могут помочь при поиске неисправностей в работе оборудования.
Вход в веб-интерфейс	Журнал включает все события, связанные с операциями входа в ГИ МЭ ИВК КОЛЬЧУГА-К.
Системные сообщения (journald)	Журнал системных сообщений, собранных с помощью journald, включает записи журнала из всей системы: загрузочные сообщения, сообщения из ядра и различных приложений.
Проверка целостности файлов	В журнале отображаются события, связанные с изменением и нарушением целостности файлов системы.
Сетевые блокировки	В журнале отображаются события, которые возникают в процессе блокировки и работы сервиса fail2ban.

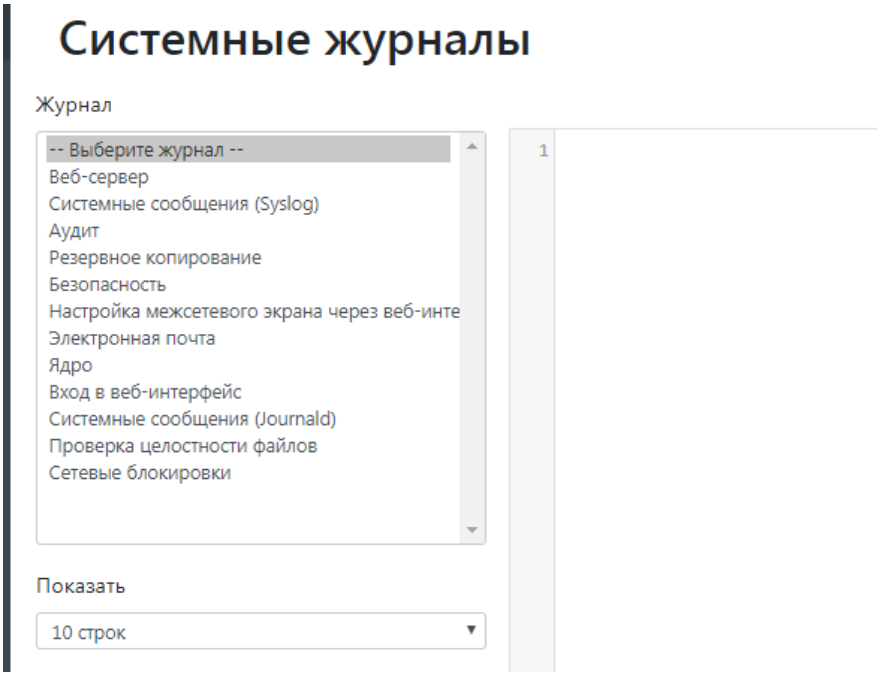


Рис. 89 – Системные журналы

С помощью выпадающего списка можно настраивать количество единовременно отображаемых строк вывода для выбранного журнала (рис. 90).

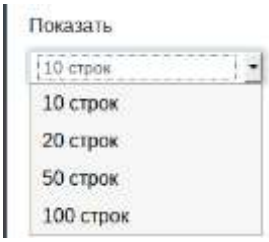


Рис. 90

Для подтверждения выбора и отображения нажать на кнопку «Обновить» (рис. 91).

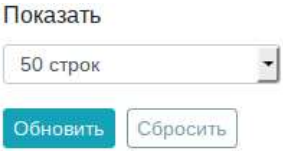


Рис. 91

В случае, если выбор неактуален, нажмите на кнопку «Сбросить». Количество отображаемых строк вернется к исходному.

Для перемещения между страницами журнала используйте кнопки стрелок

.

Пример отображения журнала «Системных сообщений (syslog)» приведен на рис. 92.

Пример отображения журнала «Аудит» приведен на рис. 93.

Рис. 92 – Журнал «Системные сообщения (syslog)»

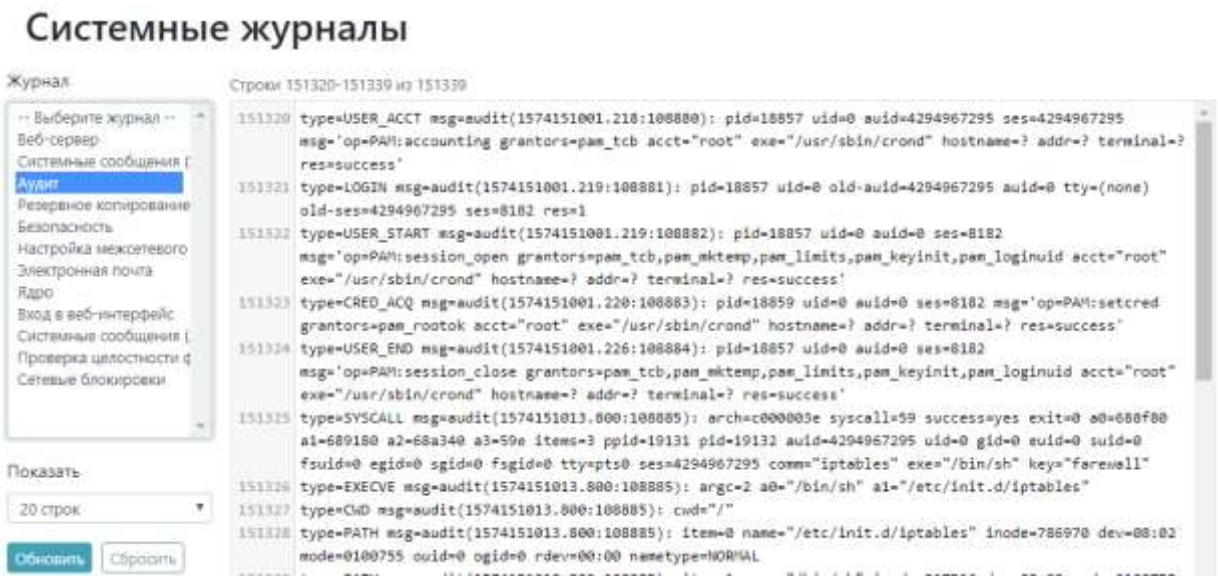


Рис. 93 – Журнал «Аудит»

9.2.12. Поддержка удаленного журналирования syslog

Для перенаправления всех сообщений на удаленный хост используется следующую запись в файле `/etc/syslog.conf`:

```
# Пример конфигурационного файла syslogd, который
# предписывает пересылать все сообщения на удалённую машину.
*.*                 @имя_хоста
```

Для перенаправления всех сообщений ядра (kernel) на удаленную машину, конфигурационный файл должен быть следующего вида:

```
# Пример конфигурационного файла syslogd.
# Все сообщения ядра пере-направляются на
# удалённую сетевую машину.
kern.*          @имя_хоста
```

Где `имя_хоста` – доменное имя либо IP-адрес.

Пример:

```
*.*                 @192.168.0.100
*.*                 @test.server.ru
kern.*               @192.168.0.100
kern.*               @test.server.ru
```

По умолчанию для отправки сообщений используется UDP порт 514.

Для отправки сообщений по другому порту, прописать в файле `/etc/services` порт в строке:

```
syslog          514/udp
```

и перезагрузить syslog сервер командой:

```
service syslogd restart
```

9.3. Настройка журналирования сервера графического интерфейса

Настройка анализаторов файлов журналов осуществляется через конфигурационный файл `/usr/share/web-kolchuga/conf/logger.conf`. Параметры задаются в формате JSON. При изменении конфигурационного файла требуется перезапуск сервера графического интерфейса – служба node.

Сервер ГИ МЭ ИВК КОЛЬЧУГА-К ведет два журнала:

- «Веб-сервер» – журналирование событий сервера ГИ – блок настроек, значение параметра `name` которого равно `default`;
- «Настройка межсетевого экрана через веб-интерфейс – журналирование событий, связанных с изменение настроек МЭ (блок настроек, значение параметра `name` которого равно `iptables`).

В настройках каждого журнала можно определить:

1) общий уровень журналирования;
2) уровни журналирования для вывода в консоль и файл (см. таблицу 8);
3) а также параметры журналирования в файл, а именно:
   - `filename` – путь в файлу журнала;
   - `maxsize` – максимальный размер файла (после превышения которого будет осуществляться ротация файла журнала);
   - `maxFiles` – количество файлов журнала для ротации.

9.4. Системные службы

МЭ ИВК КОЛЬЧУГА-К предлагает диспетчер управления службами, с помощью которого можно управлять перезапуском и остановкой служб.

Системные службы, будучи запущенными, принимают соединения, предоставляя, таким образом, различные сервисы.

Важно помнить, что изменения настроек какой-либо службы вступают в силу только после перезапуска службы. Этот модуль и позволяет осуществить такой перезапуск.

Выбор системных служб происходит из выпадающего списка (рис. 94).

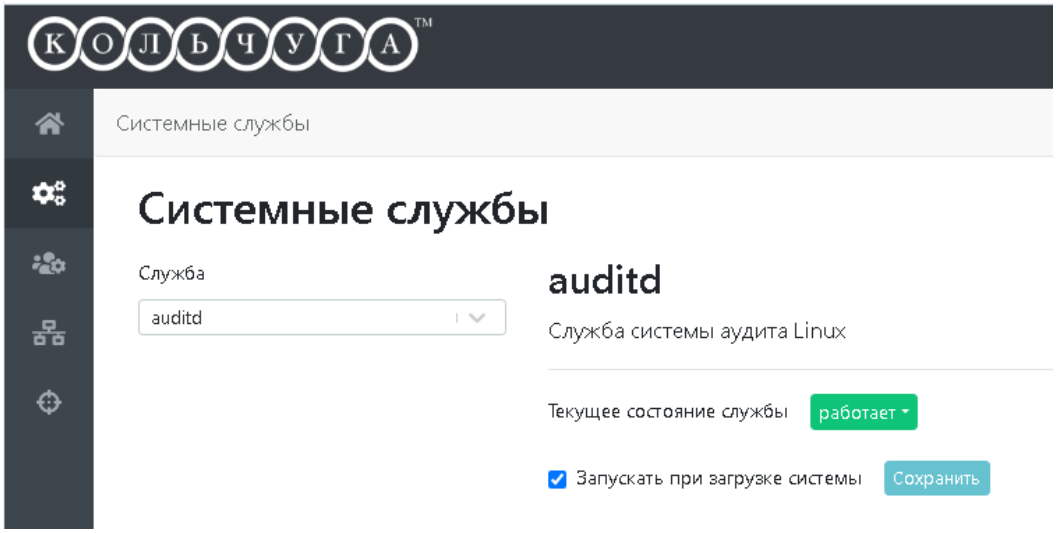


Рис. 94 – Выбор системных служб

На визуальном интерфейсе отображается название выбранной службы, управление событиями которой происходит, а также текущее состояние службы.

Выпадающий список «Текущее состояние службы» позволяет остановить либо перезапустить работающую службу или запустить остановленную (рис. 95). Изменение состояния службы действует только до перезагрузки. Если необходимо, чтобы служба запускалась автоматически при загрузке системы, отметьте соответствующий пункт «Запускать при загрузке системы».

Для подтверждения произведенных настроек нажать кнопку «Сохранить».

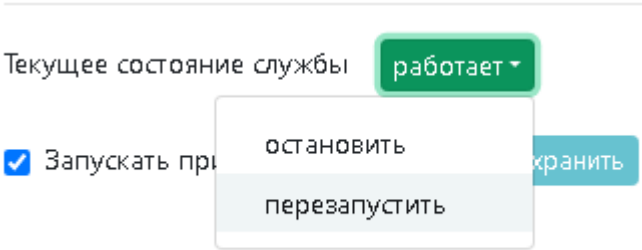


Рис. 95 – Изменение текущего состояния службы

Список доступных к управлению системных служб приведен в таблице 42.

Т а б л и ц а 42

Наименование служб	
auditd	bird
csync2	fail2ban
firehol	fireqos
ipset	iptables
netdata	network
node	osec-onboot
squid	sshd
suricata	syslogd
zabbix_agentd	

9.5. Выключение межсетевого экрана

В разделе меню «Система» перейти в панель «Выключение межсетевого экрана» для управления работоспособностью (рис. 96).

При необходимости выключить МЭ выбрать кнопку «Выключить сейчас».

При необходимости в перезагрузке нажать на кнопку «Перезагрузить».

Можно задать перезагрузку МЭ ежедневно в определенное время. Для этого нужно поставить флаг «Перезагружать межсетевой экран каждый день» и задать время. Нажать кнопку «Сохранить» для подтверждения настроек (рис. 96).

Для отмены примененных настроек нажать кнопку «Сбросить».

Для перезагрузки/выключения МЭ с использованием консольного интерфейса (удаленного консольного интерфейса или локального интерфейса управления) необходимо выполнить команды:

- для перезагрузки: `reboot`
- для выключения: `shutdown -P -h now`

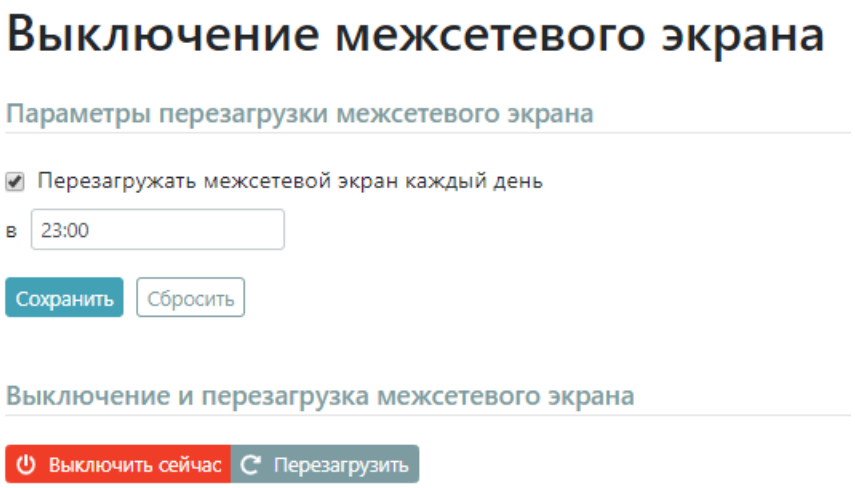


Рис. 96

9.6. Проверка целостности

9.6.1. Контрольное суммирование исполняемых файлов ПО МЭ ИВК КОЛЬЧУГА-К

МЭ ИВК КОЛЬЧУГА-К предоставляет возможность осуществления интегрального контрольного суммирования исполняемых файлов с помощью встроенного скрипта gensum. Метод вычисления хэш-сумм объектов контроля осуществляется в соответствии с алгоритмом ГОСТ Р 34.11–2012 «Информационная технология. Криптографическая защита информации. Функция хэширования».

Расчет интегральной КС исполняемых файлов МЭ ИВК КОЛЬЧУГА-К выполняется через графический интерфейс см. п. 4.6 открыть пункт «О межсетевом экране» (см. рис. 11) и нажать кнопку «Расчет контрольной суммы».

Примечание. По окончанию расчета может понадобиться обновить страницу.

Вычисленная интегральная КС сравнивается с указанной в документе «Формуляр. ЛКНВ.466217.002 ФО», входящем в комплект поставки, или при установке обновления безопасности от предприятия-изготовителя с КС, указанной в сопроводительном информационном сообщении.

9.6.2. Программный комплекс проверки целостности системы Osec

Osec является легковесным программным комплексом проверки целостности системы, который используется, чтобы обнаружить различия между двумя ее состояниями. Osec также предоставляет возможность выполнить проверку системы на наличие опасных файлов, например, с установленными битами прав смены идентификаторов пользователя (suid), группы (sgid) и с общедоступной записью.

9.6.2.1. Проверка контроля целостности в ГИ МЭ ИВК КОЛЬЧУГА-К

Проверка целостности системы заключается в поиске различий между текущим состоянием входящих в ее состав файлов и их прежним состоянием.

Раздел предназначен для выполнения следующих операций:

- выбор набора объектов файловой системы, подлежащих проверке;
- настройка периодической проверки целостности системы;
- настройка режима отчетности о производимых проверках;
- управление запуском проверки;
- получение информации о результатах произведенных проверок.

Своевременно выполняемая проверка позволит вовремя выявить опасные изменения, произошедшие в системе.

ГИ раздела проверка целостности состоит из трех вкладок (рис. 97):

- «Файлы журнала» (п. 9.6.2.2);
- «Настройка и запуск» (п. 9.6.2.3);
- «Директории и файлы» (п. 9.6.2.4).

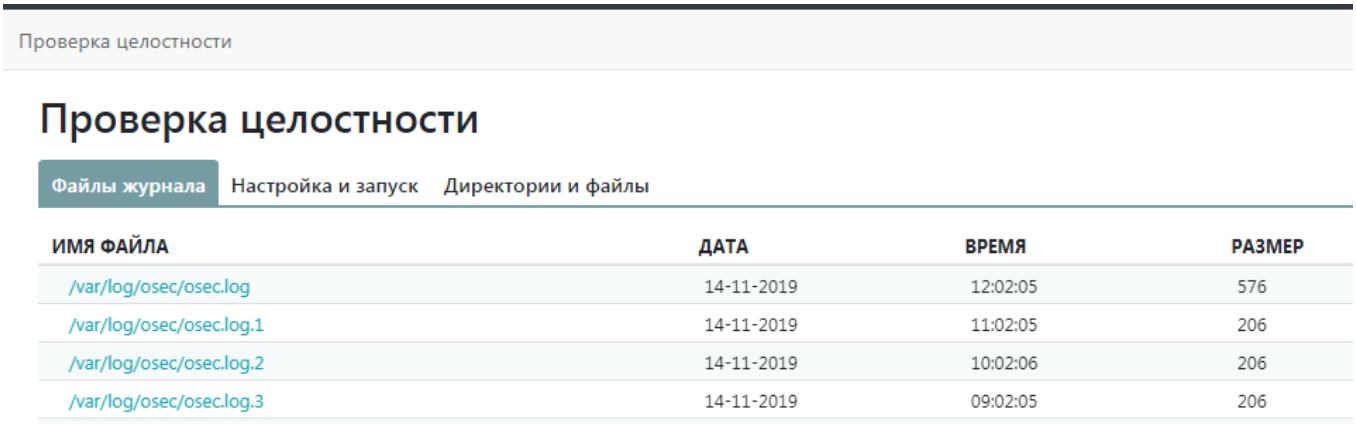


Рис. 97 – Вкладка «Файлы журнала»

9.6.2.2. Файлы журнала osec

Файлы журнала osec содержат список файлов отчетов по контролю целостности, с указанием имени файла, размера, даты и времени его создания.

При нажатии на имени файла указателем мыши на экране появится окно просмотра его содержимого (рис. 98).

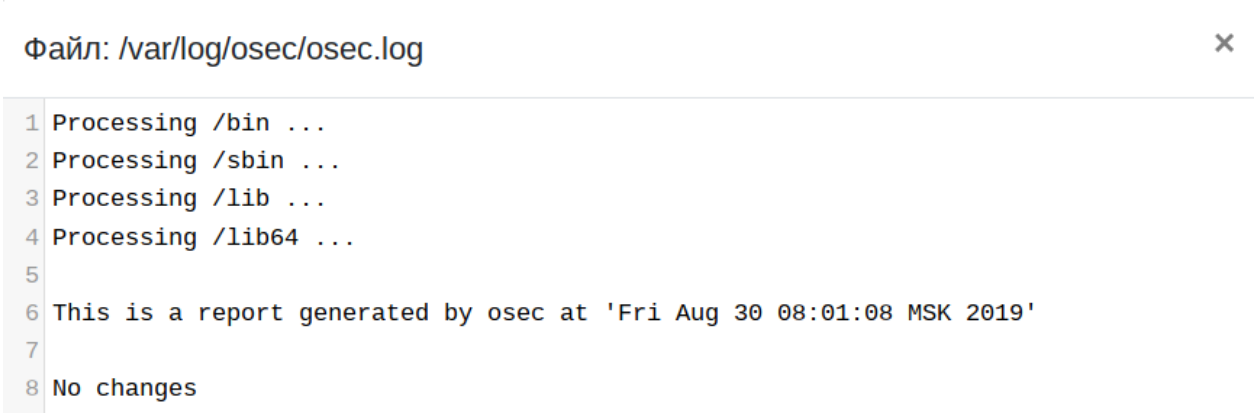


Рис. 98 – Содержимое файла отчета по контролю целостности

Для просмотра отчетов по контролю целостности с использованием консольного интерфейса необходимо выполнить команду:

```
cat /var/log/osec/osec.log
```

Содержимое файла будет идентично содержимому, приведенному на рис. 98.

9.6.2.3. Настройка и запуск osec

В подразделе «Настройка запуска» в выпадающем списке можно настроить периодичность запуска проверки контроля целостности:

- ежечасно;
- ежедневно;
- еженедельно;
- ежемесячно;
- отключить.

В подразделе «Настройка отчетности» если необходимо, можно установить флаги для записи отчетов в журнал и отправки отчета, а также управлять количеством хранимых файлов журнала (рис. 99).

В подразделе «Управление запуском» изменяется статус выполнения: «Не исполняется» и «Запустить».

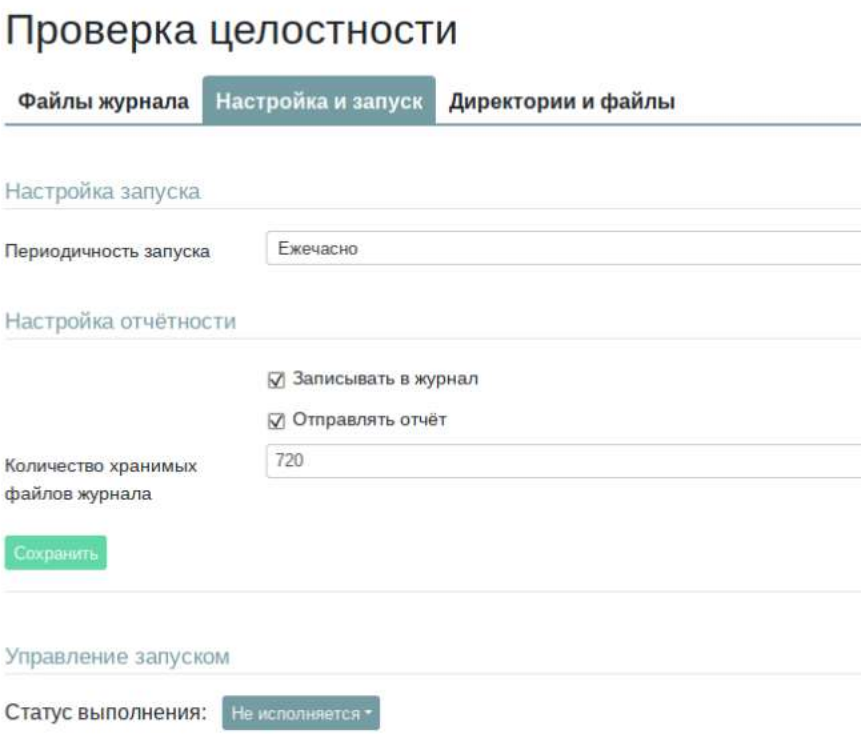


Рис. 99 – Вкладка «Настройка и запуск»

9.6.2.4. Отслеживаемые директории и файлы

Список объектов контроля целостности преднастроен для МЭ.

Данный список расположен в файле `/etc/osec/dirs.conf` и не подлежит изменению.

Просмотр сведений об объектах контроля целостности возможен через ГИ. На вкладке директории и файлы раздела «Проверка целостности» осуществляется просмотр списка объектов, анализируемых системой контроля целостности.

В списке объектов отображаются контролируемые директории с указанием пути и описанием (рис. 100).

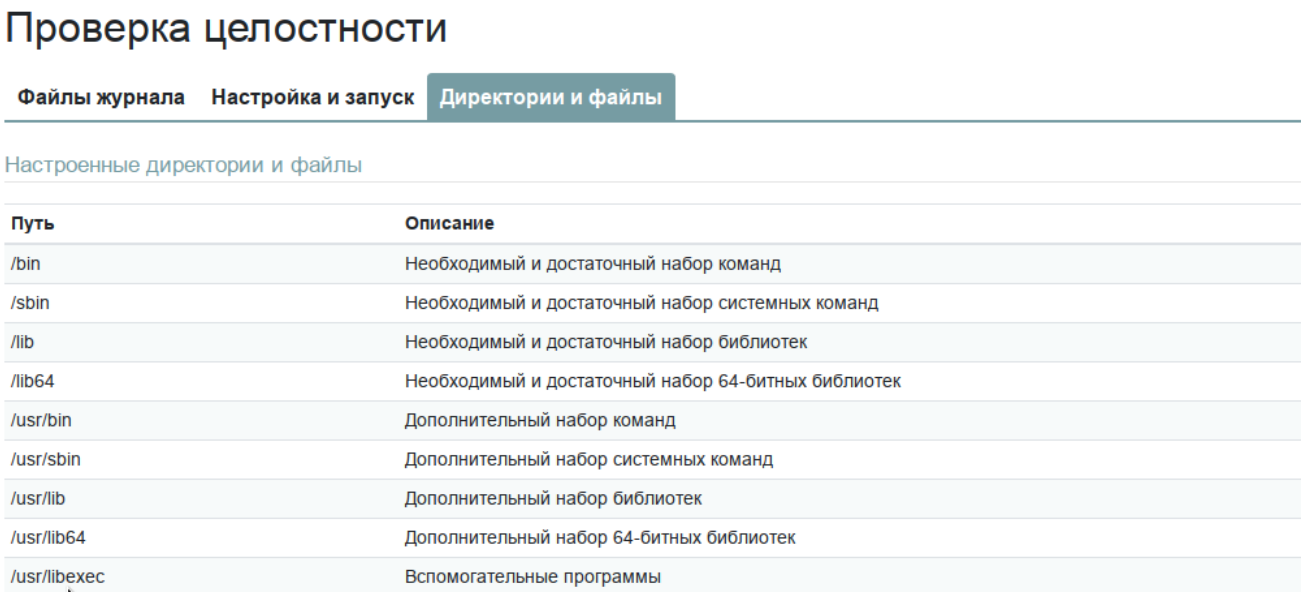


Рис. 100

Проверка целостности

Файлы журнала Настройка и запуск Директории и файлы

Настроенные директории и файлы

Путь	Описание
/bin	Необходимый и достаточный набор команд
/sbin	Необходимый и достаточный набор системных команд
/lib	Необходимый и достаточный набор библиотек
/lib64	Необходимый и достаточный набор 64-битных библиотек
/usr/bin	Дополнительный набор команд
/usr/sbin	Дополнительный набор системных команд
/usr/lib	Дополнительный набор библиотек
/usr/lib64	Дополнительный набор 64-битных библиотек
/usr/libexec	Вспомогательные программы

Рис. 100 – Вкладка «Директории и файлы»

9.7. Тестирование

Тестирование происходит во время загрузки МЭ ИВК КОЛЬЧУГА-К, которая происходит в три этапа.

На первом этапе БСВВ из MBR (первые 512 байт диска, выбранного для загрузки) загружает первичный загрузчик – First Boot Loader (FSB). FSB находит вторичный загрузчик – Second Stage Boot Loader (SSB), используя таблицу разделов, просматривая ее, обнаруживает активный раздел, после обнаружения этого раздела – загружает SSB в оперативную память и запускает его. Для корректной загрузки, активный раздел должен содержать каталог `/boot`, который должен находиться в начале диска и содержать SSB. В целом, SSB – это программа, которая выводит список вариантов загрузки (меню выбора загрузки операционной системы).

На втором этапе происходит подготовка системы для запуска служб демонов. При подготовке, загрузчик загружает в память образ ядра из каталога `/boot`.

На третьем этапе загрузки – после запуска, процесс init, согласно конфигурации в файле `/etc/inittab` (а точнее строке, начинающийся на `si::sysinit:/etc/.....`) первым делом выполняет скрипт `/etc/rc.d/rc.sysinit`, которые выполняют базовое конфигурирование системы (загрузка модулей, проверка корневой ФС и монтирование на чтение/запись, установка имени хоста, времени, монтирование оставшихся разделов, запуск сети, монтирование сетевых ФС и др.), а так же данный скрипт с помощью утилиты initlog направляет сообщения о загрузке в `/var/log/messages` (проверку правильности старта можно проверить в данном файле). При наличии сбоев при проверке ФС соответствующая запись будет в файле журнала аудита.

Для проверки корректности инициализации системы необходимо в файле журнала найти информацию по ключевому слову `rc.sysinit`.

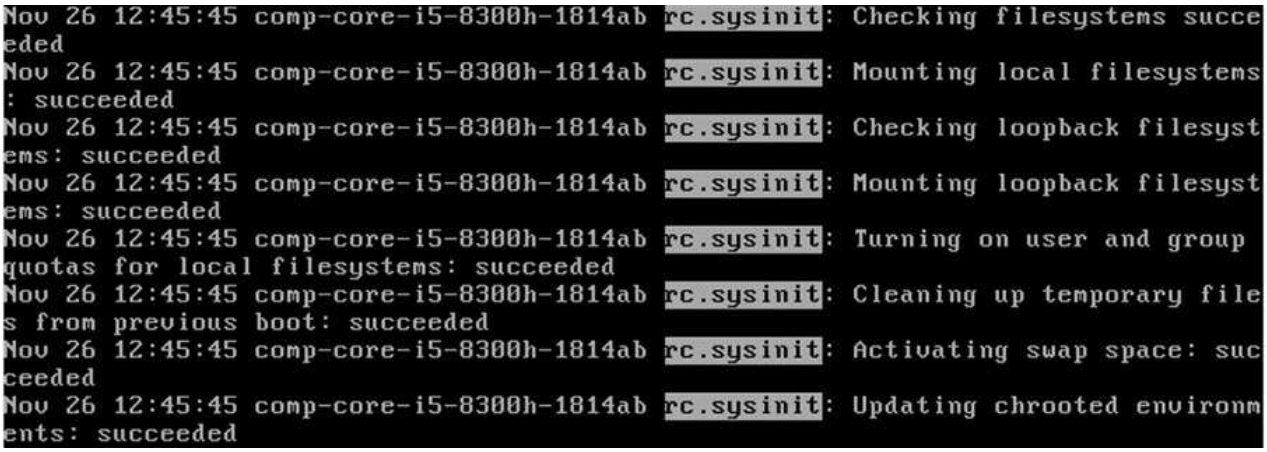


Рис. 101

На данном этапе, нет уровня выполнения. Далее, запускается скрипт инициализации `/etc/rc.d/rc`, которому передается уровень запуска в виде параметра от 0 до 6 (в соответствии с настройками из файла `/etc/inittab`, в котором указан уровень загрузки (выполнения) по умолчанию и каталог `/etc/rc.d/rc*.d`, в котором расположены скрипты запуска демонов/служб для соответствующего уровня запуска), запускает скрипты из каталога, соответствующего текущему уровню запуска.

Процесс init согласно уровню загрузки просматривает каталог `/etc/rc.d/rc0.d/` (в данном примере, цифра 0 (ноль) в имени `rc0.d` соответствует уровню загрузки – нулевому), в котором содержатся ссылки на скрипты запуска системных служб, которые, в свою очередь, расположены в `/etc/rc.d/init.d/`.

Уровни выполнения бывают следующие:

- 0: полная остановка машины;
- 1: single-user (однопользовательский) режим (используется в случае серьезных проблем или для восстановления системы);
- 2: multi-user (многопользовательский) режим, без поддержки сети;
- 3: multi-user (многопользовательский) режим с поддержкой сети (используется преимущественно на серверных системах);
- 4: неиспользуемый;

- 5: multi-user (многопользовательский) режим с поддержкой сети и графический интерфейс для входа в систему (login);
- 6: перезагрузка.

9.7.1. Самотестирование

Для проверки запуска сервисов и правильности выполнения в консольном интерфейсе от администратора выполните команду запуска сервиса check:

```
# service check
```

Процесс проверки приведен на рис. 102.

Для просмотра лога проверки воспользуйтесь командой (рис. 103):

```
# cat /var/log/check/check.log
```

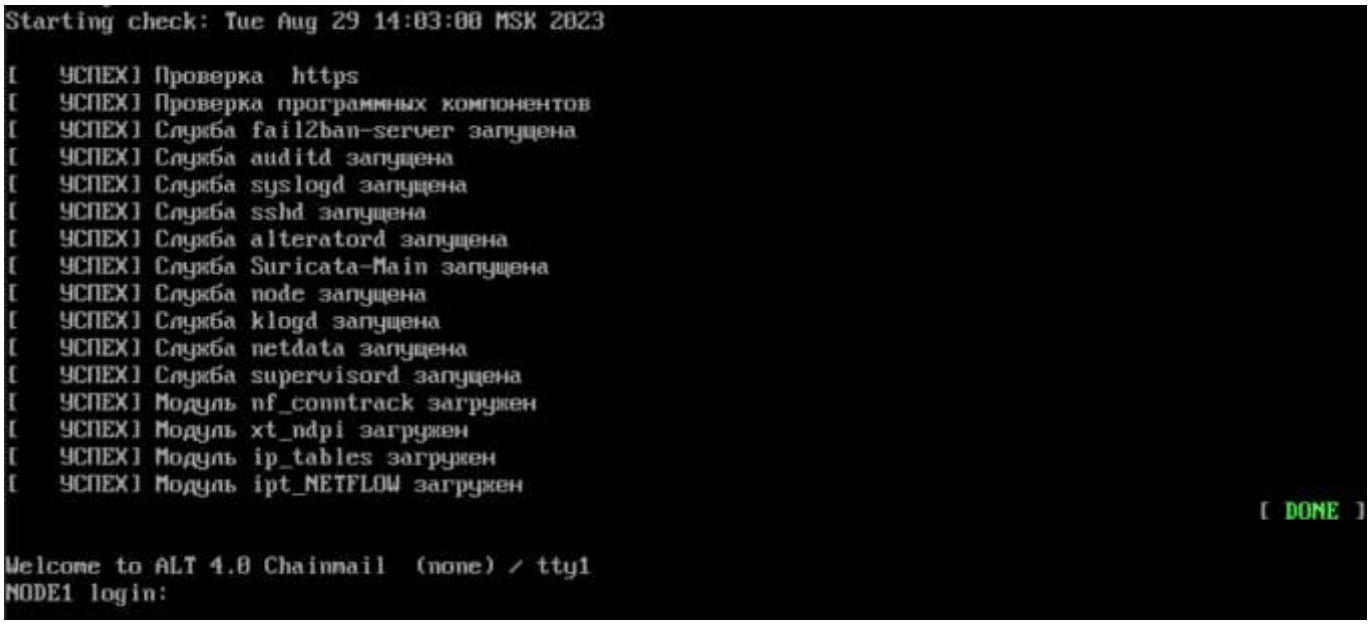


Рис. 102

```
[NODE1 admin] cat /var/log/check/check.log
Tue Aug 29 14:03:00 MSK 2023

[   УСПЕХ] Проверка  https
[   УСПЕХ] Проверка программных компонентов
[   УСПЕХ] Служба fail2ban-server запущена
[   УСПЕХ] Служба auditd запущена
[   УСПЕХ] Служба syslogd запущена
[   УСПЕХ] Служба sshd запущена
[   УСПЕХ] Служба alteratord запущена
[   УСПЕХ] Служба Suricata-Main запущена
[   УСПЕХ] Служба node запущена
[   УСПЕХ] Служба klogd запущена
[   УСПЕХ] Служба netdata запущена
[   УСПЕХ] Служба supervisord запущена
[   УСПЕХ] Модуль nf_conntrack загружен
[   УСПЕХ] Модуль xt_ndpi загружен
[   УСПЕХ] Модуль ip_tables загружен
[   УСПЕХ] Модуль ipt_NETFLOW загружен
[NODE1 admin]
```

Рис. 103 – Лог выполнения check

9.8. Обеспечение бесперебойного функционирования и восстановление

9.8.1. Резервное копирование

9.8.1.1. rsync

Резервирование происходит с помощью утилиты rsync ежечасно, с использованием утилиты cron. Управление работой данного механизма осуществляется посредством программы rsnapshot.

Файл настроек `/etc/rsnapshot/rsnapshot.conf`:

```
## Директория где будут находиться снимки
snapshot_root  /.snapshots/
## Интервалы создания снимков.
# Указывает тип снимка и сколько копий хранить
retain    hourly    6
retain    daily     7
retain    weekly    4
retain    monthly   12
# будет храниться 6 ежечасных, 7 ежедневных и 4 еженедельных
## Настройка данных для архивации
```

```
    # формат: цель для архивации | куда сохранять снимок (внутри
папки snapshot_root)
    # !!!ВАЖНО!!! указание завершающих слешей "/" в названии папок
обязательно
    # LOCALHOST
    #backup   /home/         localhost/
    backup    /etc/          localhost/
    #backup   /usr/local/    localhost/
    #backup   /var/log/rsnapshot       localhost/
    #backup   /etc/passwd    localhost/
    #backup   /home/foo/My Documents/       localhost/
    #backup   /foo/bar/ localhost/     one_fs=1,     rsync_short_args=-
urltvpog
    #backup_script /usr/local/bin/backup_pgsql.sh
    localhost/postgres/
    # You must set linux_lvm_* parameters below before using lvm
snapshots
    #backup   lvm://vg0/xen-home/ lvm-vg0/xen-home/
    ## Задание файлов исключений
    exclude        *.tmp
    exclude        ~*
    exclude        .git/
```

Указанные в файле `rsnapshot.conf` файлы и каталоги будут резервироваться в каталоге `/.snapshots/`.

9.8.1.2. Настройка резервного копирования в графическом интерфейсе

Настройка автоматического резервного копирования осуществляется с помощью файла конфигурации `/etc/rsnapshot/rsnapshot.conf` (рис. 104).

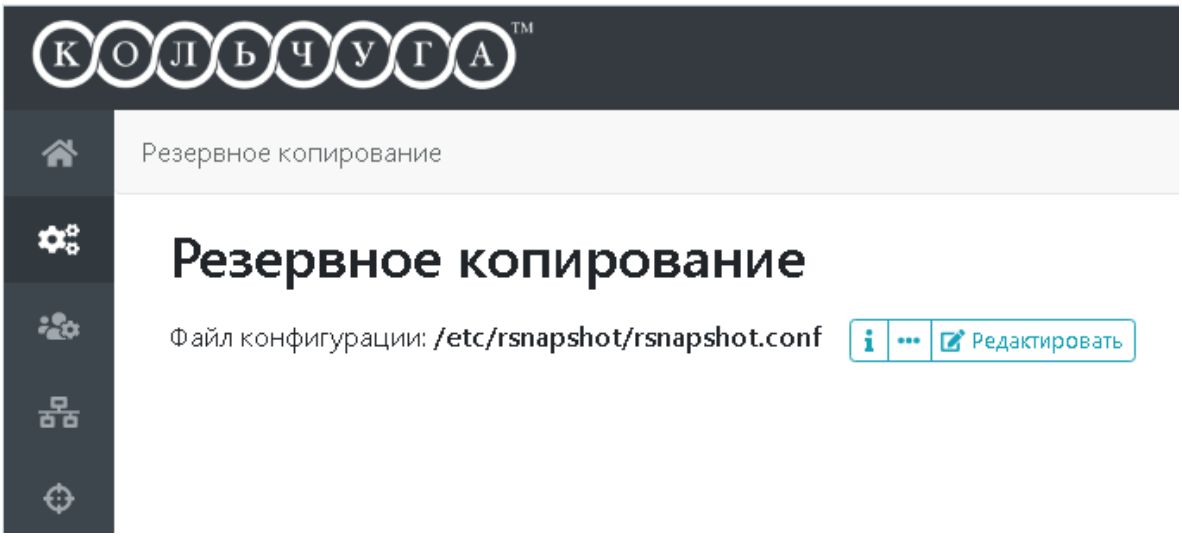
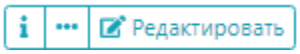


Рис. 104 – Резервное копирование

Описание действия кнопок  приведено в п. 5.4.

При редактировании файла конфигурации (рис. 105) можно указать определенные каталоги для включения в процедуру резервирования, место и период хранения резервных копий.

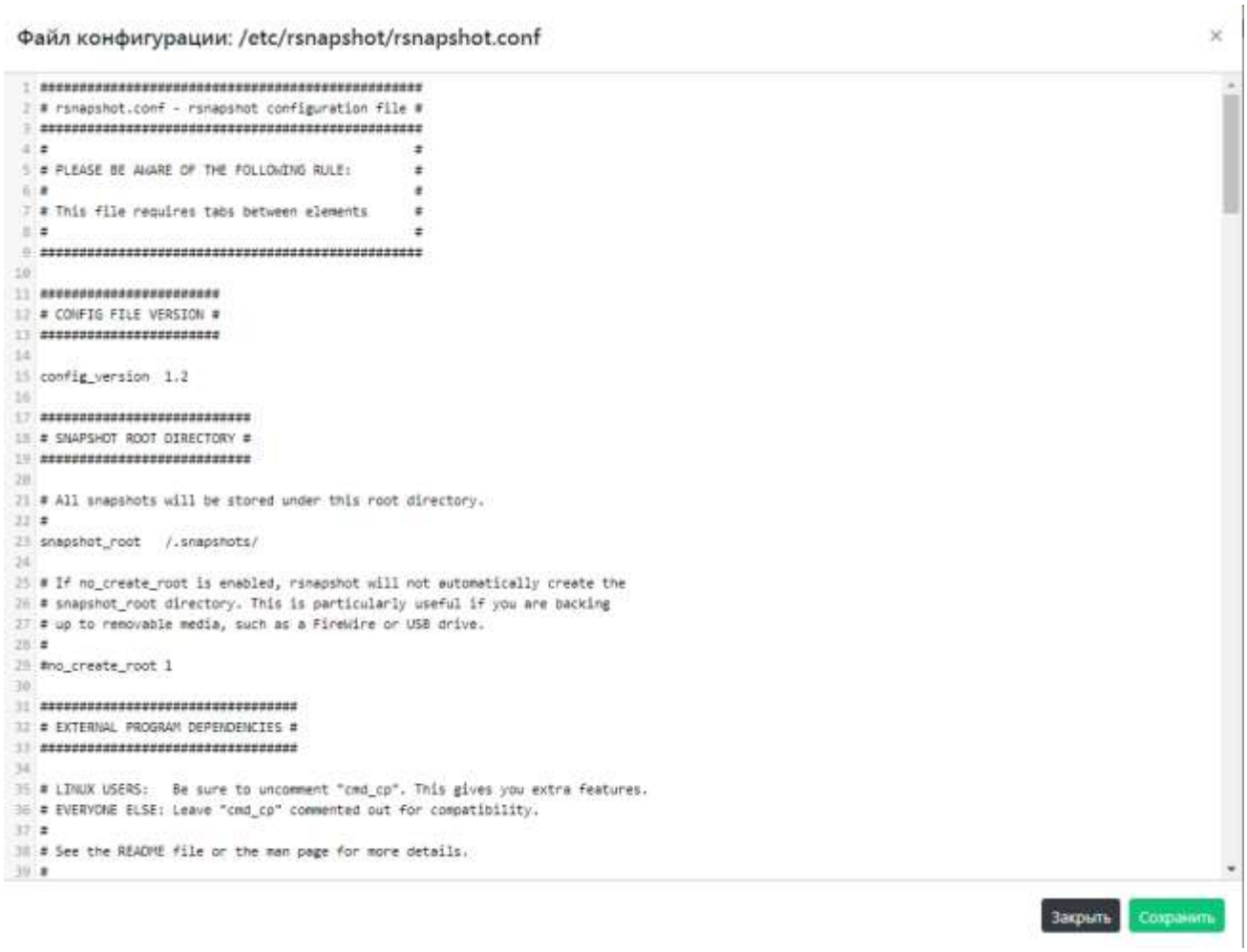


Рис. 105 – Файл конфигурации резервного копирования

9.8.2. Восстановление

Для восстановления МЭ ИВК КОЛЬЧУГА-К необходимо:

1) запустить МЭ ИВК КОЛЬЧУГА-К – нажать кнопку включения;

2) в появившемся меню загрузки выбрать пункт «Восстановление системы» (рис. 106);

Рис. 106 – Запуск восстановления системы

3) введите при запросе заводской логин и пароль администратора: **admin** пароль **chainmail1234** (рис. 107);

4) процесс восстановления автоматически запустится (рис. 108 – 109);

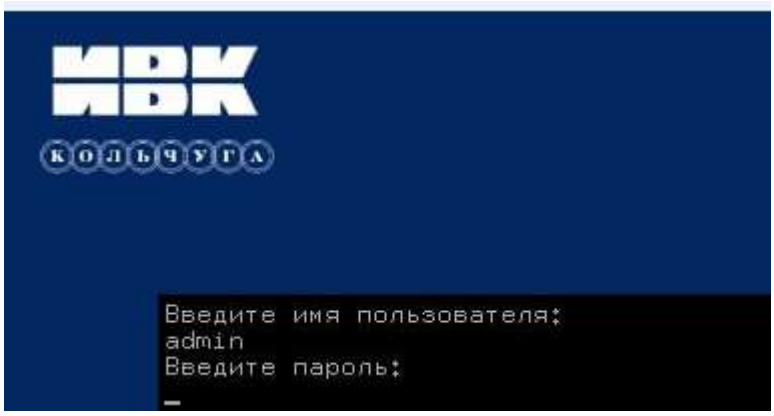


Рис. 107

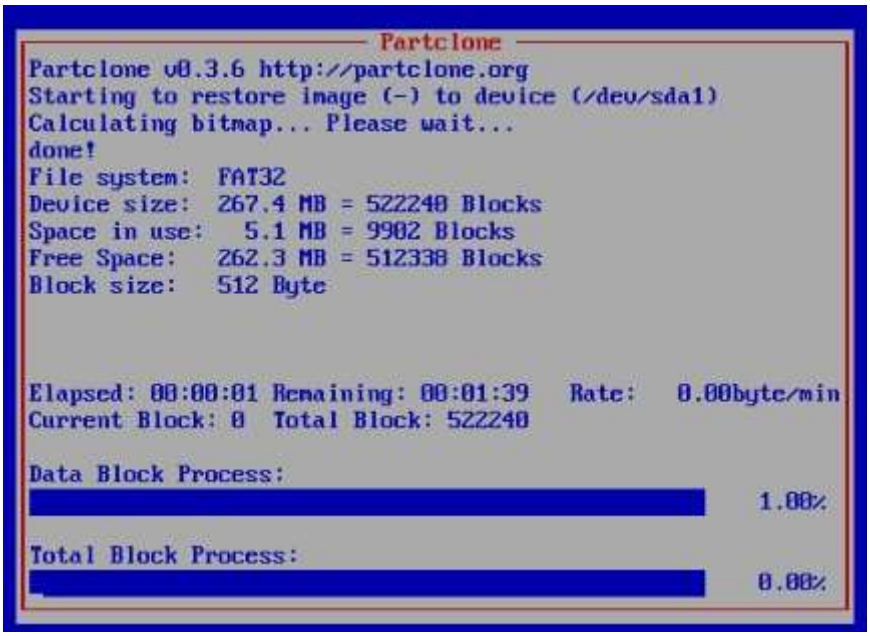


Рис. 108 – Процесс восстановления (1)

Рис. 109 – Процесс восстановления (2)

5) после успешного окончания восстановления появится сообщение `System restored!` (рис. 110), МЭ автоматически перезагрузится;
6) далее произойдет загрузка МЭ в заводской конфигурации.

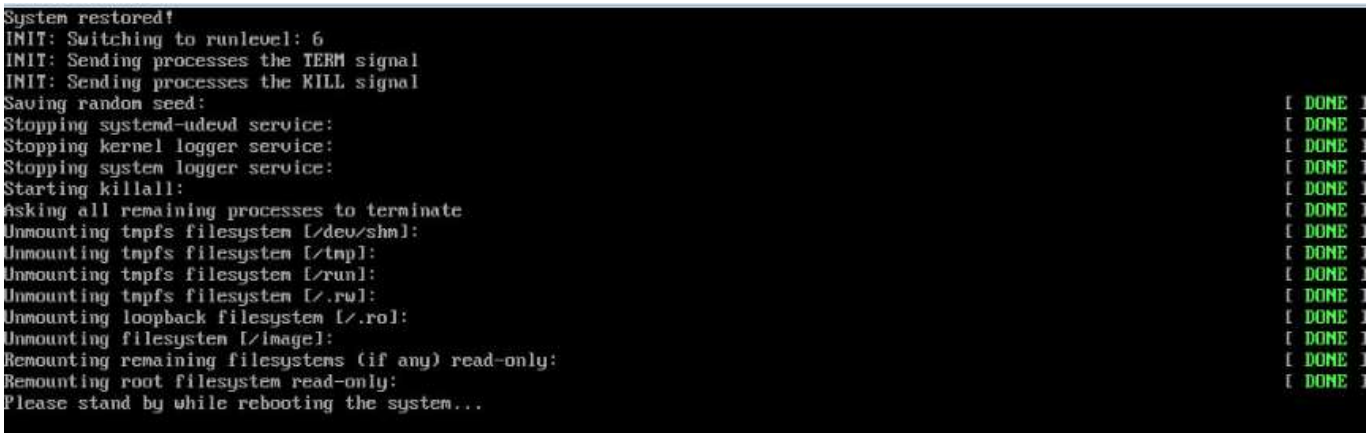


Рис. 110 – Успешное окончание восстановления

9.8.3. Кластеризация csync2

Для обеспечения отказоустойчивости и оптимизации нагрузки в МЭ ИВК КОЛЬЧУГА-К используется утилита csync2.

Для организации кластеризации, например, между двумя МЭ (NODE1 и NODE2) необходимо связать их между собой. csync2 обменивается файлами посредством шифрованного SSL-соединения, поэтому нужно создать единый csync2-сертификат, который позволит участникам группы «доверять» друг другу.

На NODE1 внести изменения в файл `/etc/hosts`: добавить информацию обо всех участвующих IP-адресах и hostname устройств (рис. 111).

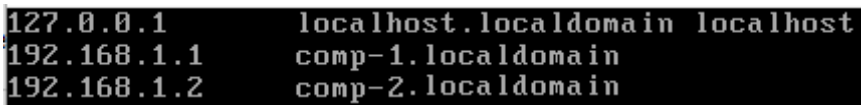


Рис. 111

Перезапустить NODE1.

На NODE1 сгенерировать сертификат вручную:

```
openssl genrsa -out /etc/csync2/csync2_ssl_key.pem 1024
openssl req -new -key /etc/csync2/csync2_ssl_key.pem -out /etc/csync2/csync2_ssl_cert.csr
openssl x509 -req -days 600 -in /etc/csync2/csync2_ssl_cert.csr -signkey /etc/csync2/csync2_ssl_key.pem -out /etc/csync2/csync2_ssl_cert.pem
```

Запустить генерацию ключа csync2:

```
csync2 -k /etc/csync2/csync2.cluster.key
```

Примечание. Выполнение этой команды может занимать длительное время.

Основные настройки производятся в конфигурационном файле утилиты `/etc/csync2/csync2.cfg`, зададим логические группы и файлы синхронизации:

```
group test {
# список хостов или IP, которые входят в эту логическую группу
host comp-1.localdomain comp-2.localdomain;
# ключ авторизации
key /etc/csync2/csync2.cluster.key;
# какие файлы \ папки необходимо синхронизировать?
include /etc/hosts; # файл hosts
include /etc/csync2.cfg; # конфигурационный файл csync2.cfg
include /mnt/test ; # любой файл \ папка для синхронизации
}
```

Запустить сервис csync2 на NODE1 командой:

```
service csync2 start
```

На NODE1 провести синхронизацию – выполнить команду:

```
csync2 -x
```

После первичной авторизации все хосты синхронизируются и указанные файлы станут одинаковыми на всех машинах логической группы.

Возможные проблемы:

1) отсутствие доступа на запись `/etc/hosts` – проверьте запуск утилиты на других машинах: `service csync2 start`;
2) ошибки с SSL – проверьте, скопирован ли на все хосты файл `/etc/csync2/csync2.cluster.key` и правильно ли указан в конфигурационном файле.

10. СЕТЬ

Состав раздела «Сеть» ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 112):

- Ethernet-интерфейсы (п. 10.1);
- PPTP-соединения (п. 10.2);
- L2TP-соединения (п. 10.3);
- маршрутизация (п. 10.4);
- межсетевой экран (раздел 7);
- прокси-сервер (п. 10.5);
- автонастройка межсетевого экрана (п. 10.6);
- ограничение трафика (п. 10.7);
- сетевой трафик (п. 10.8);
- статистика прокси-сервера (п. 10.9);
- демон маршрутизации (bird) (п. 10.10);
- агент наблюдения (п. 10.11).

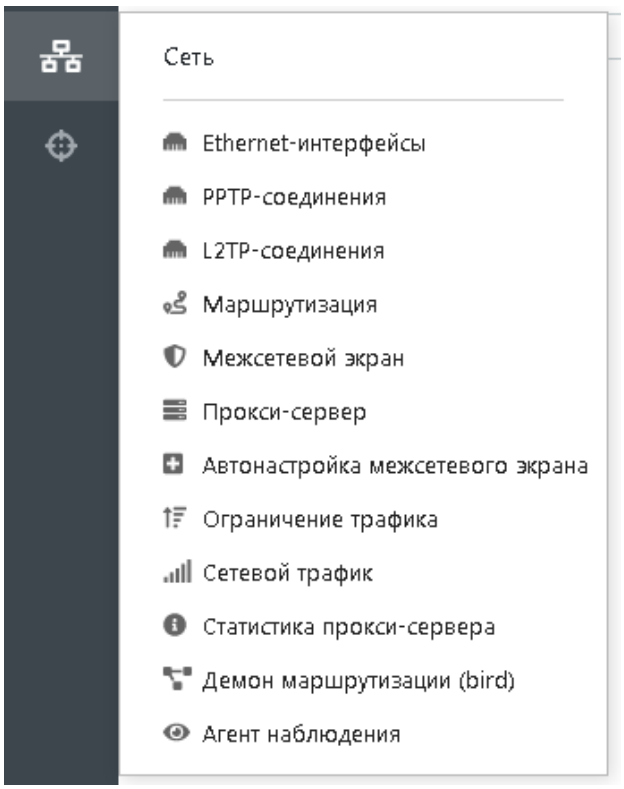


Рис. 112 – Меню раздела «Сеть»

10.1. Ethernet-интерфейсы

Локальная сеть (англ. Ethernet) – семейство технологий пакетной передачи данных между устройствами для компьютерных и промышленных сетей. Технология Ethernet позволяет использовать различные среды передачи.

VLAN – виртуальная локальная сеть, представляет собой группу хостов с общим набором требований, которые взаимодействуют так, как если бы они были подключены к широковещательному домену, независимо от их физического местонахождения. VLAN имеет те же свойства, что и физическая локальная сеть, но позволяет конечным членам группироваться вместе, даже если они не находятся в одной физической сети.

В подразделе «Ethernet-интерфейсы» (рис. 113) можно просмотреть информацию обо всех имеющихся интерфейсах и их настройках, также создать VLAN, создать объединение и создать сетевой мост.

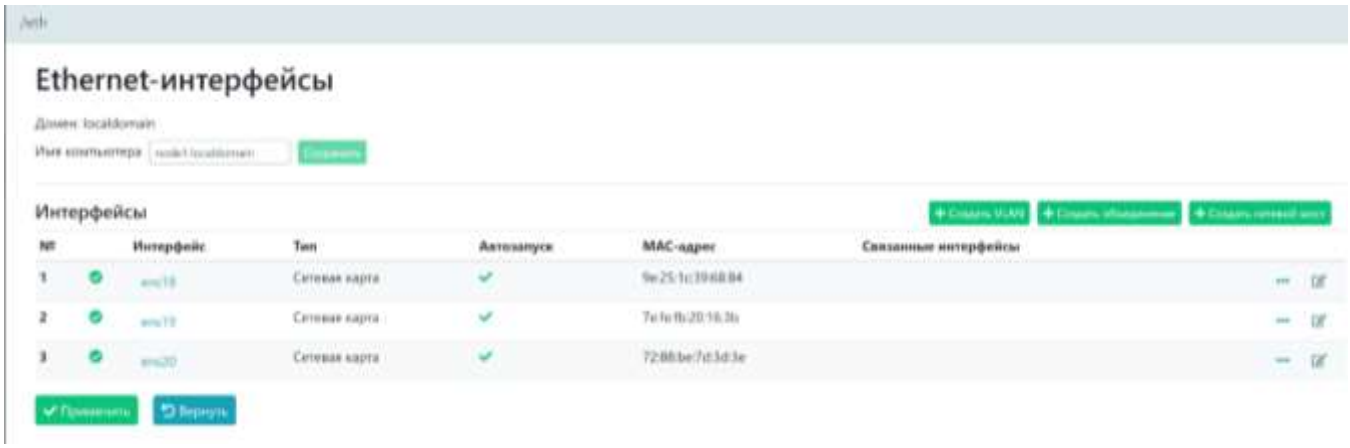


Рис. 113 – Подраздел Ethernet-интерфейсы

Виды статусов:

- – активный интефейс;
- – отключенный интерфейс.

Для активации выставленной конфигурации интерфейсов нажать кнопку Применить, для сброса всех изменений нажать кнопку Вернуть.

10.1.1. Создание VLAN

Для того чтобы создать VLAN необходимо нажать на кнопку в верхней части панели 

, на экране появится окно (рис. 114).

Рис. 114 – Окно добавление сети VLAN

Выберите интерфейс из выпадающего списка (рис. 115).

Рис. 115 – Выбор интерфейса

Далее обязательно заполнить VID (VLAN ID) – 12-битный идентификатор VLAN, использующийся в стандарте 802.1Q. Диапазон возможных значений VID от 0 до 4095.

После выбора параметров нажать на кнопку «Добавить» (рис. 116).

Рис. 116 – VID

В общем списке интерфейсах появится строка с добавленным VLAN (рис. 117).

Рис. 117

Также добавленный VLAN будет отображаться на вкладке «Конфигурация VLAN» для соответствующего интерфейса (рис. 118).

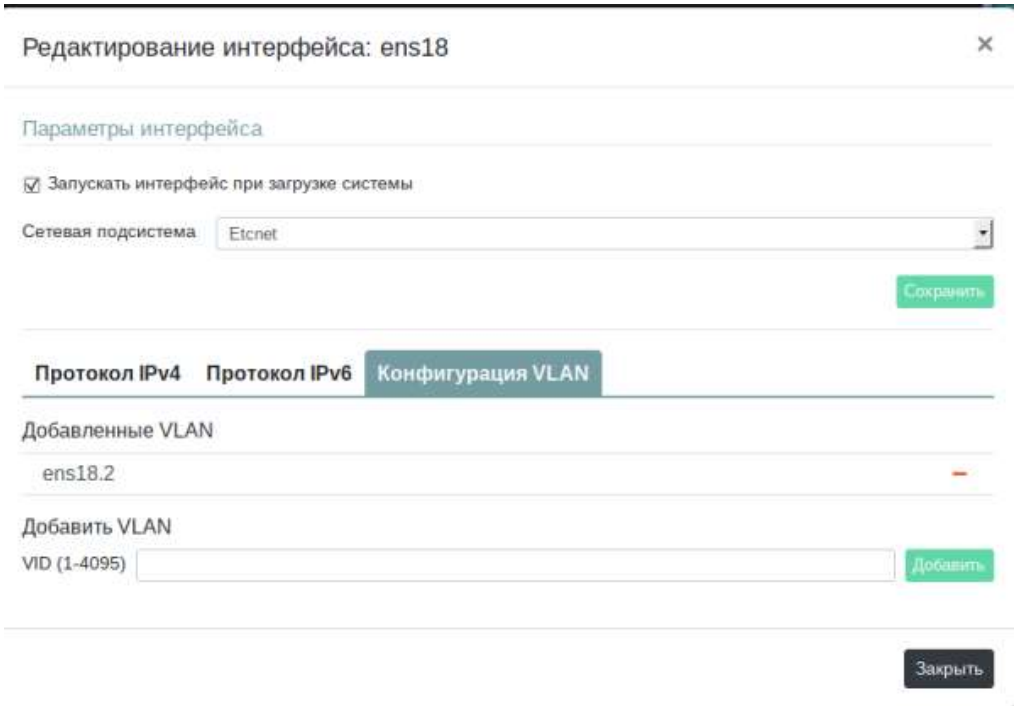


Рис. 118

10.1.2. Создание объединения

Создание объединения обеспечивает создание объединения выбранных интерфейсов (bond), позволяет создавать bond-интерфейсы с различными режимами работы. Наиболее часто используемые режимы:

- объединение пропускной способности;
- резервирование канала;
- балансировка трафика.

Для bond-интерфейсов не работает автоматическое получение IP-адресов по протоколу DHCP – возможна только ручная настройка параметров IP.

Для того чтобы создать объединение, нажмите на кнопку 

. В появившемся окне необходимо ввести имя будущего интерфейса (рис. 119).

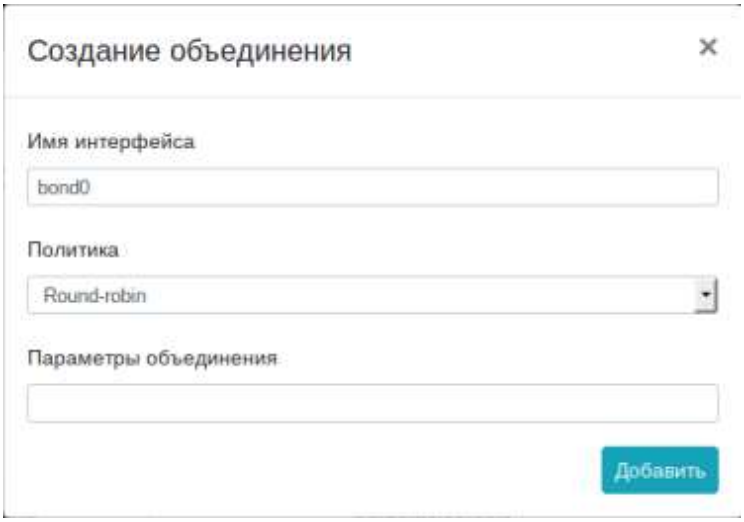


Рис. 119 – Окно добавления интерфейса

Выбрать политику – режим работы объединения из выпадающего списка (рис. 120), описание возможных типов приведены в таблице 43.

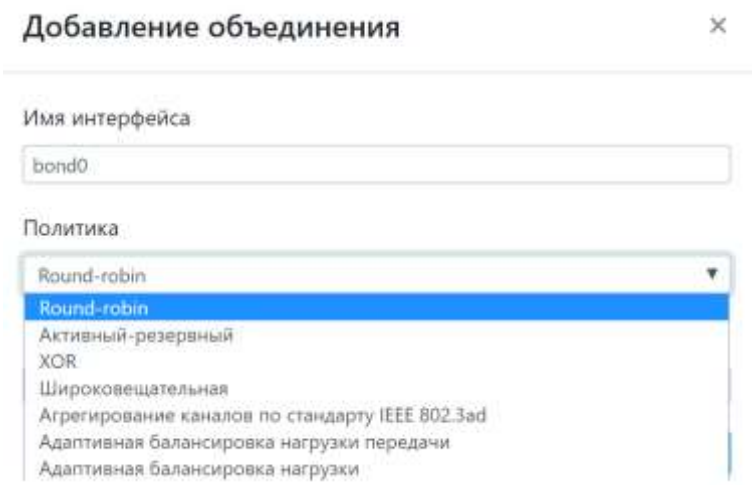


Рис. 120 – Выбор политики объединения

Таблица 43

Политика	Описание
Round-robin	Пакеты отправляются последовательно, начиная с первого доступного интерфейса и заканчивая последним. Эта политика применяется для балансировки нагрузки и отказоустойчивости.
Активный-резервный	Только один сетевой интерфейс из объединенных будет активным. Другой интерфейс может стать активным только в том случае, когда упадет текущий активный интерфейс. При такой политике MAC-адрес интерфейса объединения виден снаружи только через один сетевой порт, во избежание появления проблем с коммутатором. Эта политика применяется для отказоустойчивости.
XOR	Передача распределяется между сетевыми картами используя формулу: `[(«MAC-адрес источника» XOR «MAC-адрес назначения») по модулю «число интерфейсов»]`. Получается одна и та же сетевая карта передает пакеты одним и тем же получателям. Политика XOR применяется для балансировки нагрузки и отказоустойчивости.
Широковещательная	Передает все на все сетевые интерфейсы. Эта политика применяется для отказоустойчивости.
Агрегирование каналов по стандарту IEEE 802.3ad	Создаются агрегированные группы сетевых карт с одинаковой скоростью и дуплексом. При таком объединении передача задействует все каналы в активной агрегации, согласно стандарту ieee 802.3ad. Выбор через какой интерфейс отправлять пакет определяется политикой, по умолчанию XOR политика.

Окончание таблицы 43

Политика	Описание
	Требования: 1) поддержка ethtool в драйвере, для получения информации о скорости и дуплексе на каждом сетевом интерфейсе; 2) поддержка на коммутаторе стандарта ieee 802.3ad; 3) настройка на коммутаторе.
Адаптивная балансировка нагрузки передачи	Исходящий трафик распределяется в зависимости от загруженности каждой сетевой карты (определяется скоростью загрузки). Не требует дополнительной настройки на коммутаторе. Входящий трафик приходит на текущую сетевую карту. Если она выходит из строя, то другая сетевая карта берет себе MAC-адрес вышедшей из строя карты. Требования: 1) поддержка ethtool в драйвере, для получения информации о скорости загрузки на каждом сетевом интерфейсе.
Адаптивная балансировка нагрузки	Включает в себя политику «адаптивной балансировки нагрузки передачи» плюс осуществляет балансировку входящего трафика. Не требует дополнительной настройки на коммутаторе. Балансировка входящего трафика достигается путем ARP переговоров. Драйвер объединения (bonding) перехватывает ARP ответы, отправляемые с локальных сетевых карт наружу, и переписывает MAC-адрес источника на один из уникальных MAC-адресов сетевой карты, участвующей в объединении. Таким образом различные пиры используют различные MAC-адреса сервера. Балансировка входящего трафика распределяется последовательно (Round-robin) между интерфейсами. Требования: 1) поддержка ethtool в драйвере, для получения информации о скорости загрузки на каждом сетевом интерфейсе; 2) поддержка в драйвере замены MAC-адреса на включенном устройстве; 3) возможно придется корректировать значение параметра updelay равным или большим, чем значение задержки на коммутаторе (что бы ARP ответы небыли заблокированы на коммутаторе при переподключении ссылки, либо при добавлении новой сетевой карты в объединение).

Заполнить поле «Параметры объединения» (рис. 121). В таблице 44 приведены возможные значения параметров объединения.

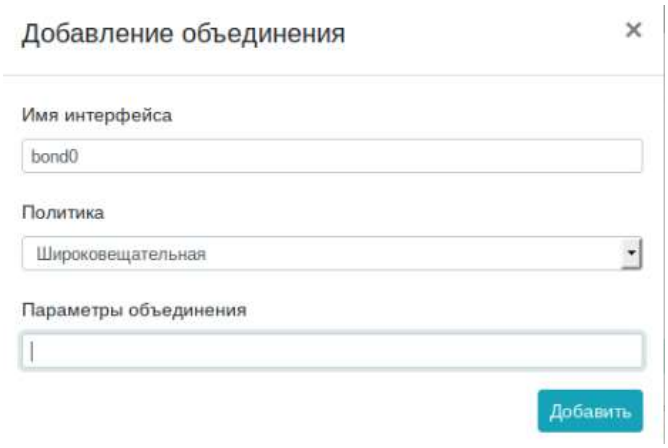


Рис. 121 – Параметры объединения

Таблица 44

Параметр	Описание	Значения	
mode	Определяет политику поведения объединенных интерфейсов.	Возможные значения:	
		balance-rr или 0	Политика round-robin. Пакеты отправляются последовательно, начиная с первого доступного интерфейса и заканчивая последним. Эта политика применяется для балансировки нагрузки и отказоустойчивости.
		active-backup или 1	Политика активный-резервный. Только один сетевой интерфейс из объединенных будет активным. Другой интерфейс может стать активным, только в том случае, когда упадет текущий активный интерфейс. При такой политике MAC-адрес bond интерфейса виден снаружи только через один сетевой порт, во избежание появления проблем с коммутатором. Эта политика применяется для отказоустойчивости.
		balance-xor или 2	Политика XOR. Передача распределяется между сетевыми картами используя формулу: [(«MAC-адрес источника» XOR «MAC-адрес назначения») по модулю «число интерфейсов»]. Получается одна и та же сетевая карта передает пакеты

Продолжение таблицы 44

Параметр	Описание	Значения	
			одним и тем же получателям. Опционально распределение передачи может быть основано и на политике «xmit_hash». Политика XOR применяется для балансировки нагрузки и отказоустойчивости.
		broadcast или 3	Широковещательная политика. Передает все на все сетевые интерфейсы. Эта политика применяется для отказоустойчивости.
		802.3ad или 4	Политика агрегирования каналов по стандарту IEEE 802.3ad. Создаются агрегированные группы сетевых карт с одинаковой скоростью и дуплексом. При таком объединении передача задействует все каналы в активной агрегации, согласно стандарту IEEE 802.3ad. Выбор через какой интерфейс отправлять пакет определяется политикой, по умолчанию XOR политика, можно использовать «xmit_hash» политику. Требования: 1) поддержка Ethtool в драйвере, для получения информации о скорости и дуплексе на каждом сетевом интерфейсе; 2) поддержка на коммутаторе стандарта IEEE 802.3ad; 3) настройка на коммутаторе.
		balance-tlb или 5	Политика адаптивной балансировки нагрузки передачи. Исходящий трафик распределяется в зависимости от загруженности каждой сетевой карты (определяется скоростью загрузки). Не требует дополнительной настройки на коммутаторе. Входящий трафик приходит на текущую сетевую карту.

Продолжение таблицы 44

Параметр	Описание	Значения	
			Если она выходит из строя, то другая сетевая карта берет себе MAC-адрес вышедшей из строя карты. Требования: Поддержка Ethtool в драйвере, для получения информации о скорости загрузки на каждом сетевом интерфейсе.
		balance-alb или 6	Политика адаптивной балансировки нагрузки. Включает в себя политику balance-tlb плюс осуществляет балансировку входящего трафика. Не требует дополнительной настройки на коммутаторе. Балансировка входящего трафика достигается путем ARP переговоров. Драйвер bonding перехватывает ARP ответы, отправляемые с локальных сетевых карт наружу, и переписывает MAC-адрес источника на один из уникальных MAC-адресов сетевой карты, участвующей в объединении. Таким образом различные пиры используют различные MAC-адреса сервера. Балансировка входящего трафика распределяется последовательно (round-robin) между интерфейсами. Требования: 1) поддержка Ethtool в драйвере, для получения информации о скорости загрузки на каждом сетевом интерфейсе; 2) поддержка в драйвере замены MAC-адреса на включенном устройстве; 3) возможно придется корректировать значение параметра updelay равным или большим, чем значение задержки на коммутаторе

Продолжение таблицы 44

Параметр	Описание	Значения	
			(что бы ARP ответы небыли заблокированы на коммутаторе при переподключении линка, либо при добавлении новой сетевой карты в bonding).
ad_select	Определяет логику выбора для агрегации по стандарту IEEE 802.3ad (эта опция добавлена в bonding начиная с версии 3.4.0).	Возможные значения:	
		stable или 0	Это значение по умолчанию. Активный агрегатор выбирается по наибольшей объединенной полосе пропускания. Перевыбор активного агрегатора осуществляется, только, когда упадут линки на всех сетевых картах в активном агрегаторе, либо если активный агрегатор не имеет сетевых карт.
		bandwidth или 1	Активный агрегатор выбирается по наибольшей объединенной полосе пропускания. Перевыбор активного агрегатора происходит если: - сетевой интерфейс добавился или удалился из объединения; - на любом интерфейсе изменилось состояние линка; - любой интерфейс изменил состояние ассоциации 802.3ad; - административно перевели состояние интерфейса bond в поднятое (up).
		count или 2	Активный агрегатор выбирается по наибольшему числу портов. Перевыбор происходит при таких же условиях, как и для значения bandwidth.

Продолжение таблицы 44

Параметр	Описание	Значения
arp_interval	Определяет ARP мониторинг канала (задается в миллисекундах). ARP мониторинг периодически проверяет на сетевых картах возможность приема и передачи трафика. Обычно для проверки генерируются ARP запросы, отправляемые на адрес, указанный в параметре «arp_ip_target». Такое поведение может быть изменено параметром «arp_validate». Если ARP мониторинг используется в режиме balance-rr или balance-xor, тогда коммутатор должен быть сконфигурирован на режим, в котором равномерно распределяются пакеты по всем линкам. Если коммутатор сконфигурирован передавать пакеты по политике XOR, тогда все ответы с arp_ip_target будут получены через один и тот же линк, что вызовет падение на остальных интерфейсах в объединении. ARP мониторинг не может использоваться одновременно с мониторингом MII (miimon).	Значение по умолчанию 0 (выключен).
arp_ip_target	Указывает IP-адреса для ARP мониторинга (используется, если arp_interval >0). На эти адреса будут отправляться ARP запросы, для определения возможности приема-передачи через интерфейсы, входящие в bonding. IP-адрес задается в формате ddd.ddd.ddd.ddd (прим. 192.168.0.1). Для указания нескольких IP-адресов, их нужно разделить запятой. Максимально можно использовать 16 IP-адресов.	Значение по умолчанию: без IP-адреса.

Продолжение таблицы 44

Параметр	Описание	Значения	
arp_validate	Определяет будут или нет проверяться ARP запросы и ответы при использовании режима active-backup. При этой опции ARP мониторинг проверяет входящие ARP запросы и ответы, и следит, что бы интерфейс был поднят, если он получает соответствующий ARP трафик. Для активного интерфейса производится проверка приходящих ARP ответов, они должны исходить от хоста, указанного в arp_ip_target. Резервный интерфейс обычно не получает такие ARP ответы, и проверка на нем выполняется путем ответа на ARP запрос, посланный через активный интерфейс. При определенной конфигурации сети, может возникнуть ситуация, при которой, резервный интерфейс не сможет получать ARP запросы, при возникновении такой ситуации проверку на резервных интерфейсах следует отключить.	Возможные значения:	
		none или 0	Проверка не выполняется (значение по умолчанию).
		active или 1	Проверка происходит только на активном интерфейсе.
		backup или 2	Проверка происходит только на резервных интерфейсах.
		all или 3	Проверка происходит на всех интерфейсах.
downdelay	Определяет время (в миллисекундах) задержки перед отключением интерфейса, если произошел сбой соединения. Эта опция действительна только для мониторинга MII (miimon). Значение параметра должно быть кратным значениям miimon. Если оно не кратно, то округлится до ближайшего кратного значения.	Значение по умолчанию 0.	
fail_over_mac	Определяет, как будут прописываться MAC-адреса на объединенных интерфейсах в режиме active-backup при переключении интерфесов. Обычным поведением является одинаковый MAC-адрес на всех интерфейсах.	Возможные значения:	
		none или 0	Отключает fail_over_mac. Устанавливает одинаковый MAC-адрес на всех интерфейсах во время переключения. Это значение по умолчанию.

Продолжение таблицы 44

Параметр	Описание	Значения	
	Этот параметр добавлен в версии bonding драйвера 3.2.0, а значение «follow» в версии 3.3.0	active или 1	MAC-адрес на bond интерфейсе будет всегда таким же как и на текущем активном интерфейсе. MAC-адреса на резервных интерфейсах не изменяются. MAC-адрес на bond интерфейсе меняется во время обработки отказа.
		follow или 2	MAC-адрес на bond интерфейсе будет таким, как на первом интерфейсе добавленном в объединение. На второй и последующем интерфейсе не устанавливается этот MAC, пока они в резервном режиме. MAC-адрес прописывается во время обработки отказа, когда резервный интерфейс становится активным, он принимает новый MAC (тот, что на bond интерфейсе), а старому активному интерфейсу прописывается MAC, который был на текущем активном (то есть MAC на bond интерфейсе должен быть всегда одним и тем же).
lacp_rate	Определяет с каким интервалом будут передаваться партнером LACPDU пакеты в режиме 802.3a.	Возможные значения:	
		slow или 0	Запрос партнера на передачу LACPDU пакетов каждые 30 секунд (значение по умолчанию).
		fast или 1	Запрос партнера на передачу LACPDU пакетов каждую 1 секунду.
max_bonds	Указывает сколько bonding устройств следует создавать драйверу. Например, если «max_bonds = 3», то в системе будут созданы bond0, bond1, bond2 интерфейсы.	Значение по умолчанию 1. Если значение 0, то загрузится только драйвер, но устройство создано не будет.	
miimon	Устанавливает периодичность MII мониторинга в миллисекундах. Определяет, как часто будет проверяться состояние линии на наличие отказов. Параметр use_carrier указывает каким образом будет определяться состояние канала.	Значение по умолчанию 0 – отключает MII мониторинг. Если нужно использовать такой мониторинг, рекомендуется сперва попробовать значение 100.	

Продолжение таблицы 44

Параметр	Описание	Значения	
num_grat_arp и num_unsol_na	Указывает количество оповещений, отправляемых соседним пирам после возникновения события отказа (самообращенные ARP запросы (gratuitous ARP) и не затребованные оповещения IPv6 (unsolicited IPv6 Neighbor Advertisements)). После того, как поднялся линк на новом сетевом интерфейсе, отправляется оповещение на интерфейсе bond и на каждом его VLAN подинтерфейсе. Если значение больше 1, то оповещение повторяется через каждый интервал мониторинга канала (интервал задается в arp_interval или miimon).	Допустимое значение от 0 до 255, значение по умолчанию 1.	
primary	Строка вида (eth0, eth1, и т. д.). Указывает какой интерфейс будет первичным. Этот интерфейс будет всегда активным, пока он доступен, переключение произойдет только в том случае, если интерфейс упадет или выключится. Этот параметр используется, когда один интерфейс имеет преимущество перед другим, например, по полосе пропускания. Параметр только для режима active-backup.		
primary_reselect	Определяет, как будет производиться возвращение на первичный интерфейс, после возобновления его работоспособности.	Возможные значения:	
		always или 0	Значение по умолчанию. Первичный интерфейс становится активным всегда, когда возобновляется работоспособность через него.
		better или 1	Первичный интерфейс становится активным, если значение скорости и дуплекса лучше значений текущего активного интерфейса.

Продолжение таблицы 44

Параметр	Описание	Значения	
		failure или 2	Первичный интерфейс становится активным, только, если на текущем активном интерфейсе произошла ошибка.
updelay	Задает время задержки в миллисекундах, перед тем как поднять линк при обнаружении восстановления канала. Этот параметр возможен только при MII мониторинге. Значение параметра должно быть кратным значениям miimon. Если оно не кратно, то округлится до ближайшего кратного значения.	Значение по умолчанию 0.	
use_carrier	Указывает как miimon будет определять состояние линии, используя контроль ввода-вывода (ioctl) MII или ETHTOOL, либо используя функцию netif_carrier_ok(). MII или ETHTOOL ioctl мене эффективны и используют устаревшие методы работы с ядром. Предпочтительней использование netif_carrier_ok, но не все драйвера поддерживают данную функцию. Если драйвер не поддерживает netif_carrier, то может возникнуть такая ситуация, что интерфейс всегда поднят, даже при отсутствии линка, в этом случае стоит использовать MII или ETHTOOL ioctl.	Значение 1 – включает использование netif_carrier_ok(). Это значение по умолчанию. Значение 0 – включает использование устаревших MII/ETHTOOL ioctl.	
xmit_hash_policy	Определяет хэш политику передачи пакетов через объединенные интерфейсы в режиме balance-xor или 802.3ad.	Значение по умолчанию: layer2. Возможные значения:	
		layer2	Использует только MAC-адреса для генерации хэша. При этом алгоритме трафик для конкретного сетевого хоста будет отправляться всегда через один и тот же интерфейс. Алгоритм совместим с 802.3ad. Формула расчета хэша: (source MAC XOR destination MAC) modulo slave count

Окончание таблицы 44

Параметр	Описание	Значения	
		layer2+3	Использует как MAC-адреса так и IP-адреса для генерации хэша. Алгоритм совместим с 802.3ad. Формула расчета хэша: (((source IP XOR dest IP) AND 0xffff) XOR (source MAC XOR destination MAC)) modulo slave count
		layer3+4	Используется IP-адреса и протоколы транспортного уровня (TCP или UDP) для генерации хэша. Алгоритм не всегда совместим с 802.3ad, так как в пределах одного и того же TCP или UDP взаимодействия может передаваться как фрагментированные так и не фрагментированные пакеты. Во фрагментированных пакетах порт источника и порт назначения отсутствуют. В результате в рамках одной сессии пакеты могут дойти до получателя не в том порядке (так как отправляются через разные интерфейсы). Некоторое оборудование, совместимое с 802.3ad может некорректно отработать такую ситуацию. Формула расчета хэша: ((source port XOR dest port) XOR ((source IP XOR dest IP) AND 0xffff) modulo slave count
resend_igmp	Указывает, какое количество отчетов о принадлежности группе (IGMP membership) отсылать при возникновении события отказа. Один отчет отсылается немедленно, после возникновения отказа, последующие пакеты отправляются с интервалом в 200 миллисекунд. Используется в режимах balance-rr (0), active-backup (1), balance-tlb (5) and balance-alb (6).	Возможные значения от 0 до 255. Значение 0 – не посылать отчеты, в случае возникновения отказа. Значение по умолчанию 1.	

После введения всех параметров необходимо нажать на кнопку «Добавить», откроется окно редактирования интерфейса (рис. 122).

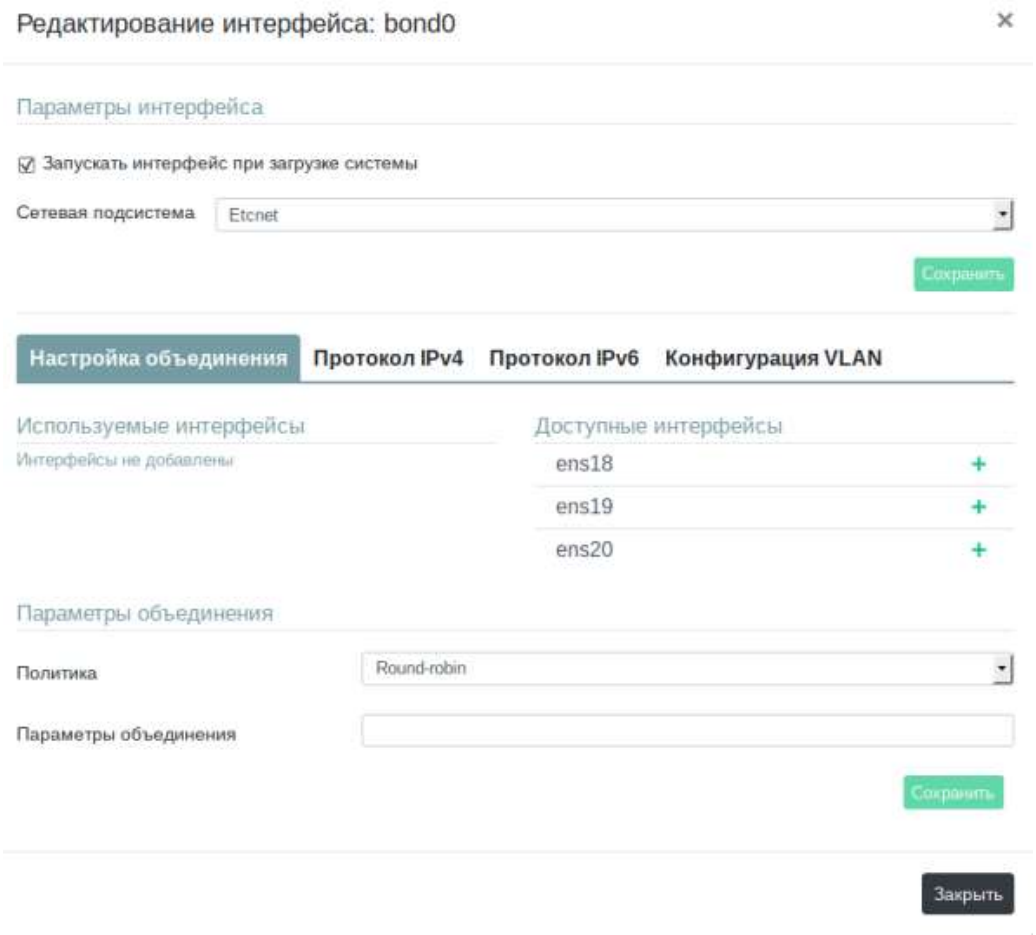


Рис. 122

Добавьте используемые интерфейсы или отредактируйте настройки и нажмите на кнопку «Сохранить».

В общем списке интерфейсах также появится строка с добавленным объединением (рис. 123).

Рис. 123

10.1.3. Создание сетевого моста

Сетевой мост – это сетевое устройство, предназначенное для объединения сегментов сети передачи данных в единую сеть.

Для того чтобы добавить сетевой мост, необходимо ввести имя интерфейса (рис. 124) и нажать на кнопку «Добавить».

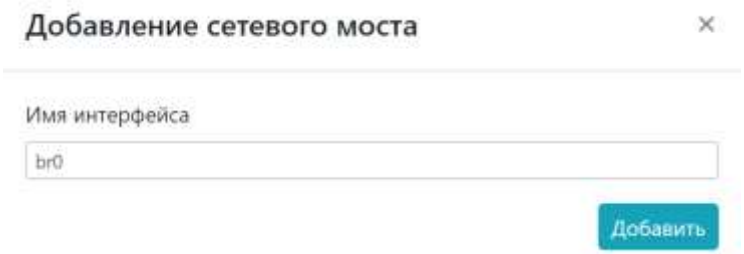


Рис. 124 – Окно добавления сетевого моста

Система уведомит об успешном создании сетевого моста (рис. 125) и появится окно редактирования настроек интерфейса сетевого моста (рис. 126).

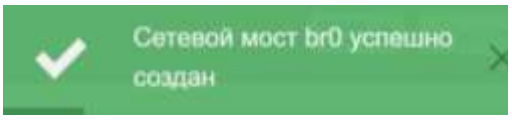


Рис. 125

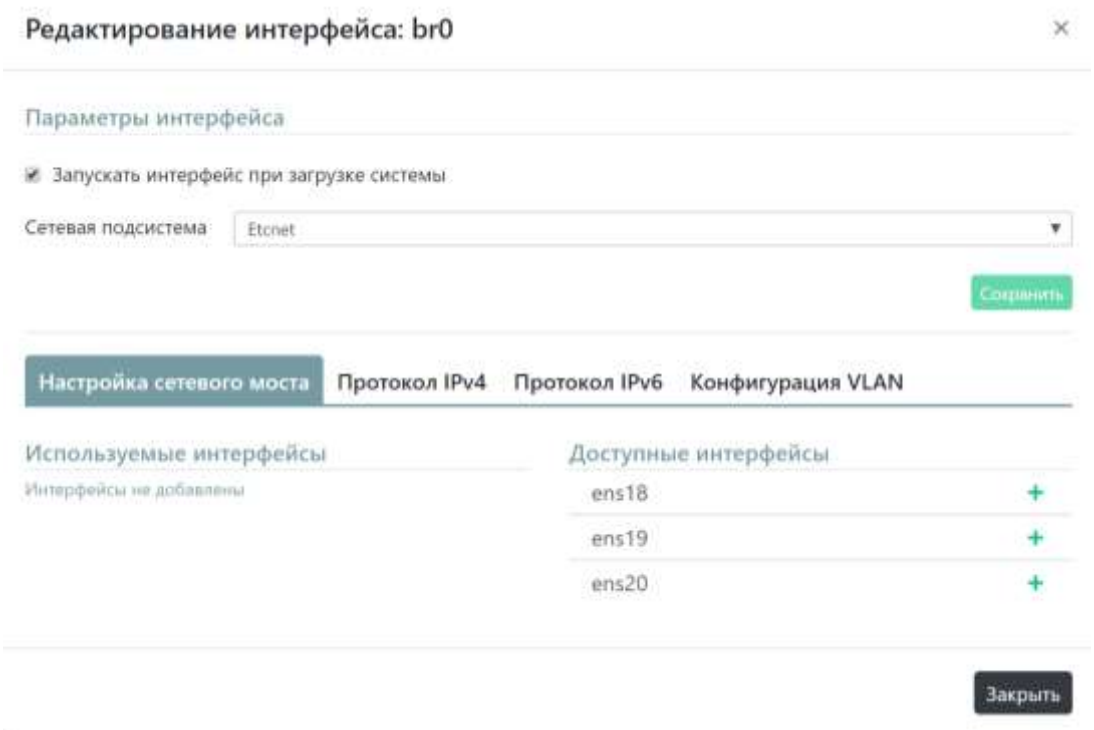


Рис. 126 – Окно редактирования настроек сетевого моста

Параметры редактирования настроек интерфейса позволяют выбрать сетевую подсистему («Etcnet» или «Не контролируется») и поставить/убрать флаг запуска интерфейса при загрузке системы. Для подтверждения произведенных настроек нажмите на кнопку «Сохранить».

На вкладке «Настройки сетевого моста» отображены «Используемые интерфейсы» и «Доступные интерфейсы». Для выбора используемого интерфейса нажать на кнопку + (рис. 127). Для удаления интерфейса из списка используемых нажмите кнопку −.

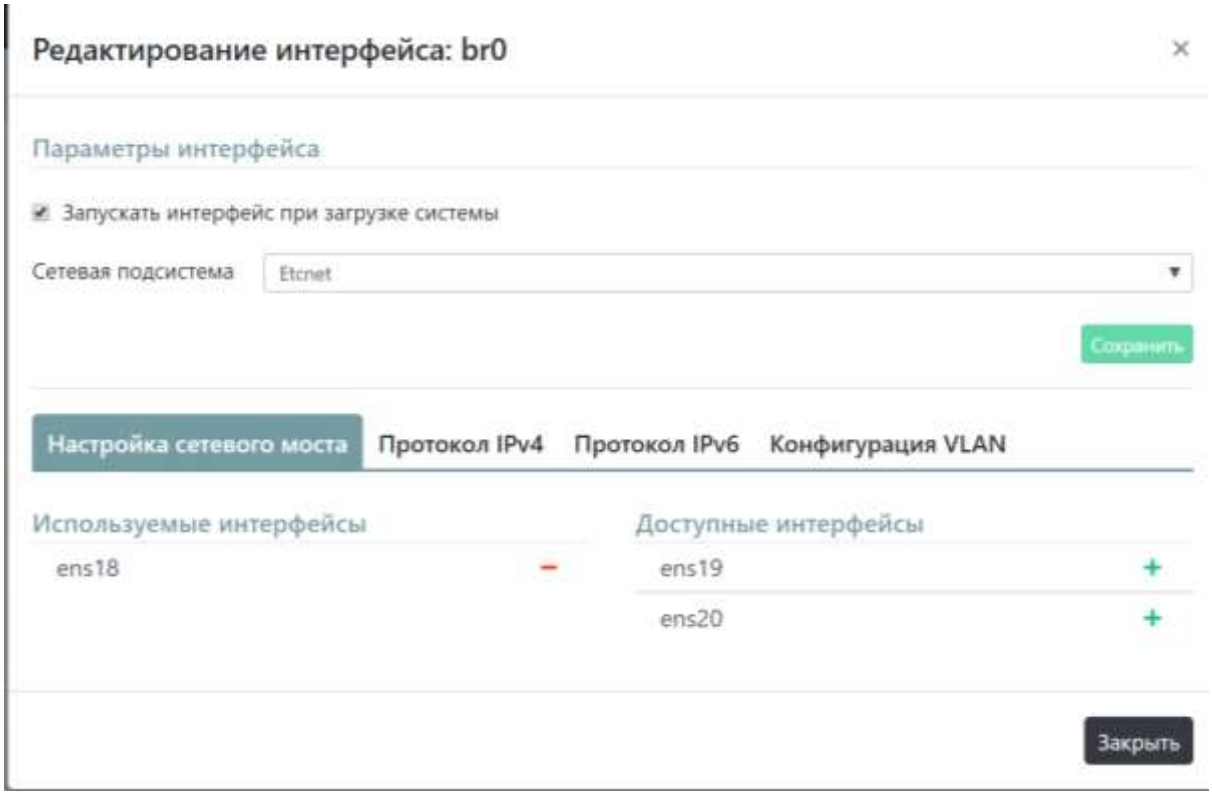


Рис. 127 – Используемые интерфейсы

Редактирование и описание настроек других вкладок интерфейса приведено в соответствующих подразделах:

- «Настройка протокола IPv4» (п. 10.1.4.1);
- «Настройка протокола IPv6» (п. 10.1.4.2);
- «Конфигурация VLAN» (п. 10.1.4.3).

Для того чтобы удалить интерфейсы, созданные при добавлении VLAN, сетевого моста или объединения нажмите кнопку  . Если кнопка удаления была выбрана ошибочно, то нажмите кнопку «Отмена» или закройте окно и операция удаления не будет произведена. Если интерфейс действительно нужно удалить, то в появившемся окне нажать на кнопку «Удалить» (рис. 128).

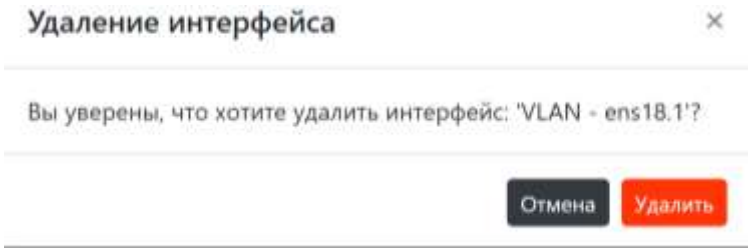


Рис. 128 – Окно удаления интерфейса

Система уведомит пользователя информационным сообщением, о успешном удалении интерфейса (рис. 129).

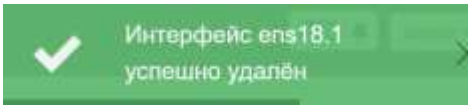


Рис. 129 – Уведомление системы о успешном удалении интерфейса

10.1.4. Данные Ethernet-интерфейса

В окне Ethernet-интерфейсы отображается домен пользователя и имя компьютера, которое можно изменить, внеся необходимое название и нажав на кнопку «Сохранить». Также представлены названия интерфейсов и их общие данные (рис. 130).

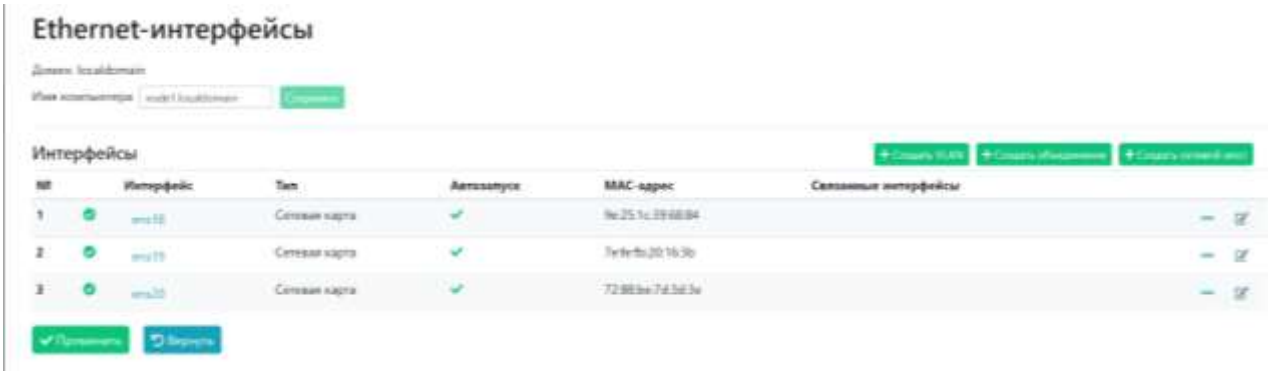


Рис. 130 – Ethernet-интерфейсы

Кнопки, расположенные справа от каждого имени интерфейса необходимы, чтобы посмотреть свойства интерфейса или его отредактировать.

Для того, чтобы ознакомиться с данными о созданном интерфейсе нажмите кнопку ••• или на название интерфейса, например, ens18. Появится всплывающее окно просмотра с описанием характеристик (рис. 131).

Кнопка ✎ или ✎ Редактировать на рис. 131 позволяет редактировать настройки интерфейса. Если изменения вносить не нужно, то нажмите на кнопку «Закрыть».

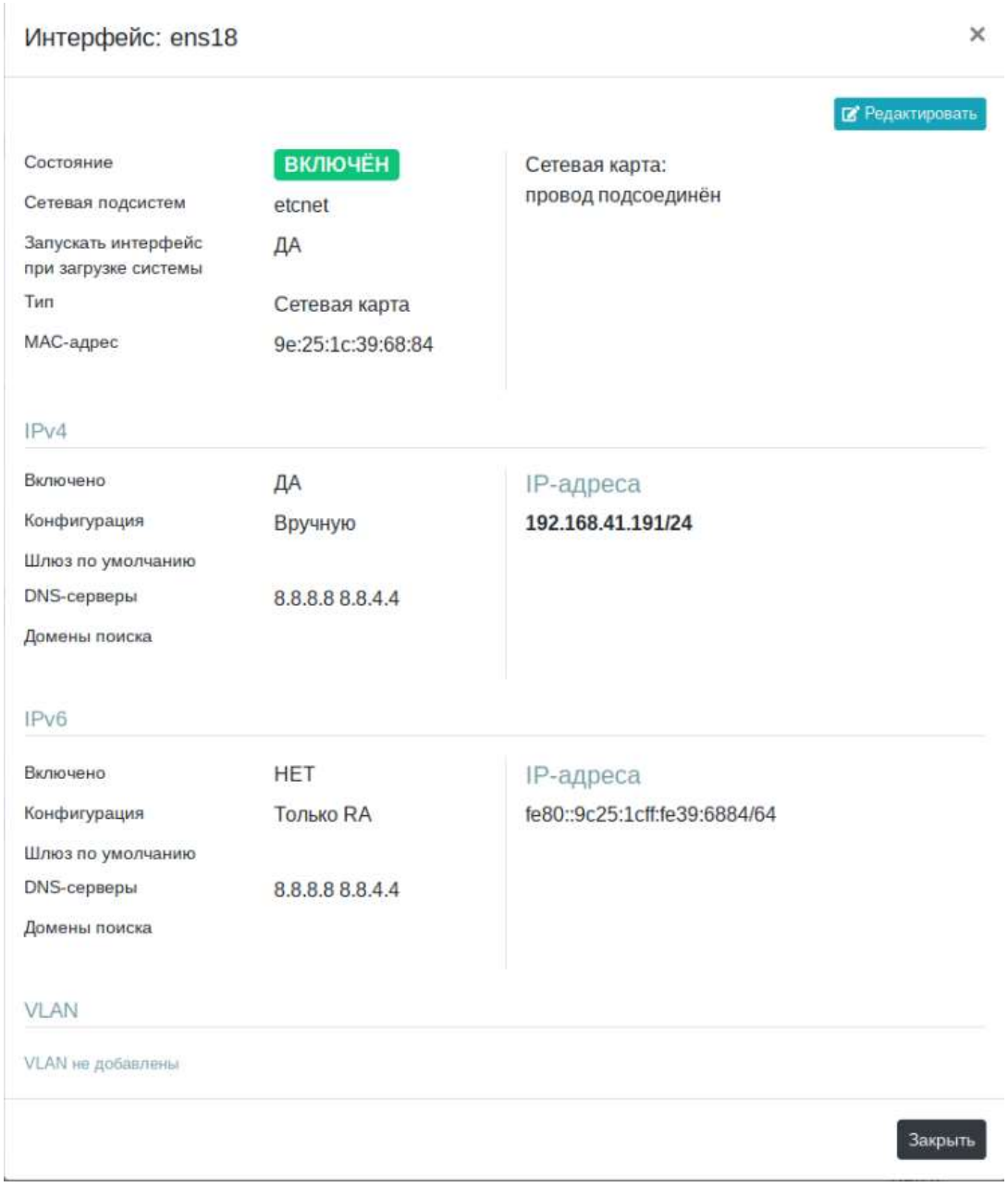


Рис. 131 – Окно просмотра настроек интерфейса

Окно редактирования интерфейса позволяет автоматически запускать интерфейс при запуске системы, выбрать сетевую подсистему, а также произвести настройку протоколов IPv4 и IPv6 (рис. 132).

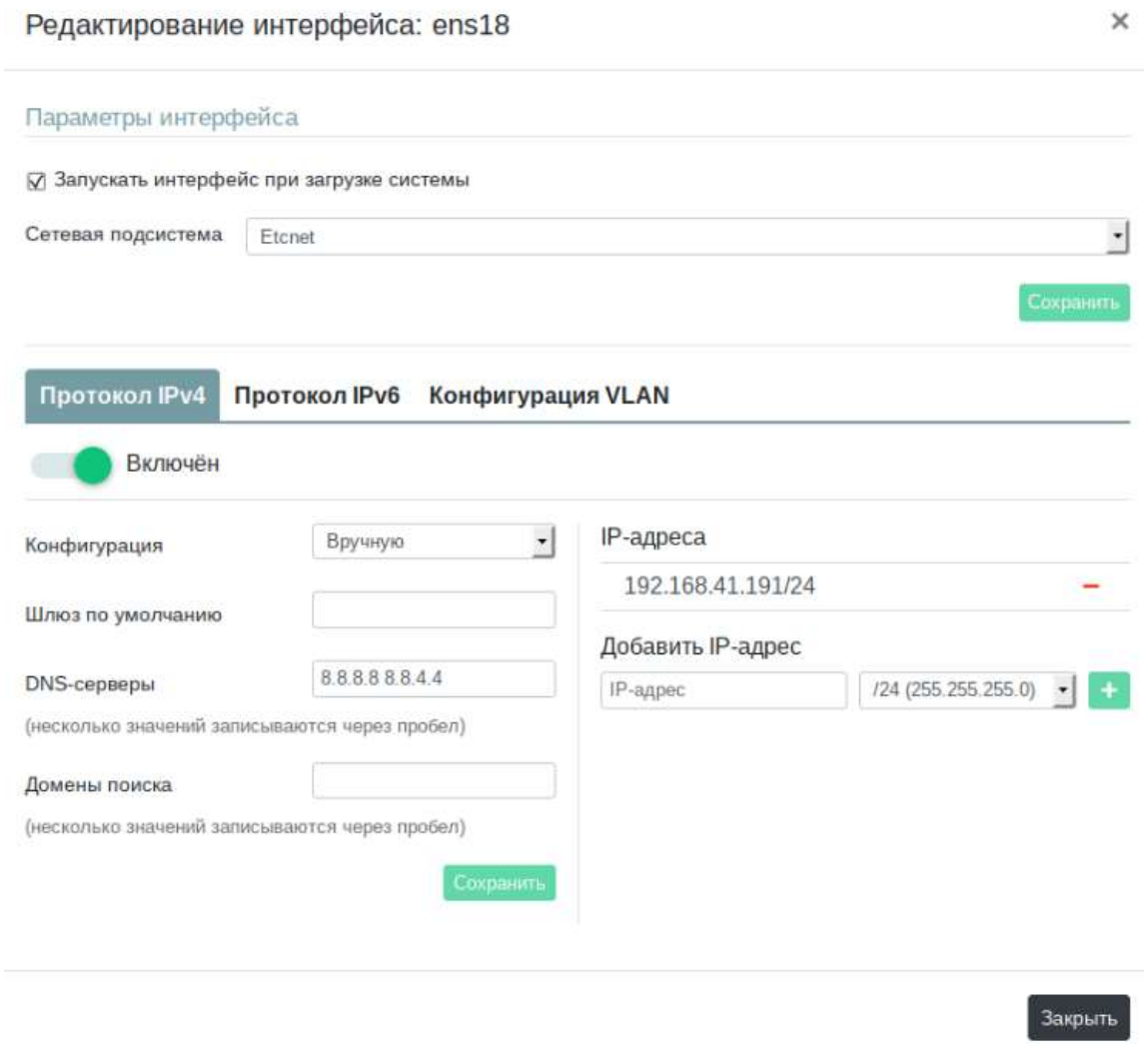


Рис. 132 – Окно редактирования интерфейса

Параметры редактирования интерфейса настроек позволяют выбрать сетевую подсистему «Etcnet» или «Не контролируется» и поставить/убрать флаг запуска интерфейса при загрузке системы.

При настройке протоколов следует выбрать необходимую вкладку для IPv4 (см. настройки в п. 10.1.4.1) или IPv6 (см. настройки в п. 10.1.4.2).

Включение протокола осуществляется сдвигом переключателя вправо (см. рис. 132).

Если необходимо выключить протокол, то сдвинуть переключатель влево (рис. 133). Все параметры редактирования перестанут быть видны и активны.

Система уведомит пользователя информационным сообщением, что версия протокола успешно выключена/включена (рис. 134).

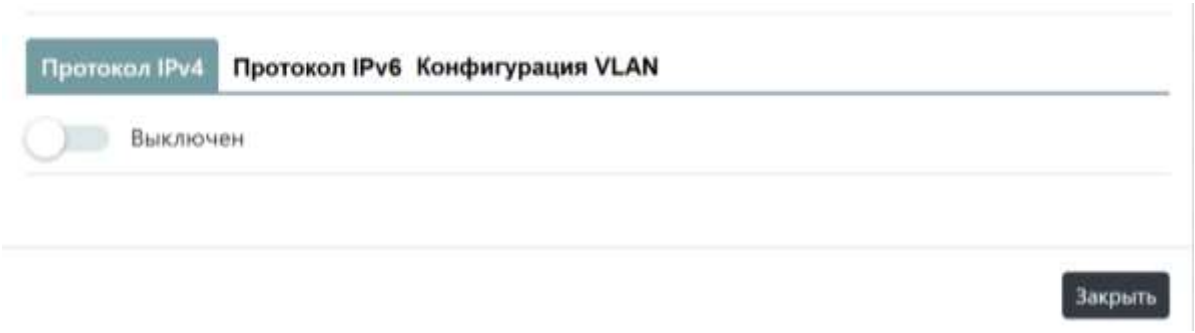


Рис. 133 – Выключение версии протокола

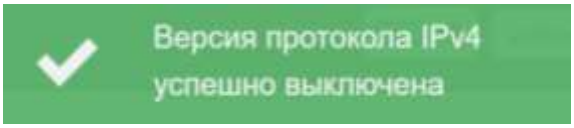


Рис. 134 – Сообщение о выключении версии протокола

Для подтверждения и активации произведенных настроек нажмите на кнопку «Сохранить».

10.1.4.1. Настройка протокола IPv4

IP (Internet Protocol) – основа стека протоколов TCP/IP. IP-адрес и Маска сети – обязательные параметры каждого хоста IP-сети. Первый параметр – уникальный идентификатор машины, от второго напрямую зависит, к каким машинам локальной сети данная машина будет иметь доступ. В случае наличия DHCP-сервера можно все вышеперечисленные параметры получить автоматически – включите «Использовать DHCP» в варианте конфигурации.

Вариант конфигурации определяется через выпадающий список:

- «Вручную»;
- «Использовать DHCP»;
- «Использовать Zeroconf».

«Шлюз по умолчанию» – выставить значение используемого шлюза по умолчанию, обязательно, если требуется выход во внешнюю сеть.

«DNS-серверы» – при работе и настройке сетевых служб часто приходится использовать символьные имена других машин в сети. Чтобы система преобразовала их в IP-адреса, требуется либо перечислить соответствия в файле `/etc/hosts`, либо воспользоваться DNS-сервером. DNS (Domain Name System – система доменных имен) – распределенная база данных, способная по запросу, содержащему доменное имя хоста (компьютера или другого сетевого устройства), сообщить IP-адрес или наоборот по данному IP-адресу сообщить доменное имя устройства. Значения DNS-серверов записываются через пробел.

«Домены поиска» – перечислить в поле наиболее часто используемые домены (например, domain), записываются через пробел.

«IP-адреса» – отображаются список добавленных IP-адресов машин из локальной сети.

Для добавления в поле «Добавить IP-адрес» вписать значение IP-адреса машины из локальной сети и справа в выпадающем списке выбрать маску подсети, нажать кнопку .

Для удаления IP-адреса из списка нажать на кнопку .

Для подтверждения настроек нажать на кнопку «Сохранить».

Если редактирование настроек произведено, нажмите кнопку «Закрыть».

10.1.4.2. Настройка протокола IPv6

На вкладке «Протокол IPv6» (рис. 135) возможен выбор только включения/отключения использования протокола через переключатель.

Или выбор вариант конфигурации через выпадающий список:

- «Только RA»;
- «Использовать DHCP»;
- «Вручную».

Добавление IP-адресов происходит автоматически.

После выполнения редактировать нажать кнопку «Сохранить» на вкладке настроек. Система уведомит пользователя, что изменения успешно сохранены (рис. 136).

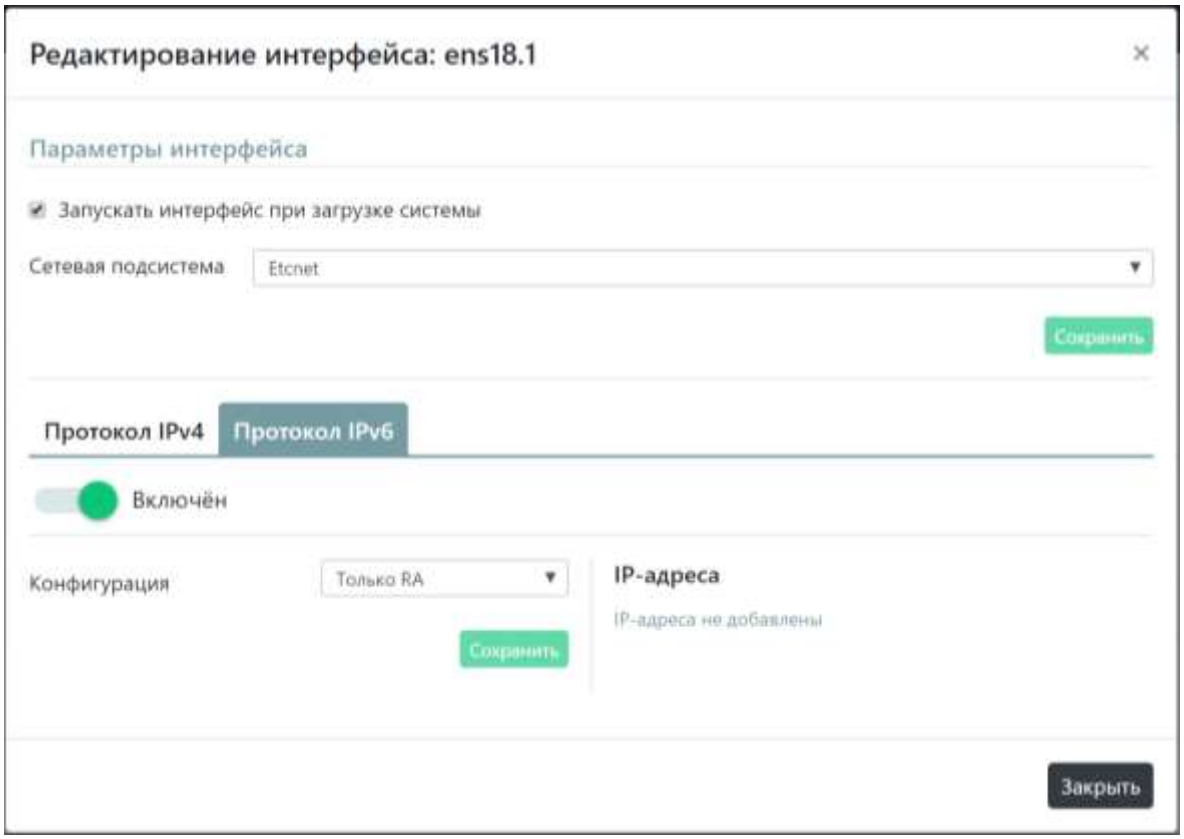


Рис. 135

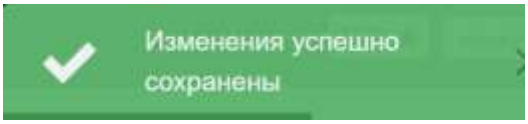


Рис. 136 – Успешное сохранение изменений

10.1.4.3. Конфигурация VLAN

При выборе интерфейса сетевой карты и открытии для редактирования вкладки «Конфигурация VLAN» можно добавить VLAN по номеру VID автоматически привязанного к этому интерфейсу.

Для этого нужно ввести в поле соответствующий номер VID и нажать кнопку «Добавить». В общем списке интерфейсов появится строка с добавленным интерфейсом типа VLAN (см. рис. 117).

10.2. PPTP-соединения

PPTP (Point-to-point tunneling protocol) – протокол для организации прямого соединения между двумя машинами в сети (рис. 137).

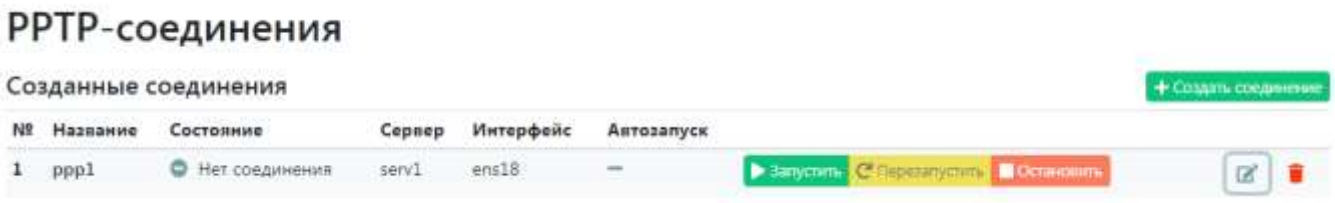


Рис. 137

Для создания нового соединения нажмите на кнопку «Создать соединение».

Имена для PPTP-соединений принято назначать в формате `ppp[номер]` (рис. 138).

Далее необходимо указать адрес удаленного PPTP-сервера (рис. 139), а также имя пользователя и пароль на этом сервере, а также выбрать «Опорный интерфейс» – интерфейс, через который будет происходить соединение.

Набор используемых параметров для каждого конкретного соединения зависит от возможностей PPTP-сервера.

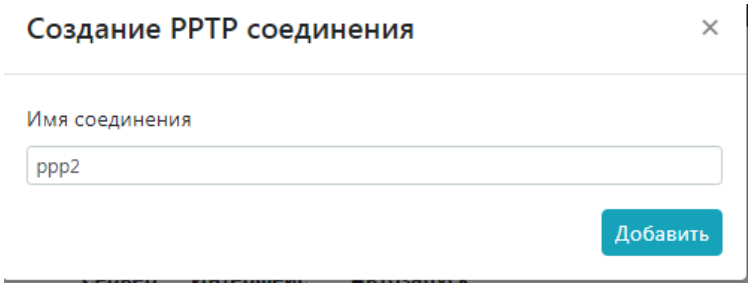


Рис. 138

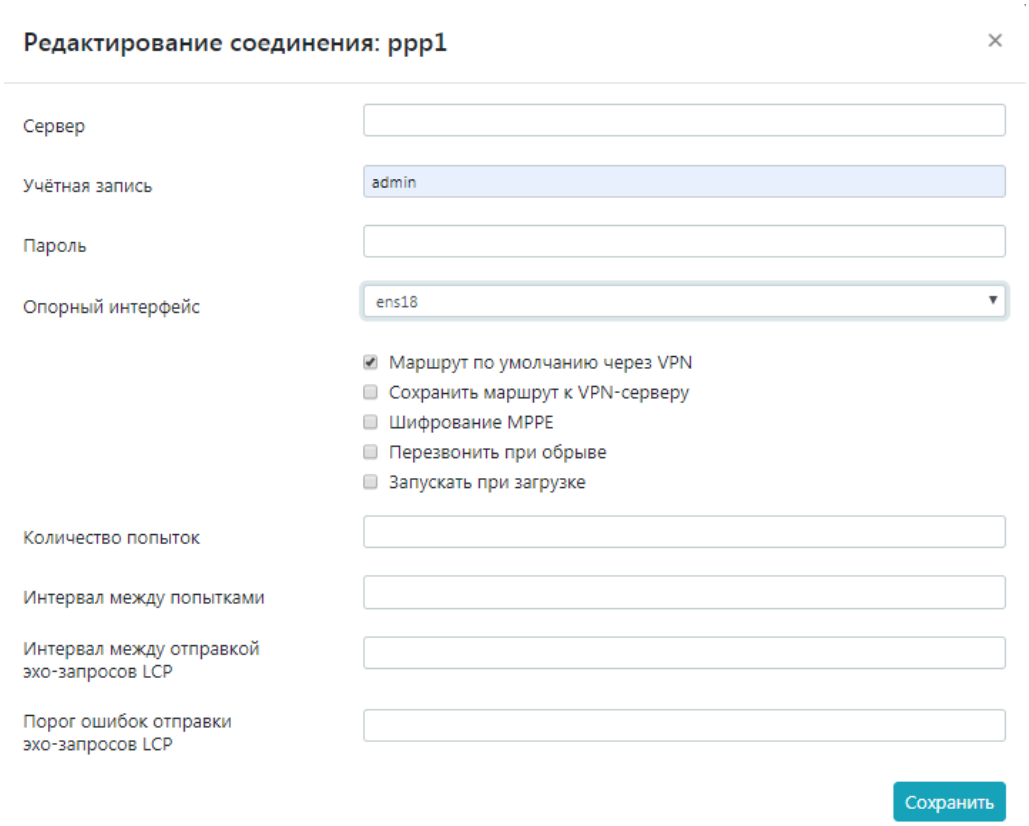


Рис. 139

10.3. L2TP-соединения

Подраздел L2TP-соединения (рис. 142) позволяет управлять L2TP-соединениями на сервере (клиентская часть L2TP).

При первом использовании модуля (когда еще нет настроенных L2TP-соединений) будет предложено создать новое L2TP-соединение задав ему имя. После того как соединение будет создано, появится сообщение об успехе создания соединения и откроется окно для редактирования его параметров (рис. 140). Обязательно указываются IP-адреса сервера L2TP и имя пользователя (рис. 141).

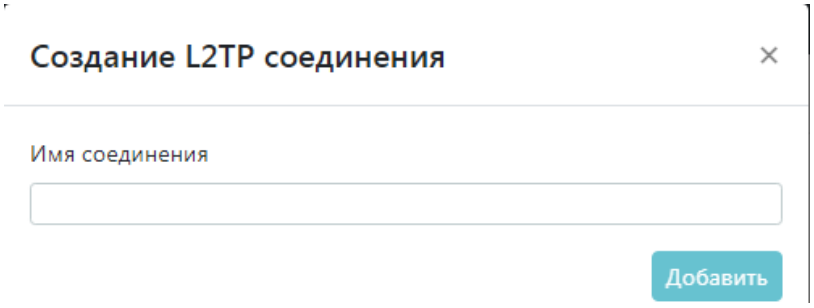


Рис. 140

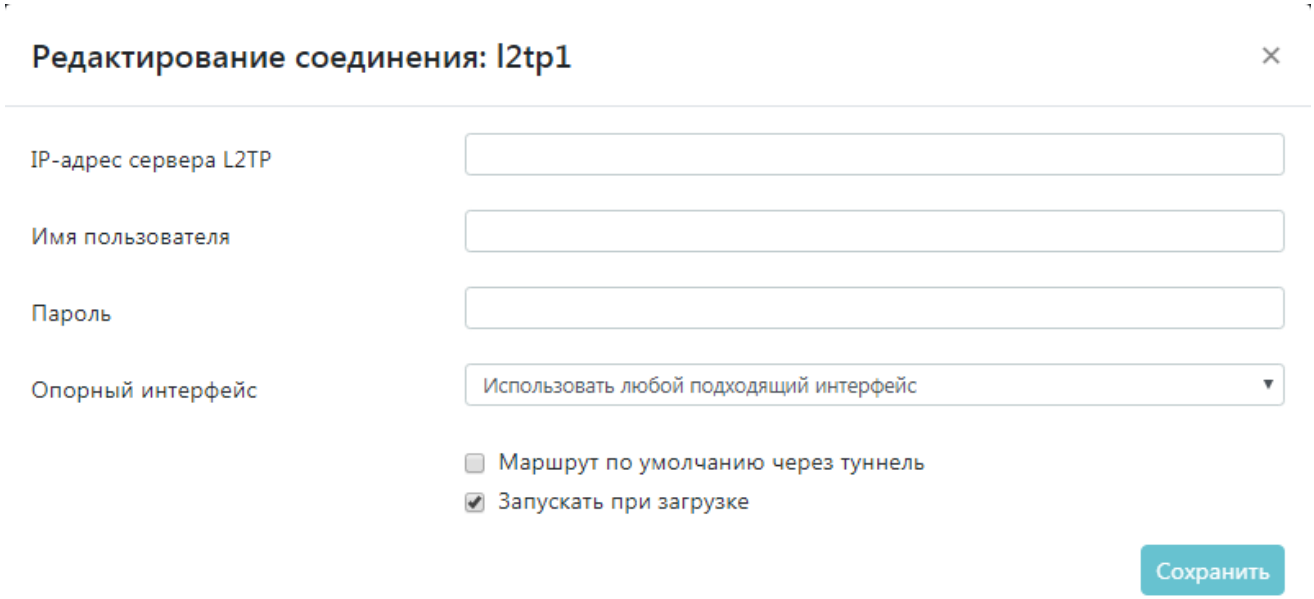


Рис. 141

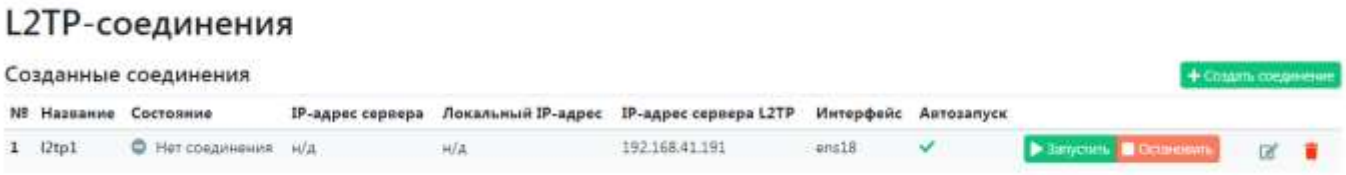


Рис. 142

10.4. Маршрутизация

Подраздел «Маршрутизация» (рис. 143) предназначен для управления статическими маршрутами на сервере. Экран модуля состоит из двух частей. В левой части отображается текущее состояние таблицы маршрутизации. В правой части отображаются текущие заданные вручную маршруты. Для добавления маршрута выберите кнопку «Создать маршрут» (рис. 144), для редактирования – (рис. 145), для удаления маршрута – .

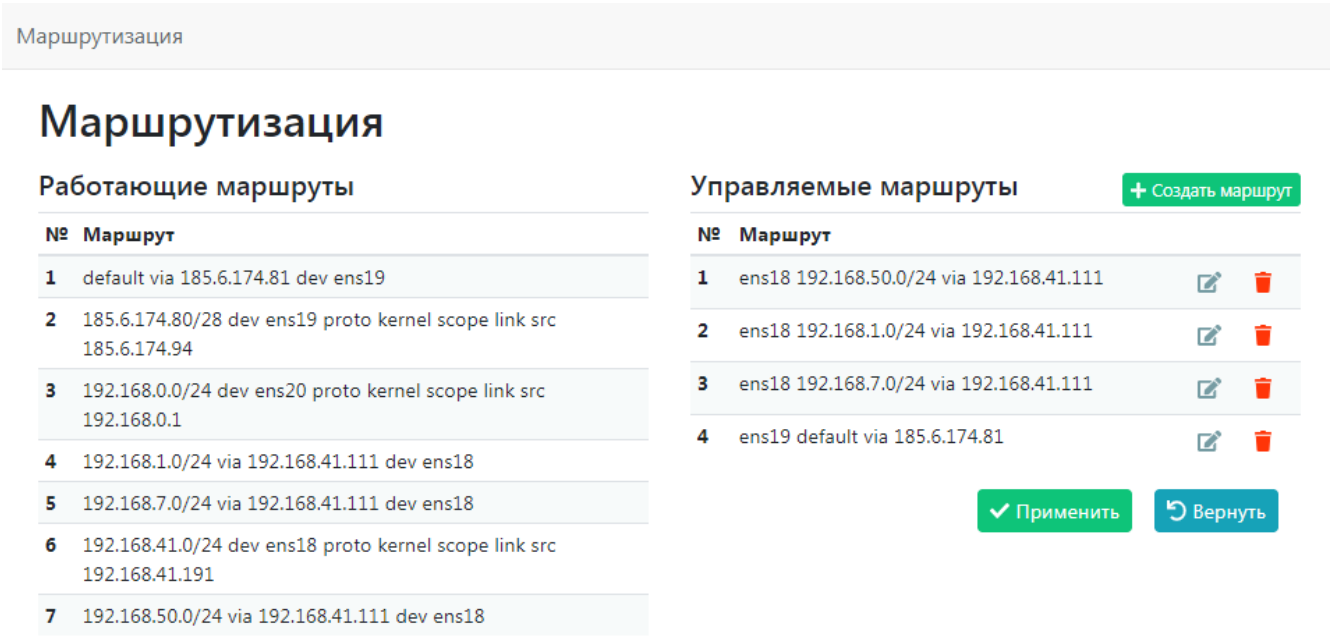


Рис. 143

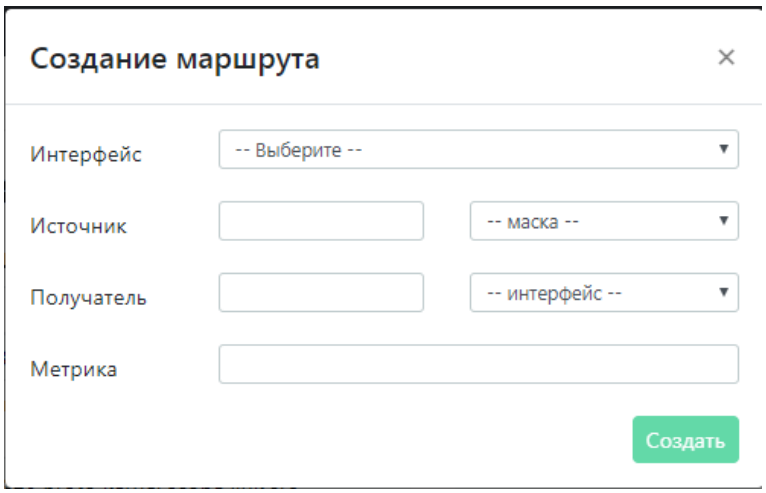


Рис. 144

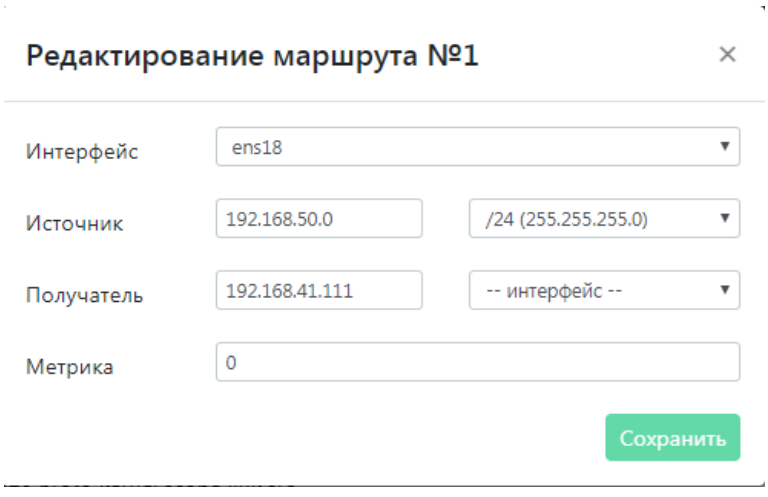


Рис. 145

10.5. Прокси-сервер

Прокси-сервер (squid) – это программа, которая получает HTTP/FTP-запросы клиентов и по ним обращается к ресурсам Интернет, позволяет контролировать содержание сетевых сообщений пользователей сети, их активности в сети интернет, выявлять внешние и внутренние угрозы информационной безопасности. Применение прокси-сервера дает возможность использовать фиктивные IP-адреса во внутренней сети (маскарадинг).

По умолчанию прокси-сервер настроен на работы в прозрачном режиме, для получения обращения к веб-ресурсам по протоколу HTTP.

Для включения работы прокси-сервера в прозрачном режиме необходимо добавить следующее правило iptables:

```
iptables -t nat -I PREROUTING -p tcp –dport 80 -j DNAT –to 127.0.0.1:3128
```

В последующем весь траффик на 80 порт (HTTP) будет автоматически передан на прокси-сервер для последующей обработки.

Пересылка каждого запроса во внешнюю сеть фиксируется прокси-сервером (рис. 146) в специальном журнале. На основании этих данных автоматически формируются отчеты о статистике использования ресурсов сети, в том числе потраченного времени и количества переданных данных (трафика) (см. п. 10.9).

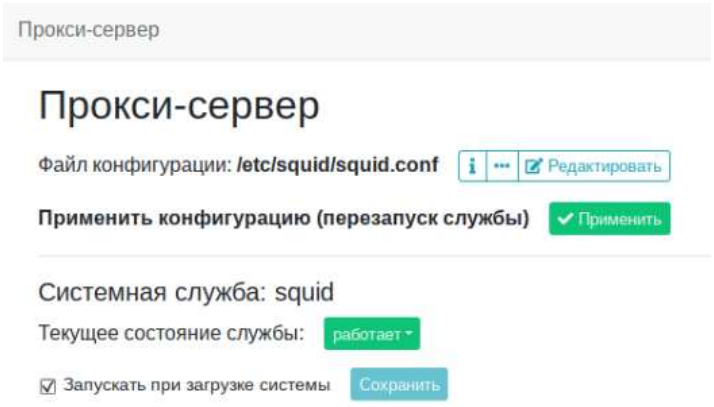


Рис. 146 – Окно файла конфигурации прокси-сервера

Основное назначение прокси-сервера – это сбор статистики и формирование отчета об объеме полученных из внешней сети данных. В зависимости от режима аутентификации пользователей при доступе к прокси-серверу, записи отчета могут содержать как имена пользователей, так и адреса локальной сети.

В данном подразделе выбирается файл конфигурации прокси-сервера, который можно также просмотреть, отредактировать, выгрузить и узнать информацию о нем.

Также возможно изменять текущее состояние службы и перезапускать ее (рис. 147).

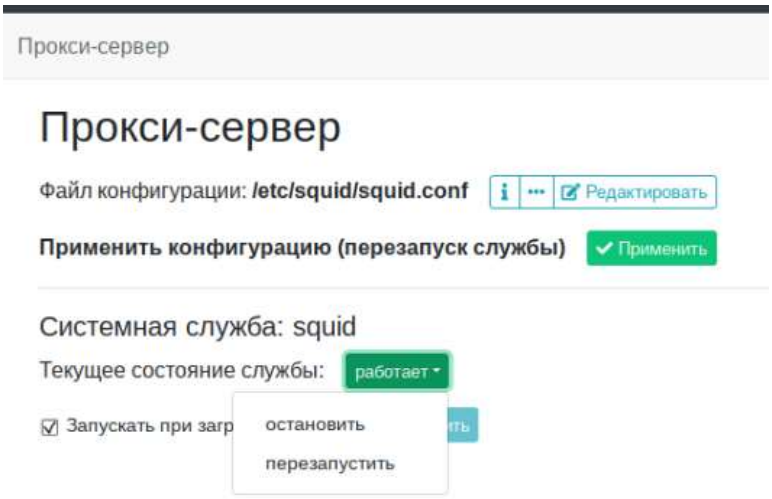


Рис. 147 – Настройки прокси-сервера

При установленном флаге «Запускать при загрузке системы» запуск будет происходить автоматически.

После выполнения всех настроек прокси-сервера необходимо нажать кнопку «Сохранить».

Для настройки прокси-сервера необходимо выполнить изменение его конфигурационного файла `squid.conf`.

При необходимости указать прокси-провайдера (тот сервер, который станет вашим neighbour):

```
cache_peer proxy.isp.ru
```

Установите объем памяти, разрешенный для кэша squid, в байтах, и каталог для дискового кэша:

```
cache_mem 65536
cache_dir ufs /usr/local/squid/cache 1024 16 256
```

где 1024 – количество Мбайт, отводимое под кэш в указанном каталоге. В этом каталоге будут храниться кэшированные файлы. Стоит ли говорить, что если у вас несколько жестких дисков, то кэш нужно разместить на самом быстром из них.

Указать хосты, которым разрешен доступ к прокси-серверу:

```
acl allowed_hosts src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
```

Указать разрешенные SSL-порты:

```
acl SSL_ports port 443 563
```

Запретить метод CONNECT для всех портов, кроме указанных в acl SSL_ports:

```
http_access deny CONNECT !SSL_ports
```

Запретить доступ всем, кроме тех, кому можно:

```
http_access allow localhost
http_access allow allowed_hosts
http_access allow SSL_ports http_access deny all
```

Прописать субъектов squid, которым разрешено пользоваться прокси-сервером в режиме авторизации (в рассматриваемом примере это test):

```
ident_lookup on
acl allowed_users test
http_access allow allowed_users
http_access deny all
```

Тэги `maximum_object_size` и `maximum_object` устанавливают ограничения на размер передаваемых объектов. Ниже приведен пример запрета доступа к любому URL, который соответствует шаблону `games`, и разрешения доступа ко всем остальным:

```
acl GaMS url_regex games
http_access deny GaMS
http_access allow all
```

Далее приведены дополнительные настройки squid в конфигурационном файле `squid.conf`:

- `http_port` – порт для запросов клиентов. С этого порта прокси-сервер будет ожидать и обрабатывать запросы клиентов. Значение по умолчанию равно 3128;
- `icp_port` – порт для общения с соседями через ICP. Если «соседей» (peer) нет, то установите `icp_port 0`. По умолчанию используется значение 3130. При использовании этого параметра нужно установить ключ `--enable-htcp` для директивы `htcp_port 4827`;
- `tcp_outgoing_address` – при отправлении информации указанный адрес будет использован в качестве исходного. Значение по умолчанию: `tcp_outgoing_address 255.255.255.255`;
- `udp_outgoing_address` – то же самое, что и предыдущая директива – но только для ICP. Значение по умолчанию;
- `udp_outgoing_address 255.255.255.255`. То же, но для ICP при приеме – директива `udp_incoming_address` со значением по умолчанию `0.0.0.0`;
- `passive_ftp on | off` – по умолчанию этот режим включен, но если прокси-сервер находится за межсетевым экраном, то параметр `passive_ftp` нужно выключить.

10.6. Автонастройка межсетевого экрана

Автоматическая настройка МЭ осуществляется с помощью загрузки файла конфигурации.

Для активации настроек конфигурации МЭ в соответствии с загруженным файлом нажмите кнопку «Применить».

Для сохранения текущей конфигурации МЭ в файл для настройки по умолчанию нажмите на кнопку «Сохранить» (рис. 148).

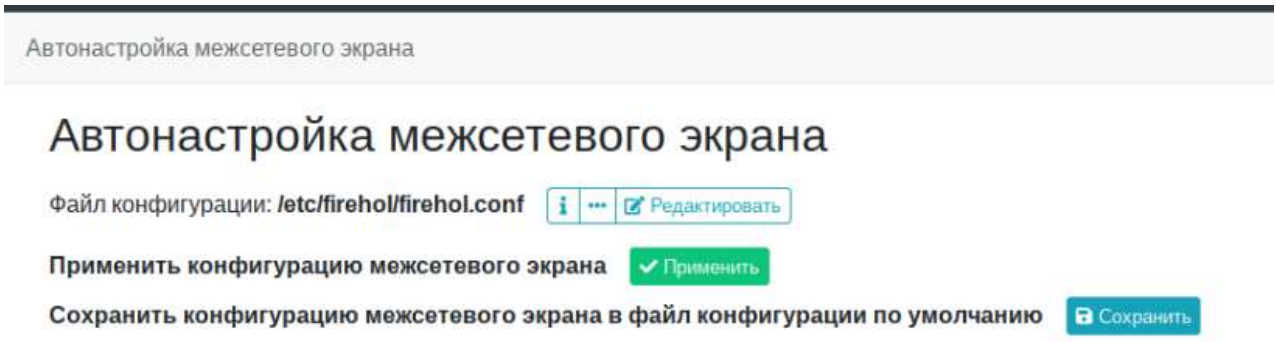


Рис. 148 – Окно автоматической настройки МЭ

10.7. Ограничение трафика

Для приоритизации информационных потоков используется инструмент FireQOS, подробнее о его возможностях приведено в п. 10.7.1.

10.7.1. FireQOS

fireqos – инструмент формирования трафика.

Синтаксис:

```
fireqos CONFIGFILE [start | debug] [ - conf-arg . . . ]
fireqos { stop | clear_all_qos }
fireqos status [name [ dump [class]]]
fireqos { dump | tcpdump } name class [ tcpdump-arg . . . ]
fireqos { drops | overlimits | requeues } name
```

При запуске без аргументов, fireqos предоставит некоторую справку по использованию.

При задании файла в параметре *CONFIGFILE* fireqos будет использовать этот файл вместо `/etc/firehol/fireqos.conf` в качестве своей конфигурации.

Параметр *name* всегда ссылается на имя интерфейса из файла конфигурации. Параметр *class* всегда ссылается на именованный класс внутри именованного интерфейса.

Можно передавать аргументы для использования конфигурационным файлом, отделяя любые конфигурационные значения от остальных аргументов с помощью --. Аргументы доступны в конфигурации с использованием стандартного синтаксиса bash, например, $1, $2 и т. д.

10.7.1.1. Команды

`start; debug`

Активирует формирование трафика на всех интерфейсах, как указано в файле конфигурации. При вызове в качестве отладки FireQOS также выводятся все команлы tc, которые он выполняет.

`stop`

Удаляет весь сформированый трафик, применяемый FireQOS (не затрагивает QoS на других интерфейсах и IFB, используемые другими инструментами).

`clear_all_qos`

Удаляет весь сформированый трафик на всех сетевых интерфейсах и удаляет все устройства IFB из системы, даже те, которые применяются другими инструментами.

`status`

Показывает активное использование указанного интерфейса. FireQOS покажет скорость трафика по всем классам, прибавляя по одной строке в секунду (аналогично vmstat, iostat и т. д.)

Если указан `dump`, tcpdump выгружает трафик в данном классе интерфейса.

`tcpdump; dump`

FireQOS временно зеркалирует трафик любого конечного класса в устройство IFB. Затем он запускает tcpdump на этом интерфейсе, чтобы выгрузить трафик на консоль.

Можно добавить любые параметры tcpdump в командную строку (чтобы сбросить трафик в файл, сопоставить подмножество трафика и т. д.), например:

```
fireqos tcpdump adsl-in voip -n
```

запустит tcpdump всего трафика на adsl-in интерфейсе, в классе voip.

параметр `-n` является параметром tcpdump.

Примечание. Когда FireQOS работает в режиме tcpdump, он блокирует сам себя и не запускается параллельно с другим FireQOS, изменяющим QoS, или сбрасывающим другой трафик. Это связано с тем, что FireQOS резервирует устройство ifb0 для мониторинга. Если бы двум процессам FireQOS было разрешено

выполнять tcpdump параллельно, то дампы были бы неправильными. Поэтому он блокирует сам себя, чтобы предотвратить подобный случай.

`drops`

Показывает количество пакетов, отбрасываемых в секунду для каждого класса для указанного интерфейса.

`overlimits`

Показывает задержку пакетов в секунду для каждого класса для указанного интерфейса.

`requeues`

Показывают пакеты, помещаемые в очередь в секунду для каждого класса для указанного интерфейса.

10.7.1.2. Конфигурация

Конфигурационный файл FireQOS по умолчанию – `/etc/firehol/fireqos.conf`.

Этот файл определяет формирование трафика, которое будет применяться fireqos.

Конфигурация состоит из нескольких определений входного и выходного интерфейсов (см. п. 10.7.1.4). Каждый интерфейс может определять любое количество (необязательно вложенных) классов (см. п. 10.7.1.3), которые формируют трафик, которому они соответствуют (см. п. 10.7.1.5).

10.7.1.2.1. Единицы измерения скорости

В FireQOS скорость может быть выражена в следующих единицах измерения:

`#bps` # байт в секунду

`#kbps; #Kbps` # Кбайт в секунду

`#mbps; #Mbps` # Мбайт в секунду

`#gbps; #Gbps` # Гбайт в секунду

`#bit` # бит в секунду

`#kbit; #Kbit;` # # Кбит в секунду (по умолчанию)

`#mbit; #Mbit` # Мбит в секунду

`#gbit; #Gbit` # Гбит в секунду

`#%` в классе используется этот процент от скорости охвата.

Примечание. Значение по умолчанию, kbit, отличается от tc, которое предполагает количество байт в секунду, если единица измерения не указана.

10.7.1.2.2. Пример

В этом примере используются операторы сравнения.

```
     # incoming traffic from my ADSL router
     interface eth2 adsl-in input rate 10500kbit adsl remote pppoe-llc
     class voip commit 100kbit pfifo
     match udp ports 5060,10000:10100 # asterisk sip and rtp match udp
ports 16393:16402 # apple facetime
     class realtime commit 10% match tcp port 22,1195:1198,1753 # ssh,
openvpn, pptp match udp port 53 # dns match proto GRE match icmp match
tcp syn match tcp ack
     class clients commit 10%
     match tcp port 20,21,25,80,143,443,465,873,993 # mail, web, ftp,
etc
     # unmatched traffic goes here ('default' is a special name) class
default max 90%
     # I define torrents beneath the default class, so they slow
     # down when the default class is willing to get bandwidth class
torrents max 90%
     match port 51414 # my torrent client
     # outgoing traffic to my ADSL router
     interface eth2 adsl-out output rate 800kbit adsl remote pppoe-llc
     class voip commit 100kbit pfifo
     match udp ports 5060,10000:10100 # asterisk sip and rtp match udp
ports 16393:16402 # apple facetime
     class realtime commit 10% match tcp port 22,1195:1198,1753 # ssh,
openvpn, pptp match udp port 53 # dns match proto GRE match icmp match
tcp syn match tcp ack
     class clients commit 10%
     match tcp port 20,21,25,80,143,443,465,873,993 # mail, web, ftp,
etc
     # unmatched traffic goes here ('default' is a special name) class
default max 90%
```

```
    # I define torrents beneath the default class, so they slow
    # down when the default class is willing to get bandwidth class
torrents max 90%
    match port 51414 # my torrent client
```

В этом примере используются настройки сервера/клиента в двунаправленном интерфейсе. Конечно, также могут быть указаны операторы сравнения. FireQOS создаст из этого два интерфейса: world-in и world-out.

```
    DEVICE=dsl0
    INPUT_SPEED="12000kbit" OUTPUT_SPEED="800kbit"
    LINKTYPE="adsl local pppoe-llc"
    # a few service definitions
    # all the rest that are used in this example
    #  are  defined  by  FireQOS  server_netdata_ports="tcp/19999"
server_rtp_ports="udp/10000:10100"
server_openvpn_ports="any/1195:1198"
server_mytorrent_ports="any/60000"
    server_mytorrenttransfers_ports="any/60001:64999"
server_myssh_ports="tcp/2222"
    # League Of Legends game (yes! I have kids)
    server_lol_ports="udp/5000:5500
tcp/8393:8400,2099,5223,5222,8088"
    interface  $DEVICE  world  bidirectional  $LINKTYPE  input  rate
$INPUT_SPEED output rate $OUTPU
    class voip commit 100kbit pfifo
    server sip client sip server rtp client stun
    class interactive input commit 20% output commit 10%
    server icmp limit 50%
    server dns client dns
    server ssh client ssh
    server myssh client myssh client teamviewer client lol
    class chat input commit 1000kbit output commit 30%
    client facetime
    server hangouts client hangouts
    client gtalk client jabber
```

```
     class vpns input commit 20% output commit 30%
     server pptp server GRE server openvpn
     class servers server netdata server http
     # a class group to favor tcp handshake over transfers class group
surfing prio keep commit 5%
     client surfing client rsync
     class synacks match tcp syn match tcp ack
     class group end
     class synacks commit 5%
     match tcp syn match tcp ack
     class default
     class background commit 4%
     client torrents server mytorrent server mytorrenttransfers
```

10.7.1.3. fireqos-class

fireqos-class – определяет класс трафика.

Синтаксис:

```
{class|class4|class6|class46} [group] name [optional-class-params]
{class|class4|class6|class46} group end
```

Существует также необязательный параметр соответствия, называемый class (см. п. 10.7.1.6.3.3).

Запись class наследует версию IPv4/IPv6 от своего встроенного интерфейса (см. п. 10.7.1.4).

Запись class4 включает в себя только трафик IPv4 в этом классе.

Запись class6 включает в себя только трафик IPv6 в этом классе.

Запись class46 включает в себя трафик как IPv4, так и IPv6 в этом классе. Фактический трафик сопоставляется классу, который определяется путем добавления сравнений (см. п. 10.7.1.5).

Последовательность, в которой классы отображены в конфигурации, определяет их приоритет. Первый класс – самый важный. Если нет иных ограничений, он получит всю доступную полосу пропускания, если это необходимо.

Второй класс менее важен, чем первый, третий еще менее важен, чем второй, и т. д. Поэтому располагайте классы в порядке их важности.

Классам может быть явно присвоен приоритет с помощью параметра prio (см. п. 10.7.1.6.2).

Примечание. Базовый Linux qdisc, используемый FireQOS, HTB, поддерживает только 8 приоритетов, от 0 до 7. Поэтому, если используете более 8 приоритетов, все последующие получат одинаковый приоритет (prio 7).

Все классы в FireQOS совместно используют полосу пропускания интерфейса. Однако у каждого класса есть фиксированная скорость (минимальная гарантированная скорость, которую он получит, если потребуется) и предел (максимальная скорость, которую может достичь этот класс, при условии наличия свободных мощностей и даже при наличии свободных мест).

Классы могут быть вложены на любой уровень с помощью синтаксиса class group.

По умолчанию FireQOS создает вложенные классы как классы, непосредственно присоединенные к своему родительскому классу. Таким образом, вложенность не приводит к каким-либо задержкам.

FireQOS также может эмулировать новое оборудование на уровне группового класса. Чтобы выполнить вложенность аппаратной эмуляции, добавьте определение канального уровня (ethernet, adsl, atm и т.д.) или просто mtu для класса group. FireQOS создаст qdisc внутри класса, где будут назначены параметры канального уровня, и дочерние классы будут присоединены к этому qdisc. Что добавляет некоторую задержку к пакетам дочерних классов, но позволяет эмулировать новое оборудование. Параметры канального уровня приведены в п. 10.7.1.6.2.

Существует специальный класс, называемый default. Классы по умолчанию могут быть заданы явно в файле конфигурации. Если они не найдены в конфигурации, FireQOS добавит по одному в конце каждого интерфейса или группы классов.

10.7.1.3.1. Параметры

`group`

Классы можно вкладывать в группы, используя `group`. Сгруппированные классы необходимо закрыть командой `class group end`. Группы классов также могут быть вложенными.

`name`

Это имя из одного слова для класса, которое используется для отображения информации о состоянии.

`optional-class-params`

Набор необязательных параметров класса, применяемых к этому классу.

Следующие необязательные параметры класса наследуются от интерфейса, в котором находится класс:

- ceil;
- burst;
- cburst;
- quantum;
- qdisc.

Если определить один из параметров на уровне интерфейса, то все классы в интерфейсе получат это значение по умолчанию. Эти значения могут быть переопределены через параметр в классе. Также наследование работает и с группами классов.

Необязательные параметры класса, отсутствующие в приведенном выше списке, не наследуются от интерфейсов.

FireQOS по умолчанию выделяет 1/100 родительской пропускной способности каждому классу. Что может быть переопределено для каждого класса, через добавление фиксации в классе или добавив минимальную скорость в родительский класс.

10.7.1.3.2. Примеры

Чтобы создать вложенный класс, вызываемый серверами, содержащий http и smtp:

```
interface eth0 lan input rate 1Gbit class voip commit 1Mbit
match udp ports 5060,10000:10100
```

```
     class group servers commit 50% # define the parent class
          match tcp # apply to all child classes
          class mail commit 50%    # 50% of parent ('servers')
          match port 25  # matches within parent ('servers')
     class web commit 50%
     match port 80
     class group end     #  end  the  group  'servers'  class  streaming
commit 30%
```

Чтобы создать вложенный класс, эмулирующий модем ADSL:

```
     interface eth0 lan output rate 1Gbit ethernet class lan
     match src 192.168.0.0/24 # LAN traffic
     class group adsl rate 10Mbit ceil 10Mbit adsl remote pppoe-llc
     match all # all non-lan traffic in this emulated hardware group
     class voip # class within adsl match udp port 5060
     class web # class within adsl
     match tcp port 80,443 class group end
```

10.7.1.4. fireqos-interface

fireqos-interface – создает определение интерфейса.

Синтаксис:

`{ interface | interface4 }` *`device name direction`* `[`*`optional-class-params`*`] { rate | commit | min }` *`speed`* `interface46 ... interface6 ...`

При записи interface или interface4 правила формирования трафика применяются только к трафику IPv4.

При записи interface6 правила формирования трафика применяются только к трафику IPv6.

При записи interface46 правила формирования трафика применяются как к трафику IPv4, так и к IPv6. Фактическое поведение класса в формировании трафика определяется путем добавления классов (см. п. 10.7.1.3).

П р и м е ч а н и е . Чтобы добиться наилучших результатов при формировании входящего трафика, не следует использовать 100% доступную пропускную способность на уровне интерфейса. Если использовать все, что есть, то при 100% использовании канала соседние маршрутизаторы начнут ставить пакеты в очередь. Это разрушит расстановку приоритетов. Вместо этого используйте 85% или 90%.

10.7.1.4.1. Параметры

`device`

Это имя интерфейса, отображаемое командой `ip link show` (например, eth0, ppp1 и т. п.).

`name`

Это имя из одного слова для этого интерфейса, которое используется для последующего получения информации о состоянии.

`direction`

Если задано значение input, формируется трафик, поступающий на интерфейс.

Если установлено значение output, формируется трафик, выходящий из интерфейса. Если установлен двунаправленный трафик, можно формировать как входной, так и выходной трафик. Если требуется различать входные и выходные параметры для каждого оператора в интерфейсе, можно добавить к ним префикс ввода или вывода следующим образом:

```
interface eth0 lan bidirectional ... class voip input commit 1Mbit output commit 2Mbit ...
```

`option-class-params`

Список необязательных параметров класса, которые можно применить к интерфейсу, см. в п. 10.7.1.6.2.

`speed`

Для интерфейса фиксированная скорость должна быть указана с помощью этой опции. Скорость может быть выражена в любых единицах, описанных в п. 10.7.1.2.

10.7.1.4.2. Примеры

Создание политики input на eth0, со скоростью до 1 Гбит трафика:

```
interface eth0 lan-in input rate 1Gbit
```

10.7.1.5. fireqos-match

fireqos-match – сравнение трафика QOS.

Синтаксис:

`{match|match4|match6|match46}` *`optional-match-params`*

При записи match версия IPv4/IPv6 наследуется от включающего его класса (см. п. 10.7.1.3).

Запись match4 означает сравнение только с трафиком IPv4.

Запись match6 означает сравнение только с трафиком IPv6.

Запись match46 означает сравнение как с трафиком IPv4, так и IPv6.

Можно добавить в конфигурацию FireQOS столько операторов сравнения, сколько захотите. Они присваивают трафик классу: по умолчанию тому классу, после которого они объявлены. Последовательность сравнений в конфигурации определяет их приоритет: первому присваивается приоритет 10, а для каждого последующего добавляется 10 (10, 20, 30, ...).

Приоритет сравнений может быть назначен явно с помощью параметра prio (см. п. 10.7.1.6.3.12).

Если один оператор сравнения генерирует несколько операторов фильтра tc, все фильтры, созданные одним оператором, будут иметь один и тот же приоритет.

П р и м е ч а н и е . Правила сравнения привязываются к родительскому классу, в котором они появляются. В конфигурации они записываются под классом, но на самом деле они привязаны к родительскому классу, так что они классифицируют трафик родительского элемента, которому они сопоставляют класс.

Также возможно сгруппировать все операторы сравнения под классами. Что позволяет расположить их в предпочтительном порядке без необходимости использования каких-либо явных параметров prio. Однако в этом случае каждый оператор сравнения должен указывать, к какому классу он относит соответствующие пакеты, используя параметр class. См. п. 10.7.1.6.3.3 и примеры ниже.

Также можно описать операторы клиента и сервера, как это позволяет FireHOL, с теми же определениями служб. Однако для FireQOS клиентские порты игнорируются. Операторы сервера соответствуют портам сервера, а операторы клиента соответствуют портам сервера на удаленной стороне.

Пример:

```
server_myrtp_ports="10000:10100"
interface eth0 lan bidirectional rate 1Gbit
```

```
class voip server sip client sip server myrtp
class dns server dns
class mail
serv class dns server dns
class mail
server smtp
```

10.7.1.5.1. Параметры

`optional-match-params`

Набор необязательных параметров сравнения (см. п. 10.7.1.6.3).

10.7.1.5.2. Примеры

Проверка трафика с классами:

```
interface eth0 lan output rate 1Gbit class voip match udp ports 5060,10000:10100
class dns match udp port 53
class mail
match tcp port 25
```

Сравнение разделяет и явно распределяет трафик по классам (обратите внимание: без параметров класса весь трафик будет классифицирован как «почта»):

```
interface eth0 lan output rate 1Gbit
class voip class dns class mail
match udp ports 5060,10000:10100 class voip match tcp port 25 class mail match tcp port 80 class web
```

10.7.1.6. Необязательные параметры

10.7.1.6.1. fireqos-params

fireqos-params – общие параметры класса/сравнения.

Синтаксис:

`prio priority`

Имена некоторых необязательных параметров одинаковы как для класса, так и для сравнения.

`prio`

Версию класса см. в п. 10.7.1.6.2.6.

Версию сравнения см. в п.10.7.1.6.3.12.

`priority`

Версию класса см. в п. 10.7.1.6.3.3.

Версию сравнения см. в п.10.7.1.6.2.17.

10.7.1.6.2. fireqos-params-class

fireqos-params-class – необязательные параметры класса.

Синтаксис:

```
rate | commit | min speed
ceil | max speed minrate
speed
{ qdisc qdisc-name | pfifo|bfifo|sfq|fq_codel|codel|none } [options "qdisc-options"]
prio { 0..7 | keep | last }
{ linklayer linklayer-name } | { adsl {local|remote} encapsulation } | ethernet | atm
mtu bytes
mpu bytes
tsize size overhead
bytes r2q factor
burst bytes cburst
bytes quantum
bytes priority |
balanced input |
output
```

Все параметры применимы к операторам interface и class.

Единицы скорости определены в п. 10.7.1.2.

10.7.1.6.2.1. input, output

Для двунаправленных интерфейсов input, output определяют направление, в котором применяются следующие за ним параметры.

Затрагиваются только следующие параметры, все остальные применяются как к входу (input), так и к выходу(output):

- minrate;

- rate, min, commit;

- ceil, max.

Если что-то из вышеперечисленного не определено ни для входа, ни для выхода, будет использоваться значение по умолчанию.

10.7.1.6.2.2. rate, min, commit

Когда классу предоставляется фиксированная скорость, это значит, что полоса пропускания будет предоставлена классу, когда он будет в ней нуждаться. Если классу не нужна полоса пропускания, она будет доступна для использования любым другим классом.

Для интерфейсов должна быть определена скорость.

Для классов rate по умолчанию равен 1/100 от пропускной способности интерфейса.

10.7.1.6.2.3. ceil, max

Определяет максимальную скорость, которую может использовать класс. Даже при наличии доступной пропускной способности класс не превысит свою максимальную скорость.

Для интерфейсов значением по умолчанию является скорость интерфейса. Для классов значением по умолчанию является ceil интерфейса.

10.7.1.6.2.4. minrate

Определяет фиксированную скорость по умолчанию для всех классов, для которых в файле конфигурации не указана скорость. Это вызывает перерасчет tc r2q.

Если минимальная скорость не указана, FireQOS присваивает значение по умолчанию, равное 1/100 скорости интерфейса.

10.7.1.6.2.5. qdisc qdisc-name, pfifo, bfifo, sfq, fq_codel, codel, none

Qdisc определяет метод распределения пропускной способности класса по его сокетам. Он применяется внутри самого класса и полезен в случаях, когда класс становится переполненным.

Qdisc полезен только при применении к классу. Его можно указать на уровне интерфейса, чтобы установить значение по умолчанию для всех включенных классов.

Чтобы передать параметры qdisc, можно указать их через переменную среды или явно для каждого класса.

Установите переменную `FIREQOS_DEFAULT_QDISC_OPTIONS_qdiscname` в файле конфигурации. Например, для sfq:

```
FIREQOS_DEFAULT_QDISC_OPTIONS_sfq="perturb 10 quantum 2000".
```

Используя эту переменную, каждый sfq получит эти параметры по умолчанию. Но все равно можно переопределить это, указав явные параметры для отдельных qdiscs, например, чтобы добавить некоторые параметры sfq, нужно написать:

```
class classname sfq options "perturb 10 quantum 2000"
```

Ключевое слово options должно появляться сразу после имени qdisc.

10.7.1.6.2.6. prio (class)

Существует также параметр сравнения prio, см. п. 10.7.1.6.3.12.

HTB поддерживает 8 приоритетов, от 0 до 7. Любому числу, меньшему 0, будет присвоен приоритет 0. Любое число выше 7 будет иметь приоритет 7.

По умолчанию FireQOS присваивает первому классу приоритет 0 и увеличивает это число на 1 для каждого класса, с которым он сталкивается в конфигурационном файле. Если классов больше 8, то все классы после 8-го получат приоритет 7. В сбалансированном режиме (см. п. 10.7.1.6.2.17) все классы по умолчанию получат приоритет 4.

FireQOS переопределяет приоритеты для каждого интерфейса и группы классов.

Определение приоритета класса зависит от распределения полосы пропускания между классами. Классы с более высокими приоритетами (более низким prio) получат всю свободную полосу пропускания. Классы с одинаковым приоритетом получат процент от свободной полосы пропускания, пропорциональный их фиксированным ставкам.

Ключевые слова keep и last заставят класс использовать приоритет класса чуть выше / ниже, чем его. Чтобы два последовательных класса имели одинаковый приоритет, просто добавьте prio keep ко второму.

10.7.1.6.2.7. linklayer – linklayer-name, ethernet, atm

linklayer – определение канального уровня, может быть задано только для интерфейсов, используется ядром для вычисления служебной информации пакетов.

10.7.1.6.2.8. adsl

adsl – это специальный параметр канального уровня, который автоматически вычисляет ATM overheads для подключения. Локально используется, когда Linux работает под управлением PPPoE, удаленный используется, когда на маршрутизаторе запущен PPPoE.

П р и м е ч а н и е . Этот параметр позволяет знать о пакетах, которые содержат ethernet заголовок, например, инкапсуляцию (все метки в одной строке являются псевдонимами):

- IPoA-VC/Mux или ipoa-vc mux, ipoa-vc или ipod-mix;
- IPoA-LLC/SNAP или ipoa-llc snap или ipoa-llc или ipoa-snap;
- Bridged-VC/Mux или bridged-vc mux или bridged-vc или bridged-mux;
- Bridged-LLC/SNAP или bridged-llc snap или bridged-llc или bridged-snap;
- PPPoA-VC/Mux или pppoa-vcmux или pppoa-vc или pppoa-mux;
- Pppoe-OOO/SNAP или pppoa-OOO snap или pppoa-ooo или pppoa-snap;
- PPPoE-VC/Mux или pppoe-vcmux или pppoe-vc или pppoe-mux;
- PPPoE-LLC/SNAP или pppoe-llc snap, или pppoe-llc, или pppoe-snap, если adsl-маршрутизатор может предоставить mtu, то добавьте также параметр mtu (см. п. 10.7.1.6.2.9).

10.7.1.6.2.9. mtu

Определяет MTU интерфейса в байтах.

FireQOS запросит интерфейс, чтобы найти его MTU. Можно переопределить это поведение, присвоив этот параметр классу или интерфейсу.

10.7.1.6.2.10. mpu

Определяет MPU интерфейса в байтах.

FireQOS не устанавливает значения по умолчанию, но можно установить свой собственный, используя этот параметр.

10.7.1.6.2.11. tsize

FireQOS не устанавливает размер по умолчанию, но можно установить свой собственный, если использовать этот параметр.

10.7.1.6.2.12. overhead

FireQOS автоматически вычисляет overhead в байтах для ADSL. Для всех остальных технологий можно указать overhead в конфигурационном файле.

10.7.1.6.2.13. r2q

FireQOS вычисляет правильный коэффициент r2q, так, что возможно управление скоростями с шагом в 1/100 от скорости интерфейса (если это возможно).

Примечание. Для HTB этот параметр игнорируется, когда задано значение quantum. По умолчанию FireQOS устанавливает quantum для интерфейса MTU, поэтому r2q, игнорируется ядром.

10.7.1.6.2.14. burst

burst определяет количество байт, которые будут отправлены одновременно с максимальной скоростью, когда классу разрешено отправлять трафик. Это что-то вроде ‘транспортной единицы’. Классу разрешается отправлять по крайней мере пакет байтов, прежде чем пытаться обслуживать любой другой класс.

Значение burst никогда не должно быть ниже, чем mtu интерфейса, а группы классов и интерфейсы никогда не должны иметь меньшее значение burst, чем их дочерние элементы. Если указан более высокий burst для дочернего класса, то это может привести к задержкам родительского класса (дочерний класс будет истощать родительский).

По умолчанию FireQOS позволяет ядру определять этот параметр путем вычисления наименьшего возможного значения. Минимальное значение зависит от скорости интерфейса и тактовой частоты процессора.

burst наследуется от интерфейсов к классам и от групповых классов к их подклассам. FireQOS не позволит установить burst в подклассе, более высоком, чем его родительский. Установка burst для подкласса выше, чем родительский, приведет к истощению родительского класса, который может привести к задержкам на срок до минуты, когда это произойдет. Чтобы эта проверка сработала, FireQOS использует только свою конфигурацию (он не спрашивает ядро, чтобы проверить,

как значение, указанное в файле конфигурации для подкласса, соотносится с фактическим значением его родительского класса).

10.7.1.6.2.15. cburst

cburst похож на burst, но с аппаратной скоростью (а не только с максимальной скоростью).

По умолчанию FireQOS позволяет ядру определять этот параметр.

burst наследуется от интерфейсов к классам и от групповых классов к их подклассам. FireQOS не позволит установить cburst в подклассе, более высоком по сравнению с его родительским классом. Установка cburst у подкласса выше, чем у его родительского, приведет к истощению родительского класса, который может привести к задержкам на срок до минуты, когда это произойдет. Чтобы эта проверка сработала, FireQOS использует только свою конфигурацию (он не спрашивает ядро, чтобы проверить, как значение, указанное в файле конфигурации для подкласса, соотносится с фактическим значением его родительского класса).

10.7.1.6.2.16. quantum

quantum определяет количество байт, которое классу разрешено отправлять одновременно, когда он заимствует свободную полосу пропускания у других классов. По умолчанию FireQOS устанавливает quantum в качестве mtu интерфейса.

quantum наследуется от интерфейсов к классам и от групповых классов к их подклассам.

10.7.1.6.2.17. priority, balanced

Эти параметры задают режим приоритета дочерних классов.

`priority`

Это режим по умолчанию, в котором FireQOS присваивает каждому классу дополнительный приоритет. В этом режиме первый класс принимает значение prio 0, второй – prio 1 и т. д. Когда класс имеет более высокий приоритет, чем другие (чем меньше число, тем выше приоритет), тот этот класс получит всю доступную свободную полосу пропускания, когда ему это понадобится. Резервная полоса пропускания

будет выделена классам с более низким приоритетом только в том случае, если классы с более высоким приоритетом в ней не нуждаются.

`balanced`

Сбалансированный режим присваивает prio 4 всем дочерним классам. Когда несколько классов имеют один и тот же prio, доступная свободная полоса пропускания распределяется между ними пропорционально их фиксированной скорости. Значение 4 можно переопределить, изменив значение `FIREQOS_BALANCED_PRIOR` в верхней части конфигурационного файла, на то, какое хотите назначить сбалансированному режиму для всех классов.

Режим приоритета может быть установлен в интерфейсах и группах классов. Эффект будет одинаков. Классы, которые определены как дочерние классы, по умолчанию получат значение родителя на основе заданного режима приоритета.

Эти параметры влияют только на приоритет по умолчанию, который будет назначен FireQOS. Значение по умолчанию используется только в том случае, если явно не используется параметр prio для класса.

Примечание. Существует также параметр сравнения prio, см. п. 10.7.1.6.3.12.

10.7.1.6.3. fireqos-params-match

fireqos-params-match – необязательные параметры сравнения.

Синтаксис:

```
at { root | name } class name syn|syns ack|acks
{ proto|protocol protocol [,protocol...] } |tcp|udp|icmp|gre|ipv6
{ tos | priority } tosid [,tosid...]
{ DSCP } classname [,classname...] mark mark [,mark...] connmark mark [,mark...] rawmark mark [,mark...] custommark name mark [,mark...]
{ port | ports } port[:range] [ ,port[:range]... ]
{ sport | sports } port[:range] [ ,port[:range]... ]
{ dport | dports } port[:range] [ ,port[:range]... ] { ip | net | host } net [,net...] src net [,net...]
dst net [,net...]
```

`{ srcmac | smac }` *mac* `{ dstmac | dmac }` *mac* `prio` *id* `input output custom` *'custom tc parameters'* `estimator` *interval decay* `police` *police* `insidegre`

Эти параметры применяются к операторам сравнения.

10.7.1.6.3.1. input, output

На двунаправленных интерфейсах input и output будут проверять текущее направление интерфейса. Если в сравнении input, но в интерфейсе output, то сравнение будет обратным. То же самое произойдет, если в сравнении output, а в интерфейсе input.

Изменяемыми параметрами являются:

- src и dst;

- sport и dport;

- src mac и dst mac.

Можно задать следующим образом:

```
interface dsl0 world bidirectional ...
class surfing ... match input sport 0:1023
```

В приведенном выше примере будет осуществлять сравнение sport 0:1023 на входном интерфейсе и автоматически dport 0:1023 на выходном интерфейсе.

10.7.1.6.3.2. at

По умолчанию сравнение привязывается к родительскому элементу его родительского класса. Например, если его родительским классом является класс непосредственно под интерфейсом, то сравнение привязывается к интерфейсу и сравнивается со всем трафиком интерфейса. Для вложенных классов дочерний класс присоединяется к родительскому классу и сравнивается со всем трафиком родительского класса.

С помощью параметра at можно добавить для сравнения к любому классу. Параметром name должно быть имя класса. Имя root привязывает сравнение к интерфейсу.

10.7.1.6.3.3. class

Определяет имя класса (name), который будет получать пакеты при совпадении. По умолчанию это имя класса, в котором отображается оператор match.

Примечание. Существует также определение класса для трафика, см. п. 10.7.1.3.

10.7.1.6.3.4. syn, syns

Сравнивает пакеты TCP SYN. Обратите внимание, что должен быть указан параметр tcp.

Если один и тот же оператор сравнения включает больше протоколов, чем TCP, то сравнение будет работать для пакетов TCP и будет автоматически проигнорировано для всех других протоколов. Например, syn игнорируется при генерации UDP-фильтра в приведенном ниже примере:

```
match tcp syn match proto
tcp,udp syn
```

10.7.1.6.3.5. ack, acks

То же, что и syn, но сравнивает небольшие TCP-пакеты с установленным битом ACK.

10.7.1.6.3.6. proto, protocol, tcp, udp, icmp, gre, ipv6

Используется для сравнения с протоколом в IP заголовке.

10.7.1.6.3.7. tos, priority

Проверка по полю TOS IPv4 или приоритет поля IPv6. TOS может быть значением/маской, любым форматом, который принимает tc, или одним из следующих:

- min-delay, minimize-delay, minimum-delay, low-delay, interactive;
- maximize-throughput, maximum-throughput, max-throughput, highthroughput, bulk;
- maximize-reliability, maximum-reliability, max-reliability, reliable;
- min-cost, minimize-cost, minimum-cost, low-cost, cheap, normal-service, normal.

Примечание. Существует также параметр класса, называемый prio, см. п. 10.7.1.6.2.6.

10.7.1.6.3.8. dscp

Проверка по значению DSCP в поле заголовка IP TOS. Имя класса (`classname`) должно быть одним из следующих:

- CS1, CS2, CS3, CS4, CS5, CS6, CS7;

- AF11, AF12, AF13;

- AF21, AF22, AF23;

- AF31, AF32, AF33;

- AF41, AF42, AF43;

- EF.

П р и м е ч а н и е . tc-filter поддерживает только параметры ToS. Вот почему в коде fireqos настроена таблица lookaside для преобразования значения DSCP в соответствующее ему значение TOS. Для получения дополнительной информации смотрите RFC 2474.

10.7.1.6.3.9. mark, connmark, custommark, rawmark

Проверка по меткам (MARK) iptables.

Соответствующие метки iptables не работают на интерфейсах входа. Их можно использовать только на выходе.

10.7.1.6.3.10. ports, sports, dports

Проверка портов в IP-заголовке. Для проверки портов источника и получателя задаются отдельные правила: dports соответствует портам назначения, sports соответствует портам источника.

10.7.1.6.3.11. ip, net, host, src, dst

Проверка пое IP-адресам в заголовке IP. IP, сеть и хост задают отдельные правила для проверки источника и получателя: src соответствует IP-адресам источника и dst – IP-адресам назначения.

П р и м е ч а н и е . Если классом проверки является IPv4, то можно использовать только IPv4-адреса. Для переопределения используйте match6 ... src/dst *IPV6_IP* Аналогично, если класс IPv6, то можно использовать только IPv6-адреса. Для переопределения используйте match4 ... src/dst *IPV4_IP*.

Можно смешивать IPv4 и IPv6 любым удобным способом. FireQOS поддерживает наследование, чтобы определить для каждого оператора, значение по умолчанию. Например:

```
     interface46 eth0 lan output rate 1Gbit # ipv4 and ipv6 enabled
class voip # класс ipv4 и ipv6, интерфейс и тот, и другой
     match udp port 53 # правило ipv4 и ipv6, поскольку оба классы
     match4 src 192.0.2.1 # правило только для ipv4
```

```
    2001:db8::1 # правило только для ipv6
    class4 realtime # класс только ipv4 проверяет src 198.51.100.1 #
правило только для ipv4, поскольку класс предназначен только для ipv4
    class6 servers # класс только ipv6 проверяет src 2001:db8::2 #
правило только для ipv6, поскольку класс предназначен только для ipv6
```

Чтобы преобразовать интерфейс IPv4 в IPv6, просто замените interface на interface6. Все правила в этом интерфейсе автоматически унаследуют новый протокол. Конечно, если используются IP-адреса для проверки пакетов, убедитесь, что они также являются IPv6-адресами.

10.7.1.6.3.12. prio (match)

П р и м е ч а н и е . Существует также параметр класса prio, см. п. 10.7.1.6.2.6.

Все операторы сравнения прикреплены к интерфейсу. Они перенаправляют трафик в свой класс, но на самом деле они выполняются для всех пакетов, которые покидают интерфейс (входные проверки на самом деле являются выходными проверками на устройстве IFB).

По умолчанию, приоритеты, выполняются в порядке, в котором отображаются в файле конфигурации, т.е. сначала выполняется первая проверка первого класса, затем остальные проверки в последовательности, в которой они отображаются, затем проверка второго класса и т.д.

Иногда необходимо контролировать порядок проверок. Например, если хотите, чтобы хосту 192.0.2.1 был присвоен первый класс, за исключением порта tcp/1234, которому должен быть присвоен второй класс, поэтому следующее не будет работать:

```
    interface eth0 lan output rate 1Gbit
    class high match host 192.0.2.1
    class low match host 192.0.2.1 port 1234 # никогда не будет проверен
```

В этом случае первой проверке присваивается приоритет 10, а второму – приоритет 20. Вторая проверка ничего не будет проверять, так как весь трафик для хоста уже проверяется в первой.

Установка явного приоритета позволяет изменить порядок, в котором выполняются проверки. FireQOS задает приоритет 10 первой проверке каждого

интерфейса, 20 – второй, 30 – третьей и т.д. Таким образом, значение по умолчанию равно 10-кратному порядковому номеру. Можно настроить prio на перезапись этого номера.

Чтобы принудительно выполнить вторую проверку перед первой, просто установите для нее более низкий приоритет. Например:

```
interface eth0 lan output rate 1Gbit
class high match host
192.0.2.1
class low
match host 192.0.2.1 port 1234 prio 1 # проверяет перед хостом
```

10.7.1.6.3.13. insidegre

При указании ключевого слово insidegre GRE (Generic Routing Encapsulation) пакет проверяется на инкапсулированную информацию в заголовке IP-пакета. insidegre доступен для:

- src;
- dst;
- protocol;
- port;
- tos;
- dscp.

Примеры:

```
interface eth0 world ... class surfing commit 128kbit ceil 1024kbit prio 7
match src 10.1.128.230 dst 8.8.8.8 insidegre match
protocol ospf insidegre match port 25 insidegre match
tos 3 insidegre match dscp ef insidegre
```

10.7.2. Приоритизация информационных потоков

Далее приведен пример настройки приоритизации.

Сначала пропишите интерфейс направления передачи трафика input и output.

Настройки производятся в файле `/etc/firehol/fireqos.conf`.

INPUT_SPEED – входящая скорость.

OUTPUT_SPEED – исходящая скорость.

В процентах указывается ширина канала для отдельных сервисов.

Пример:

```
    DEVICE=eno1
    INPUT_SPEED=95Mbit
    OUTPUT_SPEED=95Mbit
    LINKTYPE=ethernet
    interface $DEVICE world-in input rate $INPUT_SPEED $LINKTYPE
      ############### DNS, SSH, rsync, ping
############################
      class interactive commit 10%
       match tcp port 22
       match udp port 53
       match sport 873
       match proto GRE
       match icmp
    ############## VPN ##############################################

      class vpns commit 15%
        match sport 1701
        match sport 1194

    ############# Mail, web, ftp ####################################
      class surfing commit 40%
        match tcp sport 20,21,25,80,110,143,443,465,993
    ########### Torrents ############################################
      class torrents commit 1%
        match sports 6881:6999
        match sports 16384:65535 dports 16384:65535

    interface $DEVICE world-out output rate $OUTPUT_SPEED $LINKTYPE
    ################# DNS, SSH, rsync, ping ####################
      class interactive commit 10%
        match udp port 53
        match tcp port 22
        match dport 873
        match proto GRE
        match icmp
    ################# VPN ########################################
      class vpns commit 15%
        match dport 1701
        match dport 1194
    ################# Web, mail, ftp #############################
      class surfing commit 40%
        match tcp dport 20,21,25,80,110,143,443,465,993
    ########### Torrents ############################################
      class torrents commit 1%
        match dports 6881:6999
        match sports 16384:65535 dports 16384:65535
```

ВНИМАНИЕ!

При форматирование правил в конфигурационном файле `/etc/firehol/fireqos.conf` важно сохранить структуру, отступы и форматирование текста в соответствии с исходным текстом.

Для применения конфигурации после правки файла `/etc/firehol/fireqos.conf`, выполнить команды:

```
chkconfig fireqos on
service fireqos start
```

10.7.3. Ограничение трафика в графическом интерфейсе

Раздел «Сеть» (см. п. 5.3) → Подраздел «Ограничение трафика» предоставляет интерфейс для настройки ограничений пропускной способности сети через fireqos.

Перед началом использования следует добавить сети, клиенты которых подлежат контролю пропускной полосы, в «список сетей для вычисления classid». Также можно указать внутренние и внешние сети, для которых не будет действовать ограничение трафика.

Для идентификации клиентов «Ограничение трафика» использует IP-адрес. Ограничение задается размером в Кбит.

В данном подразделе выбирается файл конфигурации для ограничения трафика, который можно также просмотреть, отредактировать, выгрузить, узнать информацию о нем и осуществить применение настроек.

Системную службу ограничения трафика можно запускать автоматически при установленном флаге «Запускать при загрузке системы» (рис. 149).

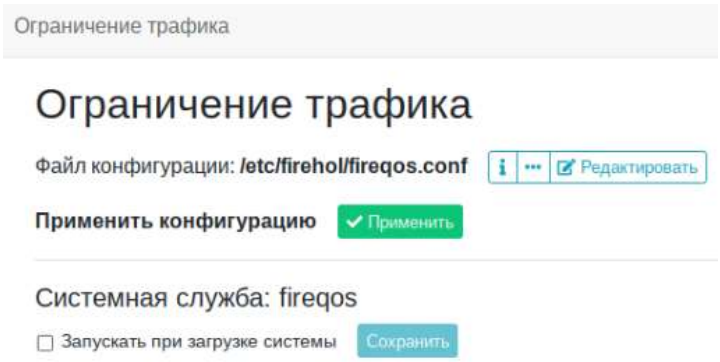


Рис. 149 – Окно ограничения трафика файла конфигурации

10.8. Сетевой трафик

Сетевой трафик – это объем информации, передаваемой через компьютерную сеть за определенный период времени. Количество трафика измеряется как в пакетах, так и в битах. Мониторинг и анализ трафика необходимы для того, чтобы более эффективно диагностировать и решать проблемы, исключая простой сетевых сервисов в течение длительного времени.

Вкладка «Статистика» подраздела «Сетевой трафик» представляет собой вывод данных NETFLOW: о временном интервале, размере, скорости передачи данных и др. Вся информация представлена в текстовом виде (рис. 150). В случае необходимости информацию можно обновить, нажав на соответствующую кнопку.

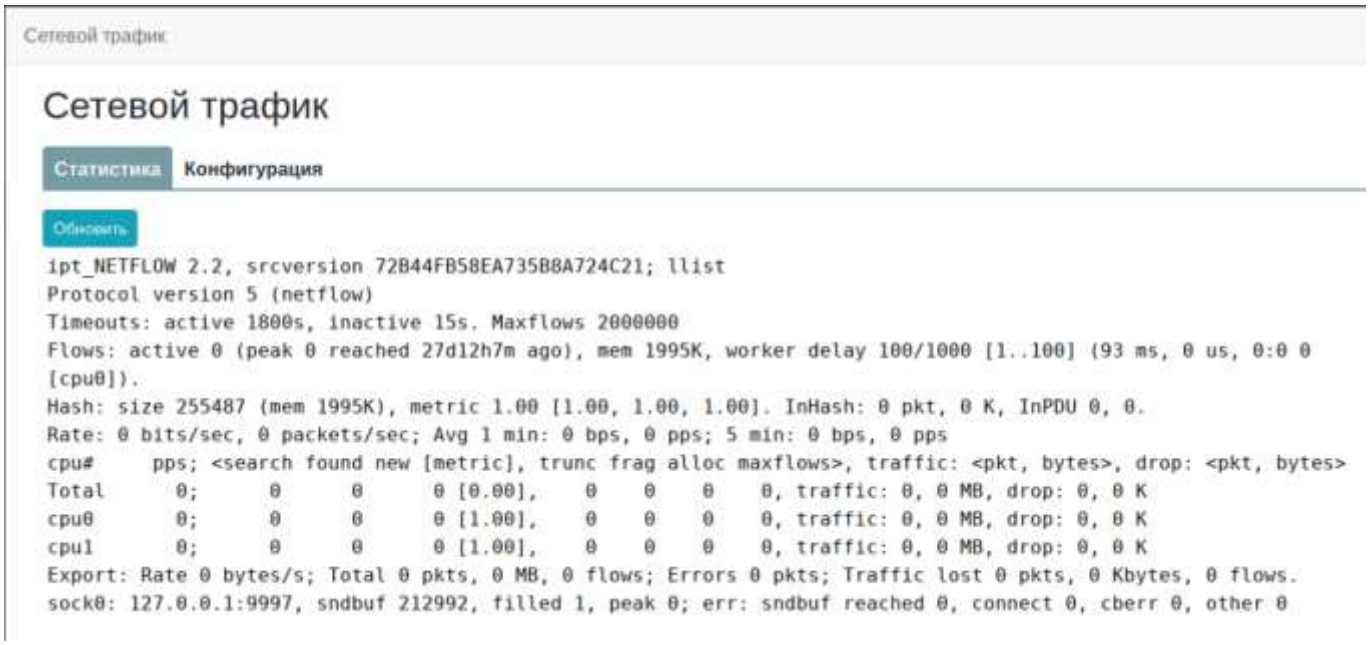


Рис. 150 – Окно статистики сетевого трафика

Вкладка «Конфигурация» используется для вывода информации по конкретной точке назначения (destination) путем ввода в соответствующее поле адреса назначения – внешний локальный адрес с добавленным номером служебного порта.

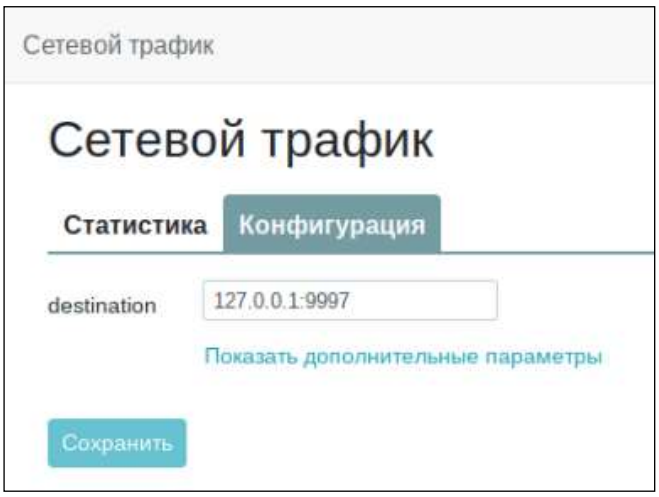


Рис. 151 – Окно конфигурации сетевого трафика

Для внесения изменений в дополнительные параметры для заданной конфигурации нажмите на строку «Показать дополнительные параметры» (рис. 152).

Среди параметров: протокол, активное/неактивное время, частота обновления, скорость, размер, количество ошибок (багов), максимальный поток, время развертывания и др.

После редактирования параметров для внесения изменений нажмите на кнопку «Сохранить».

Рис. 152 – Дополнительные параметры конфигурации

10.9. Статистика прокси-сервера

Подраздел предназначен для просмотра отчета об объеме полученных из внешней сети данных. В зависимости от режима аутентификации пользователей при доступе к прокси-серверу, записи отчета могут содержать как имена пользователей, так и адреса локальной сети. Текущая статистика отображается прокси-сервером в реальном времени и записывается в журнал для последующего анализа и формирования отчетов.

Для просмотра статистики прокси-сервера необходимо выбрать период для детализации в заданном интервале (рис. 153).

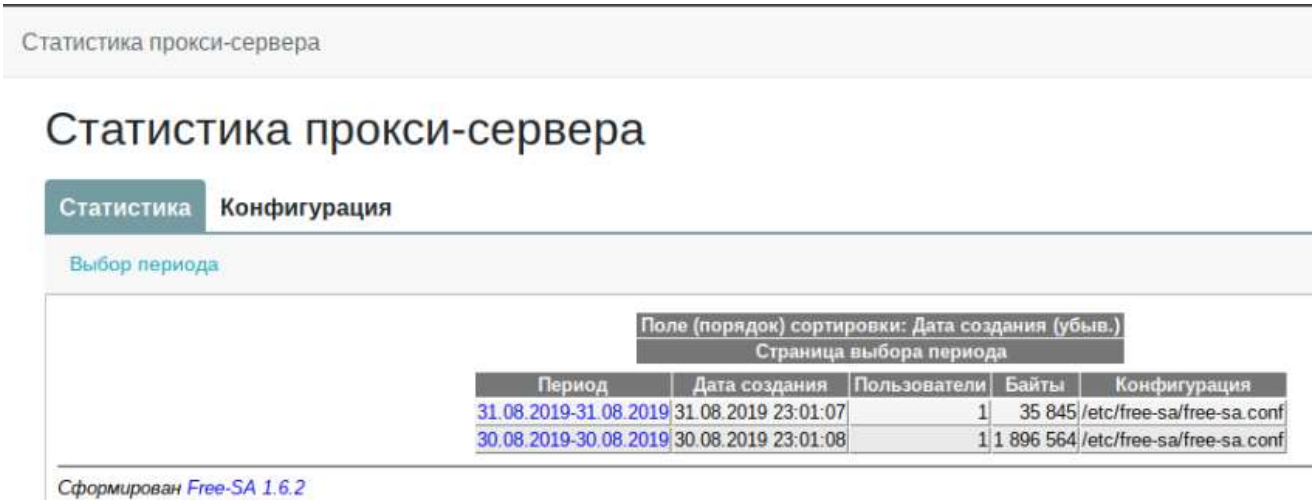


Рис. 153 – Окно статистики прокси-сервера

При нажатии на выбранный период появится информационное окно, в котором в виде списка отображаются параметры статистики (рис. 154).

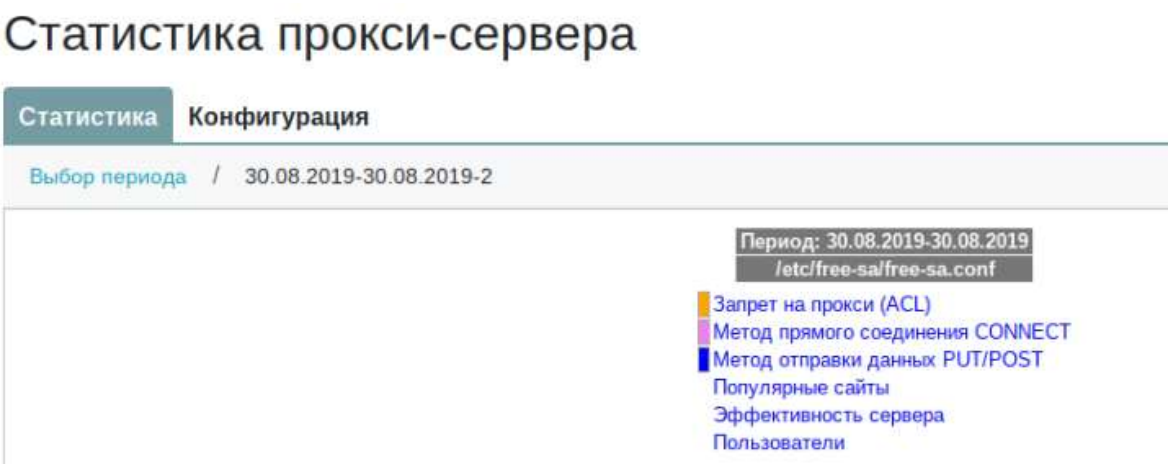


Рис. 154 – Окно просмотра статистики за выбранный период

Пример детализации отчета по выбранному параметру статистики приведен на рис. 155.

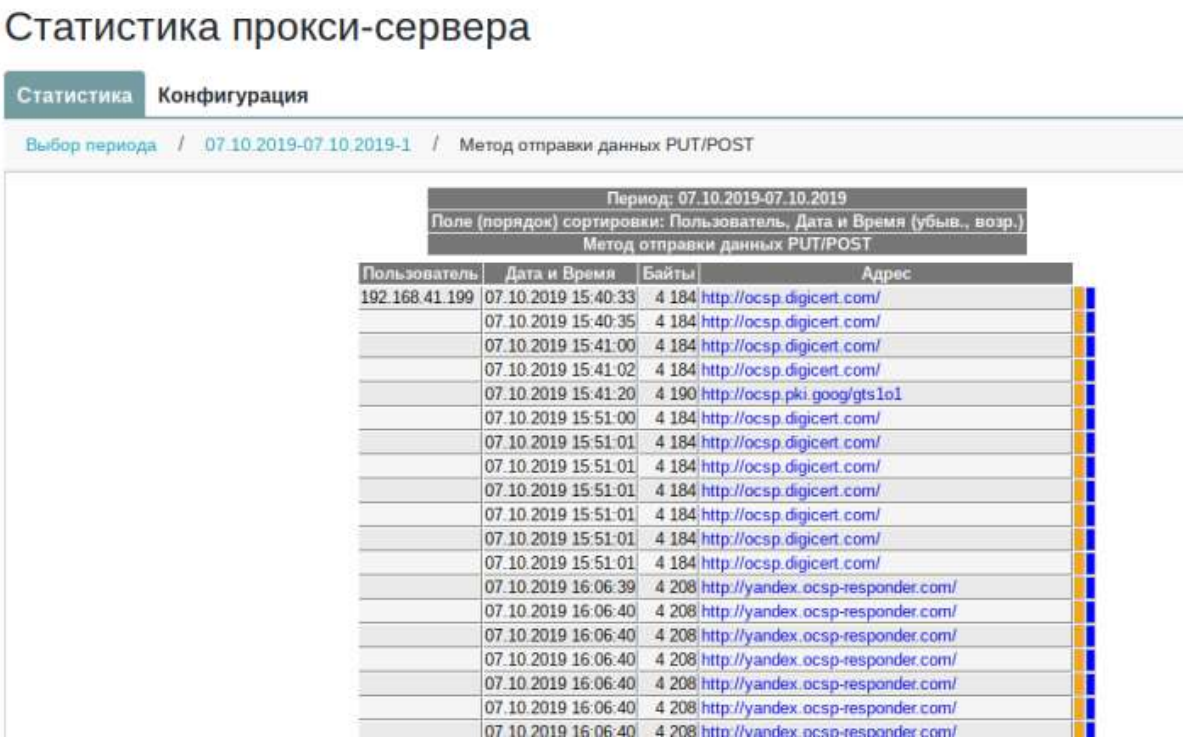


Рис. 155

Вкладка «Конфигурация» предоставляет возможность настройки конфигурационного файла формирования статистики прокси-сервера (рис. 156, рис. 157) (см. описание кнопок в п. 5.4).

Для изменения частоты сбора статистики отметьте флаг параметра «Автоматический сбор статистики каждый час» (рис. 156).

После внесения необходимых настроек нажмите на кнопку «Сохранить».

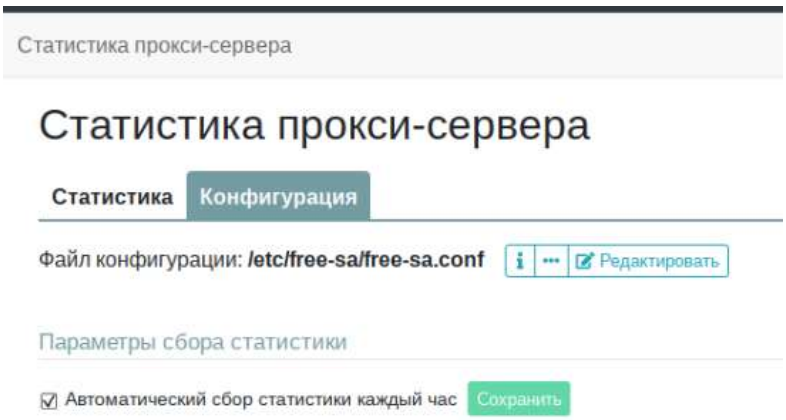


Рис. 156 – Окно конфигурации статистики прокси-сервера

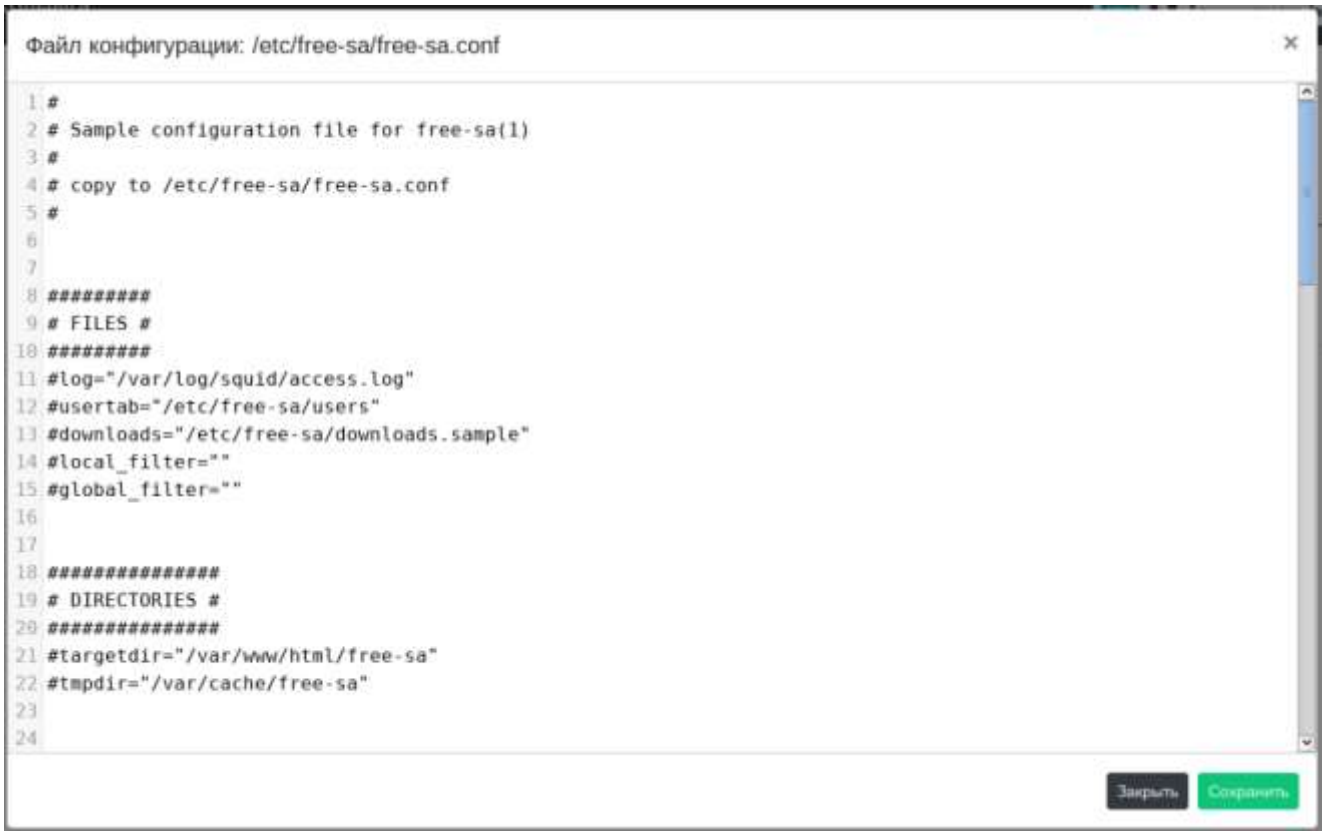


Рис. 157 – Окно просмотра файла конфигурации

10.10. Демон маршрутизации (bird)

BIRD (BIRD Internet Routing Daemon) – демон маршрутизации – программа, работающая в фоновом режиме, которая выполняет динамическую часть интернет-маршрутизации, то есть она связывается с другими маршрутизаторами, вычисляет таблицы маршрутизации и отправляет их ядру ОС, которое выполняет фактическую пересылку пакетов. BIRD предназначен для облегчения настройки связи между различными сетями, имеет расширяемую архитектуру и поддерживает различные протоколы маршрутизации, такие как: BGPv4, RIPv2, OSPFv2, OSPFv3 и виртуальный протокол Pipe для обмена маршрутами между различными таблицами маршрутизации. Для всех протоколов реализована работа с IPv6.

В подразделе «Демон маршрутизации (bird)» происходит управление файлом конфигурации работы службы BIRD (см. описание кнопок в п. 5.4) (рис. 158).

Пример просмотра окна файла конфигурации представлен на рис. 159.

Доступно изменение текущего состояния работы службы в ГИ. Службу можно остановить или запустить в ручном режиме. Для этого нужно выбрать из выпадающего списка необходимый параметр (рис. 158).

Для автоматического запуска службы при загрузке системы – установить флаг в соответствующем поле.

После внесения необходимых настроек нажмите на кнопку «Сохранить».

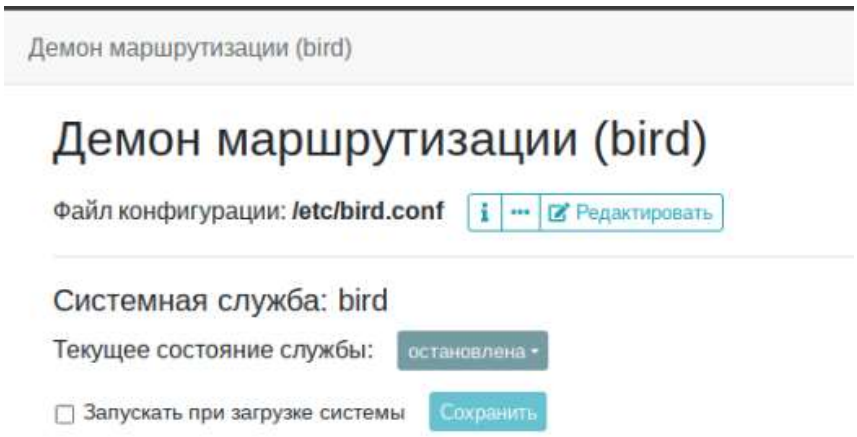


Рис. 158 – Окно системной службы BIRD

Рис. 159 – Окно информации о файле конфигурации системной службы BIRD

Пример настройки файла конфигурации `/etc/bird.conf`:

```
     #Установка формата времени для логирования
     timeformat base    iso long;
     timeformat log     iso long;
     timeformat protocol iso long;
     timeformat route   iso long;

     #Определение пути к лог-файлу
     log «/var/log/bird.log» all;
     log stderr all;

     ##### Описание настроек протоколов
# The direct protocol automatically generates device routes to
# all network interfaces. Can exist in as many instances as you wish
# if you want to populate multiple routing tables with device routes.
protocol direct {
interface «eth*», «*»; # Restrict network interfaces it works with
}

# This pseudo-protocol watches all interface up/down events.
protocol device {
scan time 10; # Scan interfaces every 10 seconds

}

# This pseudo-protocol performs synchronization between BIRD’s routing
# tables and the kernel. If your kernel supports multiple routing
tables
# (as Linux 2.2.x does), you can run multiple instances of the kernel

# protocol and synchronize different kernel tables with different BIRD
tables.
protocol kernel {
persist off; # Don’t remove routes on bird shutdown
scan time 20; # Scan kernel routing table every 20 seconds
import none; # Default is import all
export all; # Default is export none
}

     #Статический маршрут, необходимый для добавления маршрута
по умолчанию в таблицу master
     protocol static static1 {
     route 0.0.0.0/0 via «lo»;
     }
```

```
     #Статический маршрут для добавления префикса в таблицу
     protocol static static2 {
     preference 253;
     route zzz.zzz.zzz.0/23 via «lo»;
     }
     ###Конфигурация протокола OSPF:
     #Необходимо объявление ospf соседям маршрут по умолчанию и маршруты типа directly, а в таблицу master необходимо передать маршруты, полученные от ospf соседей, кроме маршрута по умолчанию
     #для этого используются соответствующие фильтры:
     filter import_OSPF {
     if ( source = RTS_OSPF && net != 0.0.0.0/0 ) then {
     print «net accepted:», net;
     accept;
     }

     reject;

     }

     filter export_OSPF {
     #Передача маршрутов connected
     if ( source = RTS_DEVICE ) then {
     print «net accepted:», net;
     ospf_metric2 = 20;
     accept;

     }
     #Передача маршрута по умолчанию, т. к. он «завернут» на интерфейс loopback, то необходимо использовать конструкцию RTS_STATIC_DEVICE, а не RTS_STATIC
     if ( source = RTS_STATIC_DEVICE && net = 0.0.0.0/0 ) then {
     print «net accepted:», net;
     ospf_metric2 = 5;
     accept;
     }

     reject;
     }

     #Конфигурация протокола OSPF
     protocol ospf ospfAS65000 {
     router id 87.244.8.18;
     export filter export_OSPF;
     import filter import_OSPF;
     area 0.0.0.0 {
```

```
     interface «eth1.4» {
     hello 10;
     retransmit 5;
     cost 10;

     transmit delay 1;

     dead count 4;
     wait 40;
     type broadcast;
     priority 0;
     };

     };
     }
     ###

     ###Настройка BGP
     #Определение функции, с помощью которой будут блокироваться сети RFC1918, по умолчанию, префиксы короче /24, префиксы /32, /0 и /7, если они вдруг появятся от bgp соседей
     function avoid_nonexist()
     #Описание префикс-листа, по которому будет происходить проверка маршрутов
     prefix set nonexist;

     {
     nonexist = [ 169.254.0.0/16+, 172.16.0.0/12+, 192.168.0.0/16+, 10.0.0.0/8+, 224.0.0.0/4+, 240.0.0.0/4+, 0.0.0.0/32-, 0.0.0.0/0{25,32}, 0.0.0.0/0{0,7} ];
     if net ~ nonexist then return false;
     return true;
     }
     #Определение функции, с помощью которой будут фильтроваться маршруты из AS
     function pref_from_myasset()

     prefix set pref_from_65000;
     {
     pref_from_65000 = [ zzz.zzz.zzz.0/23 ];
     if net ~ pref_from_65000 then return true;

     return false;
     }
```

```
###Основной провайдер

##Фильтр маршрутов и установка local preference
filter prov65002in {
if avoid_nonexist() then
{
bgp_local_pref = 340; #устанавливка local preference
accept;
}

reject;
}

filter prov65002out {
if pref_from_myasset() then
{
bgp_community = -empty-; #не отправлять никаких community
bgp_path.prepend(65000); #добавление prepend
accept;
}

reject;
}

protocol bgp bgpAS65002 {
table master;
router id yyy.yyy.yyy.2;
description «AS65002»;
local as 65000;
neighbor yyy.yyy.yyy.1 as 65002;
hold time 240;
startup hold time 240;
connect retry time 120;
keepalive time 80;
start delay time 5;
error wait time 60, 300;
error forget time 300;
next hop self;
path metric 1;
default bgp_med 0;
source address yyy.yyy.yyy.2;
export filter prov65002in;
import filter prov65002out;

}
###
```

```
     ###Резервный провайдер
     ##Фильтр маршрутов, установка local preference для маршрутов,
помеченных community на стороне второго провайдера
     filter prov65001in {

     if (65001,1400) ~ bgp_community then
     {
     bgp_local_pref = 400; # local preference для маршрутов отмеченных
community 65001:400
     accept;
     }
     bgp_local_pref = 330; #local preference для остальных маршрутов
     if avoid_nonexist() then accept;
     }
     filter prov65002out {
     if pref_from_myasset() then
     {
     bgp_community = -empty-;
     accept;
     }

     reject;
     }

     protocol bgp bgpAS65001 {
     table master;
     router id xxx.xxx.xxx.2;
     description «AS65001»;
     local as 65000;
     neighbor xxx.xxx.xxx.1 as 65001;
     hold time 240;
     startup hold time 240;
     connect retry time 120;
     keepalive time 80;
     start delay time 5;
     error wait time 60, 300;
     error forget time 300;
     next hop self;
     path metric 1;
     default bgp_med 0;
     source address xxx.xxx.xxx.2;
     export filter prov65002out;
     import filter prov65002in;

     }
     ###
```

10.11. Агент наблюдения

Zabbix-агент – программа контроля локальных ресурсов и приложений (таких как накопители, оперативная память, статистика процессора и т. д.) на сетевых системах. Zabbix-агенты являются чрезвычайно эффективными из-за использования специфических системных вызовов для сбора информации и подготовки статистики. Агент локально собирает оперативную информацию и отправляет данные zabbix-серверу для дальнейшей обработки.

ВНИМАНИЕ!

Подключение к агенту наблюдения должно осуществляться только по доверенным каналам передачи данных. Открывать порт агента наблюдения во внешнюю сеть категорически запрещается.

В данном подразделе происходит управление файлом конфигурации работы службы zabbix-агента (рис. 160) (см. описание кнопок в п. 5.4).

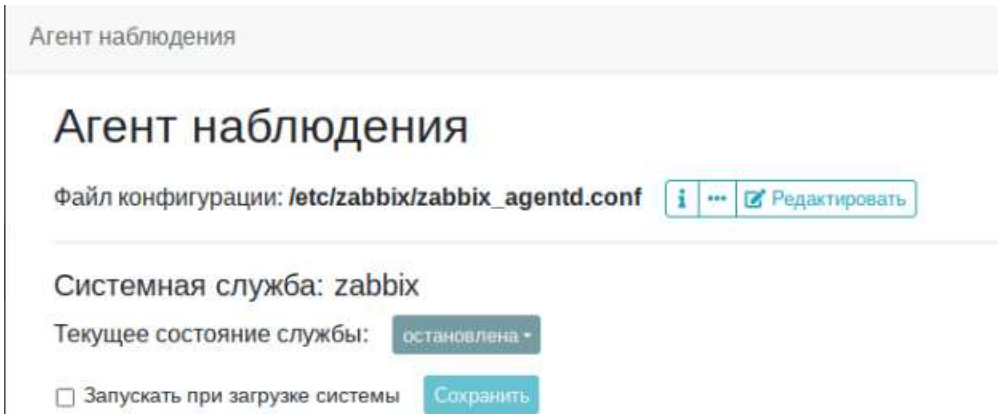


Рис. 160 – Системная служба zabbix

zabbix_agentd – демон zabbix-агента для мониторинга различных параметров сервера.

Для управления работой агента наблюдения используются опции административных функций (таблица 45), с помощью которых можно изменить уровень журналирования у процессов агента.

Таблица 45

Опция	Описание	Цель
`log_level_increase[=<цель>]`	Увеличение уровня журналирования. Действует на все процессы, если цель не указана.	Цель можно указать с помощью: - `тип процесса` – все процессы указанного типа (например, listener); - `тип процесса,N` – тип процесса и номер (например, listener,3); - `pid` – идентификатора процесса (от 1 до 65535). Для больших значений указывайте цель в виде `тип-процесса,N`.
`log_level_decrease[=<цель>]`	Уменьшение уровня журналирования. Действует на все процессы, если цель не указана.	

Список всех типов процессов агента:

- active checks – процесс для выполнения активных проверок;

- collector – процесс для сбора данных;

- listener – процесс ожидающий и выполняющий пассивные проверки.

Системную службу можно останавливать и запускать в ручном режиме.

Для автоматического запуска службы при загрузке системы – установить флаг в соответствующем поле (рис. 161).

После внесения необходимых настроек нажмите на кнопку «Сохранить».

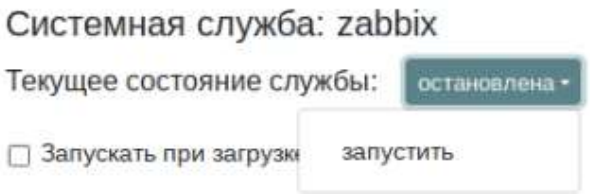


Рис. 161 – Изменение текущего состояния службы

Пример просмотра файла конфигурации представлен на рис. 162.

Файл конфигурации: /etc/zabbix/zabbix_agentd.conf

```
# This is a configuration file for Zabbix agent daemon (Unix)
# To get more information about Zabbix, visit http://www.zabbix.com

############ GENERAL PARAMETERS #################

### Option: PidFile
#   Name of PID file.
#
# Mandatory: no
# Default:
# PidFile=/var/run/zabbix/zabbix_agentd.pid

### Option: LogType
#   Specifies where log messages are written to:
#       system  - syslog
#       file    - file specified with LogFile parameter
#       console - standard output
#
# Mandatory: no
# Default:
# LogType=file

### Option: LogFile
#   Log file name for LogType 'file' parameter.
```

Рис. 162 – Окно информации о файле конфигурации системной службы zabbix

Поддерживаемые параметры в файле конфигурации zabbix-агента (`/etc/zabbix/zabbix_agentd.conf`) приведены в таблице 46.

Т а б л и ц а 46 – Параметры файла конфигурации zabbix-агента

Параметр	Обязательный	Диапазон	По умолчанию	Описание
`Alias`	нет			Задает алиас ключу элемента данных. Его можно использовать для замены длинных и сложных ключей элементов данных на более простые и короткие. Можно добавлять несколько параметров `Alias`. Разрешено указывать несколько параметров с одинаковым ключом `Alias`. Несколько ключей `Alias` могут ссылаться на один и тот же ключ. Алиасы можно использовать в `HostMetadataItem`, но нельзя в `HostnameItem` параметрах.
`AllowRoot`	нет		0	Разрешение агенту запускаться под root. Если отключено и агент запускается из-под root, то агент попытается переключиться на пользователя zabbix. Не имеет смысла, если агент запускается под обычным пользователем. 0 – не разрешать. 1 – разрешать.
`BufferSend`	нет	1 – 3600	5	Не хранить данные в буфере дольше N секунд.
`BufferSize`	нет	2 – 65535	100	Максимальное количество значений в буфере памяти. Агент будет отправлять все собранные данные zabbix-серверу или прокси при заполнении буфера.
`DebugLevel`	нет	0 – 5	3	Задает уровень журналирования: - 0 – основная информация о запуске и остановки процессов zabbix; - 1 – критичная информация; - 2 – информация об ошибках; - 3 – предупреждения; - 4 – для отладки (записывается очень много информации); - 5 – расширенная отладка (записывается еще больше информации).

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
EnableRemoteCommands	нет		0	Разрешены ли удаленные команды с zabbix-сервера. 0 – не разрешены. 1 – разрешены.
HostInterface	нет	0 – 255 символов		Опциональный параметр, который задает интерфейс хоста. Интерфейс хоста используется при процессе авторегистрации хоста сети. Агент сообщит об ошибке и не запустится, если значение превышает ограничение в 255 символов. Если на задано, значение будет извлечено с HostInterfaceItem.
HostInterfaceItem	нет			Опциональный параметр, который задает элемент данных, используемый для получения интерфейса хоста. Интерфейс хоста используется при процессе авторегистрации хоста сети. В процессе запроса авторегистрации агент запишет в журнал сообщение об ошибке, если полученное значение от указанного элемента данных превышает ограничение в 255 символов. Эта опция используется только, когда не задан HostInterface.
HostMetadata	нет	0 – 255 символов		Опциональный параметр, который задает метаданные хоста сети. Метаданные хоста сети используются только в процессе автоматической регистрации хостов сети (активный агент). Если не определено, то значение берется от HostMetadataItem. Агент выдаст ошибку и не запустится, если указанное значение выходит за лимит длины строки или не является UTF-8 строкой.
HostMetadataItem	нет			Опциональный параметр, который задает элемент данных zabbix-агент, который используется для получения метаданных хоста сети. Этот параметр используется только, если HostMetadata не определен. Поддерживаются UserParameters и алиасы. Поддерживается system.run[] независимо от значения EnableRemoteCommands.

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
				Значение `HostMetadataItem` поступает при каждой попытке авто-регистрации и используется только в процессе автоматической регистрации хостов сети (активный агент). В процессе запроса авторегистрации агент запишет в журнал предупреждающее сообщение, если полученное значение от указанного элемента данных выходит за лимит в 255 символов. Значение, полученное от указанного элемента данных должно являться UTF-8 строкой, в противном случае оно будет игнорироваться.
`Hostname`	нет		Задается `HostnameItem`	Уникальное, регистрозависимое имя хоста. Требуется для активных проверок и должно совпадать с именем хоста сети указанном на сервере. Допустимые символы: буквенно-цифровые, `'.'`, `' '`, `'_'` и `'-'`. Максимальная длина: 128.
`HostnameItem`	нет		`system.hostname`	Опциональный параметр, который задает элемент данных zabbix-агент, который используется для получения имени хоста. Этот параметр используется только, если `Hostname` не определен. Не поддерживает `UserParameters`, счетчики производительности и алиасы, но поддерживает `system.run[]`, независимо от значения `EnableRemoteCommands`.
`Include`	нет			Включает отдельные файлы или все файлы из папки с файлом конфигурации. Для включения только необходимых файлов из указанной папки, поддерживается символ звездочки для поиска совпадения по маске. Например: `/абсолютный/путь/к/файлам/конфигурации/*.conf`.
`ListenIP`	нет		0.0.0.0	Список IP-адресов разделенных запятыми, которые должен слушать агент.

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
`ListenPort`	нет	1024 – 32767	10050	Агент будет слушать этот порт для подключений с сервера.

`LoadModule`	нет			Модули, которые загружаются во время старта. Модули используются для расширения возможностей сервера. Форматы:
				`Loadmodule=<module.so>`
				`LoadModule=<path/module.so>`
				`LoadModule=</abs_path/module.so>`
				Модули должны находиться в папке указанной в параметре `LoadModulePath` или путь должен быть указан до имени модуля.
				Если путь до модуля абсолютный, тогда `LoadModulePath` игнорируется.
				Допускается добавлять несколько параметров `LoadModule`.

`LoadModulePath`	нет			Абсолютный путь к папке с модулями агента. По умолчанию зависит от опций компиляции.
`LogFile`	Да, если LogType задан как `file`, иначе нет.			Имя файла журнала.
`LogFileSize`	нет	0 – 1024	1	Максимальный размер файла журнала в Мбайт. 0 – отключение автоматической ротации журнала. П р и м е ч а н и е . Если лимит достигнут и ротация не удалась, по каким-либо причинам, существующий файл журнала очищается и начинается новый.
`LogType`	нет		file	Тип вывода журнала: - `file` – запись журнала в файл, указанный в LogFile параметре; - `system` – запись журнала в syslog; - `console` – вывод журнала в стандартный вывод.

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
LogRemoteCommands	нет		0	Включение журналирования выполняемых shell команд как предупреждений. 0 – отключено. 1 – включено.
MaxLinesPerSecond	нет	1 – 1000	20	Максимальное количество новых строк в секунду, которые агент будет отправлять серверу или прокси при обработке активных проверок 'log' и 'eventlog'. Указанное значение будет перезаписано параметром 'maxlines', указанное в ключах элементов данных 'log' и 'eventlog'. Обратите внимание: zabbix будет обрабатывать в 10 раз больше новых строк, чем указано в MaxLinesPerSecond при поиске требуемой строки в элементах данных журналов.
PidFile	нет		/tmp/zabbix_agentd.pid	Имя PID файла.
RefreshActiveChecks	нет	60 – 3600	120	Как часто обновлять список активных проверок, в секундах. Обратите внимание, что после неуспешного обновления активных проверок, следующая попытка будет предпринята через 60 секунд.
Server	да, если StartAgents задано значением 0 явно			Список разделенных запятой IP-адресов, опционально в CIDR нотации, или имен хостов zabbix-серверов и zabbix-прокси. Входящие соединения будут приниматься только с хостов указанных в этом списке. Если включена поддержка IPv6, то 127.0.0.1, ::127.0.0.1, ::ffff:127.0.0.1 обрабатываются одинаково и ::/0 разрешает все IPv4 и IPv6 адреса.0.0.0.0/0 можно использовать, чтобы разрешить любой IPv4 адрес.

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
				Обратите внимание, что «IPv4-совместимые IPv6 адреса» (`0000::/96 prefix`) поддерживаются, но являются устаревшими согласно RFC4291. Пример: `Server=127.0.0.1,192.168.1.0/24,::1,2001:db8::/32,zabbix.domain` Пробелы допускаются.
`ServerActive`	нет			Список пар `IP:порт` (или `имя хоста:порт`) zabbix-серверов или zabbix-прокси для активных проверок. Можно указывать несколько адресов разделенных запятыми, чтобы параллельно использовать несколько независимых zabbix-серверов. Пробелы допускаются. Если порт не указан, то используется порт по умолчанию. IPv6 адреса должны быть заключены в квадратные скобки, если для хоста указывается порт. Если порт порт не указан, то квадратные скобки для IPv6 адресов опциональны. Если параметр не указан, активные проверки отключены.
`SourceIP`	нет			Локальный IP-адрес для исходящих подключений.
`StartAgents`	нет	0 – 100	3	Количество пре-форков экземпляров zabbix_agentd, которые обрабатывают пассивные проверки. Если указано значение равное 0, то пассивные проверки будут отключены, и агент не будет слушать какой-либо TCP-порт. Максимальное количество 16 до версии 1.8.5.
`Timeout`	нет	1 – 30	3	Тратить не более `Timeout` секунд при обработке

Продолжение таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
`TLSAccept`	да, если заданы TLS сертификат или параметры PSK (даже при незашифрованном соединении), в противном случае – нет.			Какие принимаются входящие подключения. Используется пассивными проверками. Можно указывать несколько значений, разделенных запятой: - `unencrypted` – принимать подключения без шифрования (по умолчанию); - `psk` – принимать подключения с TLS и pre-shared ключом (PSK); - `cert` – принимать подключения с TLS и сертификатом.
`TLSCAFile`	нет			Абсолютный путь к файлу, который содержит сертификаты верхнего уровня CA(и) для верификации сертификата хоста, используется для зашифрованных соединений между zabbix компонентами.
`TLSCertFile`	нет			Абсолютный путь к файлу, который содержит сертификат или цепочку сертификатов, используется для зашифрованных соединений между zabbix компонентами.
`TLSConnect`	да, если заданы TLS сертификат или параметры PSK (даже при незашифрованном соединении), в противном случае – нет.			Как агент должен соединяться с zabbix-сервером или прокси. Используется активными проверками. Можно указать только одно значение: - `unencrypted` – подключаться без шифрования (по умолчанию); - `psk` – подключаться, используя TLS и pre-shared ключом (PSK); - `cert` – подключаться, используя TLS и сертификат.

Окончание таблицы 46

Параметр	Обязательный	Диапазон	По умолчанию	Описание
`TLSCRLFile`	нет			Абсолютный путь к файлу, который содержит отозванные сертификаты. Этот параметр используется для зашифрованных соединений между zabbix компонентами.
`TLSKeyFile`	нет			Абсолютный путь к файлу, который содержит приватный ключ агента, используется для зашифрованных соединений между zabbix компонентами.
`TLSPSKFile`	нет			Абсолютный путь к файлу, который содержит pre-shared ключ агента, используется для зашифрованных соединений с zabbix-сервером.
`TLSPSKIdentity`	нет			Строка идентификатор pre-shared ключа, используется для зашифрованных соединений с zabbix-сервером.
`TLSServerCertIssuer`	нет			Разрешенный эмитент сертификата сервера (прокси).
`TLSServerCertSubject`	нет			Разрешенная тема сертификата сервера (прокси).
`UnsafeUserParameters`	нет	0,1	0	Разрешить все символы, которые можно передать аргументами в пользовательские параметры. Не разрешены следующие символы: `\ ' " ` * ? [ ] { } ~ $! & ; ( ) < > | # @` Кроме того, не разрешены символы новой строки.
`User`	нет		zabbix	Использование привилегий указанного, существующего пользователя системы. Имеет эффект только, если запускается под root и `AllowRoot` отключен.
`UserParameter`	нет			Пользовательский параметр для мониторинга. Можно указать нескольких пользовательских параметров. Формат: `UserParameter=<ключ>,<shell команда>` Обратите внимание, что команда не должна возвращать только пустую строку или EOL. Например: `UserParameter=system.test,who|wc -l`

10.12. SNMP

Семейство стандартов простого протокола управления сетями (Simple Network Management Protocol, SNMP).

> ВНИМАНИЕ!
>
> Подключение по протоколу SNMP должно осуществляться только по доверенным каналам передачи данных. Открывать порт SNMP во внешнюю сеть категорически запрещается.

Для настройки SNMP, необходимо в файле `/etc/snmp/snmpd.conf` прописать в строчках:

```
rocommunity public  localhost
rocommunity public  10.137.210.1
```

вместо `public` прописать свое значение `rocommunity`, вместо `10.137.210.1`, свой IP-адрес. При этом данное значение должно представлять некую случайную буквенно-числовую последовательность длиной не менее 8 символов.

Перезагрузить сервис командой:

```
service snmpd restart
```

Для старта службы после перезагрузки:

```
chkconfig snmpd on
```

11. СИСТЕМА ОБНАРУЖЕНИЯ ВТОРЖЕНИЙ

11.1. Общие положения

СОВ МЭ ИВК КОЛЬЧУГА-К – выполняет сбор, декодирование, мониторинг, аудит сетевой безопасности и вывода. Представляет собой одновременно систему обнаружения вторжений (IDS) и систему предотвращения вторжений (IPS).

Работа СОВ «построена» вокруг группы правил, как предопределенных, так и созданных пользователем, применяемых к неким образом полученному трафику, может использоваться для анализа текущего трафика и файлов pcap.

Существует следующие варианты получения трафика:

- пропускать трафик через себя в inline режиме;
- получать копию трафика с порта коммутатора;
- анализировать дампы с трафиком.

В МЭ ИВК КОЛЬЧУГА-К предусмотрено два режима IPS: NFQ и AF_PACKET.

NFQ IPS режим (по умолчанию) работает следующим образом:

1) пакет попадает в iptables;
2) правило iptables направляет его в очередь NFQUEUE, например, `iptables -I INPUT -p tcp -j NFQUEUE`;
3) из очереди NFQUEUE пакеты могут обрабатываться на уровне пользователя, что и делает IPS, прогоняя пакеты по настроенным правилам (rules) и в зависимости от них может вынести один из трех вердиктов:
   - NF_ACCEPT – принять, пакет проходит дальше;
   - NF_DROP – отбросить, удалить этот пакет;
   - NF_REPEAT – повторить итерацию (повторный вызов функции с этим же пакетом), позволяет промаркировать пакеты и отправить их в начало текущей таблицы iptables, что дает возможность дальше их анализировать с помощью правил iptables.

Режим AF_PACKET отличается более высоким быстродействием в скорости обработки пакетов, но накладывает на систему ряд ограничений: она должна иметь два сетевых интерфейса и работать в качестве шлюза. Один интерфейс подключается после пограничных устройств и «смотрит» через них в незащищенные сети общего пользования (Интернет). Другой интерфейс подключается на вход защищаемого сегмента (см. п. 11.7.1.5), чтобы весь трафик проходил через СОВ и анализировался. Заблокированный пакет просто не пересылается на второй интерфейс.

По умолчанию захваченный трафик идет до декодирования одним потоком.

Стоит отметить наличие в СОВ основанных на библиотеке HTP продвинутых средств инспектирования HTTP. Они же могут быть использованы для протоколирования трафика без детектирования. Система также поддерживает декодирование IPv6, включая туннели IPv4-in-IPv6, IPv6-in-IPv6 и другие.

СОВ автоматически распознает множество протоколов, и правила не привязаны к номеру порта. Кроме того, в правилах используется концепция flowbits. Для отслеживания срабатываний используются переменные сессии, позволяющие создавать и применять различные счетчики и флаги. Многие СОВ рассматривают разные TCP-соединения как отдельные сущности и могут не увидеть связи между ними, свидетельствующей о начале атаки. СОВ пытается видеть картину целиком и во многих случаях распознает распределенный по разным соединениям вредоносный трафик.

Первым делом входящий трафик разбивается на TCP, UDP или другие транспортные потоки, после чего встроенные синтаксические анализаторы маркируют их и разбивают на высокоуровневые протоколы и их поля – нормализуя, если требуется. Полученные декодированные, разжатые и нормализованные поля протоколов затем проверяются наборами правил (сигнатур), которые выявляют, есть ли среди сетевого трафика попытки сетевых атак или вредоносные пакеты.

В качестве наборов правил СОВ для мониторинга сетевого трафика на наличие вредоносных действий, нарушений политики и угроз, используются наработки различных индивидуальных исследователей и компаний, в частности,

Suricata Emerging Threats, Sourcefire VRT, OpenSource Emerging Threats, а также Emerging Threats Pro.

Исследователи изучают появляющиеся угрозы и формируют свое понимание того, как та или иная атака может быть обнаружена на сетевом уровне за счет особенностей эксплуатации или других сетевых артефактов, а затем переводят полученное представление в одну или множество сигнатур на понятном для СОВ языке.

Правила (сигнатуры) в целом состоят из подмножества простых проверок, например, поиска подстроки или регулярного выражения, выстроенных в определенном порядке. При анализе сетевого пакета или потока сигнатура проверит все его содержимое на наличие всех допустимых комбинаций.

В СОВ МЭ ИВК КОЛЬЧУГА-К есть несколько готовых предопределенных правил (п. 11.8), в соответствии с которыми система работает в «пассивном режиме», она просто информирует администратора о возможных угрозах (п. 11.3), без каких-либо изменений в проходящем трафике, никакие сетевые пакеты не отбрасываются.

Настройки системы и правила хранятся в файлах формата YAML. Подробнее о конфигурации СОВ приведено в п. 11.7.

Данные, сформированные в результате проверок, СОВ выдаются в формате JSON, именно эти структурированные данные, представлены в разделе «СОВ» ГИ МЭ ИВК КОЛЬЧУГА-К (п. 11.3 – 11.5) для обработки, систематизации, анализа и визуализации.

11.2. Графический интерфейс СОВ

Состав раздела «СОВ» ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 163):

- события (п. 11.3);
- предупреждения (п. 11.4);
- важные (избранные) (п. 11.5);
- архив (п. 11.6);
- конфигурация (п. 11.7);
- правила (п. 11.8).

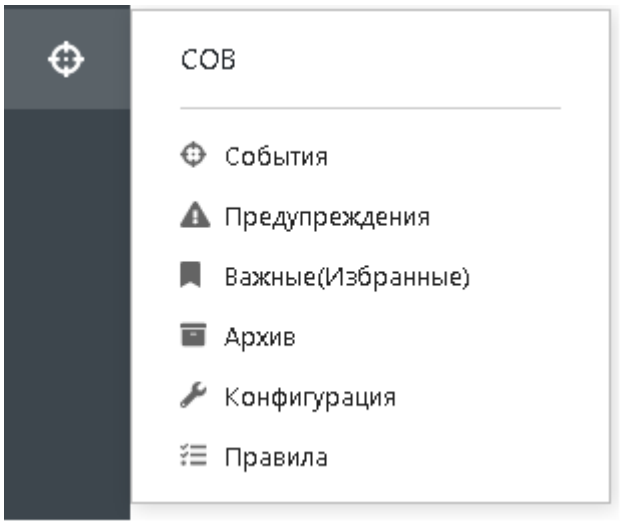


Рис. 163 – Меню раздела «СОВ»

11.3. События

В разделе «СОВ» ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 164) представлен журнал записей – результатов работы СОВ, правил и обнаружения вторжений (см. п. 11.8), для анализа событий и контроля данных о сетевом трафике в защищаемой сети.

При анализе пакета СОВ выполняет предопределенное действие, при совпадении всех указанных условий заданного правила, сгенированный сигнал или событие вместе с информацией о пакете попадает в журнальный файл.

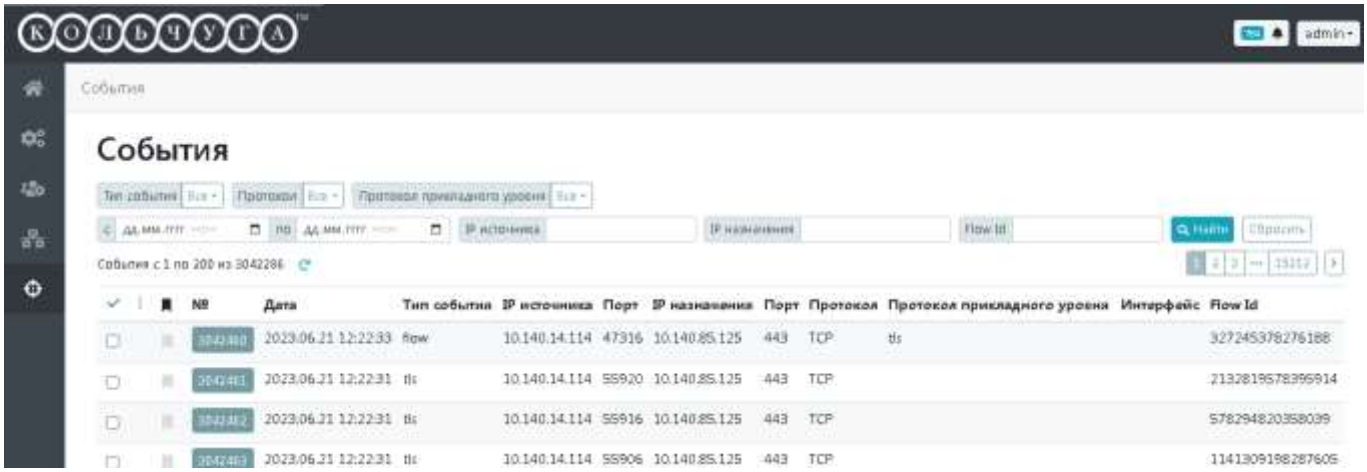


Рис. 164 – Раздел «СОВ». Вкладка «События»

11.3.1. Параметры события

Каждая запись журнала содержит следующие параметры события:

- № – номер зарегистрированного события – имеет цветовую подсветку, которая обозначает важность события:

  а) 

– высокий уровень предупреждения (alert);

  б) 

– средний уровень предупреждения (alert);

  в) 

– низкий уровень предупреждения (alert);

  г) 

– все остальные некритичные события;

- «Дата» – дата и время произошедшего события;
- «Тип события»;
- «IP источника» – сетевой адрес источника (отправителя);
- «Порт» – порт источника трафика;
- «IP назначения» – сетевой адрес получателя трафика;
- «Порт» – порт назначения трафика;
- «Протокол» – протокол трафика, который был декодирован;
- «Протокол прикладного уровня» (см. п. 11.9.34);
- «Интерфейс» – сетевой интерфейс;
- «Flow ID» – идентификатор потока трафика.

При нажатии левой кнопкой мыши на № записи события, отображается окно с более детальной информацией о параметрах события, а также JSON/текст данных, полученных в результате проверок правил (рис. 165). Параметры событий отличаются, в зависимости от типа событий.

Рис. 165 – Информация о событии

11.3.2. Действия

При анализе информации о событии доступны следующие действия:

1) в журнале (см. рис. 164):

- отметить как важное с помощью нажатия на пиктограму (рис. 166);

- выделить запись с помощью пиктограммы и применить одно из действий в выпадающем меню (рис. 167):

а) отметить как важное – позволяет поместить событие в специальную вкладку «Важные (избранные)»(см. п. 11.5);

б) удалить из важных – удаляет событие из отображения на вкладке «Важные (избранные)» (см. п. 11.5);

в) архивировать – позволяет поместить событие в специальную вкладку «Архив» (см. п. 11.6);

г) восстановить – восстанавливает событие в журнале после списания в «Архив» (см. п. 11.6);

2) при просмотре детальной информации о событии  (см. рис. 165):

- отметить как важное – позволяет поместить событие в специальную вкладку «Важные (избранные)»(см. п. 11.5);
- архивировать – позволяет поместить событие в специальную вкладку «Архив» (см. п. 11.6).

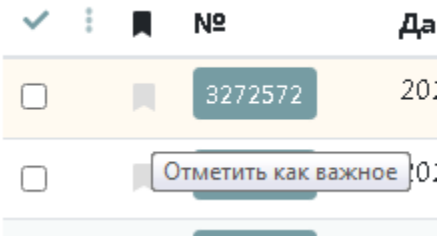


Рис. 166

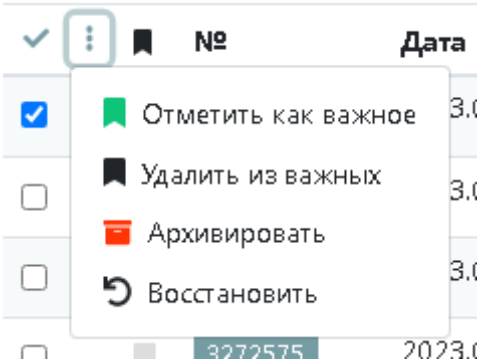


Рис. 167

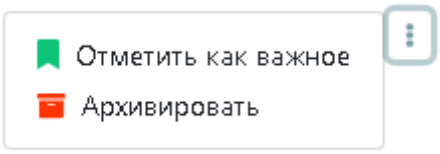


Рис. 168

При выполнении действий над выбранным событием, появится вслывающее окно, с информацией о том, что данные записи (ях) о событии (ях) обновлены (рис. 169).

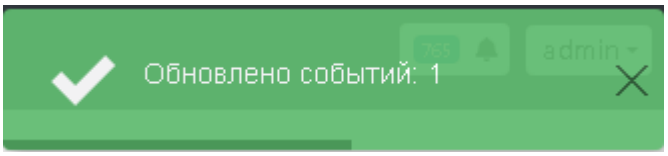


Рис. 169

11.3.3. Фильтрация событий

Для удобства работы записи в журнале можно фильтровать по следующим параметрам событий (рис. 170):

- «Дата» – дата и время произошедшего события;
- «Тип события»;
- «Протокол» – протокол трафика;
- «Протокол прикладного уровня»;
- временной период произошедшего события – время и дата задается с помощью выпадающего календаря (рис. 171);
- «IP источника» – IP-адрес источника;
- «IP назначения» – IP-адрес объекта назначения трафика;
- «Flow ID» – идентификатор потока трафика.

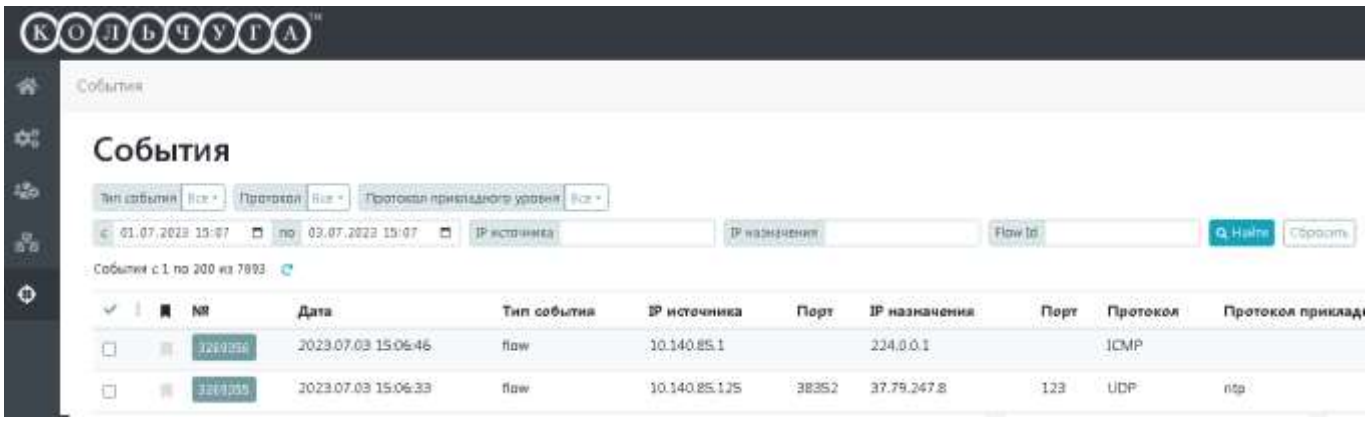


Рис. 170

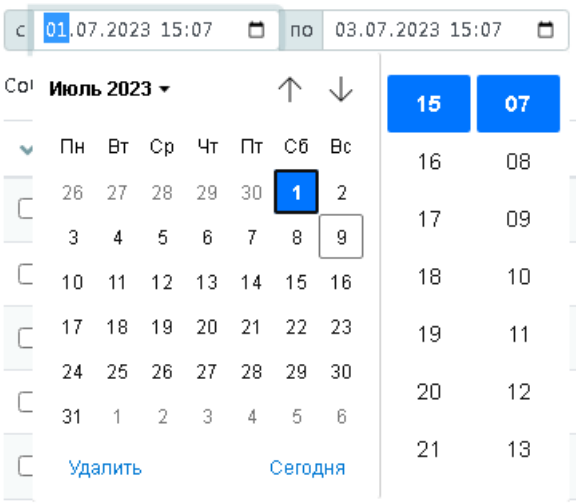
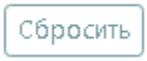


Рис. 171

Для осуществления фильтрации:

- выбрать и уточнить один или несколько параметров события;
- нажать на кнопку 

;
- дождаться осуществления обновления списка событий.

Кнопка 

отменяет фильтрацию списка событий и возвращает вкладку журнала к состоянию по умолчанию.

11.4. Предупреждения

На вкладке «Предупреждения» (рис. 172) журнала СОВ содержится список событий типа предупреждение (alert) высокого/среднего уровня важности (цветовая подсветка 

/ 

). Здесь в журнале представлена более детальная информация о сигнатуре для анализа.

Параметры записи о событии:

- № – номер зарегистрированного события – имеет цветовую подсветку, которая обозначает важность события;
- «Дата» – дата и время произошедшего события;
- «IP источника» – IP-адрес источника;
- «Порт» – порт источника трафика;
- «IP назначения» – IP-адрес объекта назначения трафика;

- «Порт» – порт назначения трафика;
- «Протокол» – протокол трафика;
- «Протокол прикладного уровня»;
- «Интерфейс» – сетевой интерфейс;
- «Flow ID» – идентификатор потока трафика;
- «ID сигнатуры» – идентификатор сигнатуры;
- «Сигнатура» – предоставляет информацию о подписи (сигнатуре) и возможном предупреждении.

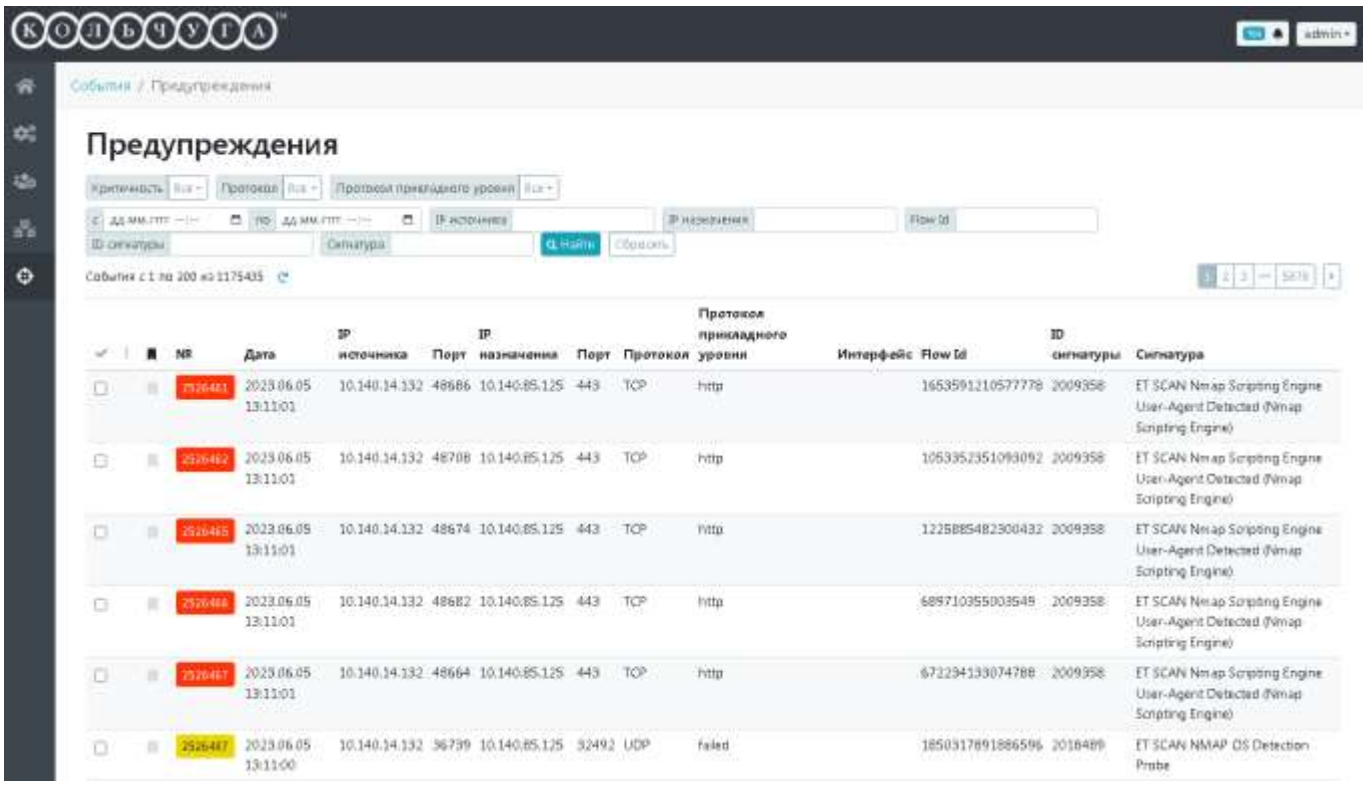


Рис. 172

При нажатии левой кнопкой мыши на № записи события, отображается окно с более детальной информацией о параметрах события, а также JSON/текст данных, полученных в результате проверок правил (см. рис. 165). Параметры событий отличаются, в зависимости от типа событий, так, например, alert (оповещение) содержит следующую информацию о событии (рис. 173):

- общие параметры;
- параметры события;
- параметры протокола;

- JSON/текст данных СОВ.

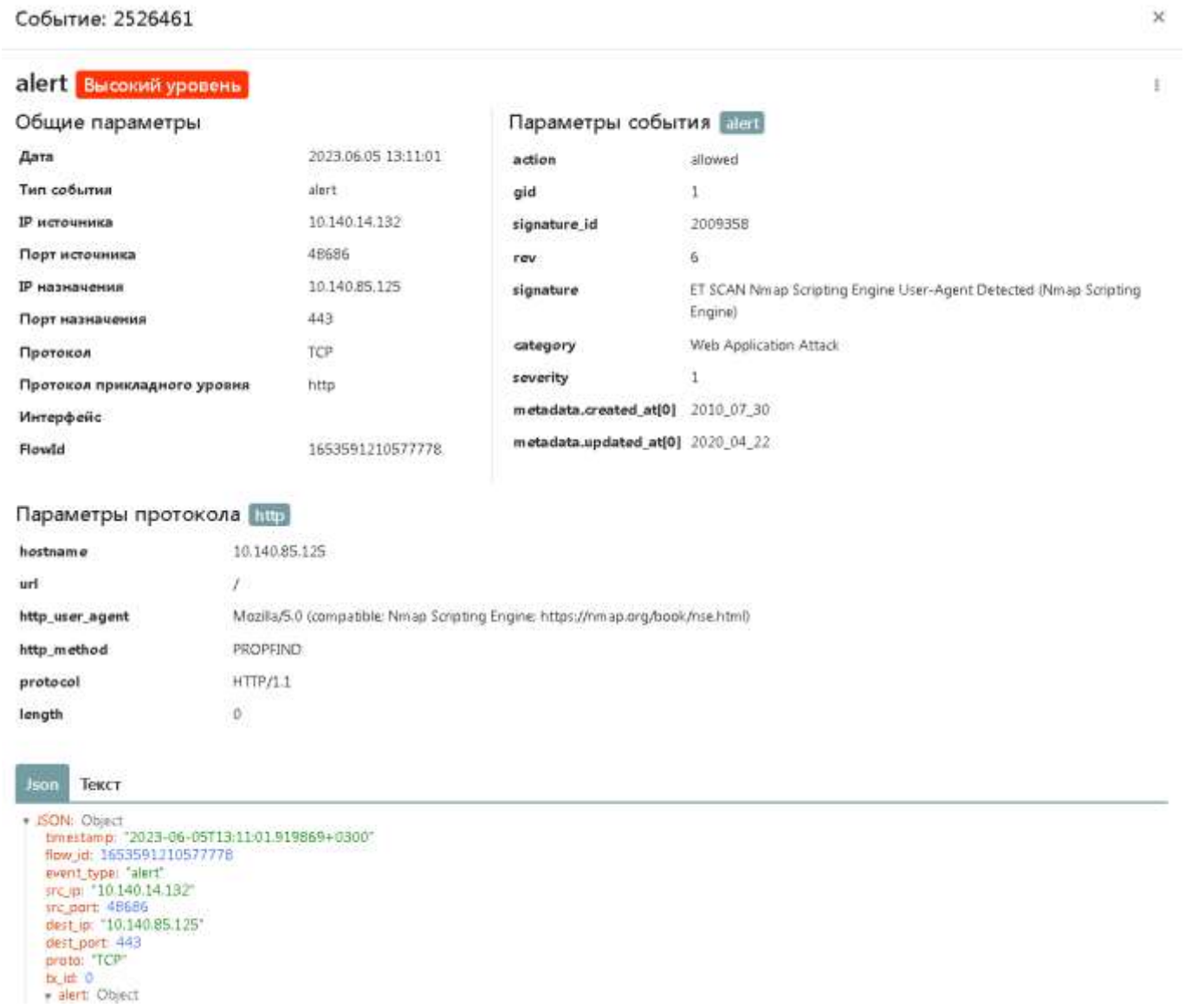


Рис. 173

Действия, доступные над записью о событии аналогичны действиям, описанным в п. 11.3.2.

Фильтрация событий осуществляется аналогично описанию, приведенном в п. 11.3.3, кроме этого доступна возможность построения выборки с использованием следующих поисковых ограничений:

- критичность – уровень критичности (важности) события alert (высокий, средний, низкий);
- «ID сигнатуры» – идентификатор сигнатуры;

- «Сигнатура» – поиск по ключевым словам, фразам в описании сигнатуры.

11.5. Важные (избранные)

Если событие было отмечено как важное , одним из способов (см. п. 11.3.2), то запись будет отображаться на этой вкладке (рис. 174).

Фильтрация происходит аналогично п. 11.3.3.

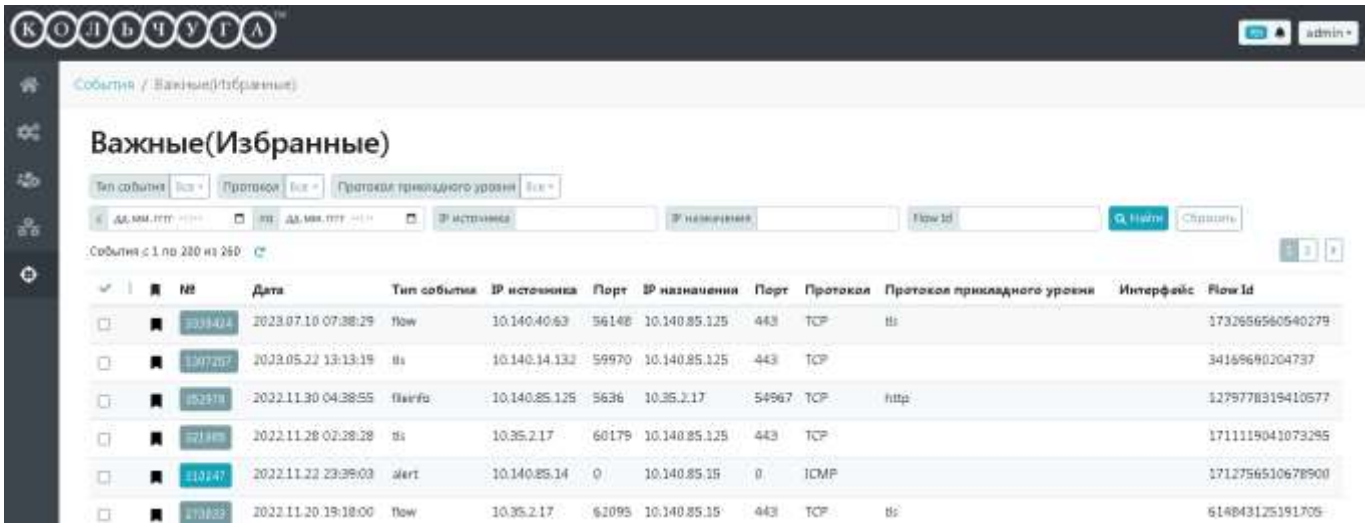


Рис. 174

11.6. Архив

Здесь содержатся все архивированные записи событий.

Если событие было перемещено в архив, одним из способов (см. п. 11.3.2), то запись будет отображаться на этой вкладке.

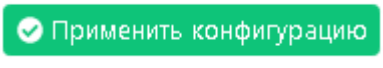
Фильтрация происходит аналогично п. 11.3.3.

11.7. Конфигурация

После внесения изменений на любой из вкладок раздела «Конфигурация» (п. 11.7.1.1 – 11.7.1.4), для применения настроек, необходимо:

- нажать 

для проверки корректности изменений;

- нажать 

для перезапуска службы СОВ и применения обновленной конфигурации.

11.7.1.1. Параметры сети

На данной вкладке доступна графическая настройка параметров конфигурации для работы СОВ, которые находятся в файле конфигурации: `/etc/suricata/suricata.yaml` (см. п. 11.7.1.4).

Для корректного применения базы сигнатур необходимо указать расположение объектов (сетей, серверов и портов), подверженных проверке.

Здесь можно указать внутренние и внешние сети, диапазоны адресов различных серверов, а также используемые порты. Всем параметрам присвоены значения по умолчанию, с которыми СОВ может корректно запуститься.

Для первичной настройки СОВ для защищаемой системы, обязательно необходимо указать IP-адрес системы в параметре «Домашняя сеть» (директива HOME_NET) на этой вкладке (рис. 175).

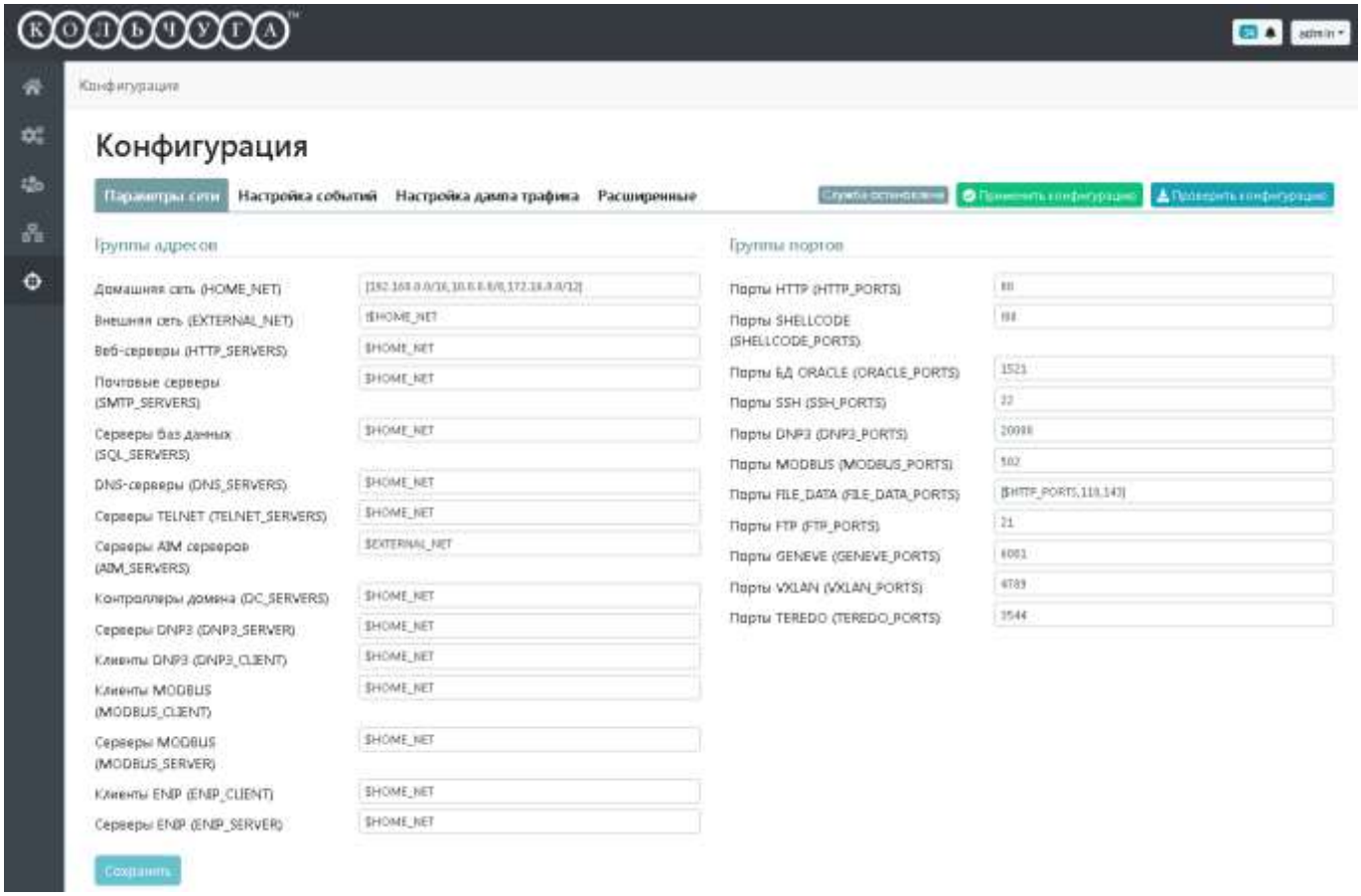


Рис. 175

При этом для параметра «Внешняя сеть» (переменная EXTERNAL_NET) значение по умолчанию «!$HOME_NET» – анализируется трафик на внешних

интерфейсах. Для анализа трафика локальной сети, можно в поле «Внешние сеть» добавить значение параметра «Домашняя сеть».

Переменные, приведенные на вкладке, используются в правилах для указания параметров сети, IP-адресов, чтобы указать какие сети проверять, а какие нет. Их можно разделить на два типа: группы адресов и группы портов, но выполняют одну и ту же функцию: изменяют правило так, чтобы оно было актуально для контролируемой сети

В качестве значений переменных «Группы адресов» можно назначать IP-адреса (поддерживаются как IPv4, так и IPv6) и диапазоны IP-адресов. Их можно комбинировать с операторами (таблица 47).

Т а б л и ц а 47

Оператор	Описание
../..	Диапазоны IP-адресов (нотация CIDR).
!	Исключение/отрицание.
[.., ..]	Группировка.

В подразделе «Группы портов» должны быть указаны актуальные порты протоколов для защищаемой системы. С приведенными выше операторами (таблица 47) можно комбинировать и порты.

Допустимыми являются следующие значения:

- номер порта;

- диапазон портов;

- группировка.

Для параметра «SHELLCODE-порты» часто используется исключение портов (например, !80). Все параметры должны быть установлены. Если нет необходимости устанавливать конкретный адрес/порт, можно ввести значение «any», которому соответствуют все IP-адреса (0.0.0.0/0)/порты, или оставить по умолчанию.

Если одна из границ диапазона не задана, вместо нее используется минимальный (0) или максимальный (65535) номер порта.

См. также информацию по адресам/портам источника и пункта назначения в п. 11.9.4 и п. 11.9.5.

Примеры использования адресов и портов в правилах приведены в таблице 48.

Таблица 48

Пример	Значение
!1.1.1.1	все IP-адреса, кроме 1.1.1.1
![1.1.1.1, 1.1.1.2]	все IP-адреса, кроме 1.1.1.1 и 1.1.1.2
$HOME_NET	Значения HOME_NET из yaml-файла
[$EXTERNAL_NET, !$HOME_NET]	EXTERNAL_NET, а не HOME_NET
[10.0.0.0/24, !10.0.0.5]	10.0.0.0/24 кроме 10.0.0.5.
[80, 81, 82]	порт 80, 81 и 82
[80: 82]	диапазон портов от 80 до 82
[1024:]	от 1024 до максимального номера порта
21	порт 21
!80	все порты, кроме 80
[80:100,!99]	диапазон от 80 до 100, исключая 99
[1:80,![2,4]]	диапазон от 1 до 80, кроме портов 2 и 4
[..., [....]]	
[..., ![.....]]	

Для корректной работы СОВ также проверьте и исправьте актуальный сетевой интерфейс (см. п. 11.7.1.4).

11.7.1.2. Настройка событий

На данной вкладке можно настроить регистрацию различных событий безопасности и выполнить настройки эвристического анализа (рис. 176).

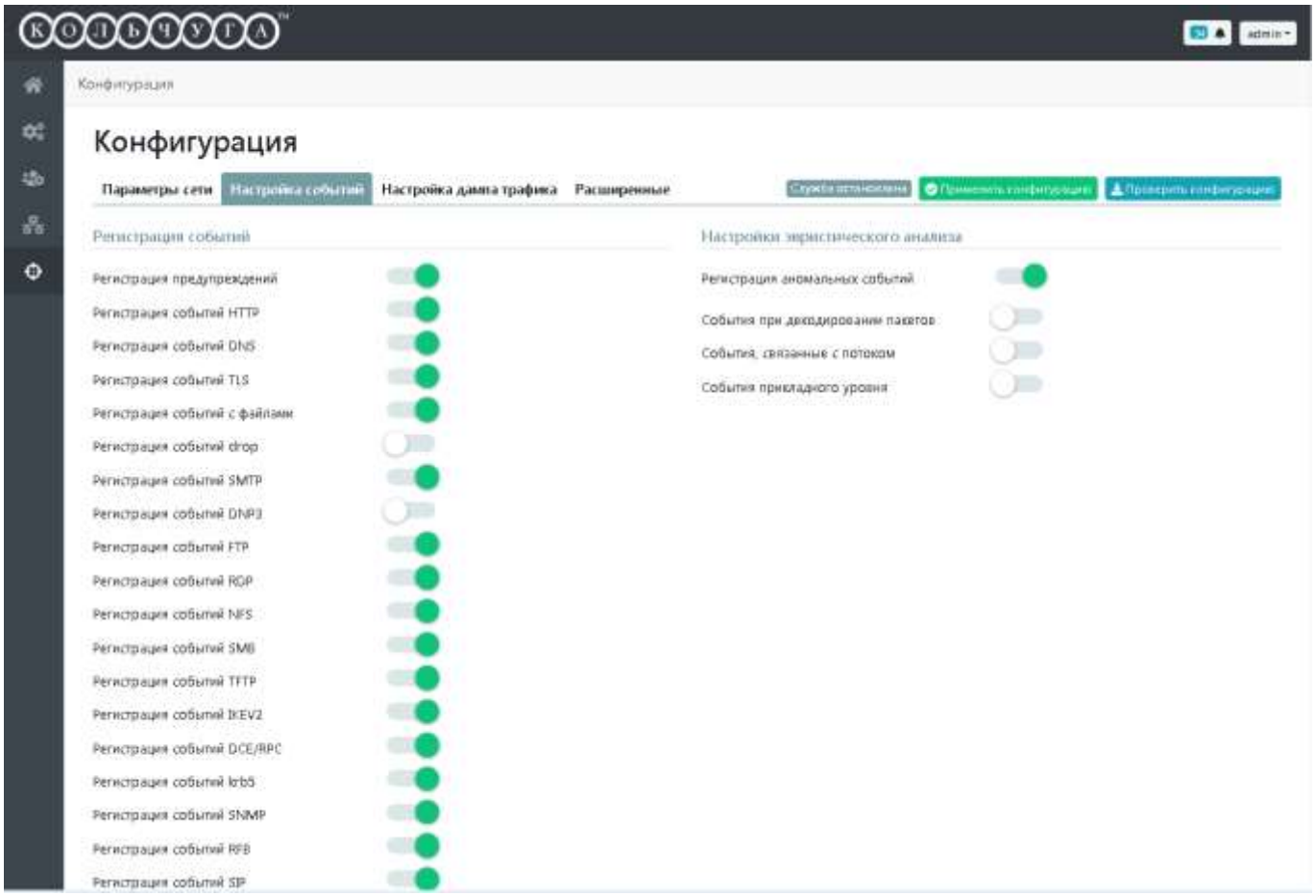


Рис. 176

Доступна регистрация событий, приведенных в таблице 49.

Таблица 49

Виды событий			
DCE/RPC	DHCP	DNP3	DNS
FTP	HTTP	HTTP2	IKEV2
MQTT	NFS	RDP	RFB
SIP	SMB	SMTP	SNMP
SSH	TFTP	TLS	drop
krb5	предупреждения	двунаправленные потоки	с файлами
	статистика	метаданные (metadata)	однонаправленные потоки (netflow)

Эвристический анализ выявления аномалий сетевого трафика может применяться в дополнение к сигнатурному анализу. При этом используются настройки эвристического анализатора, заданные по умолчанию (см. рис. 176).

Настройки эвристического анализа включают регистрацию:

- аномальных событий;
- событий при декодировании пакетов;
- событий, связанных с потоком;
- событий прикладного уровня.

Активация параметров регистрации событий и настроек выполняется с помощью переключателя  / (включить/выключить).

Для сохранения измененных значений, нажмите на кнопку 

внизу страницы, до перехода на другую вкладку ГИ МЭ ИВК КОЛЬЧУГА-К.

11.7.1.3. Настройка дампа трафика

На данной вкладке происходит настройка следующих возможностей (рис. 177):

- включить дамп трафика – активация записи дампа трафика выполняется с помощью переключателя  / (включить/выключить);
- название файла – шаблон для имени файла с дампом трафика;
- лимит размера файла (kb, mb, gb) – максимальный размер файла с дампом, можно указывать в Кбайт(kb), Мбайт (mb), Гбайт (gb), например, 5mb;
- макс. кол-во файлов – максимальное количество сохраняемых файлов с дампом трафика;
- сохранение в директорию – наименование директории для сохранения дампа трафика.

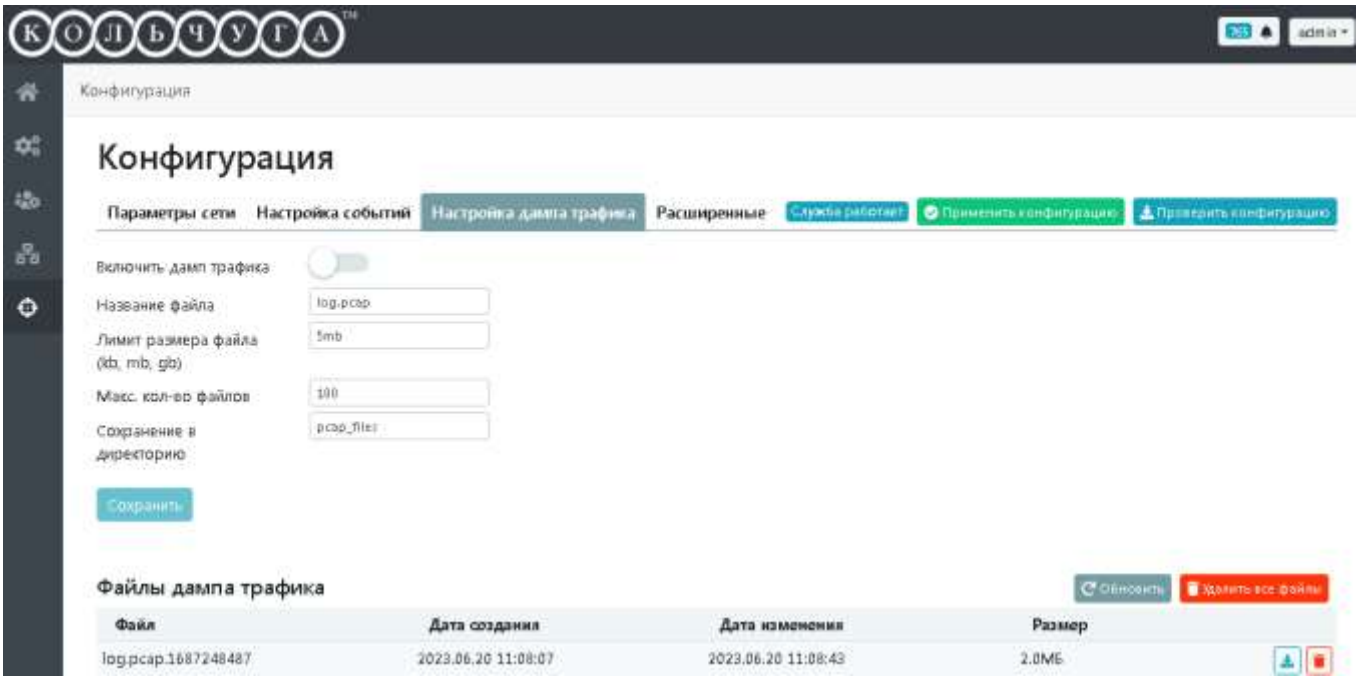


Рис. 177

Для сохранения настроек, нажмите на кнопку 

, до перехода на другую вкладку ГИ МЭ ИВК КОЛЬЧУГА-К.

Далее на странице отображается список сохраненных файлов дампа в трафика с краткой характеристикой каждого файла:

- наименование;
- дата создания;
- дата изменения;
- размер.

Есть возможность удалить отдельно каждый файл дампа – кнопка  или все сразу – кнопка 

.

Кнопка  загружает файл с дампом на компьютер, с которого выполняется вход в ГИ МЭ ИВК КОЛЬЧУГА-К.

11.7.1.4. Расширенные

Вкладка «Расширенные» (рис. 178) позволяет настраивать и просматривать файлы для конфигурации работы СОВ, по умолчанию:

- файл конфигурации: `/etc/suricata/suricata.yaml` – основной файл конфигурации СОВ;
- файл конфигурации правил: `/etc/suricata/rules.yaml` – файл текущей конфигурации правил, содержит список применяемых правил для работы СОВ, могут быть записаны наименования файлов правил, или маска, применяемая к группе наименований файлов правил одного вида (например, `ivk-*.rules`);
- файл классификации: `/etc/suricata/rules/classification.config` – файл описания возможных видов классификаций файлов правил (см. п. 11.9.8.5). Список значений «краткое имя» соответствует вариантам фильтра и значениям параметра «Классификация» подраздела «СОВ»→ «Правила» (см. п. 11.8.1).

Также на данной вкладке возможен просмотр состояния и перезапуск службы suricata, есть возможность отметить опцию «Запускать при загрузке системы».

Просмотр состояния и перезапуск службы возможен также в подразделе «Система»→ «Системные службы» (см. п. 9.4).

Рис. 178

11.7.1.5. Настройка сетевого интерфейса

В случае работы СОВ в режиме AF_PACKET необходимо указать актуальное наименование интерфейса в файле конфигурации.

Для этого, зайдите в консольный интерфейс МЭ ИВК КОЛЬЧУГА-К (см. п. 4.2.2) и определите сетевой интерфей, по которым СОВ должна проверять сетевые пакеты, например, с помощью команды:

```
ip addr
```

Пример части вывода команды о сетевом интерфейсе приведен на рис. 179.

```
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state DOW
N group default qlen 1000
    link/ether c4:54:44:d0:93:38 brd ff:ff:ff:ff:ff:ff
```

Рис. 179

Используйте эту информацию для настройки СОВ на вкладке «Расширенные» в файле конфигурации `/etc/suricata/suricata.yaml` – нажать кнопку Редактировать. Внесите изменения и нажмите «Сохранить».

В примере на рис. 179 имя интерфейса – eth0, поэтому имя интерфейса в разделе `af-packet` (`/etc/suricata/suricata.yaml`) должно совпадать с ним. Пример конфигурации интерфейса может выглядеть следующим образом:

```
af-packet:
    - interface: eth0
      cluster-id: 99
      cluster-type: cluster_flow
      defrag: yes
```

11.8. Правила СОВ

Обнаружение и предотвращение вторжений, анализ трафика СОВ осуществляет с использованием сигнатурного метода, основанного на применении набора правил.

Для запуска предупреждений об угрозах или иных действий над контролируемым трафиком, политику правил необходимо заранее предусмотреть и периодически обновлять.

Администратор СОВ может формировать правила СОВ, используя для этого готовые наборы правил (база правил), определяя только действие (см. п. 11.8.1.1), которое должно быть выполнено при срабатывании сигнатуры атаки. Редактирование поставляемых правил не предусмотрено.

Предоставляемый набор баз правил МЭ ИВК КОЛЬЧУГА-К предварительно разбит по группам. См. информацию о списке файлов правил в п. 11.8.2.

Описание правил, файлов правил и настройка их синхронизации приведена в п. 11.8.1 – 11.8.3.

После внесения изменений на любой из вкладок раздела «Правила» (п. 11.8.1 – 11.8.3), для применения настроек, необходимо:

- нажать 

для перезапуска службы СОВ и применения обновленной конфигурации.

Подробный синтаксис написания правил приведен в п. 11.9.

11.8.1. Правила

В подразделе «СОВ» → «Правила» приведен список всех возможных правил в СОВ МЭ ИВК КОЛЬЧУГА-К (рис. 180).

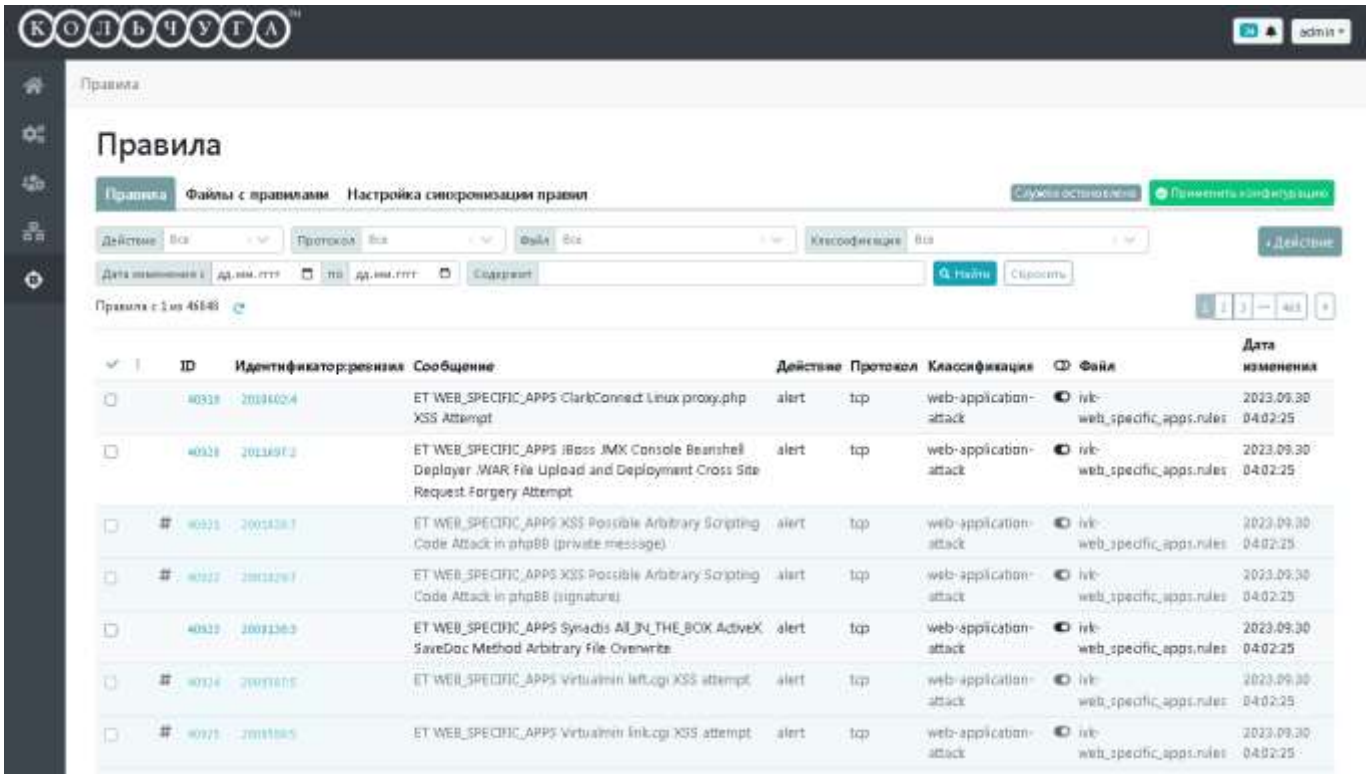


Рис. 180

Правило состоит из следующих частей:

1) действие, определяющее, что происходит при совпадении правила (см. п. 11.9.2);
2) заголовок, определяющий:
   - протокол (см. п. 11.9.3);
   - адрес источника/назначения (см. п. 11.9.4);
   - порт источника/назначения (см. п. 11.9.5);
   - направление (см. п. 11.9.6);
3) опции правила, определяющие особенности правила:
   - сообщение (см. п. 11.9.8.1);
   - ID – идентификатор (см. п. 11.9.8.2);
   - идентификатор:ревизия (см. п. 11.9.8.3);
   - классификация (см. п. 11.9.8.5);
   - дополнительные параметры (см. п. 11.9.9 – 11.9.35).

Запись правила в списке содержит также (см. рис. 180):

- признак активации правила:
  а) правило работает файл правила включен;
  б) # правило отключено, при этом файл правила включен ;
- наименование файла правила;
- дату изменения файла правила.

11.8.1.1. Управление

Интерфейс подраздела «СОВ» → «Правила» позволяет управлять включением/отключением правил, а также редактировать действия для текущих правил:

1) в списке (см. рис. 180):
   - выделить запись с правилом с помощью пиктограммы и применить одно из действий в выпадающем меню:
     а) изменить действие – позволяет изменить действие (см. п. 11.9.2), происходящее при совпадении правила;

б) включить – активирует правило;

в) выключить – правило отключено;

2) при просмотре детальной информации о правиле (см. рис. 181), при нажатии на поле «ID» или «Идентификатор:ревизия»:

- изменить действие – позволяет изменить действие (см. п. 11.9.2), происходящее при совпадении правила;
- включить – активирует правило;
- выключить – правило отключено.

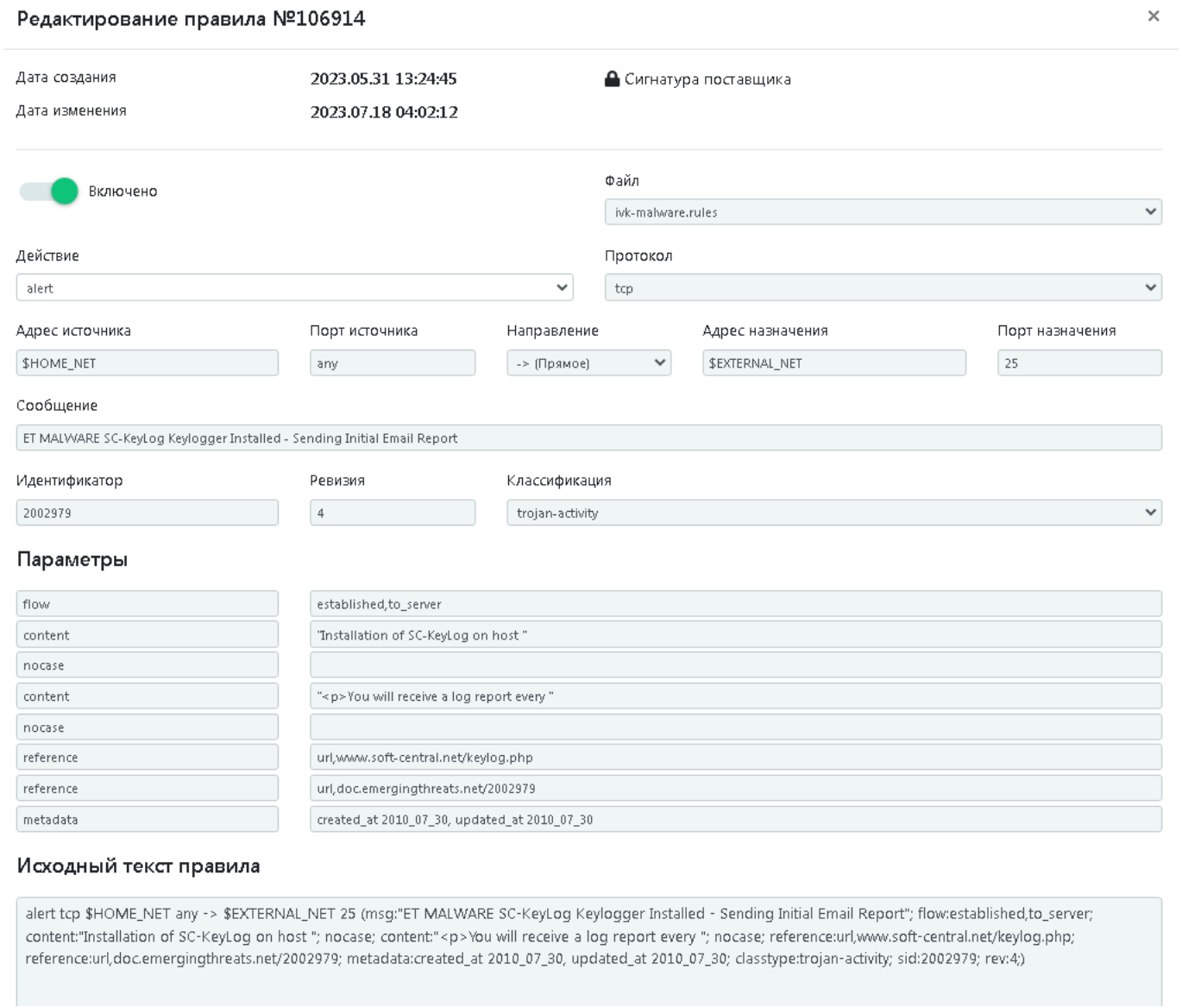


Рис. 181

11.8.1.2. Фильтрация правил

Для удобства работы со списком правил можно фильтровать в нем записи по следующим параметрам (см. рис. 180):

- «Действие» – действие, определяющее, что происходит при совпадении правила (см. п. 11.9.2);
- «Протокол» – протокол трафика (см. п. 11.9.3);
- «Файл» – наименование файла с правилом;
- классификация правила (см. п.11.9.8.5);
- «IP источника»/«IP назначения» – IP-адрес объекта источника/назначения трафика (см. п. 11.9.6);
- «Flow ID» – идентификатор потока трафика.

Для осуществления фильтрации:

- выбрать и уточнить один или несколько параметров правила;
- нажать на кнопку 

;
- дождаться осуществления обновления списка событий.

Кнопка 

отменяет фильтрацию списка событий и возвращает вкладку журнала к состоянию по умолчанию.

11.8.1.3. Создание правил в ГИ

Для создания правила в ГИ МЭ ИВК КОЛЬЧУГА-К (см. рис. 180) нажмите на кнопку «Действие»→ выбрать «Создать правило» (рис. 182).

Рис. 182

Появится окно создания нового правила (рис. 183).

Подробнее синтаксис правил и его параметры описаны в п. 11.9.

При добавлении правила СОВ в ГИ МЭ ИВК КОЛЬЧУГА-К (рис. 184):

- выберите файл для записи нового правила;

П р и м е ч а н и е . См. создание файла в п. 11.8.2 и на рис. 189.

- выберите действие (см. п. 11.9.2);
- заполните заголовок правила:

а) протокол (см. п. 11.9.3);

б) адрес источника/назначения (см. п. 11.9.4);

в) порт источника/назначения (см. п. 11.9.5);

г) направление (см. п. 11.9.6);

- добавьте необходимые параметры правила (см. п.11.9.7, 11.9.8), дополнительные параметры (см. п. 11.9.9 – 11.9.35).

Также можно выбрать файл для сохранения правила и вставить текст правила в поле «Исходный текст правила», созданный в соответствии с синтаксисом правил (см. п. 11.9), и нажать кнопку «Сохранить».

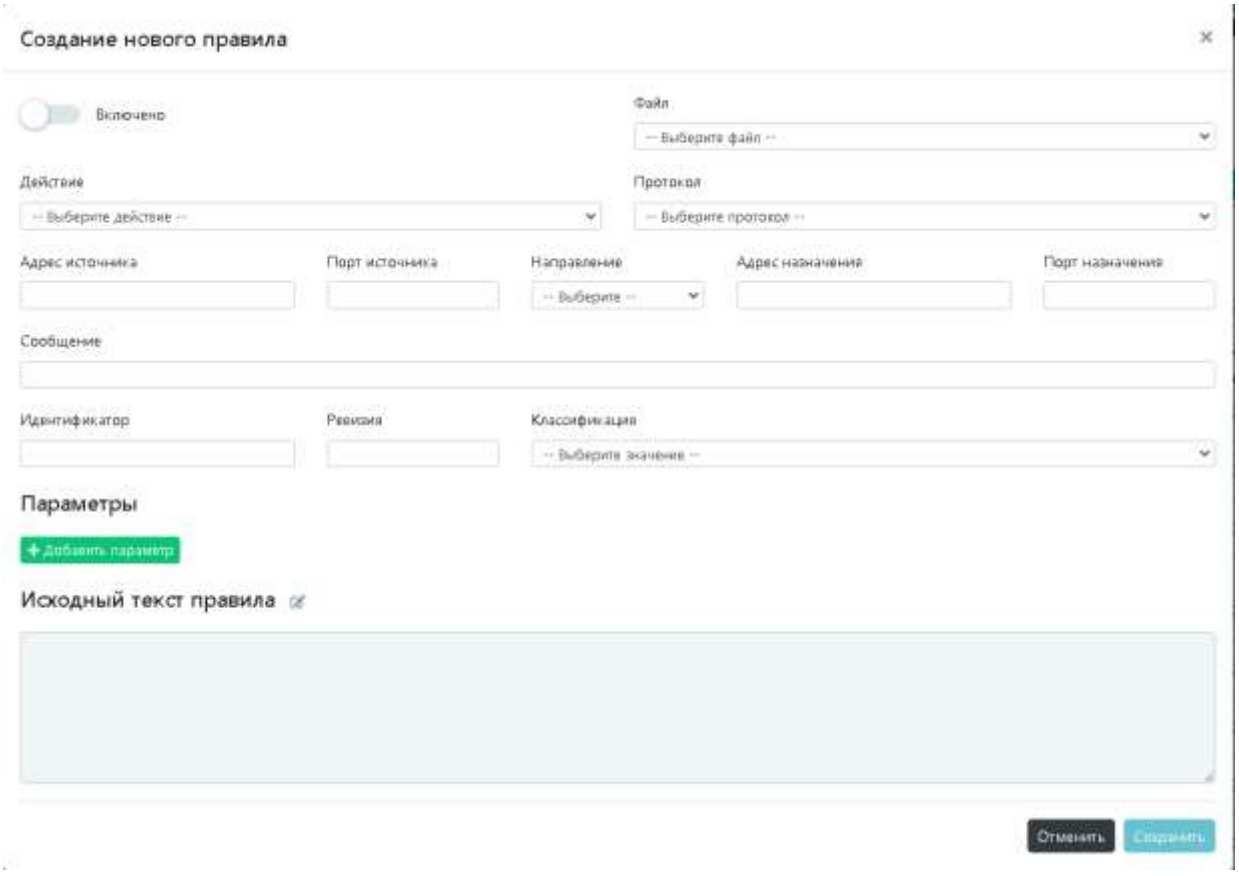


Рис. 183

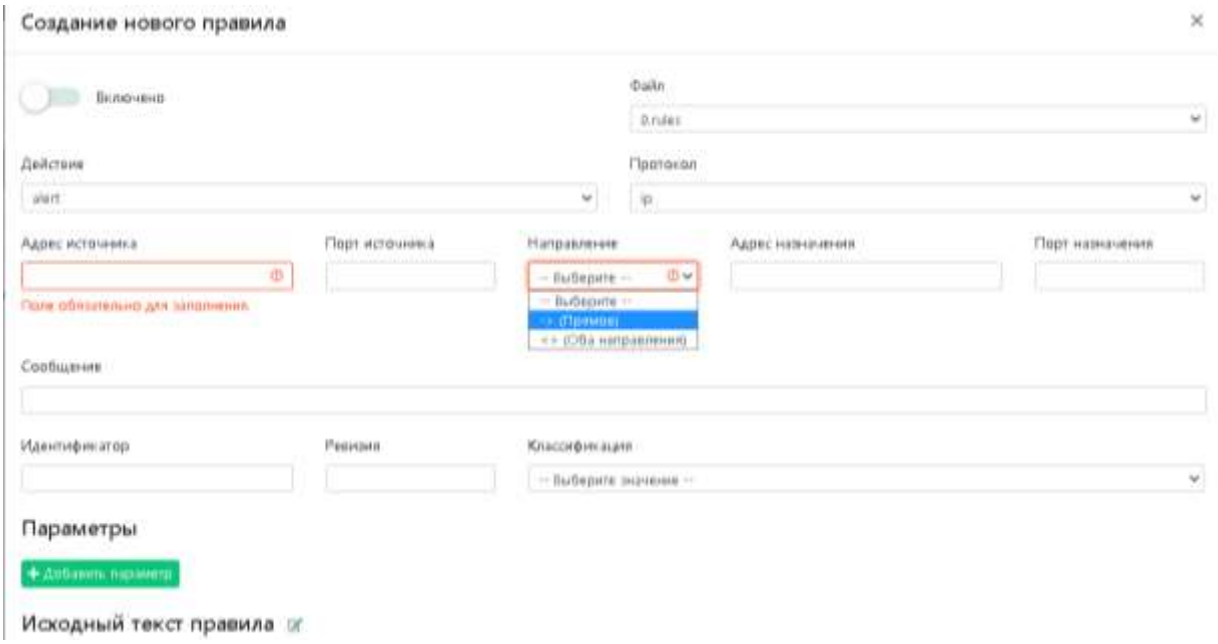


Рис. 184

Вставим следующий пример правила в поле «Исходный текст правила» и нажмем кнопку «Сохранить»:

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Outdated Firefox"; content:"User-Agent|3A| Mozilla/5.0 "; content:"Firefox/3."; distance:0; content:!"Firefox/3.6.13"; distance:-10; sid:9000000; rev:1;)
```

`content:!"Firefox/3.6.13";` – означает, что предупреждение будет генерируется, если используемая версия Firefox отличается от 3.6.13.

Результат добавления правила в ГИ представлен на рис. 185.

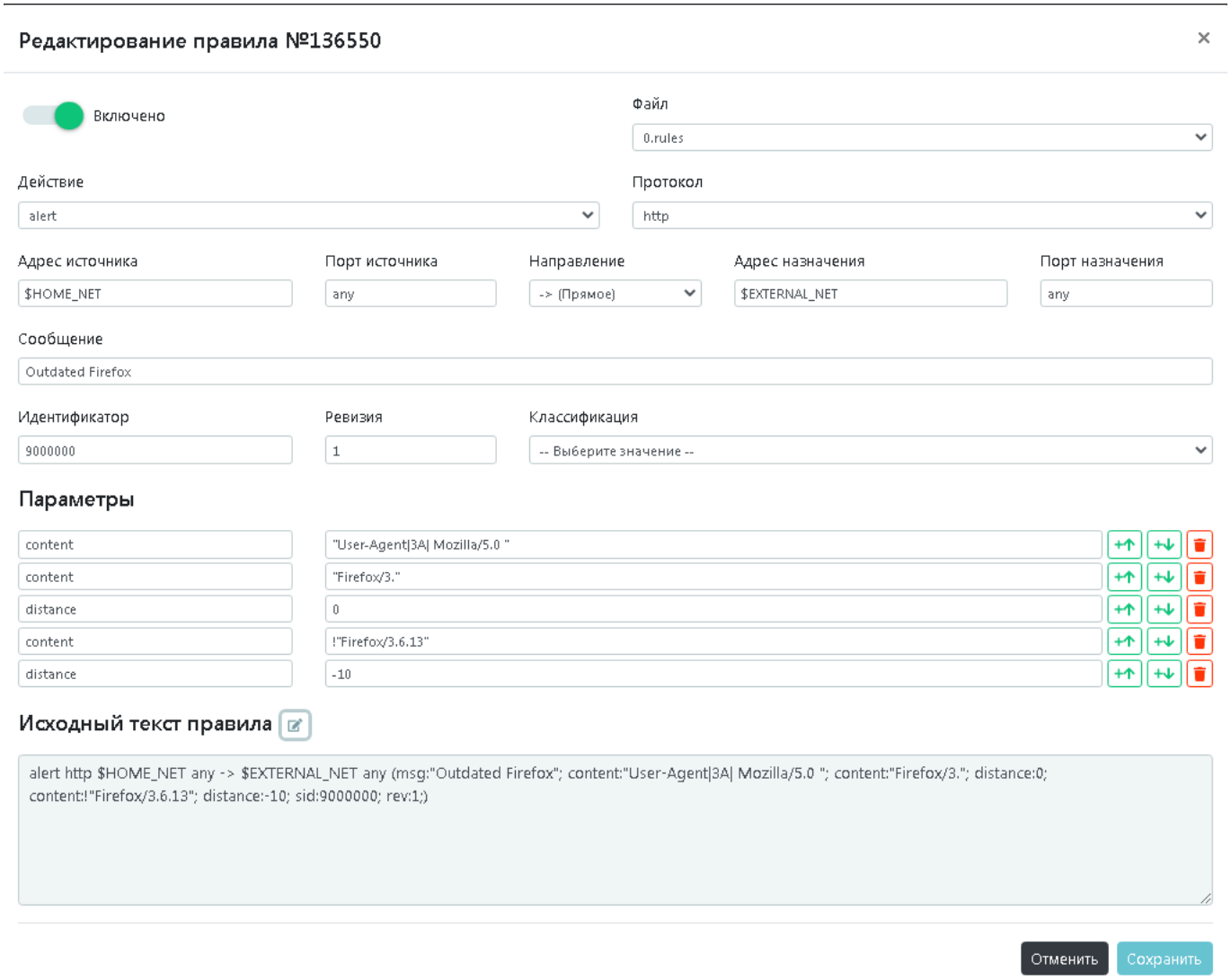


Рис. 185

Если произойдет дублирование идентификатора правила, в списке правил записи с одинаковы идентификаторами будут отмечены пиктограммой ⚠ (рис. 186).

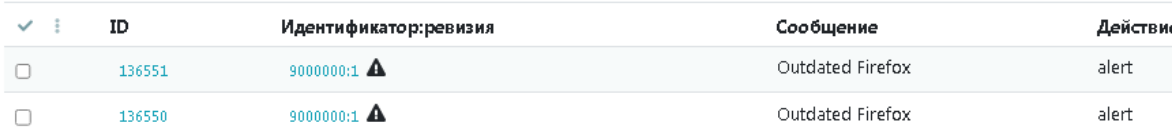


Рис. 186

11.8.2. Файлы с правилами

В подразделе приведен список всех файлов правил СОВ МЭ ИВК КОЛЬЧУГА-К (рис. 187). Правила объединены в файлы по смысловым группам.

В ГИ МЭ ИВК КОЛЬЧУГА-К отображаются правила из файлов, которые добавлены в конфигурацию.

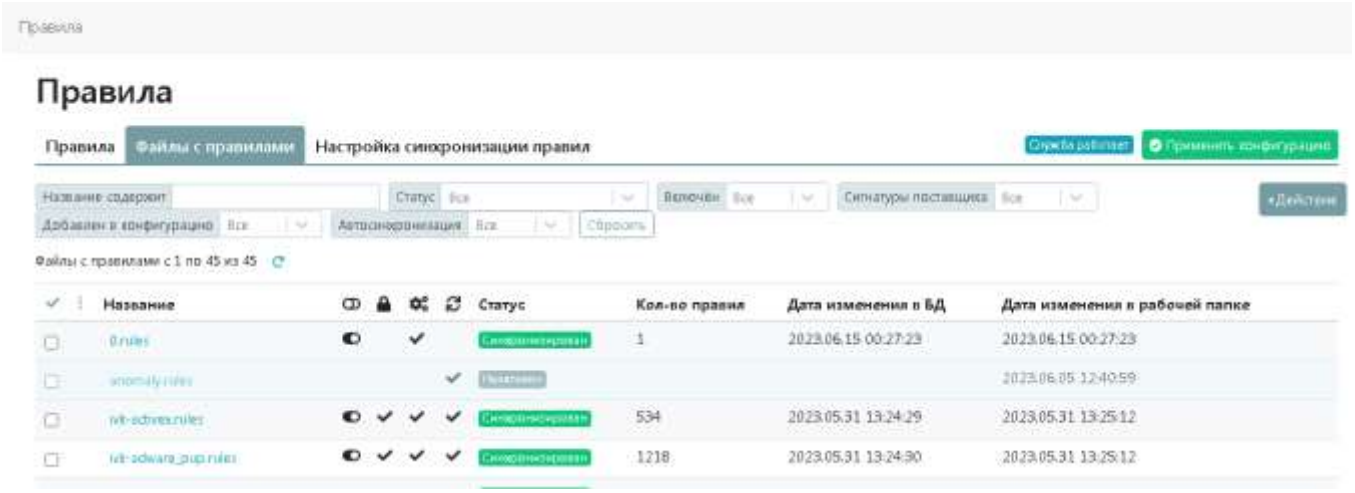


Рис. 187

Отображаемые параметры файла правила в списке:

- название – название файла правил в МЭ ИВК КОЛЬЧУГА-К;
- пиктограмы состояния (свойства) файла:
  - а) / включен или выключен;
  - б) – сигнатуры (правила) поставщика;
  - в) – добавлен в конфигурацию;
  - г) – автосинхронизация с рабочей папкой – файлы с правилами, для которых включена «автоматическая синхронизация», будут автоматически синхронизироваться с рабочей папкой при изменении правил;
  - д) – подтверждает одно из свойств, указанное в пунктах а), б), в);
- статус:
  - а) Синхронизирован – файлы правил в рабочей папке соответствуют

состоянию базы данных (БД) правил;

б) не синхронизирован – есть изменения правил в БД, состояние правил в рабочей папке отличается;

в) Неактивен неактивен;

г) в обработке – файл во временной папке, идет процесс обработки;

д) не контролируется – файл добавлен в конфигурацию и есть в рабочей папке, но его нет во временной, и соответственно нет в БД. Правила из таких файлов недоступны для редактирования в ГИ;

- количество правил – количество правил в файле;
- дата изменения в БД;
- дата изменения в рабочей папке.

В списке файлов правил доступны следующие действия:

1) в журнале (см. рис. 187):

- выделить запись с помощью пиктограммы  и применить одно из действий  в выпадающем меню (рис. 167):

а) синхронизировать – синхронизирует файлы правил временной и рабочей папок при изменении;

б) включить – включает файл с правилами;

в) выключить – выключает файл с правилами;

2) при просмотре детальной информации о файле правил Действие (см. рис. 188):

- файл во временной папке;
- файл в рабочей папке;
- проверить конфигурацию – для проверки корректности изменений;
- синхронизировать – синхронизироваться с рабочей папкой при изменении правил;
- удалить из файла конфигурации правил СОВ – перестанут быть доступны в ГИ МЭ ИВК КОЛЬЧУГА-К;
- удалить.

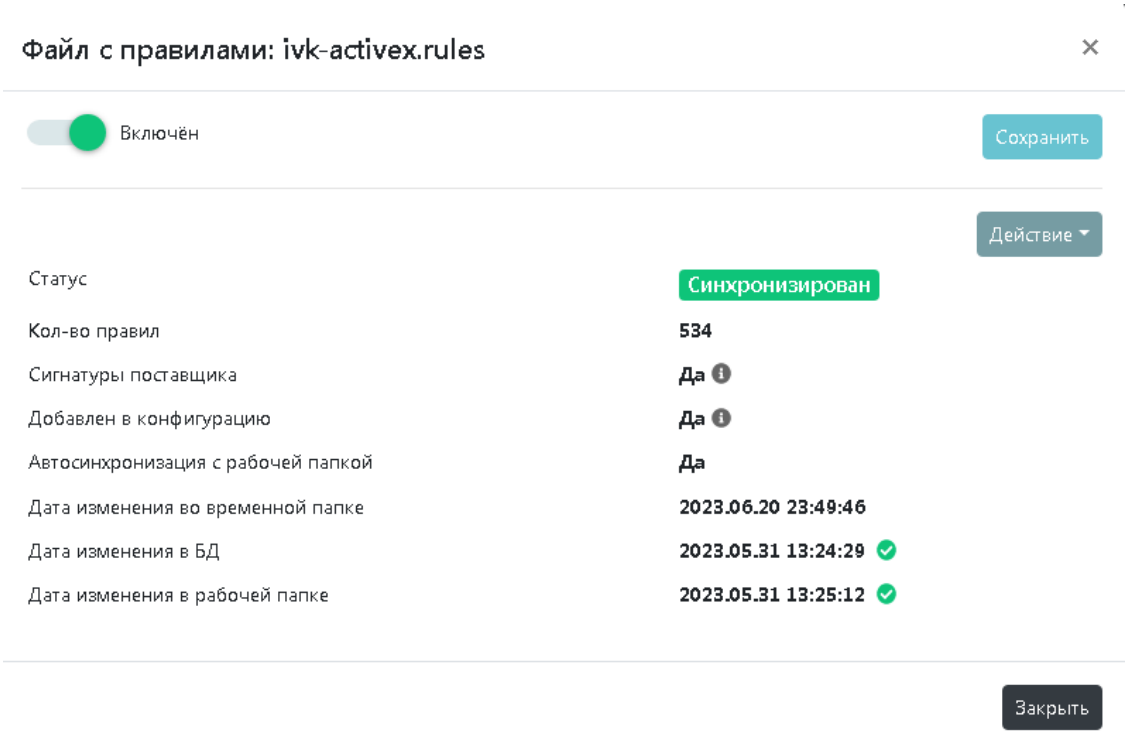


Рис. 188

Доступные действия с файлами правил в данном подразделе (рис. 189):

- создать файл – создание нового файла правил во временной папке с последующей загрузкой в БД;
- загрузить файл – загрузка нового файла правил во временную папку с последующей загрузкой в БД;
- синхронизировать – синхронизация БД и рабочей папки.

Для добавления пользовательских правил сначала создайте файл для правил (рис. 189).

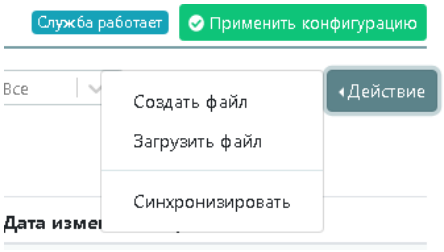


Рис. 189

Кнопка обновляет информацию списка файлов правил.

11.8.3. Настройка синхронизации правил

В данном подразделе можно с помощью кнопки + Добавить добавить файл правил по наименованию или группу файлов, используя маску (рис. 190), затем с помощью кнопки отредактировать активацию автоматической синхронизации с рабочей папкой при изменении правил (рис. 191).

Файлы правил поставщика (`ivk-*.rules`) автоматически синхронизированы и не переименовываются.

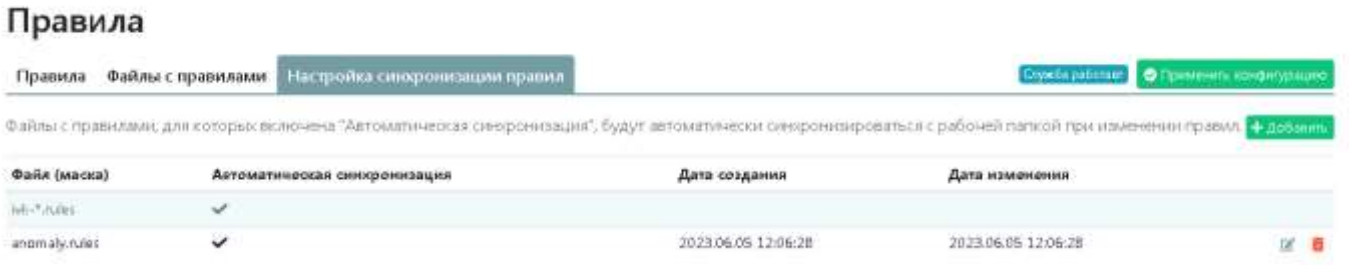


Рис. 190

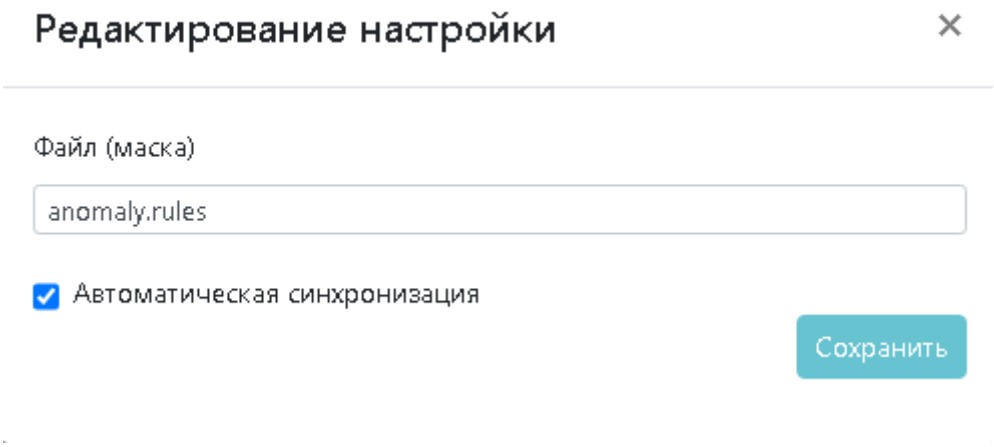


Рис. 191

11.8.4. Обновление базы решающих правил

Осуществить успешный вход на МЭ ИВК КОЛЬЧУГА-К через локальный консольный интерфейс (см. п. 4.2.2).

Для просмотра дат обновления правил СОВ выполните команду:

```
# ls -l /etc/suricata/
```

Выполнить команду для обновления базы решающих правил:

```
# update-rules
```

Пример выполнения обновления базы решающих правил приведен на рис. 192.

```
[NODE1 admin] ls -l /etc/suricata/
итого 88
drwxr-x--- 2 _suricata root  4096 авг 31 15:53 rules
drwxr-xr-x 2 _suricata root  4096 авг 31 16:52 rules_tmp
-rw------- 1 _suricata root    43 авг 23 12:56 rules.yaml
-rw-r--r-- 1 _suricata root 72612 авг 31 15:55 suricata.yaml
-rw------- 1 _suricata root  1644 авг 23 12:56 threshold.config
[NODE1 admin] update-rules
--2023-08-31 16:54:33--  ftp://repo.ivk.ru/rules.tgz
           => «rules.tgz»
Распознаётся repo.ivk.ru (repo.ivk.ru)… 185.6.174.74
Подключение к repo.ivk.ru (repo.ivk.ru)|185.6.174.74|:21... соединение установлено.
Выполняется вход под именем anonymous … Выполнен вход в систему!
==> SYST ... готово.  ==> PWD ... готово.
==> TYPE I ... готово.  ==> CWD не нужен.
==> SIZE rules.tgz ... 2830603
==> PASV ... готово.  ==> RETR rules.tgz ... готово.
Размер (байт): 2830603 (2,7M) (не достоверно)

rules.tgz                       100%[=================================================================>]   2,70M  14,1MB/s    за 0,2s

2023-08-31 16:54:38 (14,1 MB/s) - «rules.tgz» сохранён [2830603]

--2023-08-31 16:54:38--  ftp://repo.ivk.ru/rules.tgz.sum
           => «rules.tgz.sum»
Распознаётся repo.ivk.ru (repo.ivk.ru)… 185.6.174.74
Подключение к repo.ivk.ru (repo.ivk.ru)|185.6.174.74|:21... соединение установлено.
Выполняется вход под именем anonymous … Выполнен вход в систему!
==> SYST ... готово.  ==> PWD ... готово.
==> TYPE I ... готово.  ==> CWD не нужен.
==> SIZE rules.tgz.sum ... 129
==> PASV ... готово.  ==> RETR rules.tgz.sum ... готово.
Размер (байт): 129 (не достоверно)

rules.tgz.sum                   100%[=================================================================>]     129  --.-KB/s    за 0s

2023-08-31 16:54:43 (431 KB/s) - «rules.tgz.sum» сохранён [129]

[NODE1 admin] ls -l /etc/suricata/
итого 88
drwxr-x--- 2 _suricata root  4096 авг 31 15:53 rules
drwxr-xr-x 2 _suricata root  4096 авг 31 16:54 rules_tmp
-rw------- 1 _suricata root    43 авг 23 12:56 rules.yaml
-rw-r--r-- 1 _suricata root 72612 авг 31 15:55 suricata.yaml
-rw------- 1 _suricata root  1644 авг 23 12:56 threshold.config
[NODE1 admin]
```

Рис. 192 – Процесс обновление базы решающих правил

11.9. Синтаксис правил

11.9.1. Пример правила в текстовом виде

Пример правила выглядит следующим образом:

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; content:"GET"; http.uri; content:"rule"; fast_pattern; classtype:bad-unknown; sid:123; rev:1;)
```

В этом примере красным цветом обозначено действие (см. п. 11.9.2), зеленым – заголовок (включает в себя параметры, описанные в п. 11.9.3 – 11.9.6), а синим – параметры (см. п.11.9.7 – 11.9.35).

11.9.2. Действия

Список возможных действий:

- alert – сгенерировать оповещение;
- pass – прекратить дальнейшую проверку пакета;
- drop – отбросить пакет и сгенерировать предупреждение;
- reject – отклонить, отправить сообщение об ошибке RST/ICMP unreach отправителю соответствующего пакета;
- rejectsrc – то же, что и просто reject ;
- rejectdst – отклонить и отправить пакет ошибок RST/ICMP получателю соответствующего пакета;
- rejectboth – отправлять пакеты ошибок RST/ICMP обеим сторонам диалога.

П р и м е ч а н и я :

1. В режиме IPS (inline) использование любого из действий reject также включает drop.
2. Если нарушающий пакет касается TCP, это будет Reset-пакет. Для всех остальных протоколов это будет ICMP-пакет ошибок.

Правила будут загружаться в том порядке, в котором они появляются в файлах, но обрабатываться они будут в другом порядке. Правила имеют разные приоритеты, наиболее важные будут сканированы в первую очередь. Порядок по умолчанию это:

- pass (пропустить);
- drop (отбросить);
- reject (отклонить);
- alert (оповестить).

Это означает, что правило pass рассматривается перед правилом drop, а правило drop перед правилом reject и так далее.

11.9.3. Протокол

Можно выбрать следующие основные протоколы для анализа трафика правилом:

- TCP;

- UDP;

- ICMP.

Значение ip – означает «все» или «любые».

Также возможно определить различные протоколы прикладного уровня, например, приведенные в таблице 50.

Таблица 50

Протоколы	
http	http2
tls (включает ssl)	ftp
dns	smb
ssh	dce/rpc
imap	smtp
nfs	ikev2
krb5	ntp
dhcp	rfb
rdp	snmp
tftp	sip

Доступность этих протоколов, и иных настроек протоколов, зависит от того, включен ли протокол в разделе «Конфигурация» → «Настройка событий» (см. п. 11.7.1.2), изменения также сохранятся в файле конфигурации СОВ (см. п. 11.7.1.4).

Если в правиле добавлен, например, протокол «http», то СОВ гарантирует, что правило сработает только в том случае, если поток TCP содержит HTTP-трафик.

11.9.4. Источник и пункт назначения

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; контент:"GET"; http.uri; контент:"правило"; fast_pattern; classtype: trojan-activity; sid: 123; rev: 1;)
```

Первая выделенная часть – это источник трафика, вторая – его назначение, обратите внимание на направление стрелки.

С помощью источника и пункта назначения можно указать источник трафика и пункт назначения трафика соответственно. Можно назначать IP-адреса (поддерживаются как IPv4, так и IPv6) и диапазоны IP-адресов.

Обычно используются переменные файла конфигурации СОВ, такие как $HOME_NET и $EXTERNAL_NET, непосредственно IP-адреса указаные в них и будут использоваться вместо этих переменных в правилах.

Смотрите также описание значений переменных и операторов в п. 11.7.1.1.

11.9.5. Порты (источник и получатель)

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; контент:"GET"; http.uri; контент:"правило"; fast_pattern; classtype: trojan-activity; sid: 123; rev: 1;)
```

Первая выделенная часть – это порт источника, вторая – порт назначения, обратите внимание на направление стрелки направления.

Трафик входит и выходит через порты. Разные протоколы имеют разные номера портов. Например, для HTTP порт по умолчанию – 80, а порт для HTTPS – 443. Обратите внимание, что порт не определяет, какой протокол используется для связи. Скорее, он определяет, какое приложение получает данные.

Упомянутые выше порты обычно являются портами назначения. Исходные порты, т. е. приложение, отправившее пакет, обычно назначаются случайным портом операционной системой. При написании правила для службы HTTP обычно указывается, что любой пакет из любого исходного порта, предназначенный для HTTP (порт 80) соответствует записи в правиле:

```
any -> 80
```

Смотрите также описание значений переменных и операторов в п. 11.7.1.1.

11.9.6. Направление

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; контент:"GET"; http.uri; контент:"правило"; fast_pattern ; classtype: trojan-activity; sid: 123; rev: 1;)
```

Стрелка направления указывает, каким образом будет оцениваться правило. Для большинства правил используется стрелка вправо ->. Это означает, что только пакеты с указанным направлением могут соответствовать правилу. Однако также возможно определить, чтобы правило соответствовало обоим направлениям (<>).

Следующий пример иллюстрирует направление. В этом примере есть клиент с IP-адресом 1.2.3.4, использующий порт 1024. Сервер с IP-адресом 5.6.7.8, прослушивающий порт 80 (обычно HTTP). Клиент отправляет сообщение серверу, и сервер отвечает своим ответом.

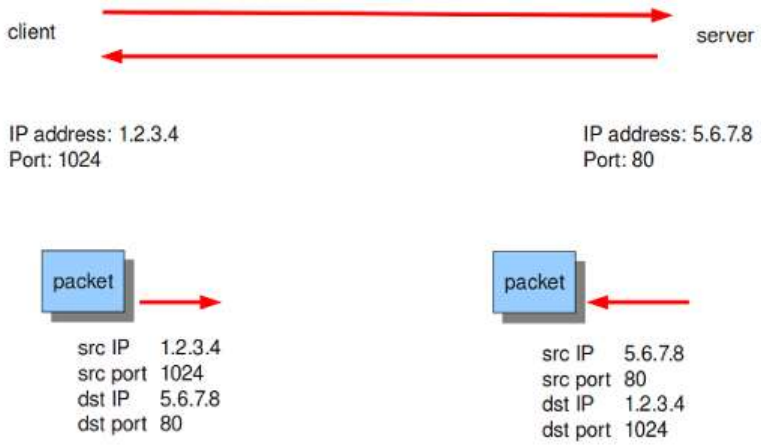


Рис. 193

Теперь предположим, что у нас есть правило со следующим заголовком:

```
alert tcp 1.2.3.4 1024 -> 5.6.7.8 80
```

Этому правилу будет соответствовать только трафик от клиента к серверу, так как направление указывает, что не нужно оценивать ответные пакеты.

«Обратного» стилевого направления не существует, т.е. нет <-.

11.9.7. Параметры правила

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"HTTP GET Request Containing Rule in URI"; flow:established,to_server; http.method; контент:"GET"; http.uri; контент:"правило"; fast_pattern; classtype: trojan-activity; sid: 123; rev: 1;)
```

Остальная часть правила состоит из опций. Они заключаются в круглые скобки и разделяются точкой с запятой. Некоторые параметры имеют настройки (см. п. 11.9.8)(например, msg), которые определяются ключевым словом параметра, за которым следует двоеточие, а затем параметры. У других нет настроек, это просто ключевое слово, например, nocase.

Синтаксис:

```
ключевое_слово: параметры;
ключевое_слово;
```

Параметры правила имеют определенный порядок, и изменение этого порядка изменит смысл правила.

Допускается запись одного правила в несколько строк, если все строки, за исключением последней, завершаются символом \.

Примечание. Символы ; и " имеют особое значение в языке правил и должны быть экранированы при использовании в значениях параметров правила. Например:

```
msg:"Message with semicolon\;";
```

Как следствие, также должна экранироваться обратная косая черта \, так как она действует как закрывающий символ.

11.9.8. Мета-ключевые слова

Мета-ключевые слова не влияют на проверку сетевого трафика СОВ; они влияют на то, как СОВ сообщает о событиях / предупреждениях.

11.9.8.1. Сообщение (msg)

Ключевое слово msg предоставляет контекстную информацию о правиле и возможном предупреждении в записи журнального файла или дампа пакета.

Синтаксис:

```
msg: "некоторое описание";
```

Примеры:

```
msg:"ET MALWARE Win32/RecordBreaker CnC Checkin";
msg:"ET EXPLOIT SMB-DS DCERPC PnP bind attempt";
```

11.9.8.2. Идентификатор подписи (sid)

Ключевое слово sid присваивает каждому правилу свой собственный идентификатор. Этот идентификатор указывается с числом, большим нуля.

Синтаксис:

```
sid:123;
```

Пример sid в подписи:

```
drop tcp $HOME_NET any -> $EXTERNAL_NET any (msg:”ET TROJAN Likely Bot Nick in IRC (USA +..)”; flow:established,to_server; flowbits:isset,is_proto_irc; content:”NICK “; pcre:”/NICK .*USA.*[0-9]{3,}/i”; reference:url,doc.emergingthreats.net/2008124; classtype:trojan-activity; sid:2008124; rev:2;)
```

Стандартная практика при написании правил заключается в том, что идентификатор указывается в качестве последнего ключевого слова (или предпоследнего, если есть `rev`).

В правилах глобального значения используются значения sid в диапазоне от 100 до 1000000. Значения менее 100 зарезервированы, а значения, превышающие 1000000, предназначены для локального использования, используйте их для идентификации собственных правил.

11.9.8.3. Ревизия (rev)

Ключевое слово sid почти каждый раз сопровождается rev – это версия правила. Если правило изменено, версия правила будет увеличена автором изменений. В правиле записывается на последнем месте, после всех ключевых слов.

Синтаксис:

```
rev:123;
```

11.9.8.4. Идентификатор группы (gid)

Ключевое слово gid можно использовать для придания различным группам подписей другое значение идентификатора (как в SID), по умолчанию использует gid 1.

В примере 1 – это gid, sid – 2008124, rev – 2:

```
10/15/09-03:30:10.219671 [**] [1:2008124:2] ET TROJAN Likely Bot Nick in IRC (USA +..) [**] [Classification: A Network Trojan was Detected] [Priority: 3] {TCP} 192.168.1.42:1028 -> 72.184.196.31:6667
```

11.9.8.5. Классификация (classtype)

Ключевое слово classtype предоставляет информацию о классификации правил и оповещений.

Класс правила (сигнатуры) определяет тип атаки, которая детектируется данной сигнатурой. Определяются также общие события, которые не относятся к атаке, но могут быть интересны в определенных случаях, например, обнаружение установления сессии TCP.

Он состоит из короткого имени, длинного имени и приоритет. Он может сказать, например, является ли правило просто информационным или речь идет о взломе и так далее. Для каждого типа класса classification.config имеет приоритет, который будет использоваться в правиле. Чем ниже значение приоритета, тем опаснее обнаруженное событие.

Пример определения типа класса:

```
config classification: web-application-attack,Web Application Attack,1
config classification: not-suspicious,Not Suspicious Traffic,3
```

По синтаксису classtype стоит перед sid и rev и после остальных ключевых слов.

Список возможных видов классификаций правил представлен в таблице 51).

Список значений «краткое имя» соответствует вариантам фильтра и значениям параметра «Классификация» подраздела «СОВ» → «Правила» (см. п. 11.8.1).

Т а б л и ц а 51 – Классификация правил

Краткое имя	Краткое описание	Перевод	Приоритет
attempted-admin	Attempted Administrator Privilege Gain	Попытка получить права администратора	Высокий (1)
attempted-user	Attempted User Privilege Gain	Попытка получения привилегий пользователя	Высокий (1)
command-and-control	Malware Command and Control Activity Detected	Обнаружена активность вредоносного ПО	Высокий (1)
credential-theft	Successful Credential Theft Detected	Обнаружена успешная кража учетных данных	Высокий (1)
domain-c2	Domain Observed Used for C2 Detected	Наблюдаемый домен используется для обнаружения C2	Высокий (1)
exploit-kit	Exploit Kit Activity Detected	Обнаружена активность набора эксплойтов	Высокий (1)
policy-violation	Potential Corporate Privacy Violation	Потенциальное нарушение корпоративной политик ИБ	Высокий (1)
shellcode-detect	Executable code was detected	Обнаружен исполняемый код	Высокий (1)
successful-admin	Successful Administrator Privilege Gain	Успешное получение прав администратора	Высокий (1)
successful-user	Successful User Privilege Gain	Успешное получение привилегий пользователя	Высокий (1)
targeted-activity	Targeted Malicious Activity was Detected	Обнаружена целенаправленная вредоносная активность	Высокий (1)
trojan-activity	A Network Trojan was detected	Обнаружен сетевой троян	Высокий (1)
unsuccessful-user	Unsuccessful User Privilege Gain	Неудачное получение привилегий пользователя	Высокий (1)
web-application-attack	Web Application Attack	Обнаружена атака на веб-приложение	Высокий (1)
attempted-dos	Attempted Denial of Service	Попытка совершения атаки Denial of Service (отказа в обслуживании)	Средний (2)
attempted-recon	Attempted Information Leak	Попытка атаки, направленной на утечку данных	Средний (2)
bad-unknown	Potentially Bad Traffic	Потенциально плохой трафик	Средний (2)
coin-mining	Crypto Currency Mining Activity Detected	Обнаружена активность по добыче криптовалюты	Средний (2)
default-login-attempt	Attempt to login by a default username and password	Попытка входа с именем/паролем по умолчанию	Средний (2)
denial-of-service	Detection of a Denial of Service Attack	Обнаружение атаки типа «отказ в обслуживании»	Средний (2)

Окончание таблицы 51

Краткое имя	Краткое описание	Перевод	Приоритет
external-ip-check	Device Retrieving External IP Address Detected	Обнаружено устройство, извлекающее внешний IP-адрес	Средний (2)
misc-attack	Misc Attack	Обнаружена атака	Средний (2)
non-standard-protocol	Detection of a non-standard protocol or event	Обнаружение нестандартного протокола или события	Средний (2)
pup-activity	Possibly Unwanted Program Detected	Возможно обнаружена нежелательная программа	Средний (2)
rpc-portmap-decode	Decode of an RPC Query	Декодирование запроса RPC	Средний (2)
social-engineering	Possible Social Engineering Attempted	Возможная попытка социальной инженерии	Средний (2)
successful-dos	Denial of Service	Отказ в обслуживании	Средний (2)
successful-recon-largescale	Large Scale Information Leak	Крупномасштабная утечка информации	Средний (2)
successful-recon-limited	Information Leak	Утечка информации	Средний (2)
suspicious-filename-detect	A suspicious filename was detected	Обнаружено подозрительное имя файла	Средний (2)
suspicious-login	An attempted login using a suspicious username was detected	Обнаружена попытка входа с использованием подозрительного имени пользователя	Средний (2)
system-call-detect	A system call was detected	Обнаружен системный вызов	Средний (2)
unusual-client-port-connection	A client was using an unusual port	Клиент использовал необычный порт	Средний (2)
web-application-activity	access to a potentially vulnerable web application	Обнаружен доступ к потенциально уязвимому веб-приложению	Средний (2)
icmp-event	Generic ICMP event	Событие ICMP	Низкий (3)
misc-activity	Misc activity	Прочая активность	Низкий (3)
network-scan	Detection of a Network Scan	Обнаружено сканирование сети	Низкий (3)
not-suspicious	Not Suspicious Traffic	Не подозрительный трафик	Низкий (3)
protocol-command-decode	Generic Protocol Command Decode	Декодирована команда протокола	Низкий (3)
string-detect	A suspicious string was detected	Обнаружена подозрительная строка	Низкий (3)
unknown	Unknown Traffic	Неизвестный трафик	Низкий (3)
tcp-connection	A TCP connection was detected	Обнаружено TCP-соединение	Информация (4)

11.9.8.6. Справка (reference)

Ключевое слово reference сообщают информацию о правиле и о проблеме, которую это правило пытается решить, позволяет включать в правила ссылки на внешние системы идентификации атак. Ключевое слово reference можно использовать в правиле несколько раз. Это ключевое слово предназначено для авторов правил и аналитиков, которые выясняют, почему правило сработало.

Синтаксис:

```
reference: type, reference
```

Пример ссылки на www.info.com:

```
reference: url, www.info.com
```

Есть также несколько систем (таблица 52), которые можно использовать в качестве справочного материала, а общеизвестным примером является CVE-база данных, которая присваивает номера уязвимостям, поэтому в справке можно использовать что-то вроде этого:

```
reference: cve, CVE-2014-1234
```

Все ссылочные типы определяются в конфигурационном файле `reference.config`.

Таблица 52

Система	Ссылка
bugtraq	http://www.securityfocus.com/bid/
cve	http://cve.mitre.org/cgi-bin/cvename.cgi?name=
nessus	http://cgi.nessus.org/plugins/dump.php3?id=
arachnids	http://www.whitehats.com/info/IDS
mcafee	http://vil.nai.com/vil/dispVirus.asp?virus_k=
url	http:/

6.2.7. Приоритет (priority)

Ключевое слово priority используется для присвоения правилу уровня приоритета. Синтаксис предполагает использование обязательного числового значения, которое может быть в диапазоне от 1 до 255. Чаще всего используются числа от 1 до 4:

- 1 высокий;

- 2 средний;

- 3 низкий;

- 4 информация.

Подписи с более высоким приоритетом будут проверяться в первую очередь. Высочайший приоритет равен 1. Обычно подписи получают значение приоритета через определенный тип classtype (см. п. 11.9.8.5), но его можно изменить с помощью ключевого слова priority.

Синтаксис:

```
priority:1;
```

11.9.8.7. Метаданные (`metadata`)

Ключевое слово `metadata` позволяет использовать дополнительную, нефункциональную информацию в правиле, не оказывающую влияние на анализ пакетов и выполняемые по отношению к ним операции. Несмотря на то, что формат является свободным, рекомендуется придерживаться указанных пар ключевых значений, поскольку их можно использовать в EVE оповещениях.

Синтаксис:

```
metadata: key value;
metadata: key value, key value;
```

11.9.8.8. Цель (target)

Ключевое слово target позволяет составителю правил указать, на какой стороне предупреждение находится цель атаки. Если оно указано, оповещение о событии будет содержать информацию об источнике и цели.

Синтаксис:

```
target:[src_ip|dest_ip]
```

Если указано значение `src_ip`, то IP-адрес источника в сгенерированном событии (поле `src_ip` в JSON) является целью атаки.

Если указано значение `dest_ip` тогда целью является IP-адрес.

11.9.9. Типы модификаторов

Некоторые ключевые слова действуют как модификаторы. Существует два типа модификаторов.

«Модификаторы содержимого», возвращаемые правилу, например:

```
alert http any any -> any any (content:"index.php"; http_uri; sid:1;)
```

В приведенном выше примере шаблон `index.php` изменен для проверки буфера HTTP `uri`.

Второй тип называется «`sticky buffer`» (липкий буфер). Он помещает имя буфера первым, а все ключевые слова, следующие за ним, применяются к этому буферу, например:

```
alert http any any -> any any (http_response_line; content: "403 Forbidden"; sid:1;)
```

В приведенном выше примере шаблон `"403 Forbidden"` проверяется на соответствие строке ответа HTTP, поскольку он следует за `http_response_line` ключевым словом.

11.9.10. Ключевые слова IP

11.9.10.1. TTL

Ключевое слово `ttl` используется для проверки времени жизни дейтаграммы IP.

Синтаксис:

```
ttl:<number>
```

Например: `ttl:10;`

Значение `ttl` (Time-to-life) определяет максимальное время пакета, которое он может находиться в Интернет-системе. Если задан 0, то пакет должен быть уничтожен. Время жизни основано на количестве прыжков (маршрутизаторов). Каждый маршрутизатор, через который проходит пакет, вычитает 1 в счетчике `TTL` пакета. Цель этого механизма состоит в том, чтобы ограничить существование пакета, чтобы пакеты не смогли оказаться в бесконечном цикле маршрутизации.

Пример использования ключевого слова `ttl` в правиле:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:”GPL MISC 0 ttl”; ttl:0; reference:url,support.microsoft.com/default.aspx?scid=kb#-#-EN-US#-#-q138268; reference:url,www.isi.edu/in-notes/rfc1122.txt; classtype:misc-activity; sid:2101321; rev:9;)
```

11.9.10.2. IPOPTS

С помощью ключевого слова `ipopts` можно проверит, есть ли указанные опции в IP заголовке. `ipopts` должно использоваться в начале правила и только один раз. Существует несколько вариантов, которые можно проверить (таблица 53).

Таблица 53

IP опции	Описание	
`Rr`	Record Route	Запись маршрута
`Eol`	End of List	Завершение списка опций
`Nop`	No Op	Нет опций
`Ts`	Time Stamp	Временная метка
`Sec`	IP Security	Опция безопасности
`esec`	IP Extended Security	Расширенные опции безопасности
`lsrr`	Loose Source Routing	Не жестко заданный отправителем маршрут
`ssrr`	Strict Source Routing	Жестко заданный отправителем маршрут
`satid`	Stream Identifier	Идентификатор потока
`Any`	any IP options are set	Любые опции

Синтаксис:

```
ipopts: <name>
```

Пример `ipopts` в правиле:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL MISC source route ssrr"; ipopts:ssrr; reference:arachnids,422; classtype:bad-unknown; sid:2100502; rev:3;)
```

11.9.10.3. sameip

Ключевое слово `sameip` позволяет детектировать пакеты с совпадающими IP-адресами для получателя и отправителя.

Синтаксис:

```
sameip;
```

Пример `sameip` в правиле, генерирует сигнал оповещения при совпадении IP-адресов получателя и отправителя:

```
alert ip any any -> any any (msg:"GPL SCAN same SRC/DST"; sameip; reference:bugtraq,2666; reference:cve,1999-0016; reference:url,www.cert.org/advisories/CA-1997-28.html; classtype:bad-unknown; sid:2100527; rev:9;)
```

11.9.10.4. ip_proto

С помощью ключевого слова `ip_proto` можно проверить идентификатор протокола в заголовке IP (таблица 54). Список протоколов находится в файле `packet-header`. Можно использовать имя или номер протокола при записи.

Синтаксис:

```
ip_proto:номер_протокола;
```

Таблица 54

Номер	Ключевое имя	Протокол	Номер	Ключевое имя	Протокол
1	ICMP	Internet Control Message Protocol	74	WSN	Wang Span Network
2	IGMP	Internet Group Management Protocol	75	PVP	Packet Video Protocol
3	GGP	Gateway-to-Gateway Protocol	76	BR-SAT-MON	Backroom SATNET Monitoring
4	IP-in-IP	IP in IP (encapsulation)	77	SUN-ND	SUN ND PROTOCOL-Temporary
5	ST	Internet Stream Protocol	78	WB-MON	WIDEBAND Monitoring
6	TCP	Transmission Control Protocol	79	WB-EXPAK	WIDEBAND EXPAK
7	CBT	Core-based trees	80	ISO-IP	International Organization for Standardization Internet Protocol
8	EGP	Exterior Gateway Protocol	81	VMTP	Versatile Message Transaction Protocol
9	IGP	Interior Gateway Protocol (any private interior gateway, for example Cisco's IGRP)	82	SECURE-VMTP	Secure Versatile Message Transaction Protocol
10	BBN-RCC-MON	BBN RCC Monitoring	83	VINES	VINES

Продолжение таблицы 54

Номер	Ключевое имя	Протокол	Номер	Ключевое имя	Протокол
11	NVP-II	Network Voice Protocol	84	IPTM	Internet Protocol Traffic Manager
12	PUP	Xerox PUP	85	NSFNET-IGP	NSFNET-IGP
13	ARGUS	ARGUS	86	DGP	Dissimilar Gateway Protocol
14	EMCON	EMCON	87	TCF	TCF
15	XNET	Cross Net Debugger	88	EIGRP	EIGRP
16	CHAOS	Chaos	89	OSPF	Open Shortest Path First
17	UDP	User Datagram Protocol	90	Sprite-RPC	Sprite RPC Protocol
18	MUX	Multiplexing	91	LARP	Locus Address Resolution Protocol
19	DCN-MEAS	DCN Measurement Subsystems	92	MTP	Multicast Transport Protocol
20	HMP	Host Monitoring Protocol	93	AX.25	AX.25
21	PRM	Packet Radio Measurement	94	OS	KA9Q NOS compatible IP over IP tunneling
22	XNS-IDP	XEROX NS IDP	95	MICP	Mobile Internetworking Control Protocol
23	TRUNK-1	Trunk-1	96	SCC-SP	Semaphore Communications Sec. Pro
24	TRUNK-2	Trunk-2	97	ETHERIP	Ethernet-within-IP Encapsulation
25	LEAF-1	Leaf-1	98	ENCAP	Encapsulation Header
26	LEAF-2	Leaf-2	99		Any private encryption scheme
27	RDP	Reliable Data Protocol	100	GMTP	GMTP
28	IRTP	Internet Reliable Transaction Protocol	101	IFMP	Ipsilon Flow Management Protocol
29	ISO-TP4	ISO Transport Protocol Class 4	102	PNNI	PNNI over IP
30	NETBLT	Bulk Data Transfer Protocol	103	PIM	Protocol Independent Multicast
31	MFE-NSP	MFE Network Services Protocol	104	ARIS	IBM's ARIS (Aggregate Route IP Switching) Protocol
32	MERIT-INP	MERIT Internodal Protocol	105	SCPS	SCPS (Space Communications Protocol Standards)
33	DCCP	Datagram Congestion Control Protocol	106	QNX	QNX

Продолжение таблицы 54

Номер	Ключевое имя	Протокол	Номер	Ключевое имя	Протокол
34	3PC	Third Party Connect Protocol	107	A/N	Active Networks
35	IDPR	Inter-Domain Policy Routing Protocol	108	IPComp	IP Payload Compression Protocol
36	XTP	Xpress Transport Protocol	109	SNP	Sitara Networks Protocol
37	DDP	Datagram Delivery Protocol	110	Compaq-Peer	Compaq Peer Protocol
38	IDPR-CMTP	IDPR Control Message Transport Protocol	111	IPX-in-IP	IPX in IP
39	TP++	TP++ Transport Protocol	112	VRRP	Virtual Router Redundancy Protocol, Common Address Redundancy Protocol (not IANA assigned)
40	IL	IL Transport Protocol	113	PGM	PGM Reliable Transport Protocol
41	IPv6	IPv6 Encapsulation (6to4 and 6in4)	114		Any 0-hop protocol
42	SDRP	Source Demand Routing Protocol	115	L2TP	Layer Two Tunneling Protocol Version 3
43	IPv6-Route	Routing Header for IPv6	116	DDX	D-II Data Exchange (DDX)
44	IPv6-Frag	Fragment Header for IPv6	117	IATP	Interactive Agent Transfer Protocol
45	IDRP	Inter-Domain Routing Protocol	118	STP	Schedule Transfer Protocol
46	RSVP	Resource Reservation Protocol	119	SRP	SpectraLink Radio Protocol
47	GRE	Generic Routing Encapsulation	120	UTI	Universal Transport Interface Protocol
48	DSR	Dynamic Source Routing Protocol	121	SMP	Simple Message Protocol
49	BNA	Burroughs Network Architecture	122	SM	Simple Multicast Protocol
50	ESP	Encapsulating Security Payload	123	PTP	Performance Transparency Protocol
51	AH	Authentication Header	124	IS-IS over IPv4	Intermediate System to Intermediate System (IS-IS) Protocol over IPv4
52	I-NLSP	Integrated Net Layer Security Protocol	125	FIRE	Flexible Intra-AS Routing Environment

Продолжение таблицы 54

Номер	Ключевое имя	Протокол	Номер	Ключевое имя	Протокол
53	SwIPe	SwIPe	126	CRTP	Combat Radio Transport Protocol
54	NARP	NBMA Address Resolution Protocol	127	CRUDP	Combat Radio User Datagram
55	MOBILE	IP Mobility (Min Encap)	128	SSCOPMCE	Service-Specific Connection-Oriented Protocol in a Multilink and Connectionless Environment
56	TLSP	Transport Layer Security Protocol (using Kryptonet key management)	129	IPLT	
57	SKIP	Simple Key-Management for Internet Protocol	130	SPS	Secure Packet Shield
58	IPv6-ICMP	ICMP for IPv6	131	PIPE	Private IP Encapsulation within IP
59	IPv6-NoNxt	No Next Header for IPv6	132	SCTP	Stream Control Transmission Protocol
60	IPv6-Opts	Destination Options for IPv6	133	FC	Fibre Channel
61		Any host internal protocol	134	RSVP-E2E-IGNORE	Reservation Protocol (RSVP) End-to-End Ignore
62	CFTP	CFTP	135	Mobility Header	Mobility Extension Header for IPv6
63		Any local network	136	UDPLite	Lightweight User Datagram Protocol
64	SAT-EXPAK	SATNET and Backroom EXPAK	137	MPLS-in-IP	Multiprotocol Label Switching Encapsulated in IP
65	KRYPTOLAN	Kryptolan	138	manet	MANET Protocols
66	RVD	MIT Remote Virtual Disk Protocol	139	HIP	Host Identity Protocol
67	IPPC	Internet Pluribus Packet Core	140	Shim6	Site Multihoming by IPv6 Intermediation
68		Any distributed file system	141	WESP	Wrapped Encapsulating Security Payload
69	SAT-MON	SATNET Monitoring	142	ROHC	Robust Header Compression
70	VISA	VISA Protocol	143	Ethernet	Segment Routing over IPv6
71	IPCU	Internet Packet Core Utility	144	AGGFRAG	AGGFRAG Encapsulation Payload for ESP
72	CPNX	Computer Protocol Network Executive	145	NSH	Network Service Header

Окончание таблицы 54

Номер	Ключевое имя	Протокол	Номер	Ключевое имя	Протокол
73	CPHB	Computer Protocol Heart Beat			

Пример `ip_proto` в правиле:

```
alert ip any any -> any any (msg:"GPL MISC IP Proto 103 PIM"; ip_proto:103; reference:bugtraq,8211; reference:cve,2003-0567; classtype:non-standard-protocol; sid:2102189; rev:4;)
```

С использованием ключевого наименования протокола, вариант примера будет выглядеть следующим образом:

```
ip_proto:PIM
```

11.9.10.5. IPv4.HDR/ IPv6.HDR

Липкий буфер для совмещения всего заголовка IPv4/IPv6.

Пример правила:

```
alert ip any any -> any any (ipv4.hdr; content:"|3A|"; offset:9; depth:1; sid:1234; rev:5;)
```

В этом примере проверяется, имеют ли 9 байт заголовка IPv4 значение `3A`, что означает, что протокол IPv4 – ICMPv6.

11.9.10.6. Идентификатор (`id`)

Ключевое слово `id` используется для проверки наличия в поле IP ID заданного значения. Идентификатор идентифицирует каждый пакет, отправленный узлом, и увеличивается на единицу, как правило, с каждым отправляемым пакетом. Идентификатор IP используется в качестве фрагмента идентификационного номера. Каждый пакет имеет идентификатор IP-адреса, и когда пакет становится фрагментированным, все фрагменты этого пакета имеют один и тот же идентификатор. Таким образом, получатель пакета знает, какие фрагменты принадлежат какому пакету.

Синтаксис:

```
id:<number>;
```

Некоторые программы (эксплойты, сканеры, старые программы) устанавливают в этом поле определенное значение, например, часто используется число 31337.

11.9.10.7. GeoIP

Ключевое слово `geoip` позволяет проверить и сопоставить источник, место назначения или исходные и конечные IPv4-адреса сетевого трафика, а также следить за тем, какой стране он принадлежит. Поддерживаются только IPv4-адреса.

Синтаксис:

```
geoip: src,RU;
geoip: both,CN,RU;
geoip: dst,CN,RU,IR;
geoip: both,US,CA,UK;
geoip: any,CN,IR;
```

Соответственно можно проверить информацию, приведенную в таблице 55.

Таблица 55

Параметр	Описание
`Both`	Оба направления совпадают с заданными `geoip`
`Any`	Одно из направлений совпадает с заданным `geoip`
`Dest`	Пункт назначения совпадает с заданным `geoip`
`Src`	Источник трафика совпадает с заданным `geoip`

11.9.10.8. `fragbits` (фрагментация IP)

С помощью ключевого слова `fragbits` можно проверить, есть ли в IP-заголовке биты фрагментации и резервные биты. Ключевое слово `fragbits` должно помещаться в начале правила. При маршрутизации сообщений из одной сети интернета к другой, может случиться так, что пакет больше, чем максимальный размер пакета, который может обработать сеть. В этом случае пакет может быть отправлен фрагментарно. Этот максимальный размер пакета называется Максимальная единица передачи (`MTU`).

Для проверки можно использовать следующие опции:

- `M – More Fragments` – проверять бит MF;

- D – Do not Fragment – проверять бит запрета фрагментации;

- R – Reserved Bit – проверять резервный бит.

Для изменения характера проверки могут использоваться перечисленные ниже модификаторы:

+ соответствует, если установлены указанные биты;

* соответствует, если установлен какой-либо из указанных битов;

! соответствует, если не установлен ни один из указанных битов.

Синтаксис: `fragbits:[*+!]<[MDR]>;`

Пример в правиле, проверяется установка флага `More Fragments`:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:”ET EXPLOIT
Invalid non-fragmented packet with fragment offset>0”; fragbits: M;
fragoffset: >0;
reference:url,doc.emergingthreats.net/bin/view/Main/2001022;
classtype:bad-unknown; sid:2001022; rev:5; metadata:created_at
2010_07_30, updated_at 2010_07_30;)
```

11.9.10.9. fragoffset

С помощью ключевого слова `fragoffset` можно сравнить смещение фрагмента дейтаграммы IP с заданным десятичным значением. Для отсечения всех первых фрагментов можно использовать ключевое слово `fragbits` и просмотр опции `More fragments` при установке `fragoffset: 0`.

Идентификатор(`id`) используется для определения того, какие фрагменты принадлежат какому пакету, а поле смещения фрагментации проясняет порядок фрагментов.

Доступны следующие модификаторы:

- < соответствует, если значение меньше указанного значения;

- > соответствует, если значение больше указанного значения;

- ! соответствует, если указанное значение отсутствует.

Синтаксис: `fragoffset:[!|<|>]<number>;`

Пример в правиле:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:”ET EXPLOIT
Invalid non-fragmented packet with fragment offset>0”; fragbits: M;
fragoffset: >0;
reference:url,doc.emergingthreats.net/bin/view/Main/2001022;
classtype:bad-unknown; sid:2001022; rev:5; metadata:created at
2010_07_30, updated_at 2010_07_30;)
```

11.9.10.10. tos

Ключевое слово `tos` позволяет проверять в пакетах поле IP TOS (тип обслуживания), может совпадать с десятичным значением поля `TOS` в IP-заголовке. Ключевое слово `tos` может иметь значение от 0 до 255. Обратите внимание, что значение поля было определено с двумя крайними правыми битами, имеющими значение 0. При указании значения для `tos` убедитесь, что значение следует за ними.

Например, вместо указания десятичного значения 34 (шестнадцатеричное число 22) дважды сдвиньте вправо и используйте десятичное число 136 (шестнадцатеричное число 88).

Можно указать и шестнадцатеричные значения, с `x` в начальне, например, x88.

Синтаксис:

```
tos:[!]<number>;
```

Пример в правиле:

```
alert ip any any -> any any (msg:"Differentiated Services Codepoint: Class Selector 1 (8)"; flow:established; tos:8; classtype:not-suspicious; sid:2600115; rev:1;)
```

Пример `tos` с отрицательными значениями:

```
alert ip any any -> any any (msg:"TGI HUNT non-DiffServ aware TOS setting"; flow:established,to_server; tos:!0; tos:!8; tos:!16; tos:!24; tos:!32; tos:!40; tos:!48; tos:!56; threshold:type limit, track by_src, seconds 60, count 1; classtype:bad-unknown; sid:2600124; rev:1;)
```

В приведенном примере проверяется отличие значения поля TOS от 0, 8, 16, 24, 32, 40, 48, 56.

11.9.11. Ключевые слова TCP

11.9.11.1. seq

Ключевое слово `seq` можно использовать в подписи для проверки порядкового номера TCP. Порядковый номер – это число, которое генерируется практически случайным образом обеими конечными точками TCP-соединения. И клиент, и сервер создают порядковый номер, который увеличивается на единицу с

каждым отправляемым байтом. Таким образом, этот порядковый номер отличается для обеих сторон. Этот порядковый номер должен быть подтвержден обеими сторонами соединения.

Через порядковые номера TCP обрабатывает подтверждение, порядок и повторную передачу. Его число увеличивается с каждым байтом данных, отправленным отправителем. Последовательность помогает отслеживать, к какому месту в потоке данных относится байт. Если флаг `SYN` установлен на 1, то порядковый номер первого байта данных равен этому числу плюс 1 (то есть 2).

Синтаксис:

```
seq:0;
```

Пример `seq` в подписи, проверяет равенство порядкового номера TCP нулю:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL SCAN NULL"; flow:stateless; ack:0; flags:0; seq:0; reference:arachnids,4; classtype:attempted-recon; sid:2100623; rev:7;)
```

11.9.11.2. ack

`Ack` используется в правиле для проверки номеров подтверждений TCP.

Синтаксис:

```
ack:1;
```

Пример в подписи:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL SCAN NULL"; flow:stateless; ack:0; flags:0; seq:0; reference:arachnids,4; classtype:attempted-recon; sid:2100623; rev:7;)
```

11.9.11.3. window

Ключевое слово `window` используется для проверки размера окна TCP. Размер окна TCP – это механизм, который управляет потоком данных. Размер окна TCP – это механизм, управляющий потоком данных. Окно устанавливается получателем (размер окна, объявленный получателем) и указывает количество байтов, которое может быть получено. Этот объем данных должен быть сначала подтвержден получателем, прежде чем отправитель сможет отправить такое же количество новых данных. Этот механизм используется для предотвращения переполнения приемника данными. Значение размера окна ограничено и может составлять от 2 до 65,535

байт. Чтобы лучше использовать пропускную способность, можно использовать большее TCP-окно.

Синтаксис:

```
window:[!]<number>;
```

Пример в правиле:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL DELETED typot trojan traffic"; flow:stateless; flags:S,12; window:55808; reference:mcafee,100406; classtype:trojan-activity; sid:2182; rev:8;)
```

11.9.11.4. tcp.mss

Ключевое слово осуществляет проверку по значению параметра TCP MSS, которая не будет осуществлена, если параметр отсутствует.

Синтаксис:

```
tcp.mss:<min>-<max>;
tcp.mss:[<|>]<number>;
tcp.mss:<value>;
```

Пример правила:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (flow:stateless; flags:S,12; tcp.mss:<536; sid:1234; rev:5;)
```

11.9.11.5. tcp.hdr

Липкий буфер для совмещения всего заголовка TCP.

Пример правила:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (flags:S,12; tcp.hdr; content:"|02 04|"; offset:20; byte test:2,<,536,0,big,relative; sid:1234; rev:5;)
```

В этом примере начинается проверка фиксированной части заголовка, поэтому осуществляется переход к параметрам переменного размера, будет выполнятся поиск MSS ((тип 2, параметр len 4) и с помощью `byte_test` будет определено, меньше ли значение параметра, чем 536. Параметр `tcp.mss` будет более эффективен в случаях, когда нет иных указанных ключевых слов.

11.9.12. Ключевые слова UDP

11.9.12.1. udp.hdr

Липкий буфер для совмещения всего заголовка UDP.

Пример правила:

```
alert udp any any -> any any (udp.hdr; content:"|00 08|"; offset:4; depth:2; sid:1234; rev:5;)
```

Этот пример соответствует полю длины заголовка UDP. В этом случае длина 8 означает, что полезной нагрузки нет. Это также можно проверить с помощью `dsize:0;`.

11.9.13. Ключевые слова ICMP

ICMP (Internet Control Message Protocol) является частью IP, который сам по себе ненадежен, когда дело доходит до доставки данных (дейтаграммы). ICMP дает обратную связь в случае возникновения проблем, что не предотвращает от происходящих проблем, но помогает в понимании того, что пошло не так и где. Если необходима надежность, протоколы, использующие IP, должны сами об этом позаботиться. В разных некоторых ситуациях будет отправлено ICMP-сообщение. Например, когда пункт назначения недоступен, если буферная емкость недостаточна для пересылки данных, или, когда дейтаграмма отправляется фрагментированной, когда этого не должно быть, и так далее.

Есть четыре важных части сообщения ICMP, по которым можно осуществить проверку, это: тип, код, идентификатор и последовательность сообщения.

11.9.13.1. itype

Ключевое слово `itype` предназначено для проверки типа сообщения ICMP (таблица 56). ICMP имеет несколько видов сообщений и использует коды для их уточнения. Разные сообщения отличаются разными именами, но важнее числовые значения.

Синтаксис:

```
itype:min<>max;
itype:[<|>]<number>;
```

Пример, выполняется проверка наличия типа сообщения ICMP больше 10:

```
itype:>10;
```

Пример в подписи:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL SCAN Broadscan Smurf Scanner"; dsize:4; icmp_id:0; icmp_seq:0; itype:8; classtype:attempted-recon; СИД:2100478; rev:4;)
```

Таблица 56

Тип	Сообщение	Перевод/примечание
0	Echo Reply	Эхо-ответ
1	Unassigned	Не назначено
2	Unassigned	Не назначено
3	Destination Unreachable	Пункт назначения недоступен
4	Source Quench	Сдерживание источника (устарело)
5	Redirect	Перенаправить
6	Alternate Host Address	Альтернативный адрес хоста (устарело)
7	Unassigned	Не назначено
8	Echo	Эхо
9	Router Advertisement	Объявление маршрутизатора
10	Router Solicitation	Запрос маршрутизатора
11	Time Exceeded	Время истекло
12	Parameter Problem	Параметр проблемы
13	Timestamp	Отметка времени
14	Timestamp Reply	Отметка времени ответа
15	Information Request	Запрос информации (устарело)
16	Information Reply	Информационный ответ (устарело)
17	Address Mask Request	Запрос маски адреса (устарело)
18	Address Mask Reply	Ответ маски адреса (устарело)
19	Reserved (for Security)	Зарезервировано (для безопасности)
20-29	Reserved (for Robustness Experiment)	Зарезервировано (для эксперимента по устойчивости)
30	Traceroute	Трассировка (устарело)
31	Datagram Conversion Error	Ошибка преобразования дейтаграммы (устарело)
32	Mobile Host Redirect	Перенаправление мобильного хоста (устарело)
33	IPv6 Where-Are-You	IPv6 «Где ты» (устарело)
34	IPv6 I-Am-Here	IPv6 «Я здесь» (устарело)
35	Mobile Registration Request	Запрос на мобильную регистрацию (устарело)
36	Mobile Registration Reply	Ответ на мобильную регистрацию (устарело)
37	Domain Name Request	Запрос доменного имени (устарело)
38	Domain Name Reply	Ответ доменного имени (устарело)
39	SKIP	Пропустить (устарело)
40	Photuris	Photuris – сеансовый ключ протокол управления

Окончание таблицы 56

Тип	Сообщение	Перевод/примечание
41	ICMP messages utilized by experimental mobility protocols such as Seamoby	Сообщения ICMP, используемые экспериментальными мобильными протоколами, такими как Seamoby
42	Extended Echo Request	Расширенный эхо-запрос
43	Extended Echo Reply	Расширенный эхо-ответ
44-252	Unassigned	Неназначенно
253	RFC3692-style Experiment 1	Эксперимент 1 RFC3692
254	RFC3692-style Experiment 2	Эксперимент 2 RFC3692
255	Reserved	Резервный

11.9.13.2. icode

С помощью ключевого слова `icode` можно проверить значение кода ICMP. Код ICMP-сообщения разъясняет сообщение. Вместе с ICMP-типом он указывает на то, какая проблема произошла. Код имеет разное назначение для каждого типа ICMP.

Синтаксис:

```
icode:[<|>]<number>;
```

В этом примере выполняется проверка наличия кода ICMP больше 5:

```
icode:>5;
```

Пример ключевого слова `icode` в правиле:

```
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg:"GPL MISC time-to-life exceeded in transit"; icode:0; itype:11; classtype:misc-activity; СИД:2100449; rev:7;)
```

Ниже в таблице 57 приведен список значений всех типов ICMP. Если код отсутствует в списке, определен только тип 0, который имеет значение кода ICMP в таблице 56.

Таблица 57

Код ICMP	Тип ICMP	Описание	Перевод
3	0	Net Unreachable	Сеть недостижима
	1	Host Unreachable	Хост недоступен
	2	Protocol Unreachable	Протокол недоступен
	3	Port Unreachable	Порт недоступен
	4	Fragmentation Needed and Don't Fragment was Set	Было установлено: требуется фрагментация/не фрагментировать
	5	Source Route Failed	Сбой исходного маршрута
	6	Destination Network Unknown	Сеть назначения неизвестна
	7	Destination Host Unknown	Хост назначения неизвестен
	8	Source Host Isolated	Изолированный исходный узел
	9	Communication with Destination Network is Administratively Prohibited	Связь с сетью назначения административно запрещена
	10	Communication with Destination Host is Administratively Prohibited	Связь с хостом назначения запрещена в административном порядке
	11	Destination Network Unreachable for Type of Service	Конечная сеть недоступна для типа услуги
	12	Destination Host Unreachable for Type of Service	Хост назначения недоступен для типа службы
	13	Communication Administratively Prohibited	Коммуникация административно запрещена
	14	Host Precedence Violation	Нарушение приоритета хоста
	15	Precedence cutoff in effect	Действующее ограничение приоритета
4	0	Redirect Datagram for the Network (or subnet)	Дейтаграмма перенаправления для сети (или подсети)
	1	Redirect Datagram for the Host	Дейтаграмма перенаправления для хоста
	2	Redirect Datagram for the Type of Service and Network	Дейтаграмма перенаправления для типа службы и сети
	3	Redirect Datagram for the Type of Service and Host	Дейтаграмма перенаправления для типа службы и хоста
6	0	Alternate Address for Host	Альтернативный адрес для хоста
9	0	Normal router advertisement	Обычное объявление роутера
	16	Does not route common traffic	Не маршрутизирует общий трафик
11	0	Time to Live exceeded in Transit	Время жизни превышено в пути
	1	Fragment Reassembly Time Exceeded	Превышено время сборки фрагмента
12	0	Pointer indicates the error	Указатель указывает на ошибку
	1	Missing a Required Option	Отсутствует обязательная опция
	2	Bad Length	Плохая длина
40	0	Bad SPI	Плохой SPI
	1	Authentication Failed	Ошибка аутентификации
	2	Decompression Failed	Декомпрессия не удалась
	3	Decryption Failed	Сбой расшифровки
	4	Need Authentication	Требуется аутентификация
	5	Need Authorization	Требуется авторизация

Окончание таблицы 57

Код ICMP	Тип ICMP	Описание	Перевод
42	0	No Error	Нет ошибки
	1-255	Unassigned	Не назначено
43	0	No Error	Нет ошибки
	1	Malformed Query	Неверный запрос
	2	No Such Interface	Нет такого интерфейса
	3	No Such Table Entry	Нет такой записи в таблице
	4	Multiple Interfaces Satisfy Query	Несколько интерфейсов удовлетворяют запросу
	5-255	Unassigned	Не назначено

11.9.13.3. icmp_id

С помощью ключевого слова `icmp_id` можно проверить значение идентификатора ICMP. Каждый ICMP-пакет получает идентификатор при отправке. Если получатель получил пакет, то он отправит ответ, используя тот же `id`, чтобы отправитель узнал его и связал с правильным ICMP-запросом.

Синтаксис:

```
icmp_id:<number>;
```

В этом примере выполняется проверка наличия нулевого значения в поле ICMP ID:

```
icmp_id:0;
```

Пример в правиле:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL SCAN Broadscan Smurf Scanner"; dsize:4; icmp_id:0; icmp_seq:0; itype:8; classtype:attempted-recon; СИД:2100478; rev:4;)
```

11.9.13.4. icmp_seq

Ключевое слово `icmp_seq` можно использовать для проверки порядкового номера ICMP. Все сообщения ICMP имеют порядковые номера, что совместно с идентификатором может быть полезно, для проверки того, какое ответное сообщение принадлежит какому сообщению запроса.

Синтаксис:

```
icmp_seq:<number>;
```

В этом примере выполняется проверка наличия сообщений ICMP с порядковым номером 0:

```
icmp_seq:0;
```

Пример в правиле:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"GPL SCAN Broadscan Smurf Scanner"; dsize:4; icmp_id:0; icmp_seq:0; itype:8; classtype:attempted-recon; СИД:2100478; rev:4;)
```

11.9.13.5. ICMPv6.HDR

Липкий буфер для совмещения всего заголовка ICMPv6.

11.9.13.6. icmpv6.mtu

Сопоставление с дополнительным значением MTU ICMPv6. Не будет соответствовать, если MTU отсутствует.

Синтаксис:

```
icmpv6.mtu:<min>-<max>;
icmpv6.mtu:[<|>]<number>;
icmpv6.mtu:<value>;
```

Пример правила:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (icmpv6.mtu:<1280; sid:1234; rev:5;)
```

11.9.14. Ключевые слова `payload`

Ключевые слова `payload` проверяют полезную нагрузку – поле данных пакета.

11.9.14.1. content

Ключевое слово `content` позволяет задавать в правиле проверку пакетов на содержание определенной информации.

Синтаксис:

```
content: ".....";
```

В правиле можно использовать несколько ключевых слов `content` для уточнения искомой информации.

Если информация, заданная content, будет обнаружена в поле данных пакета, то будет выполнятся следующая часть правила.

При записи учитывайте регистр символов.

Кроме печатных символов, есть еще специальные знаки, для их обозначения используются шестнадцатеричные обозначения. Многие языки программирования используют в качестве обозначения 0x00, где 0x означает, что речь идет о двоичном значении, однако язык правил использует |00| как обозначение. Этот тип записи также может использоваться для символов.

Пример:

```
|61| это a
|61 61| это aa
|41| это A
|21| это !
|0D| это возврат каретки (CR) ↵
|0A| это перевод строки (LF) \n
```

Есть символы, которые нельзя использовать в content, потому что они важны в правиле. Например, символы ; \ " должны быть экранированы, или для добавления этих символов в строку поиска необходимо использовать шестнадцатеричную нотацию в верхнем регистре:

```
"     |22|
;     |3B|
:     |3A|
|     |7C|
\     |5C|
```

Синтаксис для записи http:// в строке правила: content: "http|3A|//"; Если используется шестнадцатеричная нотация в подписи, убедитесь, что она всегда помещается между ||. В противном случае обозначение будет восприниматься буквально как часть содержания.

Несколько примеров:

```
content:"a|0D|bc";
content:"|61 0D 62 63|";
content:"a|0D|b|63|";
```

Правило может проверять всю полезную нагрузку на соответствие данным, указанным с помощью content или можно позволить ей проверять только

определенные части полезной нагрузки. По умолчанию правило попытается найти совпадение во всех байтах полезной нагрузки.

```
drop tcp $HOME_NET any -> $EXTERNAL_NET any \
(msg:"ET TROJAN Likely Bot Nick in IRC (USA +..)"; \
 flow:established,to_server; \
 flowbits:isset,is_proto_irc; content:"NICK "; \
pcre:"/NICK.*USA.*[0-9]{3,}/i"; \
reference:url,doc.emergingthreats.net/2008124; \
 classtype:trojan-activity; sid:2008124; rev:2;)
```

По умолчанию выполняемая проверка чувствительна к регистру, поэтому будьте точны, иначе совпадений не будет (рис. 194).

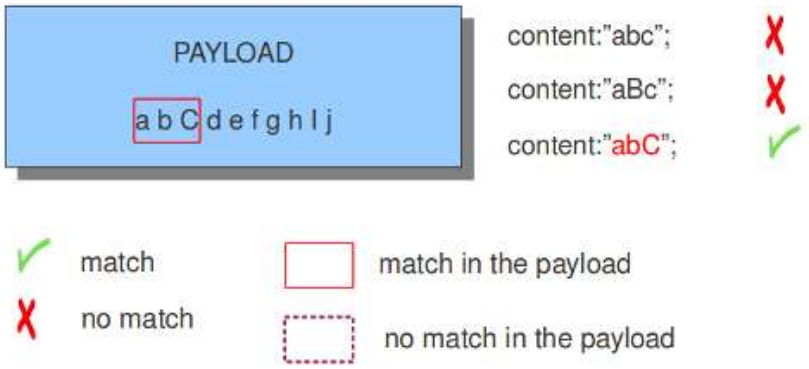


Рис. 194

Если перед строкой поиска помещен знак отрицания (!), правилу будут соответствовать пакеты, не содержащие указанных данных:

```
content: [!]"строка";
```

Например:

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Outdated Firefox on Windows"; content:"User-Agent|3A| Mozilla/5.0 |28|Windows|3B| "; content:"Firefox/3."; distance:0; content:!"Firefox/3.6.13"; distance:-10; sid:9000000; rev:1;)
```

`content:!"Firefox/3.6.13";` – означает, что предупреждение будет генерироваться, если используемая версия Firefox не 3.6.13.

11.9.14.2. nocase

Если не хотите делать различие между прописными буквами и строчными буквами, можно использовать `nocase`. Ключевое слово `nocase` – это модификатор контента.

Синтаксис:

```
nocase;
```

Размещать его нужно после контента `content`, который хотите изменить, например, рис. 195:

```
content: "abc"; nocase;
```

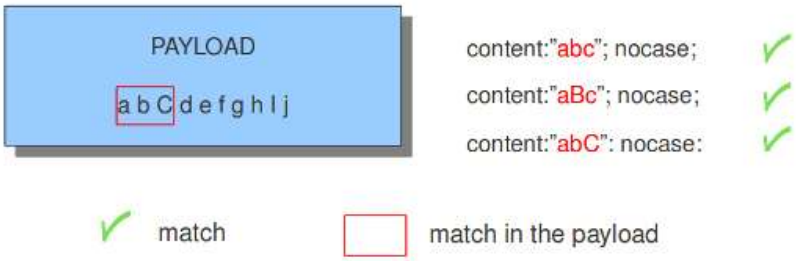


Рис. 195

11.9.14.3. depth

Ключевое слово `depth` – размер области поиска, является абсолютным модификатором содержания (`content`), следует за содержанием. Модификатор `depth` имеет обязательное числовое значение, например:

```
depth:12;
```

Число после `depth` обозначает, сколько байтов от начала поля данных пакета будет проверено (рис. 196).

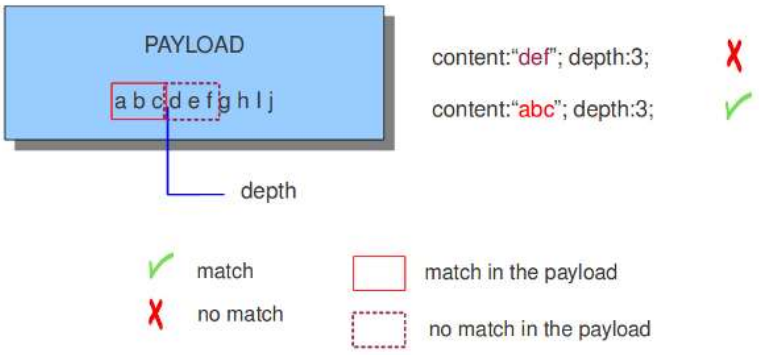


Рис. 196

11.9.14.4. startwith

Ключевое слово `startwith` похоже на `depth`. Оно не принимает аргументов и должно следовать за ключевым словом `content`, изменяя его, чтобы точно соответствовать началу буфера.

Пример:

```
content:"GET|20|"; startswith;
```

`startswith` это сокращенная нотация для:

```
content:"GET|20|"; depth:4; offset:0;
```

`startswith` нельзя смешивать с `offset`, `within` или `distance` в одном и том же шаблоне.

11.9.14.5. endswith

Ключевое слово `endwith` похоже на `isdataat:!1,relative;`. Оно не принимает аргументов и должно следовать за ключевым словом `content`. Ключевое слово `endwith` изменяет `content`, чтобы точно соответствовать концу буфера.

Пример:

```
content:".php"; endswith;
```

`endswith` это краткое обозначение для:

```
content:".php"; isdatat:!1,relative;
```

`endswith` нельзя смешивать с `offset`, `within` или `distance` в одном и том же шаблоне.

11.9.14.6. offset

Ключевое слово `offset` указывает смещение – с какого байта поля данных пакета начнется проверка. Например, `offset: 3;` проверяет четвертый байт и далее (рис. 197).

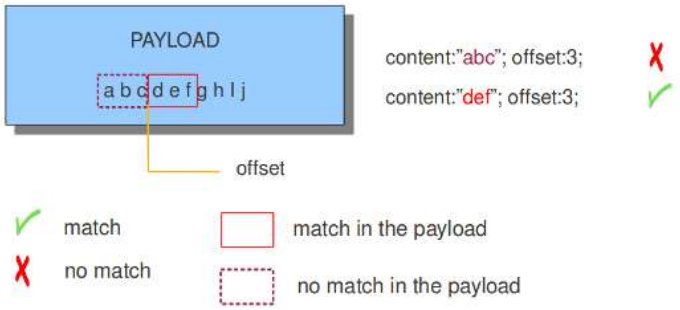


Рис. 197

Ключевые слова `offset` и `depth` могут быть объединены и часто используются вместе.

Например, если использовать следующую запись в правиле, будет проверяться полезная нагрузка от третьего байта до шестого байта (рис. 198):

```
content:"def"; offset:3; depth:3;
```

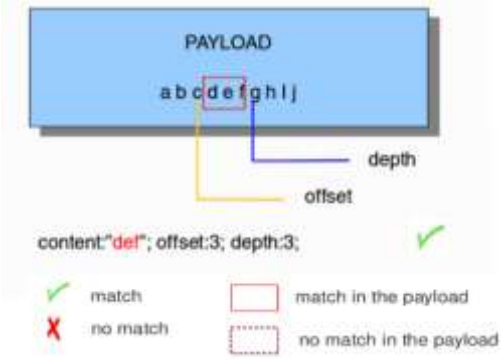


Рис. 198

11.9.14.7. distance

Ключевое слово `distance` является модификатором `content`, количество пропускаемых байтов после первого найденного соответствия. Ключевое слово `distance` имеет обязательное числовое значение, которое определяет байт в поле данных пакета (payload), относительно которого будет проверяться новое совпадение, заданное следующим `content`. `distance: 5;` означает, что шаблон может быть где угодно после предыдущего совпадения + 5 байтов. Чтобы ограничить, как далеко после последнего совпадения должна искать СОВ, используйте ключ `within`.

Примеры приведены на рис. 199 – 202.

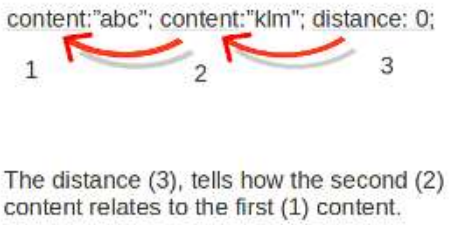


Рис. 199

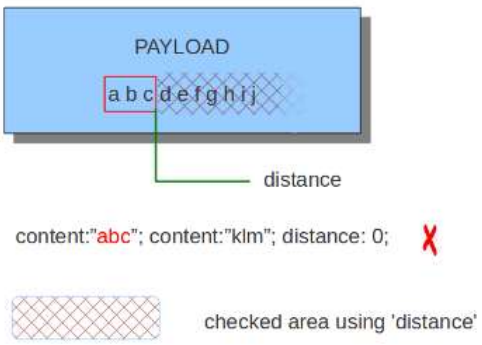


Рис. 200

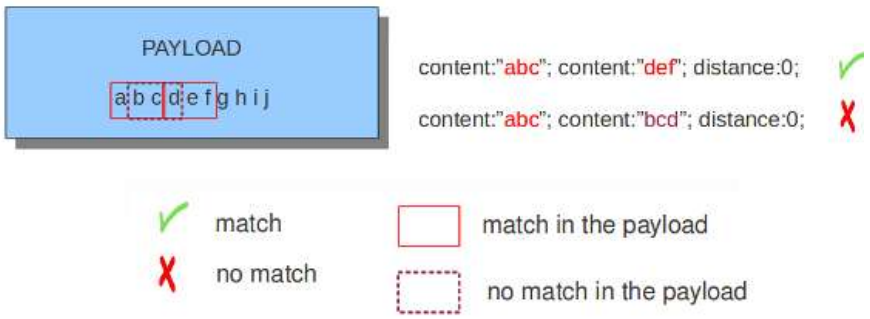


Рис. 201

Пример проверки наличия в поле данных пакета, строк abc и def (1 строка), и строки вида abcxxxxdef (2 строка), где x означает любой символ (рис. 202):

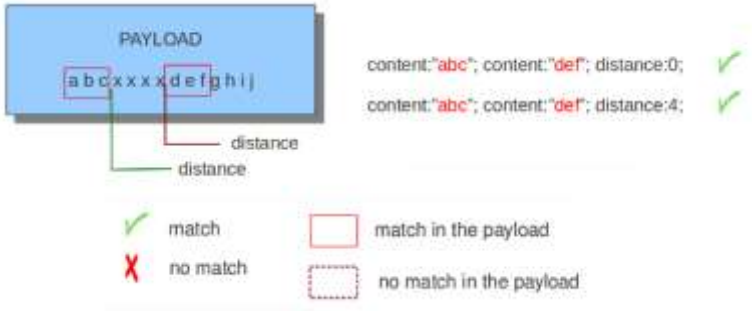


Рис. 202

`Distance` также может быть отрицательным числом (рис. 203). Его можно использовать для проверки на совпадения с частично одинаковым содержимым (см. примеры выше) или даже с полностью предшествующим содержимым. Таких же результатов можно добиться и с другими ключевыми словами.

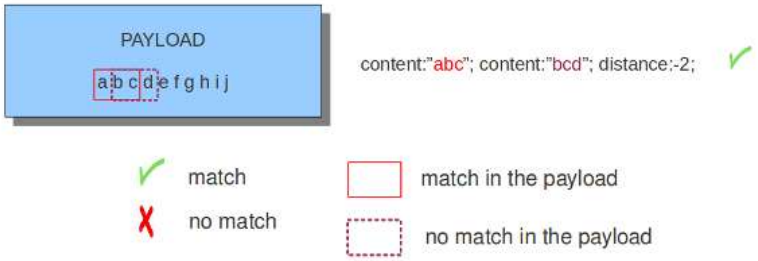


Рис. 203

11.9.14.8. within

Ключевое слово `within` – размер области поиска для `content` после первого найденного соответствия от предыдущего `content`. Ключевое слово `within` имеет обязательное числовое значение и не может быть 0. Использование `within` гарантирует, что соответствие правилу будет только в том случае, если `content` совпадает с содержимым поля данных пакета в пределах установленного количества байтов.

Пример проверки с использованием `within` представлен на рис. 204. Содержимое второго `content` должно находиться «в пределах 3» от первого `content`, поэтому второй вариант записи не найдет совпадений.

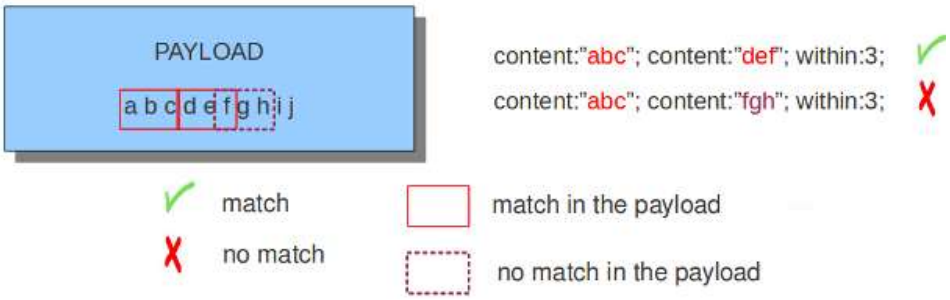


Рис. 204

Как упоминалось ранее, `distance` и `within` можно очень хорошо сочетать в правиле (рис. 205). Если необходимо проверить только определенную часть, то используйте `within`.

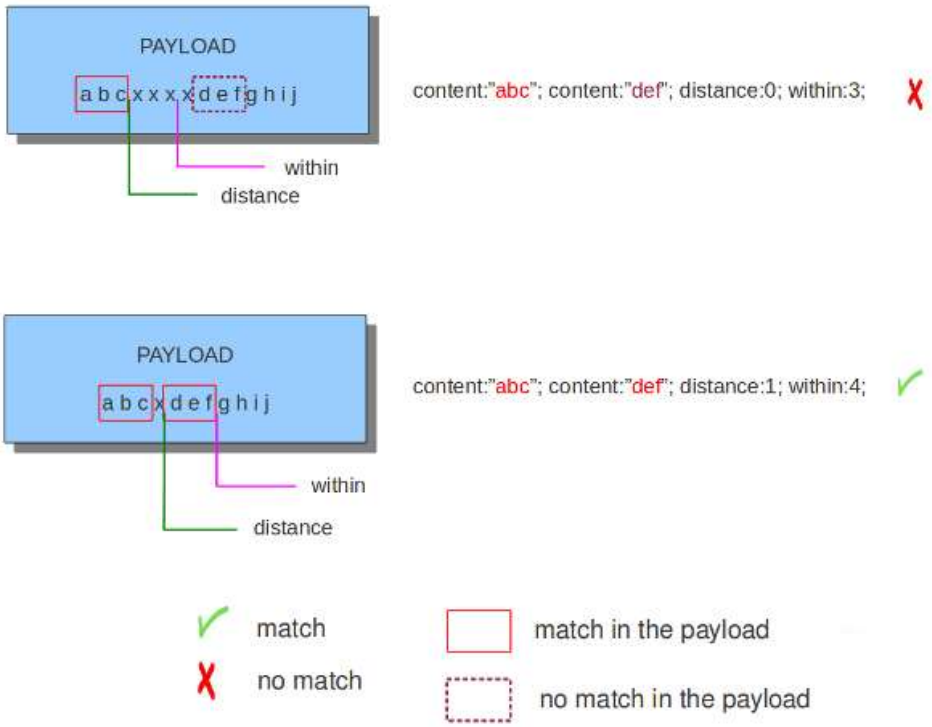


Рис. 205

11.9.14.9. isdataat

Ключевое слово `isdataat` находит и сравнивает данные в заданном участке пакета, возможно относительно завершения подстроки, найденной с помощью `content`. Значение `isdataat` задает позицию проверки. Если за ним следует ключевое слово `relative`, то проверка осуществляется в следующем фрагменте содержимого пакета заданной величины относительно последнего найденного совпадения.

Синтаксис:

```
isdataat:512;
isdataat:50, relative;
```

Первый пример иллюстрирует правило, которое исследует 512 байт содержимого пакета. Второй пример иллюстрирует правило, исследующее 50 байт после последнего совпадения.

Также можно использовать отрицание (!) перед `isdataat`.

Еще пример использования (рис. 206).

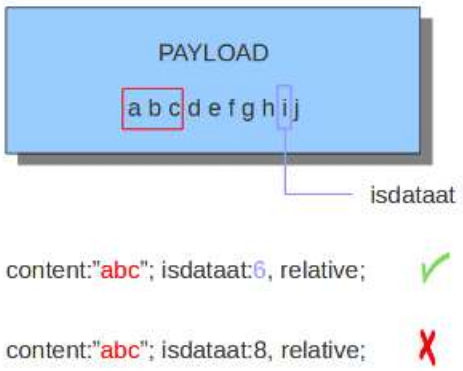


Рис. 206

11.9.14.10. bsize

С помощью ключевого слова `bsize` можно проверить размер буфера. Это повышает точность проверки содержимого, ранее это можно было сделать с помощью `isdataat`.

Синтаксис:

```
bsize:<число_байт>;
```

Пример:

```
alert dns any any -> any any (msg:”test bsize rule”; dns.query; content:”google.com”; bsize:10; sid:123; rev:1;)
```

11.9.14.11. dsize

С помощью ключевого слова `dsize` можно проверить размер поля данных пакета. Можно использовать это ключевое слово, например, для поиска аномальных размеров пакетов, которые могут быть при переполнении буфера.

Синтаксис:

```
dsize: [<>]<число_байт>;
```

Пример:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 65535 (msg:”GPL DELETED EXPLOIT LANDesk Management Suite Alerting Service buffer overflow”; dsize:>268; reference: bugtraq,23483; reference: cve,2007-1674; classtype: attempted-admin; sid:100000928; rev:1;)
```

11.9.14.12. byte_test

Ключевое слово позволяет сравнить байт с заданным значением, может использоваться применительно к двоичным значениям или их символьному представлению (таблица 58).

Синтаксис:

```
byte_test:<num of bytes>, [!]<operator>, <test value>, <offset> \
[,relative] [,<endian>][, string, <num type>][, dce] \
[, bitmask <bitmask value>];
```

Таблица 58

Параметр	Описание
<num of bytes>	Количество байт, считываемых из пакета
<operator>	Операция, выполняемая для сравнения байта с заданным значением. ! не совпадает < меньше > больше = равно <= меньше или равно >= больше или равно & побитовое И (AND) ^ побитовое ИЛИ (OR)
<value>	Значение, с которым выполняется сравнение (принимается шестнадцатеричное или десятичное число)
<offset>	Смещение в поле данных пакета, с которого начинается сравнение
[relative]	Отсчет смещения от конца предыдущего найденного соответствия
[endian]	Задает порядок следования: - big (старший разряд слева); - little (старший разряд справа).
[string] <num>	Тип считываемых значений: - hex – преобразованная строка, представленная в шестнадцатеричном формате; - dec – преобразованная строка, представленная в десятичном формате; - oct – преобразованная строка, представленная в восьмеричном формате.
[dce]	Разрешить модулю DCE определять порядок байтов
<bitmask>	Применяет оператор побитовое И (AND) к преобразованным байтам

Примеры:

```
alert tcp any any -> any any \
      (msg:"Byte_Test Example - Num = Value"; \
      content:"|00 01 00 02|"; byte_test:2,=,0x01;)

alert tcp any any -> any any \
      (msg:"Byte_Test Example - Num = Value relative to content"; \
content:"|00 01 00 02|"; byte_test:2,=,0x03,relative;)

alert tcp any any -> any any \
   (msg:"Byte_Test Example - Num != Value"; content:"|00 01 00 02|"; \
byte_test:2,!=,0x06;)

alert tcp any any -> any any \
      (msg:"Byte_Test Example - Detect Large Values"; \
     content:"|00 01 00 02|"; byte_test:2,>,1000,relative;)

alert tcp any any -> any any \
      (msg:"Byte_Test Example - Lowest bit is set"; \
      content:"|00 01 00 02|"; byte_test:2,&,0x01,relative;)

alert tcp any any -> any any (msg:"Byte_Test Example - Compare to
String"; \
      content:"foobar"; byte_test:4,=,1337,1,relative,string,dec;)
```

11.9.14.13. byte_math

Ключевое слово `byte_math` добавляет возможность выполнять математические операции над извлеченными значениями с существующей переменной или заданным значением.

Когда `relative` включен, должна быть предыдущая проверка `content` или `pcre`.

Результат может быть сохранен в переменной результата и использоваться другими параметрами позже в правиле (таблица 59).

Таблица 59

Ключевое слово	Модификатор
content	offset, depth, distance, within
byte_test	offset, value
byte_jump	offset
isdataat	offset

Синтаксис (таблица 60):

```
byte_math:bytes <num of bytes>, offset <offset>,  \
oper <operator>, rvalue <rvalue>,result <result_var>  \
[, relative] [, endian <endian>] [, string <number-type>] \
[, dce] [, bitmask <value>];
```

Таблица 60

Параметр	Описание
<num of bytes>	Количество байт, считываемых из пакета
<offset>	Смещение в поле данных пакета, с которого начинается сравнение
oper <operator>	Математическая операция для выполнения: +, -, *, /, <<, >>
<value>	Значение, с которым выполняется сравнение (принимается шестнадцатеричное или десятичное число)
rvalue <rvalue>	Значение для выполнения математической операции
result <result-var>	Хранит вычисленное значение
[relative]	Отсчет смещения от конца предыдущего найденного соответствия
[endian <type>]	Задает порядок следования: - big (старший разряд слева); - little (старший разряд справа).
[string <num_type>]	Тип считываемых значений: - hex – преобразованная строка, представленная в шестнадцатеричном формате; - dec – преобразованная строка, представленная в десятичном формате; - oct – преобразованная строка, представленная в восьмеричном формате.

Окончание таблицы 60

Параметр	Описание
[dce]	Разрешить модулю DCE определять порядок байтов
[bitmask] <value>	К извлеченному значению будет применен оператор побитовое И (AND). Результат будет сдвинут вправо на количество битов, равное количеству завершающих нулей в маске.

Синтаксис:

```
alert tcp any any -> any any \
  (msg:"Testing bytemath_body"; \
  content:"|00 04 93 F3|"; \
  content:"|00 00 00 07|"; distance:4; within:4; \
  byte_math:bytes 4, offset 0, oper +, rvalue \
  248, result var, relative;)

alert udp any any -> any any \
  (byte_extract: 1, 0, extracted_val, relative; \
  byte_math: bytes 1, offset 1, oper +, rvalue extracted_val, \
   result var;  byte_test: 2, =, var, 13; \
  msg:"Byte extract and byte math with byte test verification";)
```

11.9.14.14. byte_jump

Ключевое слово byte_jump определяет размер области данных в байтах <num of bytes> из <offset> и перемещает указатель, для следующего считывания информации из содержимого пакета.

Синтаксис (таблица 61):

```
byte_jump:<num of bytes>, <offset> [, relative] \
[, multiplier <mult_value>] [, <endian>][, string, <num_type>] \
[, align][, from_beginning][, from_end] \
     [, post_offset <value>][, dce][, bitmask <value>];
```

Таблица 61

Параметр	Описание
<num of bytes>	Количество байт, считываемых из пакета
<offset>	Смещение в поле данных пакета, с которого начинается сравнение
[relative]	Отсчет смещения от конца предыдущего найденного соответствия
[multiplier] <value>	Умножает количество вычисленных байтов на значение параметра <value> и пропускает полученное количество байтов.
[endian]	Задает порядок следования: - big (старший разряд слева); - little (старший разряд справа).
[string <num_type>]	Тип считываемых значений: - hex – преобразованная строка, представленная в шестнадцатеричном формате; - dec – преобразованная строка, представленная в десятичном формате; - oct – преобразованная строка, представленная в восьмеричном формате.
[align]	Округляет число конвертируемых байтов по следующей 32-битовой границе
[from_beginning]	Задает отсчет пропускаемых байтов от начала поля данных пакета, а не от текущей позиции в пакете.
[from_end]	Задает отсчет с конца поля данных пакета, а не от текущей позиции в пакете.
[post_offset] <value>	После выполнения операции перехода выполняется смещение на дополнительное количество байтов, указанное в <value>
[dce]	Разрешить модулю DCE определять порядок байтов
[bitmask] <value>	Оператор побитовое И (AND) будет применен к <value> и преобразованным байтам, затем будет выполнена операция перехода

Пример:

```
alert tcp any any -> any any \
      (msg:"Byte_Jump Example"; \
     content:"Alice"; byte_jump:2,0; content:"Bob";)
```

```
alert tcp any any -> any any \
     (msg:"Byte_Jump Multiple Jumps"; \
     byte_jump:2,0; byte_jump:2,0,relative; content:"foobar"; \
    distance:0; within:6;)

alert tcp any any -> any any \
     (msg:"Byte_Jump From the End -8 Bytes"; \
     byte_jump:0,0, from_end, post_offset -8; \
     content:"|6c 33 33 74|"; distance:0 within:4;)
```

11.9.14.15. byte_extract

Ключевое слово `byte_extract` извлекает <num of bytes> по определенному <offset> и сохраняет его в переменную <var_name>. Значение в <var_name> можно использовать в любом модификаторе, который принимает число как параметр, а в случае `byte_test` его можно использовать как значение.

Синтаксис (таблица 62, таблица 63):

```
byte_extract:<num of bytes>, <offset>, <var_name>, [,relative] \
[,multiplier <mult-value>][,<endian>] [, dce] [, string [, <num_type>] \
     [, align <align-value];
```

Таблица 62

Параметр	Описание
<num of bytes>	Количество байт, выбранных из пакета для преобразования
<offset>	Количество байтов в полезной нагрузке
<var_name>	Имя переменной, в которой будет храниться значение
[relative]	Смещение относительно конца последнего найденного соответствия
[multiplier] <value>	Несколько извлеченных байт <multi-value> перед сохранением
[endian]	Тип считываемого числа: - big (старший байт по самому низкому адресу); - little (старший байт по самому высокому адресу).

Окончание таблицы 62

Параметр	Описание
string <num>	hex – преобразованная строка, представленная в шестнадцатеричном формате. dec – преобразованная строка, представленная в десятичной дроби. oct – преобразованная строка, представленная в восьмеричном формате.
[dce]	Разрешить модулю DCE определять порядок байтов
align <align-value>	Округлить извлеченное значение до следующей границы байта <align-value> после умножения (если есть); <align-value> может быть 2 или 4

Таблица 63

Ключевое слово	Модификатор
content	offset, depth, distance, within
byte_test	offset, value
byte_jump	offset
isdataat	offset

Пример:

```
alert tcp any any -> any any \
      (msg:"Byte_Extract Example Using distance"; \
      content:"Alice"; byte_extract:2,0,size; content:"Bob";  \
distance:size; within:3; sid:1;)

alert tcp any any -> any any \
      (msg:"Byte_Extract Example Using within"; \
      flow:established,to_server; content:"|00 FF|"; \
      byte_extract:1,0,len,relative; content:"|5c 00|";  \
distance:2; within:len; sid:2;)

alert tcp any any -> any any \
      (msg:"Byte_Extract Example Comparing Bytes"; \
      flow:established,to_server; content:"|00 FF|"; \
      byte_extract:2,0,cmp_ver,relative; content:"FooBar";  \
distance:0; byte_test:2,=,cmp_ver,0; sid:3;)
```

11.9.14.16. rpc

Ключевое слово rpc можно использовать для проверки приложений RPC, номеров версий и процедур в запросах SUNRPC CALL.

Для номера версии и процедуры допускается использование подстановочного знака *, которому соответствуют любые значения номеров/процедур.

RPC (удаленный вызов процедур) – это приложение, которое позволяет компьютерной программе выполнять процедуру на другом компьютере (или в адресном пространстве), используется для межпроцессного взаимодействия.

Синтаксис:

```
rpc:<номер приложения>, [<номер версии>|*], [<номер процедуры>|*]>;
```

Пример в правиле:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap request yppasswdd"; rpc:100009,*,*; reference:bugtraq,2763; classtype:rpc-portmap-decode; sid:1296; rev:4;)
```

11.9.14.17. replace

Модификатор `replace` – заменяет `content` – можно использовать только в `ips`, регулирует сетевой трафик. Он изменяет содержимое (`content`), за которым следует («`abc`»), на другое («`def`»), см. пример на рис. 207.

Рис. 207

Модификатор `replace` должен содержать столько же символов, сколько и заменяемое им содержимое. Его можно использовать только с отдельными пакетами, не будет работать для нормализованных буферов, таких как HTTP uri или проверки контента в повторно собранном потоке.

Контрольные суммы будут пересчитаны СОВ и изменены после использования ключевого слова `replace`.

11.9.14.18. pcre

Ключевое слово `pcre` (совместимы с `Perl Compatible Regular Expressions`) позволяет использовать регулярные выражения в правилах.

Сложность `pcre` негативно влияет на производительность. Поэтому, чтобы избавить СОВ от необходимости часто проверять на `pcre`, в `pcre` основном используется с `content`, в этом случае содержимое должно совпасть раньше, прежде чем будет проверен параметр `pcre`.

Синтаксис:

```
pcre:"/<regex>/opts";
```

Также может использоваться знак отрицания (!) при необходимости:

```
pcre:!"/<regex>/opts";
```

В следующем примере будет проверяется, содержит ли полезная нагрузка шесть последовательных чисел:

```
pcre:"/[0-9]{6}/";
```

Пример использования в правиле:

```
drop tcp $HOME_NET any -> $EXTERNAL_NET any \
(msg:”ET TROJAN Likely Bot Nick in IRC (USA +..)”; \
 flow:established,to_server; flowbits:isset,is_proto_irc; \
 content:”NICK “; pcre:”/NICK .*USA.*[0-9]{3,}/i”; \
reference:url,doc.emergingthreats.net/2008124; \
classtype:trojan-activity; sid:2008124; rev:2;)
```

Есть несколько качеств `pcre`, которые можно изменить:

- по умолчанию `pcre` чувствителен к регистру;
- . (точка) является частью регулярного выражения. Она соответствует каждому байту, кроме символов новой строки;
- по умолчанию полезная нагрузка будет проверяться как одна строка.

Эти качества можно изменить с помощью следующих символов – Perl-совместимыми модификаторов:

- `i` – игнорировать регистр символов;
- `s` – позволяет включать в проверку символы новой строки;
- `m` – по умолчанию строка трактуется как одна большая последовательность символов. С помощью специальных символов ^ и `$` можно задать проверку

для начала или конца строки. При наличии модификатора `m` символы ^ и `$` задают проверку в начале или в конце каждой новой строки (относительно символа перевода строки в буфере), а также в начале и в конце буфера.

Чтобы использовать эти модификаторы, необходимо добавить их в `pcre` после `regex`. Так:

```
pcre: "/<regex>/i";
```

Pcre-совместимые модификаторы

Есть несколько модификаторов, совместимых с `pcre`, которые также могут изменять свойства `pcre`. Это:

- `A` – выполнять проверку только в начале буфера (аналогично ^);
- `E` – осуществляет проверку только в самом конце строки (`$`), если отсутствует модификатор `E` символ `$` задает поиск до символа новой строки в конце буфера;
- `G` – инвертирует жадность квантификаторов (количество повторов).

Примечание. Следующие символы должны быть экранированы внутри содержимого `pcre`: `; \ "`

У СОВ есть свои специфические модификаторы `pcre`:

- `R` – проверяет относительно конца предыдущего найденного совпадения, аналог `distance:0;`;
- `U` – задает поиск в декодированном буфере URI, аналог `uricontent` и `content` в сочетании с `http_uri`. `U` можно комбинировать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-uri. Больше о нормализации HTTP URI приведено в п. 11.9.18.3.

Примеры приведены на рис. 208 – 211.

content:"/index."; http_uri; content:"htm"; http_uri; distance:0; ✓

content:"index."; http_uri; pcre:"/html?$/UR"; ✓

content:"index."; http_uri; pcre:"/^/index\.html?/$/U"; ✓

Рис. 208

content:"/index."; http_uri; content:"htm"; http_uri; distance:0; ✓

content:"index."; http_uri; pcre:"/html?$/UR"; ✓

content:"index."; http_uri; pcre:"/^/index\.html?/$/U"; ✓

Рис. 209

content:"/index."; http_uri; content:"htm"; http_uri; distance:0; ✓

content:"index."; http_uri; pcre:"/html?$/UR"; ✗

content:"index."; http_uri; pcre:"/^/index\.html?/$/U"; ✗

Рис. 210

Рис. 211

- `I` – осуществляет поиск в HTTP-raw-uri, в сочетании с `http_raw_uri`. `I` можно комбинировать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-raw-uri. Больше о нормализации HTTP URI см. в п. 11.9.18.3;
- `P` – осуществляет поиск в теле HTTP-запроса, в сочетании с `http_client_body`. `P` можно сочетать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны быть в теле HTTP-запроса;
- `Q` – осуществляет поиск в теле HTTP-ответа, в сочетании с `http_server_body`. `Q` можно сочетать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны быть в теле ответа HTTP;
- `H` – осуществляет поиск в HTTP-заголовке. `H` можно сочетать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны быть в теле заголовка HTTP;
- `D` – осуществляет поиск в ненормализованном заголовке, в сочетании с `http_raw_header`. `D` можно сочетать с `/R`. Обратите внимание, что `R` относится к предыдущему совпадению, поэтому оба совпадения должны быть в заголовке HTTP-raw;

- M – осуществляет поиск в методе запроса, в сочетани с http_method. M можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-метода;
- C – осуществляет поиск в HTTP-cookie, в сочетании с http_cookie. C можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-cookie;
- S – осуществляет поиск в коде статистики HTTP, в сочетании с http_stat_code. S можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере кода статистики HTTP;
- Y – осуществляет поиск в HTTP-stat-msg, в сочетании с http_stat_msg. Y можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-stat-msg;
- B – применяется в правилах, но только для совместимости, СОВ не использует B, но поддерживает его, поэтому ошибок не возникает;
- O – переопределяет установленный лимит совпадений pcre;
- V – осуществляет поиск в HTTP-User-Agent, в сочетании с http_user_agent. V можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-User-Agent;
- W – осуществляет поиск в HTTP-узле, в сочетании с http_host. W можно сочетать с /R. Обратите внимание, что R относится к предыдущему совпадению, поэтому оба совпадения должны находиться в буфере HTTP-Host.

11.9.15. Преобразования

Ключевые слова преобразования превращают данные в липком буфере во что-то другое. Ключевые слова преобразования превращают данные в липком буфере во что-то другое. Некоторые преобразования поддерживают параметры для большего контроля над процессом трансформации.

Пример:

```
alert http any any -> any any (file_data; strip_whitespace; \
    content:"window.navigate("; sid:1;)
```

Этот пример будет совпадать по трафику, даже если между ними есть один или несколько пробелов между `navigate` и (.

Преобразования могут быть объединены в цепочку. Они обрабатываются в том порядке, в котором они появляются в правиле. Выходные данные каждого преобразования выступают в качестве входных данных для следующего.

Пример:

```
alert http any any -> any any (http_request_line;
compress_whitespace; to_sha256; content:"|54A9 7A8A B09C 1B81 3725 2214
51D3 F997 F015 9DD7 049E E5AD CED3 945A FC79 7401|"; sid:1;)
```

П р и м е ч а н и е . Не все липкие буферы поддерживают преобразования.

11.9.15.1. dotprefix

Берет буфер и добавляет к нему «.» для облегчения точной проверки домена. Например, входная строка `hello.google.com` будет изменена и станет `.hello.google.com`. Кроме того, добавление точки позволяет `google.com` сопоставить с `content:".google.com"`.

Пример:

```
alert dns any any -> any any (dns.query; dotprefix; \
    content:".microsoft.com"; sid:1;)
```

Этот пример будет соответствовать `windows.update.microsoft.com` и `maps.microsoft.com.au`, но не с `windows.update.fakemicrosoft.com`.

Это правило можно использовать для точного поиска по домену, пример:

```
alert dns any any -> any any (dns.query; dotprefix; \
    content:".microsoft.com"; endswith; sid:1;)
```

Этот пример будет соответствовать `windows.update.microsoft.com`, но не `windows.update.microsoft.com.au`.

Наконец, это правило можно использовать для точного поиска по TLD; пример:

```
alert dns any any -> any any (dns.query; dotprefix; \
   content:".co.uk"; endswith; sid:1;)
```

Этот пример будет соответствовать `maps.google.co.uk`, но не `maps.google.co.nl`.

11.9.15.2. strip_whitespace

Удаляет все пробелы, рассматриваемые вызовом `isspace()` в языке C.

Пример:

```
alert http any any -> any any (file_data; strip_whitespace; \
   content:"window.navigate("; sid:1;)
```

11.9.15.3. compress_whitespace

Сжимает все последовательные пробелы в один пробел.

11.9.15.4. to_md5

Берет буфер, вычисляет хеш MD5 и передает необработанное хеш-значение.

Пример:

```
alert http any any -> any any (http_request_line; to_md5; \
   content:"|54 A9 7A 8A B0 9C 1B 81 37 25 22 14 51 D3 F9 97|";
sid:1;)
```

П р и м е ч а н и е . Зависит от компиляции `libnss`.

11.9.15.5. to_sha1

Берет буфер, вычисляет хеш SHA-1 и передает необработанное хеш-значение.

Пример:

```
  alert http any any -> any any (http_request_line; to_sha1; \
     content:"|54A9 7A8A B09C 1B81 3725 2214 51D3 F997 F015 9DD7|";
sid:1;)
```

П р и м е ч а н и е . Зависит от компиляции `libnss`.

11.9.15.6. to_sha256

Берет буфер, вычисляет хеш SHA-256 и передает необработанное значение хеш-функции.

Пример:

```
alert http any any -> any any (http_request_line; to_sha256;
content:"|54A9 7A8A B09C 1B81 3725 2214 51D3 F997 F015 9DD7 049E
E5AD CED3 945A FC79 7401|"; sid:1;)
```

Примечание. Зависит от компиляции libnss.

11.9.15.7. pcrexform

Берет буфер, применяет требуемое регулярное выражение и выводит первое зафиксированное выражение.

Примечание. Для этого преобразования требуется обязательная строка параметра, содержащая регулярное выражение.

Этот пример предупреждает, если http.request_line содержит /dropper.php. Пример:

```
alert http any any -> any any (msg:"HTTP with pcrexform"; \
http.request_line; pcrexform:"[a-zA-Z]+\s+(.*)\s+HTTP"; \
content:"/dropper.php"; sid:1;)
```

11.9.15.8. url_decode

Декодирует данные в кодировке URL, т.е. заменяет «+» пробелом и «%HH» его значением. Не декодирует кодировку Unicode «% uZZZZ».

11.9.16. Ключевые слова Flow (поток)

11.9.16.1. Flowbits

Flowbits состоит из двух частей. Первая часть описывает действие, которое он собирается выполнить, вторая часть – это имя бита потока (flowbit).

Множество пакетов принадлежат одному потоку. СОВ хранит эти потоки в памяти. Flowbits может гарантировать, что предупреждение будет сгенерировано, например, когда совпадают два разных пакета или если оба пакета совпадают. Когда второй пакет совпадет, СОВ должна знать, был ли совпал ли первый пакет. Flowbits помечает поток, если пакет соответствует, поэтому СОВ «знает», что он должен генерировать предупреждение, когда второй пакет также совпадет.

`Flowbits` может иметь разные действия:

`flowbits: set, name`

Установит условие/'имя', если оно присутствует, в потоке.

`flowbits: isset, name`

Может использоваться в правиле, чтобы убедиться, что оно создает оповещение, когда правило соответствует и условие задано в потоке.

`flowbits: toggle, name`

Отменяет текущую настройку. Так, например, если условие установлено, оно будет отменено, и наоборот.

`flowbits: unset, name`

Может использоваться для отмены условия в потоке.

`flowbits: isnotset, name`

Может использоваться в правиле, чтобы убедиться, что оно создает предупреждение, когда оно совпадает, а условие не задано в потоке.

`flowbits: noalert`

Это правило не создает никаких предупреждений.

Пример приведен на рис. 212.

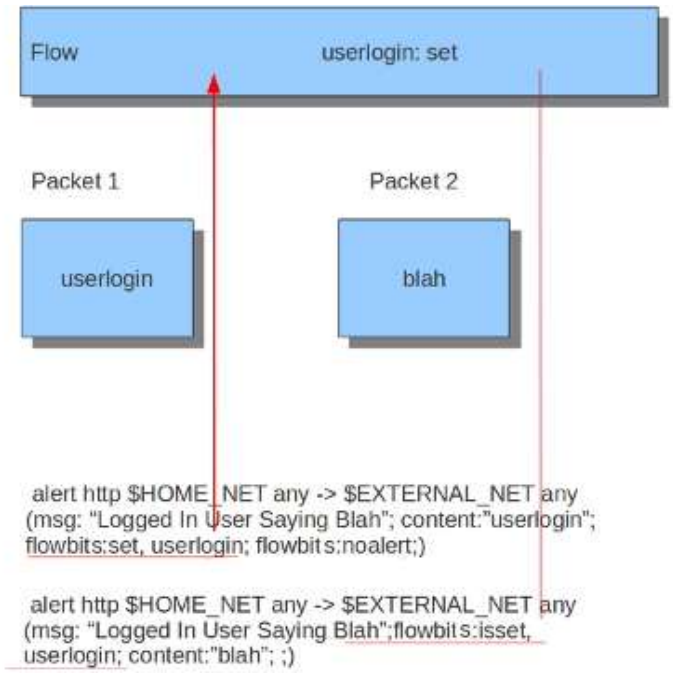


Рис. 212

Когда посмотрите на первое правило, то заметите, что оно сгенерировало бы оповещение, если бы не «Flowbits»: noalert» в конце этого правила. Целью этого правила является проверка совпадения с «userlogin» и отметка это в потоке, поэтому нет необходимости генерировать оповещение.

Второе правило не действует без первого правила. Если первое правило совпадает, flowbits задает следующее указанное условие, которое должно присутствовать в потоке. Теперь со вторым правилом можно проверить, соответствует ли предыдущий пакет первому условию. Если в этот момент второе правило совпадет, то предупреждение будет сгенерировано.

Можно использовать flowbits несколько раз в правиле и комбинировать различные функции.

11.9.16.2. Flow

Ключевое слово flow используется вместе со сборкой потоков TCP, можно использовать для применения правила к некоторым направлениям потока трафика, можно создать правила только для клиентов или только для серверов. Ключевое слово flow также можно использовать, чтобы указать, что правило должно проверять только пакеты перестроения потока (only_stream) или не переключается между пакетами (no_stream) (таблица 64).

Таблица 64

Параметр	Описание
to_client	Переключается на серверные отклики.
to_server	Переключается на клиентские запросы.
from_client	Переключается на клиентские запросы (то же, что и to_server).
from_server	Переключается на серверные отклики (то же, что и to_client).
established	Переключается только на организованные соединения TCP.
not_established	Переключается на пакеты, которые не являются частью установленного соединения.
stateless	Переключается независимо от состояния обработчика потока (stream processor) и может быть полезно для детектирования пакетов, направленных на аварийное завершение работы системы.
only_stream	Переключается на пакеты перестроения потока.
no_stream	Не переключается между пакетами перестроения потока.
only_frag	Переключается на собранные из фрагментов пакеты.
no_frag	Переключается на нефрагментированные пакеты.

Можно комбинировать несколько вариантов потока, например:

```
flow:to_client, established
flow:to_server, established, only_stream
flow:to_server, not_established, no_frag
```

Определение `established` зависит от протокола:

- TCP соединение будет установлено после трехстороннего рукопожатия (рис. 213);

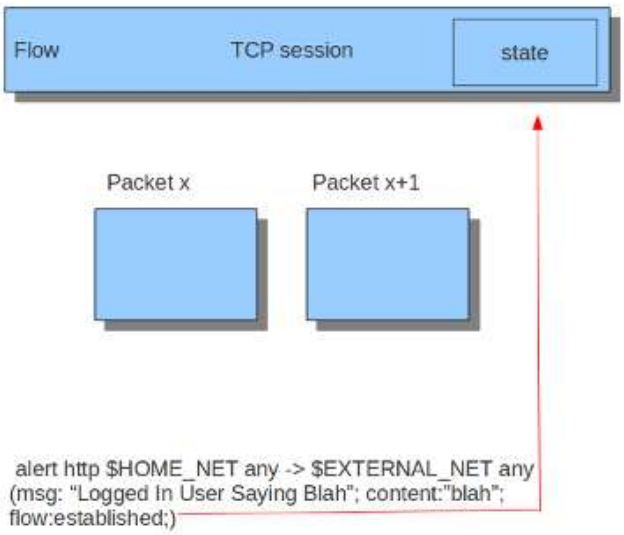


Рис. 213

- для других протоколов (например, UDP) соединение будет считаться установленным после просмотра трафика с обеих сторон соединения (рис. 214).

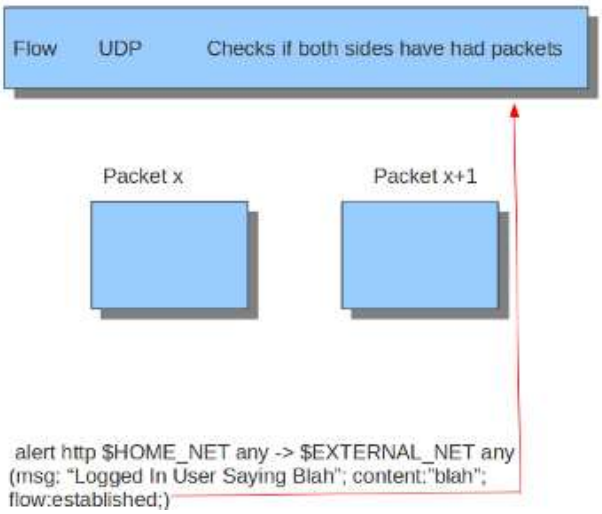


Рис. 214

11.9.16.3. Flowint

`Flowint` позволяет хранить математические операции с использованием переменных. Он работает так же, как и `flowbits`, но с добавлением математических возможностей и того факта, что целое число можно хранить и манипулировать им, а не просто установленным флагом. Можно использовать это для ряда очень полезных вещей, таких как подсчет вхождений, добавление или вычитание вхождений, выполнение пороговой обработки в потоке по отношению к нескольким факторам, таким образом, можно выполнять эти операции между потоками.

Синтаксис:

```
flowint: name, modifier[, value];
```

Определяет переменную (не обязательно) или проверяет, установлена она или нет.

```
flowint: name, < +,-,=,>,<,>=,<=,==, != >, value;
flowint: name, (isset|isnotset);
```

Сравните или измените переменную. Доступны: сложение, вычитание, сравнение больше или меньше, больше или равно и меньше или равно. Элемент для сравнения может быть целым числом или другой переменной.

Например, если необходимо подсчитать, сколько раз встречается имя пользователя в указанном потоке и не оповещать об этом:

```
alert tcp any any -> any any (msg:"Counting Usernames"; \
content:"jonkman"; lowint: usernamecount, +, 1; noalert;)
```

Правило будет подсчитывать каждое вхождение и увеличивать переменную `usernamecount`, а не генерировать предупреждение для каждого.

Теперь предположим, что необходимо создать оповещение, если в потоке есть более пяти совпадений:

```
alert tcp any any -> any any (msg:"More than Five Usernames!"; \
content:"jonkman"; flowint: usernamecount, +, 1; \
flowint:usernamecount, >, 5;)
```

Таким образом, предупреждение будет получено только в том случае, если количество имен пользователей (`usernamecount`) больше пяти.

Итак, теперь предположим, что хотим получить оповещение, как указано выше, и нет, если было больше случаев выхода из системы с этим именем пользователя. Предполагая, что выход из системы указывается с помощью «Jonkman Logout», например:

```
alert tcp any any -> any any (msg:"Username Logged out"; \
content:"logout jonkman"; flowint: usernamecount, -, 1; \
flowint:usernamecount, >, 5;)
```

Итак, теперь предупреждение будет получено только в том случае, если для этого указанного имени пользователя будет более пяти активных входов в систему.

Это довольно упрощенный пример, но показывает силу того, что такая простая функция может сделать в написании правил. Может применяться для отслеживания входа в систему, IRC конечных автоматов, отслеживания вредоносных программ и обнаружение входа в систему методом подбора пароля.

Допустим, отслеживается протокол, который обычно допускает пять неудачных попыток входа в систему за одно соединение, и у нас есть уязвимость, из-за которой злоумышленник может продолжить вход после этих пяти попыток, и нам нужно знать об этом:

```
alert tcp any any -> any any (msg:"Start a login count"; \
content:"login failed"; flowint:loginfail, notset; \
flowint:loginfail, =, 1; noalert;)
```

Таким образом, обнаруживаем первоначальный сбой и если переменная еще не установлена, то присваиваем ей значение 1:

```
alert tcp any any -> any any (msg:"Counting Logins"; \
content:"login failed"; flowint:loginfail, isset; \
flowint:loginfail, +, 1; noalert;)
```

Теперь увеличиваем счетчик, если он был установлен:

```
alert tcp any any -> any any (msg:"More than Five login fails"; \
content:"login failed"; flowint:loginfail, isset; \
flowint:loginfail, >, 5;)
```

Теперь сгенерируем предупреждение, если столкнемся с пятью неудачными попытками входа в систему в одном потоке. Но также предположим, что нам также

нужно оповещение, если есть два успешных входа в систему и один неудачный после этого:

```
alert tcp any any -> any any (msg:"Counting Good Logins";          \
      content:"login successful"; flowint:loginsuccess, +, 1; noalert;)
```

Здесь считаются успешные входы, так что теперь будем считать успешные входы и неудачные:

```
alert tcp any any -> any any (msg:"Login fail after two successes";   \
       content:"login failed"; flowint:loginsuccess, isset;          \
          flowint:loginsuccess, =, 2;)
```

Вот еще несколько общих примеров:

```
alert tcp any any -> any any (msg:"Setting a flowint counter"; \
content:"GET"; flowint:myvar, notset; \
     flowint:maxvar,notset;  flowint:myvar,=,1; flowint: maxvar,=,6;)
alert tcp any any -> any any (msg:"Adding to flowint counter";       \
    content:"Unauthorized"; flowint:myvar,isset; flowint: myvar,+,2;)
alert tcp any any -> any any \
msg:"when flowint counter is 3 create new counter"; \
    content:"Unauthorized"; flowint:myvar, isset; flowint:myvar,==,3; \
    flowint:cntpackets,notset; flowint:cntpackets, =, 0;)
alert tcp any any -> any any \
(msg:"count the rest without generating alerts"; \
    flowint:cntpackets,isset; flowint:cntpackets, +, 1; noalert;)
alert tcp any any -> any any (msg:"fire this when it reach 6";         \
    flowint: cntpackets, isset;                                  \
    flowint: maxvar,isset; flowint: cntpackets, ==, maxvar;)
```

11.9.16.4. Stream_size

Опция `stream_size` совпадает по трафику в соответствии с зарегистрированным количеством байт по порядковым номерам. Есть этого ключевого слова есть несколько модификаторову:

- > больше, чем;
- < меньше, чем;
- = равно;
- != не равно;

- >= больше или равно;

- <= меньше или равно.

Синтаксис:

```
stream_size:<server|client|both|either>, <modifier>, <number>;
```

Пример в правиле:

```
alert tcp any any -> any any (stream_size:both, >, 5000; sid:1;)
```

11.9.17. Ключевое слово bypass

Ключевое слово bypass можно использовать в сигнатурах, чтобы исключить трафик из дальнейшей оценки.

Ключевое слово bypass полезно в тех случаях, когда ожидается большой поток.

Обход потока при сопоставлении http-трафика.

Пример:

```
alert http any any -> any any (content:"ya.ru"; \
    http_host; bypass; sid:10001; rev:1;)
```

11.9.18. Ключевые слова HTTP

Существуют дополнительные модификаторы содержимого, которые могут предоставлять информацию для возможностей указанного протокола на прикладном уровне. Более подробную информацию можно найти в п. 11.9.14, эти ключевые слова гарантируют, что правило проверяет только отдельные части сетевого трафика. Например, чтобы проверить указанный URI запрос, файлы cookie или тело HTTP-запроса/ответа и т. д.

Все ключевые слова HTTP являются модификаторами. Обратите внимание на разницу между модификаторами содержимого и липкими буферами.

Модификатор «липкий буфер» размещается первым, а все ключевые слова, следующие за ними, применяются к этому буферу, например:

```
alert http any any -> any any (http.response_line; content:"403 Forbidden"; sid:1;)
```

«Модификаторы контента» оглядываются назад в правиле, например:

```
alert http any any -> any any (content:"index.php"; http_uri; sid:1;)
```

Доступны ключевые слова HTTP-запроса/ответа приведенные в таблице 65, все они имеют тип «липкий буфер».

Таблица 65

Ключевое слово	Направление
file_data	Ответ
http. request _line	Запрос
http.accept	Запрос
http.accept_enc	Запрос
http.accept_lang	Запрос
http.connection	Запрос
http.content_len	Оба
http.content_type	Оба
http.cookie	Оба
http.header.raw	Оба
http.header	Оба
http.header_names	Оба
http.host.raw	Запрос
http.host	Запрос
http.location	Ответ
http.method	Запрос
http.protocol	Оба
http.referer	Запрос
http.request_body	Запрос
http.response_body	Ответ
http.response_line	Ответ
http.server	Ответ
http.start	Оба
http.stat_code	Ответ
http.stat_msg	Ответ
http.uri.raw	Запрос
http.uri	Запрос
http.user_agent	Запрос

11.9.18.1. HTTP Primer

Важно понимать структуру HTTP-запросов и ответов. Ниже приведен простой пример HTTP-запроса и ответа.

HTTP-запрос:

```
GET /index.html HTTP/1.0\r\n
```

GET – это метод запроса. Примеры методов: GET, POST, PUT, HEAD и т. д. Путь URI – /index.html, а версия HTTP – HTTP/1.0. Сейчас используются 1.0 и 1.1 версии чаще всего.

HTTP-ответ:

```
HTTP/1.0 200 OK\r\n
<html>
<title> some page </title>
</HTML>
```

В этом примере HTTP/1.0 – это версия HTTP, 200 – код состояния ответа, а OK – сообщение о состоянии ответа.

Хотя файлы cookie отправляются в заголовке HTTP, их нельзя найти ключевым словом http.header. Для поиска файлов cookie используется ключевое слово http.cookie.

Аналогично метод HTTP принадлежит буферу методов, заголовки HTTP – буферу заголовков и т. д. Буфер – это определенная часть запроса или ответа, которую СОВ извлекает в памяти для проверки.

Все ранее описанные ключевые слова можно использовать в сочетании с буфером в правиле. Ключевые слова distance и within являются относительными модификаторами, поэтому их можно использовать только в пределах одного и того же буфера. Нельзя связать поиск content с разными буферами с относительными модификаторами.

11.9.18.2. http.method

С помощью модификатора содержимого http.method можно проверить поле метод в HTTP. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами содержимого, такими как: depth, distance, offset, nocase и within.

Примеры методов: GET, POST, PUT, HEAD, DELETE, TRACE, OPTIONS, CONNECT и PATCH.

Пример метода в HTTP-запросе приведен на рис. 215.

```
GET / HTTP/1.1
Host: www.google.com
Connection: keep-alive
Accept:
application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
```

Рис. 215

Пример записи и проверки методов приведен на рис. 216, рис. 217.

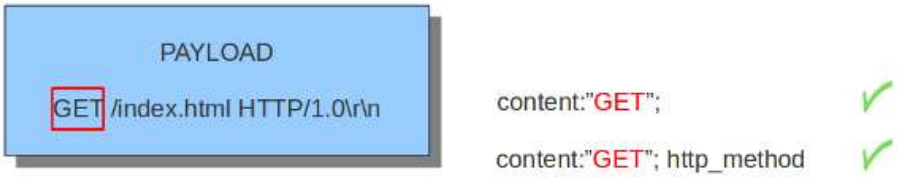


Рис. 216

Рис. 217

11.9.18.3. http.uri и http.uri.raw

С помощью модификаторов содержимого `http.uri` и `http.uri.raw` можно проверить буфер URI-запроса. Ключевые слова можно использовать в сочетании со всеми ранее упомянутыми модификаторами содержимого, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

В СОВ `uri` имеет два вида: необработанный `uri.raw` и нормализованный `uri`. Пробел, например, может быть обозначен шестнадцатеричной записью %20. Преобразовать пробел в обозначение и означает нормализацию. `uri.raw` и

нормализованный `uri` являются отдельными буферами. Таким образом, `uri.raw` проверяет необработанный буфер `uri.raw` и не может проверять нормализованный буфер `uri`.

Пример URI в HTTP-запросе:

```
GET /index.html HTTP/1.0\r\n
```

Примеры правил для `http.uri` приведены на рис. 218.

Рис. 218

11.9.18.4. urilen

Ключевое слово `urilen` используется для проверки длины URI-запроса. Можно использовать операторы < и >, которые обозначают соответственно меньше и больше.

Синтаксис:

```
urilen:3;
```

Примеры записи:

```
urilen:1;
urilen:>1;
urilen:<10;
urilen:10<>20;          (больше 10 и меньше 20)
```

Пример в правиле:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET TROJAN Possible Vundo Trojan Variant reporting to Controller"; flow:established,to_server; content:"POST "; depth:5; uricontent:"/frame.html?"; urilen: > 80; classtype:trojan-activity; reference:url,doc.emergingthreats.net/2009173; reference:url,www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/VIRUS/TROJAN_Vundo; sid:2009173; rev:2;)
```

Также можно добавить `normal` или `raw`, чтобы определить, какой тип буфера использовать (нормализованный или необработанный буфер).

11.9.18.5. http.protocol

`http.protocol` проверяет поле протокола HTTP-запроса или ответа. Если строка запроса «`GET / HTTP/1.0rn`», то этот буфер будет содержать «`HTTP/1.0`».

Пример:

```
alert http any any -> any any (flow:to_server; http.protocol; content:"HTTP/1.0"; sid:1;)
```

11.9.18.6. http.request_line

`http.request_line` используется для проверки всей строки HTTP-запроса.

Пример:

```
alert http any any -> any any (http.request_line; content:"GET / HTTP/1.0"; sid:1;)
```

11.9.18.7. http.header и http.header.raw

С помощью модификатора содержимого `http.header` можно проверить буфер HTTP-заголовка. Он проверяет все извлеченные заголовки в одном буфере, за исключением указанных ранее в документации, которые не могут соответствовать этому буферу и имеют собственный модификатор (например, `http.cookie`). Модификатор можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

Примечание. Буфер `http.header` нормализован. Любые конечные пробелы и символы табуляции удаляются, чтобы избежать этого, используйте ключевое слово `http.header.raw`.

Пример заголовка в HTTP-запросе:

```
GET / HTTP/1.1
Host: www.google.com
Connection: keep-alive
```

Примеры использования приведены на рис. 219.

Рис. 219

11.9.18.8. http.cookie

С помощью модификатора содержимого `http.cookie` можно проверить буфер `cookie` в HTTP-запросе. Модификатор можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

Обратите внимание, что файлы `cookie` передаются в заголовках HTTP, но извлекаются в выделенный буфер и проверяются с использованием их собственного указанного модификатора `http.cookie`.

Пример `cookie` HTTP-запроса приведен на рис. 220.

```
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US)
AppleWebKit/534.16
(KHTML, like Gecko) Ubuntu/10.10 Chromium/10.0.618.0
Chrome/10.0.618.0
Safari/534.16
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
Cookie:
PREF=ID=efe36c63a3bfa6a4:U=aa0cf39996084d7e:TM
=1252314621:LM=1292956821:GM=1:S=dYtecyNBioer
A47b
```

Рис. 220

Пример использования `http.cookie` приведен на рис. 221.

Рис. 221

11.9.18.9. http.user_agent

Модификатор содержимого `http.user_agent` проверяет значение User-Agent в заголовке HTTP-запроса. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

Обратите внимание, что ключевое слово `pcre` также может проверять этот буфер при использовании модификатора `/V`.

Нормализация: начальные пробелы не являются частью этого буфера. Таким образом, запись вида, «`User-Agent: rn`» приведет к пустому буферу `http.user_agent`.

Пример User-Agent в заголовке HTTP-запросе (рис. 222).

Рис. 222

Пример `http.user_agent` (рис. 223).

Рис. 223

Примечания:

1. Буфер `http.user_agent` нормализован и НЕ должен включать имя заголовка, двоеточие или начальные пробелы, то есть он не будет включать «`User-Agent:`».

2. Буфер `http.user_agent` не должен содержать CRLF (0x0D 0x0A) в конце. Если необходима проверять конец буфера, используйте ключевое слово `isdataat` или `pcre` (хотя производительность `pcre` будет хуже).

3. Если запрос содержит несколько заголовков «`User-Agent`», значения будут объединены в буфере `http.user_agent` в порядке сверху вниз с запятой и пробелом («,») между каждым из них.

Пример запроса:

```
GET /test.html HTTP/1.1
User-Agent: SuriTester/0.8
User-Agent: GGGG
```

Содержание `http.user_agent` буфера будет:

```
SuriTester/0.8, GGGG
```

11.9.18.10. http.accept

Липкий буфер для проверки значения Accept в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (http.accept; content:"image/gif"; sid:1;)
```

11.9.18.11. http.accept_enc

Липкий буфер для проверки значения Accept-Encoding в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (http.accept_enc; content:"gzip"; sid:1;)
```

11.9.18.12. http.accept_lang

Липкий буфер для проверки значения Accept-Language в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (http.accept_lang; content:"en-us"; sid:1;)
```

11.9.18.13. http.connection

Липкий буфер для проверки значения Connection в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (http.connection; content:"keep-alive"; sid:1;)
```

11.9.18.14. http.content_type

Липкий буфер для проверки значения Content-Type в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Используйте `flow:to_server` или `flow:to_client` чтобы принудительно проверить запрос или ответ.

Примеры:

```
alert http any any -> any any (flow:to_server; \
      http.content_type; content:"x-www-form-urlencoded"; sid:1;)

alert http any any -> any any (flow:to_client; \
      http.content_type; content:"text/javascript"; sid:2;)
```

11.9.18.15. http.content_len

Липкий буфер для проверки значения Content-Length в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Используйте `flow:to_server` или `flow:to_client` чтобы принудительно проверить запрос или ответ.

Примеры:

```
alert http any any -> any any (flow:to_server; \
      http.content_len; content:"666"; sid:1;)

alert http any any -> any any (flow:to_client; \
      http.content_len; content:"555"; sid:2;)
```

Чтобы выполнить числовую проверку Content-Length, можно использовать `byte_test`.

Пример, проверить и оповестить, если Content-Length больше или равно 8079:

```
alert http any any -> any any (flow:to_client; \
      http.content_len; byte_test:0,>=,8079,0,string,dec; sid:3;)
```

11.9.18.16. http.referer

Липкий буфер для проверки значения Referer в заголовке HTTP-запроса. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (http.referer; content:".php"; sid:1;)
```

11.9.18.17. http.start

Проверяет начало HTTP-запроса или ответа, содержит строку запроса/ответа, а также заголовки запроса/ответа. Используйте `flow:to_server` или `flow:to_client`, чтобы принудительно проверить запрос или ответ.

Используйте `flow:to_server` или `flow:to_client` чтобы принудительно проверить запрос или ответ.

Пример:

```
alert http any any -> any any (http.start; \
content:"HTTP/1.1|0d 0a|User-Agent"; sid:1;)
```

Буфер содержит нормализованные заголовки и завершается дополнительными символами `\r\n`, указывающими на конец заголовков.

11.9.18.18. http.header_names

Проверяет буфер, содержащий только имена HTTP-заголовков. Полезно для проверки отсутствия заголовка или проверки определенного порядка заголовков.

Буфер начинается с `\r\n` и заканчивается дополнительным `\r\n`.

Пример:

```
\\r\\nHost\\r\\n\\r\\n
```

Пример в правиле:

```
alert http any any -> any any (http.header_names; \
content:"|0d 0a|Host|0d 0a|"; sid:1;)
```

Пример, проверки, что присутствует только Host:

```
alert http any any -> any any (http.header_names; \
       content:"|0d 0a|Host|0d 0a 0d 0a|"; sid:1;)
```

Пример, проверки, что User-Agent находится непосредственно после Host:

```
alert http any any -> any any (http.header_names; \
       content:"|0d 0a|Host|0d 0a|User-Agent|0d 0a|"; sid:1;)
```

Пример, проверки, что User-Agent находится после Host, но не обязательно сразу после:

```
alert http any any -> any any (http.header_names; \
content:"|0d 0a|Host|0d 0a|"; content:"|0a 0d|User-Agent|0d 0a|"; \
   distance:-2; sid:1;)
```

11.9.18.19. http.request_body

С помощью модификатора содержимого `http.request_body` можно выполнять проверку тела HTTP-запроса. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within` и т. д.

Пример `http.request_body` в HTTP-запросе приведен на рис. 224.

```
Host: nowhereasdfasdf.com
Connection: Keep-Alive
Cache-Control: no-cache

type=playerStart&position=tidal
```

Рис. 224

Пример правила с `http.request_body` приведен на рис. 225.

Рис. 225

Примечание. Объем проверки тела запроса/клиента контролируется в `libhtp` с помощью параметра `request-body-limit`.

11.9.18.20. http.stat_code

С помощью модификатора содержимого `http.stat_code` можно выполнять проверку кода состояния HTTP. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within` и т. д.

Пример `http.stat_code` в HTTP-ответе:

```
HTTP/1.1 302 Found
```

Пример использования в правилах приведен на рис. 226.

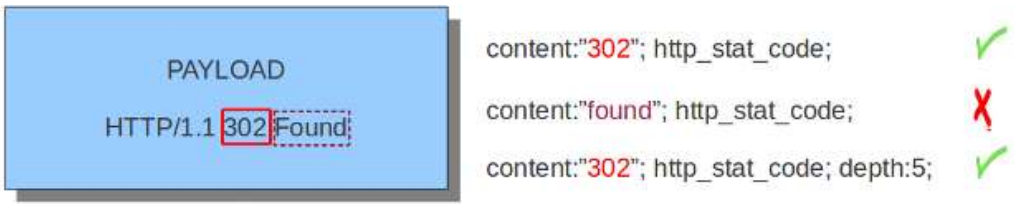


Рис. 226

11.9.18.21. http.stat_msg

С помощью модификатора содержимого `http.stat_msg` можно выполнять проверку пояснения к коду состояния HTTP. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

Пример `http.stat_code` в HTTP-ответе:

```
HTTP/1.1 302 Found
```

Пример использования в правилах приведен на рис. 227.

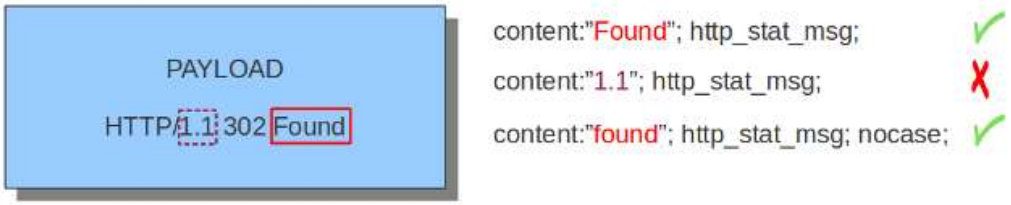


Рис. 227

11.9.18.22. http.response_line

`http.response_line` заставляет проверять всю строку HTTP-ответа.

Пример:

```
alert http any any -> any any (http.response_line;  \
content:"HTTP/1.0 200 OK"; sid:1;)
```

11.9.18.23. http.response_body

С помощью модификатора содержимого `http.response_body` можно проверить тело HTTP-ответа. Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset`, `nocase` и `within`.

Примечание. Объем проверки тела ответа/сервера контролируется в `libhtp` с помощью параметра `response-body-limit`.

Примечания:

1. Использование `http.response_body` похоже на наличие совпадений содержимого, которые идут после `file_data`, за исключением того, что оно не устанавливает постоянный (если не сбросить) указатель обнаружения на начало тела ответа сервера, то есть это не липкий буфер.

2. `http.response_body` будет соответствовать данным, декодированным `gzip`, точно так же, как и `file_data`.

3. Поскольку `http.response_body` совпадает с ответом сервера, его нельзя использовать с директивами потока `to_server` или `from_client`.

4. Соответствует модификатору `pcre`: `Q`.

11.9.18.24. http.server

Липкий буфер для проверки значения заголовка S`erver` в HTTP. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (flow:to_client; \
      http.server; content:"Microsoft-IIS/6.0"; sid:1;)
```

11.9.18.25. http.location

Липкий буфер для проверки значения заголовка Location в HTTP. Содержит только значение указанного поля заголовка. `\r\n` после заголовка не являются частью буфера.

Пример:

```
alert http any any -> any any (flow:to_client; \
      http.location; content:"http://www.google.com"; sid:1;)
```

11.9.18.26. http.host и http.host.raw

С модификатором содержимого `http.host` можно проверить только нормализованное имя хоста. `http.host.raw` проверяет необработанное имя хоста.

Ключевое слово можно использовать в сочетании со всеми ранее упомянутыми модификаторами, такими как `depth`, `distance`, `offset` и `within`.

Ключевое слово `nocase` не разрешено, имейте это в виду, и указывайте значение ключевого слова в нижнем регистре при необходимости.

Примечания:

1. `http.host` не содержит порт, связанный с хостом (например, `abc.com:1234`). Чтобы сопоставить хост и порт или применить операцию отрицания к хосту и порту, используйте `http.host.raw`.

2. Буферы `http.host` и `http.host.raw` заполняются либо из URI (если в запросе присутствует полный URI, как в запросе прокси), либо из заголовка HTTP Host. Если присутствуют оба, используется URI.

3. Буферы `http.host` и `http.host.raw` не будут включать имя заголовка, двоеточие или начальный «Host:».

4. Буферы `http.host` и `http.host.raw` не включает в проверку CRLF (0x0D 0x0A) в конце. Если необходимо проверять до конца буфера, используйте `isdataat` или `pcre` (хотя производительность `pcre` будет хуже).

5. Буфер `http.host` нормализован и требует, чтобы все буквы были строчными.

6. Содержимое, к которому применяется `http.host`, должно быть написано строчными буквами или иметь установленный флаг `nocase`.

7. `http.host.raw` соответствует ненормализованному буферу, поэтому при сопоставлении учитывается регистр (если не задано `nocase`).

8. Если запрос содержит несколько заголовков «Host», значения будут объединены в буферах `http.host` и `http.host.raw` в порядке их просмотра сверху вниз с запятой и пробелом («,») между каждым из их.

9. Соответствующий модификатор `pcre` (`http_host`): `W`.

10. Соответствующий модификатор `pcre` (`http_raw_host`): `Z`.

Пример запроса:

```
GET /test.html HTTP/1.1
Host: ABC.com
Accept: */*
Host: efg.net
```

`http.host` буфер:

```
abc.com, efg.net
```

`http.host.raw` буфер:

```
ABC.com, efg.net
```

11.9.18.27. file_data

С помощью `file_data` тело HTTP-ответа проверяется так же, как и с `http.response_body`. Ключевое слово `file_data` – это липкий буфер.

Пример:

```
alert http any any -> any any (file_data; content:"abc"; content:"xyz";)
```

Ключевое слово `file_data` влияет на всю проверку содержимого, пока не встретится ключевое слово `pkt_data` или не будет достигнут конец правила. Это делает его полезным для множественной проверки содержимого тела HTTP-ответа, избавляя от необходимости изменять каждую проверку по отдельности.

Поскольку тело HTTP-ответа может быть очень большим, оно проверяется небольшими порциями.

То, какая часть тела ответа/сервера проверяется, контролируется в `libhtp` с помощью параметра `response-body-limit`.

Если тело HTTP представляет собой flash-файл, сжатый с помощью «`deflate`» или «`lzma`», его можно распаковать, а `file_data` может совпадать с данными распаковки. Распаковка `flash` должна быть включена в конфигурацию `libhtp`:

```
# Decompress SWF files.
# 2 types: 'deflate', 'lzma', 'both' will decompress deflate and lzma
# compress-depth:
# Specifies the maximum amount of data to decompress,
# set 0 for unlimited.
# decompress-depth:
# Specifies the maximum amount of decompressed data to obtain,
# set 0 for unlimited.
swf-decompression:
  enabled: yes
  type: both
  compress-depth: 0
  decompress-depth: 0
```

Примечания:

1. Если тело HTTP использует `gzip` или `deflate`, `file_data` будет соответствовать распакованным данным.

2. На отрицательное соответствие влияет проверка по частям. Например. `content:!"<html";` не может совпасть в первом фрагменте, но, возможно, совпадет во втором. Чтобы избежать этого, используйте настройку `depth`. При настройке `depth` учитывается размер тела. Предполагая, что минимальный размер проверки тела ответа больше 1 Кбайт, `content:!"<html"; depth:1024;` может совпасть только в том случае, если шаблон `<html` отсутствует в первом проверяемом фрагменте.

3. `file_data` также может использоваться с SMTP.

11.9.19. Ключевые слова `File`

COB поставляется с несколькими ключевыми словами правил для проверки различных свойств файла. Они зависят от правильно настроенного извлечения файлов.

11.9.19.1. filename

Сравнение с указанным именем файла.

Синтаксис:

```
filename:<string>;
```

Пример:

```
filename:"secret";
```

11.9.19.2. fileext

Проверяет расширение имени файла.

Синтаксис:

```
fileext:<string>;
```

Пример:

```
fileext:"jpg";
```

11.9.19.3. filemagic

Проверяет информацию, которую `libmagic` возвращает о файле.

Синтаксис:

```
filemagic:<string>;
```

Пример:

```
filemagic:"executable for MS Windows";
```

11.9.19.4. filestore

Сохраняет файлы на диск, если сигнатура сработала.

Синтаксис:

```
filestore:<direction>,<scope>;
```

`direction` может быть:

- `request/to_server` – сохранить файл в направлении запрос к серверу;
- `response/to_client` – сохранить файл в направлении ответ клиенту;
- `both`: сохранить оба направления.

`scope` может быть:

- `file` – хранить только соответствующий файл (для `filename`, `fileext`, `filemagic`);
- `tx` – сохранить все файлы из соответствующей HTTP-транзакции
- `ssn/flow` – хранить все файлы из сеанса/потока TCP.

Если направление (`direction`) и область действия (`scope`) не указаны, направление будет таким же, как правило, и область действия будет для каждого файла.

11.9.19.5. filemd5

Сопоставляет md5 файла со списком контрольных сумм md5.

Синтаксис:

```
filemd5:[!]filename;
```

`filename` расширяется и включает правило `dir`. По умолчанию должен хранится в `/etc/suricata/rules/filename`. Используйте восклицательный знак, для применения отрицания, это позволяет создать белый список.

Примеры:

```
filemd5:md5-blacklist;
filemd5:!md5-whitelist;
```

Формат файла простой. Это текстовый файл с одним md5 на строку в начале строки в шестнадцатеричной записи. Если в строке есть дополнительная информация, она игнорируется.

Вывод из сумм md5 в следующем порядке:

```
2f8d0355f0032c3e6311c6408d7c2dc2  util-path.c
b9cf5cf347a70e02fde975fc4e117760  util-pidfile.c
02aaa6c3f4dbae65f5889eeb8f2bbb8d  util-pool.c
dd5fc1ee7f2f96b5f12d1a854007a818  util-print.c
```

Может быть запись просто md5:

```
2f8d0355f0032c3e6311c6408d7c2dc2
b9cf5cf347a70e02fde975fc4e117760
02aaa6c3f4dbae65f5889eeb8f2bbb8d
dd5fc1ee7f2f96b5f12d1a854007a818
```

Каждая запись md5 использует 16 байт памяти.

11.9.19.6. filesha1

Сопоставляет SHA1 файла со списком контрольных сумм SHA1.

Синтаксис:

```
filesha1:[!]filename;
```

`filename` расширяется и включает правило `dir`. По умолчанию должен хранится в `/etc/suricata/rules/filename`. Используйте восклицательный знак, для применения отрицания, это позволяет создать белый список.

Примеры:

```
filesha1:sha1-blacklist;
filesha1:!sha1-whitelist;
```

Формат файла аналогичен записи файла с суммами MD5 (см. п. 11.9.19.5).

11.9.19.7. filesha256

Сопоставляет SHA256 файла со списком контрольных сумм SHA256.

Синтаксис:

```
filesha256:[!]filename;
```

filename расширяется и включает правило dir. По умолчанию должен хранится в /etc/suricata/rules/filename. Используйте восклицательный знак, для применения отрицания, это позволяет создать белый список.

Примеры:

```
filesha256:sha256-blacklist;
filesha256:!sha256-whitelist;
```

Формат файла аналогичен записи файла с суммами MD5 (см. п. 11.9.19.5).

11.9.19.8. filesize

Проверяет размер файла при его передаче.

Синтаксис:

```
filesize:<value>;
```

Возможные единицы измерения: KB, MB, GB, без каких-либо единиц измерения по умолчанию используются байты.

Примеры:

```
filesize:100; # ровно 100 байт
filesize:100<>200; # больше 100 и меньше 200 байт
filesize:>100MB; # больше 100 Мбайт
filesize:<100MB; # меньше 100 Мбайт
```

Примечание. Используется для файлов, которые не полностью отслеживаются из-за потери пакетов или stream.reassembly.depth проверяются, достигнуто ли значение «больше, чем». Это связано с тем, что СОВ может знать, что файл больше заданного значения (поскольку уже его видел), но не может знать, находился ли окончательный размер в пределах диапазона, точного соответствовал или меньшего значения.

11.9.20. Ключевые слова DNS

Рассмотрим еще несколько модификаторов содержимого (см. также п. 11.9.14). Они обеспечивают проверку правилом определенной части сетевого трафика.

11.9.20.1. dns.opcode

Это ключевое слово проверяет код операции флагах заголовка DNS.

Синтаксис:

```
dns.opcode:[!]<number>
```

Проверяет DNS-запросы и ответы с кодом операции 4:

```
dns.opcode:4;
```

Проверяет DNS-запросы, где код операции НЕ равен 0:

```
dns.opcode:!0;
```

11.9.20.2. dns.query

С помощью `dns.query` проверяются запросы DNS-запросов. Ключевое слово `dns.query` работает немного иначе, чем обычные модификаторы содержимого. При использовании в правиле оно влияет на все содержимое, следующее за ним.

Пример:

```
alert dns any any -> any any (msg:”Test dns.query option”; dns.query; content:”google”; nocase; sid:1;)
```

Ключевое слово `dns.query` влияет на все последующее содержимое, пока не будет использовано `pkt_data` или не будет достигнут конец правила.

11.9.20.2.1. Нормализованный буфер

Буфер содержит буквенное доменное имя:

- значение `<length>` (как показано в необработанном DNS-запросе) представляют собой буквенные символы «.»;
- нет начального значения `<length>`;
- нет завершающего байта NULL (0x00) (используйте для отрицания `isdataat` для проверки конца).

Пример DNS-запроса для «`mail.google.com`» (для удобства чтения шестнадцатеричные значения закодированы между каналами):

- фрагмент DNS-запроса:

```
|04|mail|06|google|03|com|00|
```

- буфер `dns.query`:

```
mail.google.com
```

11.9.21. Ключевые слова SSL/TLS

В СОВ предусмотрено несколько ключевых слов правил для проверки различных свойств рукопожатия TLS/SSL. Проверки предусматривают включение строк.

11.9.21.1. tls.cert_subject

Проверка поля Subject сертификата TLS/SSL.

Примеры:

```
tls.cert_subject; content:"CN=*.googleusercontent.com";
isdataat:!1,relative;
tls.cert_subject; content:"google.com"; nocase; pcre:"/google.com$/";
```

tls.cert_subject – это липкий буфер.

tls.cert_subject может быть использовано как быстрый шаблон.

11.9.21.2. tls.cert_issuer

Проверка поля Issuer сертификата TLS/SSL.

Примеры:

```
tls.cert_issuer; content:"WoSign"; nocase; isdataat:!1,relative;
tls.cert_issuer; content:"StartCom"; nocase; pcre:"/StartCom$/";
```

tls.cert_issuer – это липкий буфер.

tls.cert_issuer может быть использовано как быстрый шаблон.

11.9.21.3. tls.cert_serial

Проверяет серийный номер сертификата.

Пример:

```
alert tls any any -> any any (msg:"match cert serial"; \
  tls.cert_serial; content:"5C:19:B7:B1:32:3B:1C:A1"; sid:200012;)
```

tls.cert_serial – это липкий буфер.

tls.cert_serial может быть использовано как быстрый шаблон.

11.9.21.4. tls.cert_fingerprint

Проверяет SHA-1 fingerprint сертификата.

Пример:

```
alert tls any any -> any any (msg:"match cert fingerprint"; \
  tls.cert_fingerprint; \
content:"4a:a3:66:76:82:cb:6b:23:bb:c3:58:47:23:a4:63:a7:78:a4:a1:18"; \
  sid:200023;)
```

tls.cert_fingerprint – это липкий буфер.

tls.cert_fingerprint может быть использовано как быстрый шаблон.

11.9.21.5. tls.sni

Проверяет поле Server Name Indication TLS/SSL.

Примеры:

```
tls.sni; content:"oisf.net"; nocase; isdataat:!1,relative;
tls.sni; content:"oisf.net"; nocase; pcre:"/oisf.net$/";
```

`tls.sni` – это липкий буфер.

`tls.sni` может быть использовано как быстрый шаблон.

11.9.21.6. tls_cert_notbefore

Проверяет поле NotBefore в сертификате.

Пример:

```
alert tls any any -> any any (msg:"match cert NotBefore"; \
  tls_cert_notbefore:1998-05-01<>2008-05-01; sid:200005;)
```

11.9.21.7. tls_cert_notafter

Проверяет поле NotAfter в сертификате.

Пример:

```
alert tls any any -> any any (msg:"match cert NotAfter"; \
  tls_cert_notafter:>2015; sid:200006;)
```

11.9.21.8. tls_cert_expired

Оценивается дата действия сертификата, если срок действия истек – возвращает `true`.

Синтаксис:

```
tls_cert_expired;
```

11.9.21.9. tls_cert_valid

Оценивает срок действия сертификата и, если он не истек, возвращает `true`. При этом не выполняется проверка цепочки сертификатов, что является противоположностью `tls_cert_expired`.

Синтаксис:

```
tls_cert_valid;
```

11.9.21.10. tls.certs

Выполняет «необработанную» проверку для каждого сертификата в цепочке сертификатов TLS.

Пример:

```
  alert tls any any -> any any (msg:"match bytes in TLS cert"; tls.certs; \
    content:"|06 09 2a 86|"; sid:200070;)
```

`tls.certs` – это липкий буфер

`tls.certs` может быть использовано как быстрый шаблон.

11.9.21.11. tls.version

Проверяет версию TLS/SSL.

Поддерживаемые значения: “1.0”, “1.1”, “1.2”, “1.3”.

Также возможно проверить версии с использованием шестнадцатеричной записи.

Примеры:

```
tls.version:1.2;
tls.version:0x7f12;
```

Первый пример проверяет TLSv1.2, а последний пример проверяет TLSv1.3 в шестнадцатеричной записи.

11.9.21.12. ssl_version

Проверяет версию протокола SSL/TLS.

Поддерживаемые значения: sslv2, sslv3, tls1.0, tls1.1, tls1.2, tls1.3.

Пример:

```
alert tls any any -> any any (msg:"match TLSv1.2"; \
  ssl_version:tls1.2; sid:200030;)
```

Также возможна проверка по нескольким версиям одновременно.

Пример:

```
alert tls any any -> any any (msg:"match SSLv2 and SSLv3"; \
  ssl_version:sslv2,sslv3; sid:200031;)
```

11.9.21.13. tls.store

Записывает TLS/SSL сертификат на диск.

11.9.21.14. ssl_state

Ключевое слово `ssl_state` проверяет состояние соединения SSL.

Возможные состояния: `client_hello`, `server_hello`, `client_keyx`, `server_keyx` и `unknown`.

Можно указать несколько состояний с | (OR) для проверки любого из указанных.

11.9.22. Ключевые слова SSH

В СОВ предусмотрено несколько ключевых слов правил для проверки различных элементов SSH соединений.

11.9.22.1. ssh.proto

Проверка версии протокола SSH. `ssh.proto` – это липкий буфер, который можно использовать в качестве быстрого шаблона.

Синтаксис:

```
ssh.proto;
```

Пример, проверка SSH версии 2.0:

```
alert ssh any any -> any any (msg:”match SSH protocol version”; ssh.proto; content:”2.0”; sid:1000010;)
```

11.9.22.2. ssh.software

Проверка строки `software` SSH баннепа. `ssh.software` – это липкий буфер, который можно использовать в качестве быстрого шаблона.

Синтаксис:

```
ssh.software;
```

Пример, проверяет, что строка `software` содержит `openssh`:

```
alert ssh any any -> any any (msg:”match SSH software string”; ssh.software; content:”openssh”; nocase; sid:1000020;)
```

11.9.22.3. ssh.protoversion

Проверяет версию используемого протокола SSH. Значение `2_compat` включает SSH версию 1.99.

Синтаксис:

```
ssh.protoversion:[0-9](\.[0-9])?|2_compat;
```

Пример, проверки SSH соединений с SSH версиями 2 или 1.99:

```
alert ssh any any -> any any (msg:”SSH v2 compatible”; ssh.protoversion:2_compat; sid:1;)
```

Пример, проверяет только SSH версию 1.10:

```
alert ssh any any -> any any (msg:”SSH v1.10”; ssh.protoversion:1.10; sid:1;)
```

11.9.22.4. ssh.hassh

Проверяет хеш (`md`5 хеш-алгоритм клиента).

Пример:

```
alert ssh any any -> any any (msg:"match hassh"; \
    ssh.hassh; content:"ec7378c1a92f5a8dde7e8b7a1ddf33d1";\
    sid:1000010;)
```

`ssh.hassh` – это липкий буфер.

`ssh.hassh` может быть использовано как быстрый шаблон.

11.9.22.5. ssh.hassh.string

Проверяет хеш строку (хеш-алгоритм клиента).

Пример:

```
alert ssh any any -> any any (msg:"match hassh-string"; \
    ssh.hassh.string; content:"none,zlib@openssh.com,zlib"; \
    sid:1000030;
```

`ssh.hassh.string` – это липкий буфер.

`ssh.hassh.string` может быть использовано как быстрый шаблон.

11.9.22.6. ssh.hassh.server

Проверяет хеш (`md`5 хеш-алгоритм сервера).

Пример:

```
alert ssh any any -> any any (msg:"match SSH hash-server"; \
   ssh.hassh.server; content:"b12d2871a1189eff20364cf5333619ee"; \
   sid:1000020;)
```

`ssh.hassh.server` – это липкий буфер.

`ssh.hassh.server` может быть использовано как быстрый шаблон.

11.9.22.7. ssh.hassh.server.string

Проверяет хеш строку (хеш-алгоритм сервера).

Пример:

```
alert ssh any any -> any any (msg:”match SSH hash-server-string”;
ssh.hassh.server.string;    content:”umac-64-etm@openssh.com,umac-
128-etm@openssh.com”; sid:1000040;)
```

`ssh.hassh.server.string` – это липкий буфер.

`ssh.hassh.server.string` может быть использовано как быстрый шаблон.

11.9.23. Ключевое слово `modbus`

Ключевое слово `modbus` можно использовать для проверки различных свойств запросов Modbus.

Существует три способа использования этого ключевого слова:

- проверка свойств функций с опцией `function`;
- проверка прямого доступа к данным с опцией `access`;
- проверка по идентификатору устройства с опцией `unit` или с предыдущей опцией `function` или `access`.

С помощью опции `function` можно проверить:

- действие, основанное на поле кода функции и коде подфункции, когда это применимо;
- одну из трех категорий функций Modbus;
- общедоступные функции, которые определены публично (настройка «`public`»);
- пользовательские функции (настройка «`user`»);

- зарезервированные функции, предназначенные для проприетарных расширений Modbus (ключевое слово «reserved »);
- одна из двух подгрупп общественных функций:
    а) назначенные функции, определение которых уже дано в спецификации Modbus (ключевое слово «assigned»);
    б) не назначенные функции, которые зарезервированы для использования в будущем (ключевое слово «unassigned»).

Синтаксис:

```
modbus: function <value>
modbus: function <value>, subfunction <value>
modbus: function [!] <assigned | unassigned | public | user | reserved | all>
```

Где ! – отрицание.

Примеры:

```
modbus: function 21             # Функция записи файла
modbus: function 4, subfunction 4  # Режим только прослушивания (Диагностика) modbus: function assigned        # определяется спецификацией прикладного протокола Modbus V1.1b3
modbus: function public          # подтвержденная сообществом Modbus.org
modbus: function user           # для внутреннего использования и не поддерживается спецификацией
modbus: function reserved        # используется некоторыми компаниями для устаревших продуктов и недоступна для общего пользования
modbus: function !reserved       # все функции, кроме зарезервированных
```

С опцией access можно проверить:

- тип доступа к данным (чтение или запись);
- доступ к одной из первичных таблиц (Discretes Input, Coils, Input Registers и Holding Registers);
- диапазон адресов доступа;
- письменное значение.

Синтаксис:

```
modbus: function <value>
modbus: function <value>, subfunction <value>
modbus: function [!] <assigned | unassigned | public | user | reserved | all>
```

Где ! – отрицание.

Примеры:

```
modbus: access <read | write>
modbus: access read <discretes | coils | input | holding>
modbus: access read <discretes | coils | input | holding>, address <value>
modbus: access write < coils | holding>
modbus: access write < coils | holding>, address <value>
modbus: access write < coils | holding>, address <value>, value <value>
```

Параметр _<value>_ проверяет адрес или значения при доступе или записи следующим образом:

```
address 100     # адрес равен 100
address 100<>200 # адрес больше 100 и меньше 200
address >100    # адрес больше 100
address <100    # адрес меньше 100
```

Примеры:

```
modbus: access read                          # Read access
modbus: access write                         # Write access
modbus: access read input                    # Read access to Discretes Input table
modbus: access write coils                   # Write access to Coils table
modbus: access read discretes, address <100  # Read access at address smaller than 100 of Discretes Input table
modbus: access write holding, address 500, value >200  # Write value greater than 200 at address 500 of Holding Registers table
```

С помощью опцией `unit` можно проверить:

- подчиненный адрес MODBUS удаленного устройства, подключенного к подсети за мостом или шлюзом. IP-адрес назначения идентифицирует сам мост, и мост использует идентификатор модуля MODBUS для пересылки запроса на нужное ведомое устройство.

Синтаксис:

```
modbus: unit <value>
modbus: unit <value>, function <value>
modbus: unit <value>, function <value>, subfunction <value>
modbus: unit <value>, function [!] <assigned | unassigned | public | user | reserved | all>
modbus: unit <value>, access <read | write>
modbus: unit <value>, access read <discretes | coils | input | holding>
modbus: unit <value>, access read <discretes | coils | input | holding>, address <value>
modbus: unit <value>, access write < coils | holding>
modbus: unit <value>, access write < coils | holding>, address <value>
modbus: unit <value>, access write < coils | holding>, address <value>, value <value>
```

Параметр _`<value>`_ проверяет адрес или значения при доступе или записи следующим образом:

```
unit 10     # идентификатор равен 10
unit 10<>20 # идентификатор больше 10 и меньше 20
unit >10    # идентификатор больше10
unit <10    # идентификатор меньше 10
```

Примеры:

```
modbus: unit 10                                    # Unit identifier 10
modbus: unit 10, function 21                       # Unit identifier 10 and write File record function
modbus: unit 10, function 4, subfunction 4         # Unit identifier 10 and force Listen Only Mode (Diagnostics) function
modbus: unit 10, function assigned                 # Unit identifier 10 and assigned function
modbus: unit 10, function !reserved                # Unit identifier 10 and every function but reserved function
```

```
    modbus: unit 10, access read                              # Unit
identifier 10 and Read access
    modbus: unit 10, access write coils                        # Unit
identifier 10 and Write access to Coils table
    modbus: unit >10, access read discretes, address <100    # Greater
than unit identifier 10 and Read access at address smaller than 100 of
Discretes Input table
    modbus: unit 10<>20, access write holding, address 500, value >200   #
Greater than unit identifier 10 and smaller than unit identifier 20 and
Write value greater than 200 at address 500 of Holding Registers table
```

Примечания:

1. Адрес чтения и записи начинается с 1. Поэтому, если в системе использует начало с 0, то нужно добавить 1 к значениям адреса.

2. В соответствии с руководством по внедрению обмена сообщениями MODBUS в протоколе TCP/IP версии 1.0b рекомендуется держать TCP-соединение открытым с удаленным устройством, а не открывать и закрывать его для каждой транзакции MODBUS/TCP. В этом случае важно установить глубину пересборки потока как неограниченную (`stream.reassembly.depth: 0`).

3. Согласно руководству по обмену сообщениями MODBUS в TCP/IP версии 1.0b, адреса ведомых устройств MODBUS в последовательной линии назначаются от 1 до 247 (десятичное число). Адрес 0 используется как широковещательный адрес.

11.9.24. Ключевые слова DNP3

Ключевые слова DNP3 можно использовать для проверки полей в декодированных сообщениях DNP3. Ключевые слова основаны на ключевых словах Snort DNP3 и должны быть совместимы на 100 %.

11.9.24.1. dnp3_func

Это ключевое слово будет проверять кодовое название функции приложения, найденного в запросе и ответах DNP3. Его можно указать как целочисленное значение или символьное имя кода функции.

Синтаксис:

```
dnp3_func:<value>;
```

Где значением (`<value>`) является целое число от 0 до 255 включительно.

Варианты кодовых названий функций представлены в таблице 66.

Таблица 66

Кодовые названия функций		
abort_file	activate_config	assign_class
authenticate_err	authenticate_file	authenticate_req
authenticate_resp	close_file	cold_restart
confirm	delay_measure	delete_file
direct_operate	direct_operate_nr	disable_unsolicited
enable_unsolicited	freeze_at_time	freeze_at_time_nr
freeze_clear	freeze_clear_nr	get_file_info
immed_freeze	immed_freeze_nr	initialize_appl
initialize_data	open_file	operate
read	record_current_time	response
save_config	select	start_appl
stop_appl	unsolicited_response	warm_restart
write		

11.9.24.2. dnp3_ind

Это ключевое слово проверяет внутренние флаги индикаторы DNP3 в заголовке ответа приложения.

Синтаксис:

```
dnp3_ind:<flag>{,<flag>...}
```

Где `flag` – название внутреннего индикатора (таблица 67).

Это ключевое слово будет проверять по любому из перечисленных флагов. Чтобы осуществить проверку по нескольким флагам (поиск типа AND), используйте `dnp3_ind` для каждого флага, который должен быть установлен.

Таблица 67

flag			
all_stations	class_1_events	class_2_events	class_3_events
need_time	local_control	device_trouble	device_restart
no_func_code_support	object_unknown	parameter_error	event_buffer_overflow
already_executing	config_corrupt	reserved_2	reserved_1

Примеры:

```
dnp3_ind:all_stations;
dnp3_ind:class_1_events,class_2_events;
```

11.9.24.3. dnp3_obj

Это ключевое слово проверяет объекты данных приложения DNP3.

Синтаксис:

```
dnp3_obj:<group>,<variation>
```

Где значениями <group> и <variation> является целое число от 0 до 255 включительно.

11.9.24.4. dnp3_data

Это ключевое слово проверяет параметры содержимого в повторно собранном буфере приложения. Повторно собранный прикладной буфер представляет собой фрагмент DNP3 с удаленными CRC (которые встречаются каждые 16 байтов) и будет полным фрагментом, возможно, повторно собранным из нескольких кадров канального уровня DNP3.

Синтаксис:

```
dnp3_data;
```

Пример:

```
dnp3_data; content:"|c3 06|";
```

11.9.25. Ключевые слова ENIP/CIP

Ключевые слова enip_command и cip_service могут использоваться для проверки различных свойств ENIP-запросов.

Существует три способа использования этого ключевого слова:

- проверка команды ENIP с настройкой «enip_command»;

- проверка CIP Service с настройкой «cip_service»;

- проверка ENIP и службы CIP с «enip_command» и «cip_service» вместе.

Для команды ENIP проверяется поле команды, найденное в инкапсуляции ENIP.

Для сервиса CIP используется не более 3 значений, разделенных запятыми, представляющих сервис, класс и атрибут. Эти значения описаны в спецификации CIP. Классы CIP связаны с их сервисами, а атрибуты CIP связаны с их сервисами. Если нужно только проверить до сервиса, укажите только значение сервиса. Если хотите проверить атрибут CIP, то необходимо указать все 3 значения.

Синтаксис:

```
enip_command:<value>
cip_service:<value(s)>
enip_command:<value>, cip_service:<value(s)>
```

Примеры:

```
enip_command:99
cip_service:75
cip_service:16,246,6
enip_command:111, cip_service:5
```

11.9.26. Ключевые слова FTP/FTP-DATA

11.9.26.1. ftpdata_command

Проверяет ftp-канал передачи данных на основе команд FTP. В настоящее время поддерживаются команды RETR (получить (`get`) файл) и STOR (положить (`put`) файл).

Синтаксис:

```
ftpdata_command:(retr|stor)
```

Примеры:

```
ftpdata_command:retr
ftpdata_command:stor
```

Пример правила:

```
alert ftp-data any any -> any any (msg:"FTP store password"; filestore; filename:"password"; ftpdata_command:stor; sid:3; rev:1;)
```

11.9.26.2. ftpbounce

Обнаружение атак с отказами по FTP.

Синтаксис:

```
ftpbounce
```

11.9.27. Ключевые слова Kerberos

11.9.27.1. krb5_msg_type

Тип сообщения Kerberos (целое число).

Значения определены в RFC4120.

Обычные значения:

- 10 (AS-REQ);
- 11 (AS-REP);
- 12 (TGS-REQ);
- 13 (TGS-REP);
- 14 (AP-REQ);
- 15 (AP-REP);
- 30 (ERROR).

Синтаксис:

```
krb5_msg_type:<number>
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 AS-REQ message"; krb5_msg_type:10; sid:3; rev:1;)
```

11.9.27.2. krb5_cname

Имя клиента Kerberos, указанное в билете (для сообщений AS-REQ и TGS-REQ).

Если имя клиента из сообщения Kerberos состоит из нескольких частей, имя сравнивается с каждой частью, и проверка будет успешной, если какие-либо из них идентичны.

Проверка чувствительна к регистру.

Синтаксис:

```
krb5_cname; content:"name";
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 des server name"; krb5_cname; content:"des"; sid:4; rev:1;)
```

`krb5_cname` – это липкий буфер.

`krb5_cname` может быть использовано как быстрый шаблон.

11.9.27.3. krb5_sname

Имя сервера Kerberos, указанное в билете (для сообщений AS-REQ и TGS REQ) или сообщении об ошибке.

Если имя сервера из сообщения Kerberos состоит из нескольких частей, имя сравнивается с каждой частью, и проверка будет успешной, если какие-либо из них идентичны.

Проверка чувствительна к регистру.

Синтаксис:

```
krb5_sname; content:"name";
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 krbtgt server name"; krb5_sname; content:"krbtgt"; sid:5; rev:1;)
```

`krb5_sname` – это липкий буфер.

`krb5_sname` может быть использовано как быстрый шаблон.

11.9.27.4. krb5_err_code

Код ошибки Kerberos – целое число.

Это ключевое слово проверяет только Kerberos сообщения об ошибках.

Список кодов ошибок определены в RFC4120.

Синтаксис:

```
krb5_err_code:<number>
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 error C_PRINCIPAL_UNKNOWN"; krb5_err_code:6; sid:6; rev:1;)
```

11.9.27.5. krb5.weak_encryption (событие)

Событие возникает, если параметры шифрования, выбранные сервером, являются слабыми или устаревшими. Например, используется размер ключа меньше 128 или используются устаревшие шифры, такие как DES.

Синтаксис:

```
app-layer-event:krb5.weak_encryption
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 weak encryption parameters"; flow:to_client; app-layer-event:krb5.weak_encryption; classtype:protocol-command-decode; sid:2226001; rev:1;)
```

11.9.27.6. krb5.malformed_data (событие)

Событие, возникающее в случае ошибки декодирования протокола.

Синтаксис:

```
app-layer-event:krb5.malformed_data
```

Пример правила:

```
alert krb5 any any -> any any (msg:"Kerberos 5 malformed request data"; flow:to_server; app-layer-event:krb5.malformed_data; classtype:protocol-command-decode; sid:2226000; rev:1;)
```

11.9.28. Ключевые слова SNMP

11.9.28.1. version

Версия SNMP протокола (целое число). Ожидаемые значения – 1, 2 (для версии 2c) или 3.

Синтаксис:

```
snmp.version:[op]<number>
```

Версия может быть точно проверена или сравнена с помощью параметра `_op_ setting`:

```
snmp.version:3    # ровно 3
snmp.version:<3   # меньше 3
snmp.version:>=2  # больше или равно 2
```

Пример правила:

```
alert snmp any any -> any any (msg:"old SNMP version (<3)"; snmp.version:<3; sid:1; rev:1;)
```

11.9.28.2. snmp.community

SNMP `community strings` аналогично паролям для сообщений SNMP в версиях 1 и 2c. В версии 3 `community strings`, скорее всего, будет зашифрована. Проверки по ключевому слову не будет, если значение недоступно.

Значением по умолчанию для `community strings`, доступной только для чтения, часто является «`public`», а доступной для чтения и записи – «`private`».

Проверка чувствительна к регистру.

Синтаксис:

```
snmp.community; content:"private";
```

Пример правила:

```
alert snmp any any -> any any (msg:"SNMP community private"; snmp.community; content:"private"; sid:2; rev:1;)
```

`snmp.community` – это липкий буфер.

`snmp.community` может быть использовано как быстрый шаблон.

11.9.28.3. snmp.pdu_type

SNMP PDU тип (целое число).

Обычные значения:

- 0: GetRequest;
- 1: GetNextRequest;
- 2: Response;
- 3: SetRequest;
- 4: TrapV1 (устарелое);
- 5: GetBulkRequest;
- 6: InformRequest;
- 7: TrapV2;
- 8: Report.

Проверку по этому ключевому слово не будет, если значение не будет доступно (например, зашифрованное SNMP v3 сообщение).

Синтаксис:

```
snmp.pdu_type:<number>
```

Пример правила:

```
alert snmp any any -> any any (msg:"SNMP response"; snmp.pdu_type:2; sid:3; rev:1;)
```

11.9.29. Ключевые слова Base64

СОВ поддерживает декодирование данных в кодировке `base64` из буферов и проверку декодированных данных.

Это достигается за счет использования двух ключевых слов, `base64_decode` и `base64_data`. Для создания оповещения необходимо использовать оба ключевых слова.

11.9.29.1. base64_decode

Декодирует данные `base64` из буфера и делает их доступными для функции `base64_data`.

Синтаксис:

```
base64_decode:bytes <value>, offset <value>, relative;
```

Параметр `bytes` указывает, сколько байтов СОВ должна декодировать и предоставить для `base64_data`. Декодирование останавливается в конце буфера.

Опция смещения (`offset`) указывает, сколько байтов СОВ должна пропустить перед декодированием. Байты пропускаются относительно начала буфера полезной нагрузки, если относительное значение не установлено.

Опция `relative` запускает декодирование относительно предыдущего совпадения контента. Поведение по умолчанию должно начинаться с начала буфера. Эта опция позволяет пропустить байты смещения относительно предыдущего совпадения.

Примечание. Относительно `relative` и `base64_decode`: проверка содержимого, которого необходимо декодировать, `relative` должно быть первой проверкой в потоке.

11.9.29.2. base64_data

`base64_data` – это липкий буфер.

Проверяет содержимое данных, ранее декодированных с помощью `base64_decode`.

Вот пример правила проверки строки «`test`», закодированной в `base64`, которая находится внутри буфера `http_uri`.

Декодирование `relative` начинается относительно известной строки «`somestring`» с известным смещением 1. Это должно быть первое появление «`somestring`» в буфере.

Пример содержимого буфера:

```
http_uri = "GET /en/somestring&dGVzdAo=&not_base64"
```

Пример правила:

```
alert http any any -> any any (msg:"Example"; http.uri; \
content:"somestring"; base64_decode:bytes 8, offset 1, relative; \
        base64_data; content:"test"; sid:10001; rev:1;)
```

Пример содержимого буфера:

```
http_uri = "GET /en/somestring&dGVzdAo=&not_base64"
```

Пример правила:

```
alert http any any -> any any \
msg:"Example"; content:"somestring"; http_uri; \
    base64_decode:bytes 8, offset 1, relative; \
    base64_data; content:"test"; sid:10001; rev:1;)
```

11.9.30. Ключевые слова SIP

Ключевые слова SIP реализованы в виде липких буферов и могут использоваться для проверки полей в сообщениях SIP (таблица 68).

Таблица 68

Ключевое слово	Направление
sip.method	Запрос
sip.uri	Запрос
sip.request_line	Запрос
sip.stat_code	Ответ
sip.stat_msg	Ответ
sip.response_line	Ответ
sip.protocol	Оба

11.9.30.1. sip.method

Это ключевое слово проверяет метод, найденного в SIP-запросе.

Синтаксис:

```
sip.method; content:<method>;
```

Примеры методов:

- INVITE;
- BYE;
- REGISTER;
- CANCEL;
- ACK;

- OPTIONS.

Пример:

```
sip.method; content:"INVITE";
```

11.9.30.2. sip.uri

Это ключевое слово проверяет `uri`, найденное в SIP-запросе.

Синтаксис:

```
sip.uri; content:<uri>;
```

Где `<uri>` – это `uri`, соответствующий схеме SIP URI.

Пример:

```
sip.uri; content:"sip:sip.url.org";
```

11.9.30.3. sip.request_line

Это ключевое слово заставляет проверять всю строку SIP-запроса.

Синтаксис:

```
sip.request_line; content:<request_line>;
```

Где `<request_line>` – частичная или полная строка.

Пример:

```
sip.request_line; content:"REGISTER sip:sip.url.org SIP/2.0"
```

11.9.30.4. sip.stat_code

Это ключевое слово проверяет код состояния, найденный в SIP-ответе.

Синтаксис:

```
sip.stat_code; content:<stat_code>
```

Где `<status_code>` принадлежит к одной из следующих групп кодов:

- 1xx – предварительные ответы;
- 2xx – успешные ответы;
- 3xx – ответы на переадресацию;
- 4xx – ответы на сбой клиента;
- 5xx – ответы на сбой сервера;
- 6xx – реагирование на глобальные сбои.

Пример:

```
sip.stat_code; content:"100";
```

11.9.30.5. sip.stat_msg

Это ключевое слово проверяет состояние сообщения, найденного в SIP-ответе.

Синтаксис:

```
sip.stat_msg; content:<stat_msg>
```

Где `<stat_msg>` – это фраза причины, связанная с кодом состояния.

Пример:

```
sip.stat_msg; content:"Trying";
```

11.9.30.6. sip.response_line

Это ключевое слово заставляет проверять всю строку SIP-ответа.

Синтаксис:

```
sip.response_line; content:<response_line>;
```

Где `<response_line>` – это частичная или полная строка.

Пример:

```
sip.response_line; content:"SIP/2.0 100 OK"
```

11.9.30.7. sip.protocol

Это ключевое слово проверяет поле протокола из строки запроса или ответа SIP.

Если строка ответа ‘SIP/2.0 100 OK’, то этот буфер будет содержать ‘SIP/2.0’

Синтаксис:

```
sip.protocol; content:<protocol>
```

Где `<protocol>` – это версия SIP протокола.

Пример:

```
sip.protocol; content:"SIP/2.0"
```

11.9.31. Ключевые слова RFB

Ключевые слова `rfp.name` и `rfb.sectype` могут использоваться для со различных свойств RFB (Remote Framebuffer, т.е. VNC) взаимодействий.

11.9.31.1. rfb.name

Проверяет значение имени поля RFB `desktop`.

Примеры:

```
rfb.name; content:"Alice's desktop";
```

```
rfb.name; pcre:"/.* \(screen [0-9]\)$/";
```

`rfb.name` – это липкий буфер.

`rfb.name` может быть использовано как быстрый шаблон.

11.9.31.2. rfb.secresult

Проверяет значение результата RFB `security`, например, `ok`, `fail`, `too many` или `unknown`.

Примеры:

```
rfb.secresult: ok;
rfb.secresult: unknown;
```

11.9.31.3. rfb.sectype

Проверяет значение поля RFB `security type`, например, 2 для проверки подлинности VNC challenge-response, 0 для отсутствия проверки подлинности и 30 для проверки аутентификации пользователя в удаленном рабочем столе Apple.

Это ключевое слово принимает числовой аргумент после двоеточия и поддерживает дополнительные спецификаторы, такие как:

- > (больше чем)
- < (меньше чем)
- >= (больше или равно)
- <= (меньше или равно)

Примеры:

```
rfb.sectype:2;
rfb.sectype:>=3;
```

11.9.32. Ключевые слова MQTT

Различные ключевые слова могут использоваться для проверки полей в фиксированных и переменных заголовках сообщений MQTT, а также значений полезной нагрузки.

11.9.32.1. mqtt.protocol_version

Проверяет значение поля версии протокола MQTT в фиксированном заголовке.

Синтаксис:

```
mqtt.protocol_version:<min>-<max>;
mqtt.protocol_version:[<|>]<number>;
mqtt.protocol_version:<value>;
```

Примеры:

```
mqtt.protocol_version:5;
```

11.9.32.2. mqtt.type

Проверяет тип сообщения MQTT (также: тип управляющего пакета).

Допустимыми значениями являются:

- CONNECT;
- CONNACK;
- PUBLISH;
- PUBACK;
- PUBREC;
- PUBREL;
- PUBCOMP;
- SUBSCRIBE;
- SUBACK;
- UNSUBSCRIBE;
- UNSUBACK;
- PINGREQ;
- PINGRESP;
- DISCONNECT;
- AUTH;
- UNASSIGNED.

Где UNASSIGNED относится к типу кода сообщения 0.

Примеры:

```
mqtt.type:CONNECT;
mqtt.type:PUBLISH;
```

11.9.32.3. mqtt.flags

Проверяет комбинации флагов заголовка MQTT, разделенных запятыми (,). Флаги могут иметь префикс ! для обозначения отрицания, т. е. флаг с префиксом ! проверяет, что он не установлен.

Допустимые флаги:

- `dup` (повторное сообщение);
- `retain` (сообщение должно остаться у брокера).

Примеры:

```
mqtt.flags:dup,!retain;
mqtt.flags:retain;
```

11.9.32.4. mqtt.qos

Проверяет код запроса Quality of Service в фиксированном заголовке MQTT.

Допустимые значения:

- 0 («`fire and forget`») сообщение отправляется без дальнейших действий или подтверждения;
- 1 (хоть одна доставка);
- 2 (ровно одна доставка).

Примеры:

```
mqtt.qos:0;
mqtt.qos:2;
```

11.9.32.5. mqtt.reason_code

Проверка числового значения кода причины (таблица 69), который используется в MQTT 5.0 для некоторых типов сообщений.

Примеры:

```
# проверка попыток отписки от темы, на которую не подписаны
mqtt.type:UNSUBACK; mqtt.reason_code:17;

# проверка одобренных публикаций, но без подписчиков
mqtt.type:PUBACK; mqtt.reason_code:16;

# проверка попыток подключения заблокированных клиентов
mqtt.CONNACK; mqtt.reason_code:138;
```

```
# проверка неудачных попыток подключения из-за неверных учетных данных
mqtt.CONNACK; mqtt.reason_code:134;

# проверка на соединения, прерванные из-за отключения сервера
mqtt.DISCONNECT; mqtt.reason_code:139;
```

Это ключевое слово также доступно под псевдонимом `mqtt.connack.return_code`.

Таблица 69

Код причины		Описание	Пакеты
десятичный	шестнад-цатеричный		
0	0x00	Success	CONNACK, PUBACK, PUBREC, PUBREL, PUBCOMP, UNSUBACK, AUTH
0	0x00	Normal disconnection	DISCONNECT
0	0x00	Granted QoS 0	SUBACK
1	0x01	Granted QoS 1	SUBACK
2	0x02	Granted QoS 2	SUBACK
4	0x04	Disconnect with Will Message	DISCONNECT
16	0x10	No matching subscribers	PUBACK, PUBREC
17	0x11	No subscription existed	UNSUBACK
24	0x18	Continue authentication	AUTH

Продолжение таблицы 69

Код причины		Описание	Пакеты
десятичный	шестнад-цатеричный		
25	0x19	Re-authenticate	AUTH
128	0x80	Unspecified error	CONNACK, PUBACK, PUBREC, SUBACK, UNSUBACK, DISCONNECT
129	0x81	Malformed Packet	CONNACK, DISCONNECT
130	0x82	Protocol Error	CONNACK, DISCONNECT
131	0x83	Implementation specific error	CONNACK, PUBACK, PUBREC, SUBACK, UNSUBACK, DISCONNECT
132	0x84	Unsupported Protocol Version	CONNACK
133	0x85	Client Identifier not valid	CONNACK
134	0x86	Bad User Name or Password	CONNACK
135	0x87	Not authorized	CONNACK, PUBACK, PUBREC, SUBACK, UNSUBACK, DISCONNECT
136	0x88	Server unavailable	CONNACK
137	0x89	Server busy	CONNACK, DISCONNECT
138	0x8A	Banned	CONNACK
139	0x8B	Server shutting down	DISCONNECT
140	0x8C	Bad authentication method	CONNACK, DISCONNECT
141	0x8D	Keep Alive timeout	DISCONNECT
142	0x8E	Session taken over	DISCONNECT
143	0x8F	Topic Filter invalid	SUBACK, UNSUBACK, DISCONNECT
144	0x90	Topic Name invalid	CONNACK, PUBACK, PUBREC, DISCONNECT
145	0x91	Packet Identifier in use	PUBACK, PUBREC, SUBACK, UNSUBACK
146	0x92	Packet Identifier not found	PUBREL, PUBCOMP
147	0x93	Receive Maximum exceeded	DISCONNECT
148	0x94	Topic Alias invalid	DISCONNECT
149	0x95	Packet too large	CONNACK, DISCONNECT
150	0x96	Message rate too high	DISCONNECT

Окончание таблицы 69

Код причины		Описание	Пакеты
десятичный	шестнад-цатеричный		
151	0x97	Quota exceeded	CONNACK, PUBACK, PUBREC, SUBACK, DISCONNECT
152	0x98	Administrative action	DISCONNECT
153	0x99	Payload format invalid	CONNACK, PUBACK, PUBREC, DISCONNECT
154	0x9A	Retain not supported	CONNACK, DISCONNECT
155	0x9B	QoS not supported	CONNACK, DISCONNECT
156	0x9C	Use another server	CONNACK, DISCONNECT
157	0x9D	Server moved	CONNACK, DISCONNECT
158	0x9E	Shared Subscriptions not supported	SUBACK, DISCONNECT
159	0x9F	Connection rate exceeded	CONNACK, DISCONNECT
160	0xA0	Maximum connect time	DISCONNECT
161	0xA1	Subscription Identifiers not supported	SUBACK, DISCONNECT
162	0xA2	Wildcard Subscriptions not supported	SUBACK, DISCONNECT

11.9.32.6. mqtt.connack.session_present

Проверяет флаг MQTT CONNACK `session_present`. Значения могут быть `yes` (да), `true` (правда), `no` (нет) или `false` (ложь).

Примеры:

```
mqtt.CONNACK; mqtt.connack.session_present:true;
```

11.9.32.7. mqtt.connect.clientid

Проверка самоназначенного идентификатора клиента в сообщении MQTT CONNECT.

Примеры:

```
mqtt.connect.clientid; pcre:"/^mosq.*/";
```

```
mqtt.connect.clientid; content:"myclient";
```

Ключевое слово `mqtt.connect.clientid` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.8. mqtt.connect.flags

Проверяет комбинацию флагов MQTT CONNECT, разделенных запятыми (,). Флаги могут иметь префикс ! для обозначения отрицания, т. е. осуществляется проверка что не содержится данных отмеченных флагом с префиксом !.

Допустимые флаги:

- `username` (сообщение содержит имя пользователя);
- `password` (сообщение содержит пароль);
- `will` (сообщение содержит определение `will`);
- `will_retain` (`will` должно быть сохранено у брокера);
- `clean_session` (начать с новой сессии).

Примеры:

```
mqtt.connect.flags:username,password,!will;
mqtt.connect.flags:username,!password;
mqtt.connect.flags:clean_session;
```

11.9.32.9. mqtt.connect.password

Проверка на наличие пароля учетных данных в MQTT CONNECT message.

Примеры:

```
mqtt.connect.password; pcre:"/^123[0-9]*/";
mqtt.connect.password; content:"swordfish";
```

Ключевое слово `mqtt.connect.password` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.10. mqtt.connect.username

Проверка на наличие имени пользователя учетных данных в MQTT CONNECT message.

Примеры:

```
mqtt.connect.username; content:"benson";
```

Ключевое слово `mqtt.connect.username` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.11. mqtt.connect.willmessage

Проверка `will message` в MQTT CONNECT message, если `will` определен.

Примеры:

```
mqtt.connect.willmessage; pcre:"/^fooba[rz]/";
mqtt.connect.willmessage; content:"hunter2";
```

Ключевое слово `mqtt.connect.willmessage` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.12. mqtt.connect.willtopic

Проверка `will topic` в MQTT CONNECT `message`, если `will` определен.

Примеры:

```
mqtt.connect.willtopic; pcre:"/^hunter[0-9]/";
```

Ключевое слово `mqtt.connect.willtopic` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.13. mqtt.publish.message

Проверка полезной нагрузки, которая будет опубликована в MQTT PUBLISH message.

Примеры:

```
mqtt.type:PUBLISH; mqtt.publish.message; pcre:"/uid=[0-9]+/";
```

Проверка на публикацию JPEG изображений:

```
mqtt.type:PUBLISH; mqtt.publish.message; content:"|FF D8 FF E0|";
startswith;
```

Ключевое слово `mqtt.publish.message` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.14. mqtt.publish.topic

Проверка темы, которая будет опубликована MQTT PUBLISH message.

Примеры:

```
mqtt.publish.topic; content:"mytopic";
```

Ключевое слово `mqtt.publish.topic` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.15. mqtt.subscribe.topic

Проверка по любой из тем, на которую подписались в MQTT SUBSCRIBE message.

Примеры:

```
mqtt.subscribe.topic; content:"mytopic";
```

Ключевое слово `mqtt.subscribe.topic` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.32.16. mqtt.unsubscribe.topic

Проверка по любой из тем, от которой отписались в MQTT UNSUBSCRIBE message.

Примеры:

```
mqtt.unsubscribe.topic; content:"mytopic";
```

Ключевое слово `mqtt.unsubscribe.topic` – это липкий буфер и может быть использовано в качестве быстрого шаблона.

11.9.33. Ключевые слова HTTP2

Фреймы HTTP2 группируются в транзакции на основе идентификатора потока, если он не равен 0. Для фреймов с идентификатором потока 0, эффекты которых являются глобальными для соединения, транзакция создается для каждого фрейма.

11.9.33.1. http2.frametype

Проверка типа фрейма, присутствующего в транзакции.

Примеры:

```
http2.frametype:GOAWAY;
```

11.9.33.2. http2.errorcode

Проверяет код ошибки в GOWAY или RST_STREAM frame.

Примеры:

```
http2.errorcode: NO_ERROR;
http2.errorcode: INADEQUATE_SECURITY;
```

11.9.33.3. http2.priority

Проверка значения поля приоритета HTTP2, присутствующего в PRIORITY или HEADERS фрейма.

Это ключевое слово принимает числовой аргумент после двоеточия и поддерживает дополнительные спецификаторы, такие как:

- > (больше чем);

- < (меньше чем);

- `x-y` (диапазон значений между `x` и `y`).

Примеры:

```
http2.priority:2;
http2.priority:>100;
http2.priority:32-64;
```

11.9.33.4. http2.window

Проверка значения поля HTTP2, присутствующего в фрейме WINDOWUPDATE.

Это ключевое слово принимает числовой аргумент после двоеточия и поддерживает дополнительные спецификаторы, такие как:

- > (больше чем);

- < (меньше чем);

- `x-y` (диапазон значений между `x` и `y`).

Примеры:

```
http2.window:1;
http2.window:<100000;
```

11.9.33.5. http2.size_update

Проверка размера таблицы динамических заголовков HTTP2.

Это ключевое слово принимает числовой аргумент после двоеточия и поддерживает дополнительные спецификаторы, такие как:

- > (больше чем);

- < (меньше чем);

- `x-y` (диапазон значений между `x` и `y`).

Примеры:

```
http2.size_update:1234;
http2.size_update:>4096;
```

11.9.33.6. http2.settings

Проверка имени и значения HTTP2 заданного в фрейме SETTINGS.

Это ключевое слово принимает числовой аргумент после двоеточия и поддерживает дополнительные спецификаторы, такие как:

- > (больше чем);

- < (меньше чем);

- `x-y` (диапазон значений между `x` и `y`).

Примеры:

```
http2.settings:SETTINGS_ENABLE_PUSH=0;
http2.settings:SETTINGS_HEADER_TABLE_SIZE>4096;
```

11.9.33.7. http2.header_name

Проверка имени заголовка HTTP2 в HEADER (или PUSH_PROMISE, или CONTINUATION).

Примеры:

```
http2.header_name; content:"agent";
```

`http2.header_name` – это липкий буфер.

Ключевое слово `http2.header_name` может быть использовано как быстрый шаблон.

11.9.33.8. http2.header

Проверяет имя и значение заголовка HTTP2 в HEADER (или PUSH_PROMISE, или CONTINUATION). Имя и значение объединяются с помощью

«:», двоеточия и пробела. Каждое двоеточие в имени или значении должно быть экранировано двойным двоеточием «:» для обнаружения.

Примеры:

```
http2.header; content:"agent: nghttp2";
http2.header; content:"custom-header: I love:colons";
```

Ключевое слово `http2.header` – это липкий буфер и может быть использовано как быстрый шаблон.

11.9.34. Ключевые слова прикладного уровня

11.9.34.1. app-layer-protocol

Проверка на обнаружение протокола прикладного уровня (уровня приложений).

Синтаксис:

```
app-layer-protocol:[!]<protocol>;
```

Примеры:

```
app-layer-protocol:ssh;
app-layer-protocol:!tls;
app-layer-protocol:failed;
```

Специальное значение «`failed`» может быть использовано при проверке потоков, в которых не удалось обнаружить протокол. Это может произойти, если СОВ не удалось распознать протокол.

11.9.34.1.1. Условия помощи

Обнаружение протокола может дать сбой в нескольких случаях:

- обе стороны проверены и совпадений не обнаружено;
- обнаружение на стороне A не удалось, сторона B вообще не имеет трафика (например, канал передачи данных FTP);
- обнаружение на стороне A не удалось, сторона B имеет настолько мало данных, что обнаружение неубедительно.

В последних двух случаях устанавливается событие уровня приложения: `appplayer_proto_detection_skipped`.

11.9.34.2. app-layer-event

Проверка по событиям, генерируемым анализаторами уровня приложения и механизмом обнаружения протоколов.

Синтаксис:

```
app-layer-event:<event name>;
```

Примеры:

```
app-layer-event:applayer_mismatch_protocol_both_directions;
app-layer-event:http.gzip_decompression_failed;
```

11.9.34.2.1. Обнаружение протокола

11.9.34.2.1.1. applayer_mismatch_protocol_both_directions

Направления `toserver` и `toclient` имеют разные протоколы. Например, клиент общается по протоколу HTTP с SSH сервером.

11.9.34.2.1.2. applayer_wrong_direction_first_data

В некоторых реализациях протокола в СОВ есть требование в отношении первого направления данных. Парсер HTTP является примером этого.

11.9.34.2.1.3. applayer_detect_protocol_only_one_direction

Обнаружить протокол удалось только в одном направлении. Для FTP и SMTP это ожидаемо.

11.9.34.2.1.4. applayer_proto_detection_skipped

Обнаружение протокола было пропущено из-за условий помощи (см. п. 11.9.34.1.1).

11.9.35. Ключевое слово `IP Reputation`

`IP Reputation` можно использовать в правилах с помощью нового ключевого слова правила `iprep`.

11.9.35.1. iprep

Директива `iprep` соответствует информации о репутации IP-адреса хоста.

Синтаксис:

```
iprep:<side to check>,<category>,<operator>,<reputation score>
```

Где:

`side to check: <any|src|dst|both>` (сторона проверки)

`category:` короткое название категории

`operator: <, >, =`

`reputation score: 1-127` (показатель репутации)

Пример:

```
alert ip $HOME_NET any -> any any \
msg:"IPREP internal host talking to CnC server"; \
flow:to_server; iprep:dst,CnC,>,30; sid:1; rev:1;)
```

Это правило будет предупреждать, когда система, заданная в `$HOME_NET`, действует как клиент при общении с любым IP-адресом в категории CnC, для которого показатель репутации установлен выше 30.

11.9.35.1.1. IP-only

11.9.35.1.2. Ключевое слово `iprep` совместимо с правилами `IP-only`. Это означает, что такое правило, как:

```
alert ip any any -> any any (msg:"IPREP High Value CnC"; \
iprep:src,CnC,>,100; sid:1; rev:1;)
```

будет проверяться только один раз для каждого направления потока.

11.9.36. Наборы данных

Используя набор данных и ключевое слово `datarep`, можно проверять большие объемы данных с любым липким буфером.

Например, для сопоставления с черным списком DN под названием dnsbl:

```
dns.query; dataset:isset,dns-bl;
```

Эти ключевые слова знают о преобразованиях, поэтому чтобы найти DNS-запрос в черном списке MD5:

```
dns.query; to_md5; dataset:isset,dns-bl;
```

11.9.36.1. Глобальная конфигурация (опционально)

Наборы данных могут быть дополнительно определены в основной конфигурации. Наборы также могут быть объявлены в синтаксисе правила.

Пример наборов для отслеживания уникальных значений:

```
datasets:
  ua-seen:
    type: string
    state: ua-seen.lst
  dns-sha256-seen:
    type: sha256
    state: dns-sha256-seen.lst
```

Правило работы с вышеперечисленным:

```
alert dns any any -> any any (msg:”dns list test”; dns.query; to_sha256; dataset:isset,dns-sha256-seen; sid:123; rev:1;)
```

```
alert http any any -> any any (msg: “http user-agent test”; http.user_agent; dataset:set,ua-seen; sid:234; rev:1;)
```

Также возможно опционально определить глобальные значения `memcap` и `hashsize` по умолчанию.

Пример:

```
datasets:
  defaults:
    memcap: 100mb
    hashsize: 2048
  ua-seen:
    type: string
    load: ua-seen.lst
```

Или определить `memcap` и `hashsize` для каждого набора данных.

Пример:

```
datasets:
  ua-seen:
    type: string
    load: ua-seen.lst
    memcap: 10mb
    hashsize: 1024
```

11.9.36.2. Ключевые слова `Rule`

11.9.36.2.1. dataset

Наборы данных бинарны: что-то есть в наборе, другое нет.

Синтаксис:

```
dataset:<cmd>,<name>,<options>;
dataset:<set|isset|isnotset>,<name> \
[, type <string|md5|sha256>, save <file name>, load <file name>, \
state <file name>, memcap <size>, hashsize <size>];
```

`type <type>`

тип данных: string, md5, sha256.

`load <file name>`

имя файла для загрузки данных при запуске СОВ.

`state`

устанавливает имя файла для загрузки и сохранения набора данных.

`save <file name>`

расширенный параметр для установки имени файла для сохранения данных в памяти при выходе из СОВ.

`memcap <size>`

максимальный предел памяти для соответствующего набора данных.

`hashsize <size>`

допустимый размер хеша для соответствующего набора данных.

Примечание. Нельзя смешивать `load` и `state` или `save` и `state`.

11.9.36.2.2. datarep

`datarep` (Data Reputation) позволяет сопоставлять данные со списком репутации.

Синтаксис:

```
datarep:<name>,<operator>,<value>, \
   [, load <file name>, type <string|md5|sha256>, \
memcap <size>, hashsize <size>];
```

Примеры правил могут выглядеть так:

```
alert dns any any -> any any (dns.query; to_md5; datarep:dns_md5, >, 200, load dns_md5.rep, type md5, memcap 100mb, hashsize 2048; sid:1;)
alert dns any any -> any any (dns.query; to_sha256; datarep:dns_sha256, >, 200, load dns_sha256.rep, type sha256; sid:2;)
alert dns any any -> any any (dns.query; datarep:dns_string, >, 200, load dns_string.rep, type string; sid:3;)
```

В этих примерах строка DNS-запроса проверяется по трем различным спискам репутации. Список MD5, список SHA256 и список необработанных строк (буфер). Правила будут совпадать только в том случае, если данные есть в списке, а значение репутации выше 200.

11.9.36.3. Перезагрузка правил

Наборы, определенные в yaml, или наборы, которые используют только `state` или `save`, считаются динамическими наборами. Они не перезагружаются во время перезагрузки правил.

Наборы, определенные в правилах с использованием только `load`, считаются статическими тестами. Ожидается, что они не изменятся во время выполнения. Во время перезагрузки правил они перезагружаются с диска. Перезагрузка вступает в силу после завершения всего процесса перезагрузки правила.

11.9.36.4. Unix сокеты

11.9.36.4.1. dataset-add

Это команда Unix сокет для добавления данных в набор. При успешном добавлении становится активным мгновенно.

Синтаксис:

```
dataset-add <set name> <set type> <data>
```

Где:

`set name`
: имя уже определенного набора данных.

`type`
: тип данных: string, md5, sha256.

`data`
: данные для добавления в сериализованной форме (base64 для строки, шестнадцатеричная запись для md5/sha256).

Пример добавления «google.com» для установки «myset»:

```
dataset-add myset string Z29vZ2xlLmNvbQ==
```

11.9.36.4.2. dataset-remove

Это команда Unix сокет для удаления данных из набора. При успехе, удаляется мгновенно.

Синтаксис:

```
dataset-remove <set name> <set type> <data>
```

Где:

`set name`

имя уже определенного набора данных.

`type`

тип данных: string, md5, sha256.

`data`

данные для удаления в сериализованной форме (base64 для строки, шестнадцатеричная запись для md5/sha256).

11.9.36.5. Форматы файлов

Наборы данных используют простой формат CSV, в котором данные располагаются построчно в файле.

11.9.36.5.1. Типы данных

`string`

в файле как строка в кодировке base64.

`md5`

в файле как строка в шестнадцатеричной кодировке.

`sha256`

в файле как строка в шестнадцатеричной кодировке.

6.33.5.2. `dataset`

Наборы данных имеют простую структуру, в которой на строку файла приходится одна часть данных.

Синтаксис:

```
<data>
```

Например, ua-seen с типом string:

```
TW96aWxsYS80LjAgKGNvbXBhdGlibGU7ICk=
```

который при передаче в `base64 -d` раскрывает свое значение:

```
Mozilla/4.0 (compatible; )
```

11.9.36.5.2. datarep

Формат `datarep` следует за набором данных, и ожидает, что есть еще 1 поле CSV.

Синтаксис:

`<data>,<value>`

12. ОБЩИЕ НАСТРОЙКИ СЕРВЕРА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА

Настройка работы сервера ГИ МЭ ИВК КОЛЬЧУГА-К осуществляется через конфигурационный файл:

`/usr/share/web-kolchuga/conf/server.conf`

Параметры задаются в формате JSON. При изменении конфигурационного файла требуется перезапуск сервера ГИ МЭ ИВК КОЛЬЧУГА-К – служба node.

12.1. Настройка параметров HTTPS

Параметры работы сервера ГИ по протоколу HTTPS расположены в блоке `ssl` конфигурационного файла. Доступны для настройки следующие параметры:

- `enable` – включение работы сервера по протоколу HTTPS;
- `port` – порт HTTPS;
- `keyPath` – путь к файлу открытого ключа;
- `certPath` – путь к файлу сертификата.

В случае, если открытый ключ и сертификат содержатся в одном файле, указывается путь к файлу в обоих параметрах `keyPath` и `certPath`.

12.2. Настройка продолжительности сессии пользователя

Продолжительность сессии пользователя в ГИ настраивается в параметре `jwtExpiresIn` блока `auth`. Значение указывается в миллисекундах. По умолчанию продолжительность сессии пользователя равна 30 минутам.

12.3. Настройка электронной почты

Настройка почтового ящика для отправки уведомлений осуществляется в блоке email конфигурационного файла.

Во вложенном параметре `config` определяются параметры почтового сервера, а именно:

- `host` – адрес почтового сервера;
- `port` – порт почтового сервера;
- `secure` – флаг использования TLS при соединении с сервером;

- `auth` – данные для аутентификации: user – имя учетной записи, pass – пароль учетной записи;
- `tls` – параметры TLS при соединении с сервером: `rejectUnauthorized` – если установлено `false`, то соединение будет установлено с сертификатом, не прошедшим проверку подлинности.

Во вложенном параметре `defaultMessageOptions` определяются параметры по умолчанию для отправки почты:

- `from` – отправитель почтового сообщения;
- `to` – получатель почтового сообщения.

12.4. Настройка перенаправления системной почты

Настройка перенаправления системной почты осуществляется в блоке `systemMailRelay` конфигурационного файла. Доступны для настройки следующие параметры:

- `enable` – активация перенаправления системной почты;
- `replaceHeaders` – флаг замены заголовков системных почтовых сообщений на определенные в параметре `defaultMessageOptions` настроек электронной почты;
- `users` – список пользователей, почтовые ящики которых необходимо просматривать для перенаправления почты;
- `deleteOnRelay` – флаг удаления перенаправленных почтовых сообщений (если указать `false`, то сообщения будут помечаться, как прочитанные, и удаляться не будут из почтового ящика пользователя).

13. ПРОЦЕДУРЫ ОБНОВЛЕНИЯ

В случае обнаружения уязвимостей в программных модулях МЭ ИВК КОЛЬЧУГА-К устранение уязвимости осуществляется путем установки обновления, либо путем принятия иных организационно-технических мер, направленных на затруднение возможности эксплуатации уязвимости. При этом сами меры носят временный характер, а их использование допустимо до момента выпуска соответствующего обновления.

На официальном сайте предприятия-изготовителя (`https://ivk.ru`) или по электронной почте потребители получают:

- информацию о мерах, направленных на нейтрализацию выявленных уязвимостей МЭ ИВК КОЛЬЧУГА-К;
- выпускаемые обновления МЭ ИВК КОЛЬЧУГА-К, включая инструкцию по установке, КС и используемый алгоритм подсчета.

Предприятие-изготовитель обеспечивает гарантированную доставку конечным пользователям файла с обновлениями или обновленной версией ПО МЭ ИВК КОЛЬЧУГА-К, например, через электронную ссылку для самостоятельного скачивания обновления с официального сайта.

При получении обновлений МЭ ИВК КОЛЬЧУГА-К перед их установкой необходимо проверить подлинность и целостность полученных обновлений с использованием указанных алгоритмов.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АПМДЗ – аппаратно-программный модуль доверенной загрузки;

АС – автоматизированная система;

АРМ – автоматизированное рабочее место;

БСВВ – базовая систем ввода-вывода;

ГИ – графический интерфейс;

КСЗ – комплекс средств защиты;

КС – контрольная сумма;

МЭ – межсетевой экран;

ОС – операционная система;

ОЗУ – оперативное запоминающее устройство;

ПО – программное обеспечение;

ПАК – программно-аппаратный комплекс;

СОВ – система обнаружения вторжений;

СКЗИ –средство криптографической защиты информации;

ФБ – функции безопасности;

ФБО – функции безопасности объекта;

ФТБ – функциональные требования безопасности;

ЭД – эксплуатационная документация.

Лист регистрации изменений									
Номера листов (страниц)					*Всего листов (страниц) в докум.*	*№ документа*	*Входящий № сопроводительного докум. и дата*	*Подп.*	*Дата*
Изм.	*измененных*	*замененных*	*новых*	*аннулированных*					