УТВЕРЖДЁН
RU.РДПТ.00012-32 34 01-ЛУ

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
EcoDPIOS-DC

Руководство оператора

RU.РДПТ.00012-32 34 01

Листов 161

2023

АННОТАЦИЯ

В настоящем руководстве описан порядок установки и первичной настройки специализированного встраиваемого программного обеспечения EcoDPIOS-DC.

Некоторые команды и значения параметров могут отличаться в более поздних или более ранних версиях программного обеспечения. Для получения информации об актуальной версии программного обеспечения и документации обратитесь на сайт компании РДП.РУ или в службу технической поддержки.

СОДЕРЖАНИЕ

1. НАЗНАЧЕНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Программное обеспечение EcoDPIOS-DC (ПО EcoDPIOS-DC) представляет собой специализированное встраиваемое ПО с микросервисной архитектурой, применяемое для построения распределённой системы сбора и анализа статистических данных, которая используется в составе Автоматизированной системы обеспечения безопасности российского сегмента информационно-телекоммуникационной сети Интернет первого этапа (АСБИ первого этапа).

Схема основных компонентов системы сбора и анализа статистических данных, построенной на базе ПО EcoDPIOS-DC, приведена в Приложении А. Голубым цветом выделены программные средства EcoDPIOS-DC (см. Приложение А).

Программные средства EcoDPIOS-DC включают:

– Систему предварительного формирования списков (далее – СПФС), размещенную на первом уровне эшелона. СПФС содержит сервисы для обработки данных от устройств EcoFilter и EcoBalancer.

– Систему централизованной обработки списков (далее – СЦОС), размещенную в Централизованной системе управления оборудованием (далее – ЦСУО). СЦОС содержит сервисы для дальнейшей обработки данных, полученных от СПФС. Связана с устройствами EcoHighway и EcoFilter второго уровня эшелона.

1.1. СПФС

Функции СПФС:

– получение журналов блокировок и журналов трафика от устройств фильтрации;
– разбор содержимого журналов блокировок и журналов трафика;
– получение точек маршрутов прохождения пользовательского трафика от устройств фильтрации;
– хранение разобранной информации в локальной базе данных;
– формирование предварительных чёрных и белых списков фильтрации;
– отправка сформированных предварительных чёрных и белых списков в СЦОС для дальнейшего анализа и формирования окончательных списков;
– предоставление статистических данных по NTP-ответам;
– сбор списков IP-адресов на основании разрешения заданных доменных имен для заданных протоколов;

– анализ подозрительного трафика на наличие DDoS-угрозы (в рамках одного узла ТСПУ) на соответствие преднастроенным правилам, выявление аномалий трафика;
– предоставление API для мониторинга и сбора метрик, API загрузки списков IP-адресов или SNI.

В состав СПФС входят следующие сервисы:

Таблица 1 – Сервисы СПФС

№	Название	Описание
1	drop-log-reader	Сервис приема журнала гистограммных и debug логов от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
2	accounting-log-reader	Сервис приема журнала accounting логов (информация об установленных сессиях) от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
3	clickstream-log-reader	Сервис приема журнала clickstream логов (информация о HTTP/HTTPS сессиях) от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
4	dns-log-reader	Сервис приема журнала dns логов от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
5	proto-log-reader	Сервис приема журнала proto логов (информация о распознанных протоколах) от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
6	shortlist-log-reader	Сервис приема журнала shortlist логов (лог блокировок по единому реестру и по ip/url) от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
7	packets-routes-log-reader	Сервис получения данных по точкам маршрутов пользовательского трафика от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse
8	netflow-log-reader	Сервис получения данных NetFlow от устройств EcoFilter-Balancer и EcoHighway, их обработки и отправки в режиме реального времени в сервис log-proxy-reader или их записи в СУБД ClickHouse
9	acl-creator-black	Сервис формирования предварительных чёрных списков по подготовленным данным логов устройств фильтрации, собранным в СУБД ClickHouse сервисами drop-log-reader
10	hrandom-creator	Сервис предварительной обработки данных журналов гистограммных логов устройств фильтрации, собранных в СУБД ClickHouse
11	dns-prober-creator	Сервис подготовки пользовательских загружаемых списков ip адресов, соотнесенных с доменными именами, с применением функционала socks proxy, настроенного на устройстве фильтрации
12	ddos-detector	Сервис противодействия DDoS-атакам. Получает обработанные пакеты от accounting-log-reader, clickstream-log-reader и proto-log-reader через events-collector-queue. Анализ подозрительного трафика (в рамках одного узла

№	Название	Описание
		ТСПУ) на соответствие преднастроенным правилам, выявление аномалий трафика
13	log-proxy-reader	Сервис получения обработанных статистических данных с применением аналитических запросов на структурированных больших данных, подготовленных сервисом drop-log-reader в СУБД ClickHouse. Предоставляет API для чтения журналов сессий и блокировок
14	events-collector-queue	Сервис передачи обработанных пакетов от accounting-log-reader, clickstream-log-reader и proto-log-reader сервису ddos-detector. Запрашивает информацию о протоколах в сервисе list-of-protocols (СЦОС). Предоставляет интерфейс для приёма сообщения журналов и интерфейс подписки на очереди сообщений
15	qoe-tracker	Сервис для расчета значений метрик QoE для списка отслеживаемых ресурсов и предоставления их для чтения через протокол HTTP в формате метрик Prometheus

1.2. СЦОС

Функции СЦОС:

- получение сформированных предварительных чёрных и белых списков фильтрации от СПФС;
- хранение предварительных списков в буферной базе данных;
- формирование основных белых и чёрных списков фильтрации;
- хранение основных белых и чёрных списков;
- формирование серых списков фильтрации;
- формирование и обеспечение доступа к данным по маршрутам пользовательского трафика;
- хранение и обеспечение доступа к спискам доменов и соответствующих им id протоколов;
- отправка сформированных списков на оборудование балансировки;
- обеспечение доступа оборудованию фильтрации к чёрным спискам;
- предоставление API для работы с записями ACL, для просмотра истории добавления записей в ACL, получения списков данных по сигнатурам, для получения данных о маршрутах пакетов трафика абонента, для наполнения и просмотра списков доменов и соответствующих id, получения итоговых списков фильтрации, для управления списками протоколов и игнорируемых протоколов.

В состав СЦОС входят следующие сервисы:

Таблица 2 – Сервисы СЦОС

№	Название	Описание
1.	packets-routes	Сервис получения данных по точкам маршрутов прохождения пользовательского трафика из СПФС. Предоставляет API для получения маршрутов пользовательского трафика
2.	list-of-protocols	Сервис хранения и централизованного доступа к актуальному списку протоколов и актуальному списку игнорируемых протоколов. Сервис list-of-protocols может передавать список протоколов сервису events-collector-queue (СПФС)
3.	list-for-dns-prober	Сервис хранения и доступа к спискам доменов и соответствующих им id протоколов для сервиса dns-prober-creator из СПФС. Сервис предоставляет API для наполнения и просмотра списка доменов и соответствующих им id протоколов
4.	acl-list-cache-dns-prober v4 + v6	Сервисы кэширующих чёрных списков dns-prober IPv4 и IPv6, собирающие данные по ip адресам и связанным с ними кодам протоколов, получаемых от сервиса dns-prober-creator из СПФС
5.	protocols-prober-data	Сервис кэширования и доступа к агрегированным пользовательским загружаемым спискам, получаемым от acl-differ-dns-prober v4+v6. Предоставляет API для получения пользовательских загружаемых списков IPv4 и IPv6
6.	acl-differ-dns-prober v4+v6	Сервисы обеления списков dns-prober IPv4 и IPv6. Работают с сервисами кэширующих черных списков dns-prober IPv4 и IPv6 и белыми списками СЦОС IPv4 и IPv6
7.	acl-differ-web	Сервис, предоставляющий API для скачивания подготовленного списка устройствами фильтрации
8.	acl-inverter v4+v6	Сервисы формирования белых списков исключений для использования в сервисе acl-differ-web
9.	acl-differ v4+v6	Сервисы получения отбелённых чёрных списков (также - сервисы сравнения списков) для IPv4, IPv6
10.	acl-list-white v4+v6	Сервисы белого списка IPv4, 6 (также - сервисы хранения основного белого списка). Сервисы предоставляют API для наполнения списка
11.	acl-list-black v4+v6	Сервисы чёрного списка IPv4, IPv6 (также - сервисы хранения основного черного списка)
12.	acl-creator-from-cache-black v4+v6	Сервисы формирования чёрных списков по данным acl-list для IPv4, IPv6

№	Название	Описание
13.	acl-list-cache-black v4+v6	Сервисы хранения промежуточных данных IPv4, IPv6
14.	acl-manager	Сервис управления записями IPv4 и IPv6 в VRF (также – сервис доставки списков), является источником актуальных списков портов и подсетей для EcoHighway
15.	envoy	Web-grpc gateway для UI acl-list
16.	rkn-creator	Сервис, формирующий чёрный список по данным реестра РКН. Список используется устройствами EcoHighway
17.	rkn-resources-list	Сервис, формирующий список URL по данным реестра РКН. Список используется устройствами фильтрации
18.	acl-list-ismon v4 + v6	Сервисы обновляемых списков префиксов IPv4, IPv6 российского сегмента сети Интернет. Предоставляют API для наполнения списка префиксов IPv4, IPv6

2. УСЛОВИЯ ПРИМЕНЕНИЯ

Для корректной работы ПО EcoDPIOS-DC должно быть установлено на серверную платформу, отвечающую приведённым ниже требованиям (из расчёта обеспечения производительности 500 Гбит/с) (см. Таблица 3).

Таблица 3 – Требования к оборудованию серверной платформы

Наименование	**Характеристики**
Процессор	Intel® Xeon® Gold 6314U 3.4 ГГц (32 ядра)
Оперативная память	512 ГБ DDR4 3200 МГц ECC Registered
Накопитель	1 ТБ NVMe SSD
Порты служебные	не менее 1 порта 1GbE RJ45 (Management/IPMI, Console) не менее 1 порта 10G SFP+ (LOG)
Порты данных	2 порта 10GbE SFP+ (XL710)
Питание	2 блока питания постоянного или переменного тока, работающие по схеме с резервированием

3. УСТАНОВКА СПФС

3.1. Подготовка к установке

Ниже перечислены программные инструменты, которые требуются для установки СПФС, и настройки, которые необходимо предварительно выполнить:

- Docker версии не ниже 19.03 с отключенным docker-proxy;
- Docker Compose с поддержкой спецификации файла docker-compose.yaml 3.7 или выше;
- для работы контейнеров может потребоваться настройка SELinux и FirewallD;
- выполнить в утилите sysctl указанные ниже настройки. Рекомендуется сохранить эти настройки в файле sysctl.conf для восстановления после перезагрузки:

```
net.ipv4.udp_mem=750000000 1400000000 1500000000

net.core.rmem_max=1500000000
net.core.wmem_max=1500000000

net.core.rmem_default=750000000
net.core.wmem_default=750000000

net.ipv4.udp_rmem_min=750000000
net.ipv4.udp_wmem_min=750000000

net.core.netdev_budget=600
net.core.netdev_max_backlog=900000
net.core.netdev_tstamp_prequeue=0

# Не валидировать SrcIP
net.ipv4.conf.<NIC>.rp_filter=0
где <NIC> - это имя интерфейса, на который приходят журналы.
```

- Добавить в автозагрузку следующие параметры:

```
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag
cpupower frequency-set --governor performance
```

- смонтировать раздел для хранения БД с опциями **noatime** и **nodiratime**;
- на устройствах EcoFilter в разделе конфигурации **clickstream** задать **log_format binary**. Пример настроек:

```
system.clickstream# show
enable
log_interface default
server_ip_and_port 172.31.255.231:5552
source_port 5552

system.qoe_log# show
```

```
enable
log_interface default
syn_log on
server_ip_and_port 172.31.255.231:5551
source_port 5551
log_format binary

system.dns_log# show
enable
log_interface default
server_ip_and_port 172.31.255.231:5556
source_port 5556
log_format binary
```

– В файле docker-compose.yaml необходимо выполнить настройку секций x-cpuset-0 и x-cpuset-1. Секции содержат настройки cpuset. Для корректной настройки cpuset необходимо определить к какой numa-ноде относится сетевая карта, которая принимает поток событий от EcoFILTER. Сервисы *-log-reader (запущенные из образа drop-log-reader) должны быть запущены на CPU, связанных с той же numa-нодой, которой принадлежит сетевая карта. Остальные сервисы должны быть запущены на CPU, связанных с другой numa-нодой. Для сервисов *-log-reader определена секция x-cpuset-0, для остальных сервисов x-cpuset-1. Узнать привязку сетевой карты к конкретной numa-ноде можно следующей командой (eno2 - это пример имени интерфейса, на который приходят журналы):

```
cat /sys/class/net/eno2/device/numa_node
0
```

Распределение CPU по numa-нодам можно получить следующей командой:

```
numactl --hardware
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 24 25 26 27 28 29 30 31 32 33
34 35
node 0 size: 128618 MB
node 0 free: 85764 MB
node 1 cpus: 12 13 14 15 16 17 18 19 20 21 22 23 36 37 38 39 40 41 42
43 44 45 46 47
node 1 size: 128996 MB
node 1 free: 9579 MB
node distances:
node   0   1
  0:  10  21
  1:  21  10
```

По приведённым выше примерам секции x-cpuset-0 и x-cpuset-1 будут иметь следующий вид:

```
x-cpuset-0:
  &cpuset0
  cpuset: 0-11,24-35
```

```
x-cpuset-1:
  &cpuset1
  cpuset: 12-23,36-47
```

Для площадок, где нагрузка на accounting-log-reader превышает 200k pps входящих пакетов, необходимо изменить следующие параметры

- в секции env:

```
  - MAX_INPUT_PACKETS_PER_SECOND=800000
  - GOMAXPROCS=12
```

- на тестовом стенде указанные параметры позволяют получить обработку 330k pps входящих пакетов. Потребление ОЗУ составит ~40Гб.
- на сетевых картах рекомендуется увеличить размер кольцевого буфера на приём. Можно установить 1024/2048/4096.

Пример команды:

```
ethtool -G <NIC> rx 1024
```

где <NIC> - имя интерфейса, на который приходят журналы.

3.2. Процедура установки

Для установки необходимо выполнить действия:

1. Распаковать архив operator-docker-compose.tar и перейти в директорию operator-docker-compose.

2. Загрузить образы контейнеров, отправив для каждого образа команду

```
docker load -i <имя образа>.tar
```

или команду для загрузки сразу всех образов

```
ls *.tar | xargs -n 1 docker load -i.
```

3. Задать специфические параметры в файле **docker-compose.yml**. Для всех параметров, которые необходимо задать, в файле **docker-compose.yml** даны описания (например, параметры SITE_NAME, ACLLISTHOST, ACLLISTPORT).

3.3. Настройка сервиса формирования предварительных чёрных списков

За формирование предварительных чёрных списков отвечает сервис **acl-creator-black**. Для правильной работы данного сервиса все необходимые параметры задают в файле конфигурации **acl.creator.config.yaml**. Ниже показана структура файла **acl.creator.config.yaml** и дано описание всех содержащихся в нём параметров.

```
clickhouse:
  host: clickhouse
```

```
  port: 9000

acllist:
  ipv4:
    host:
    port:
    secure:

  ipv6:
    host:
    port:
    secure:

timeouts:
  loop:
  error:
  maxError:

logLevel:

siteName:

defaultLifetime:

sql:
  - name:
    lifetime:
    params:
      - name:
        values:

  - name:
    lifetime:
    params:
      - name:
        values:
      - name:
        value:
```

Секция **clickhouse** содержит параметры подключения к СУБД ClickHouse, работающей в составе СПФС. Изменений не требует.

```
clickhouse:
 host: clickhouse
 port: 9000
```

Секция **acllist** содержит параметры подключения к расположенным в СЦОС кэширующим серверам acl-list, на которые сервис **acl-creator** будет передавать предварительные чёрные списки IPv4- и IPv6-адресов. В этой секции следует задать IP-адреса и порты кэширующих серверов. При необходимости можно включить шифрование соединения (параметр **secure = true**; по умолчанию **false**) (см. п. 5).

Пример:

```
acllist:
```

```
  ipv4:
    host: 192.168.202.1
    port: 30643
    secure: false

  ipv6:
    host: 192.168.202.1
    port: 30663
    secure: false
```

В секции **timeouts** задаются задержки в секундах между итерациями по формированию списков фильтрации:

- **loop** – задержка между успешными итерациями (периодичность обновления списков);
- **error** – задержка между итерациями, если на предыдущей возникла ошибка;
- **maxError** – максимальная задержка между итерациями в случае последовательного возникновения какой-либо ошибки.

```
timeouts:
  loop: 30s
  error: 60s
  maxError: 300s
```

Переменная **logLevel** определяет детализацию логирования событий в работе сервиса acl-creator. Ниже перечислены уровни детализации журнала событий от самого низкого к самому высокому:

- **panic** – экстренные предупреждения, требующие немедленных действий оператора;
- **fatal** – критическое состояние;
- **error** – ошибки в работе;
- **warning** – предупреждения, не требующие немедленных действий оператора;
- **info** – информационные сообщения;
- **debug** – информация для отладки кода;
- **trace** – информация для трассировки кода.

По умолчанию задан уровень детализации **info**. При выборе того или иного уровня детализации будут регистрироваться все события, соответствующие выбранному уровню, и события всех уровней ниже.

Переменная **siteName** задаёт имя площадки, на которой запущены сервисы acl-list.

Переменная **defaultLifetime** определяет время жизни записи, используемое по умолчанию, если иное значение не задано в секции **sql**.

Секция **sql** используется для определения параметров. Особое внимание при изменениях следует уделить значениям переменных. Если переменная может принимать массив значений, то задаётся параметр **values**. Если переменная может принимать одно значение, то задаётся параметр **value**. Пример:

```
sql:
  - name: black
    params:
      - name: threshold
        value: 60
```

3.4. Настройка сервиса dns-prober-creator

Сервис dns-prober-creator – это сервис подготовки пользовательских загружаемых списков IPv4 и IPv6 адресов, соотнесенных с доменными именами.

Сервис использует функциональность прокси-сервера устройства фильтрации трафика, настройка сервиса позволяет учитывать количество доступных фильтров и ограничения функциональности проксирования.

Для корректной работы сервиса в секции dns-prober-creator файла docker-compose.yml необходимо задать следующие переменные окружения:

- ACL_LIST_CACHE_PROBER_ADDRESS_V4 - инстанс кэширующего чёрного списка для IPv4 acl-list-cache-dns-prober-v4 из СЦОС.
- ACL_LIST_CACHE_PROBER_V4_USE_TLS - использовать ли TLS для подключения к инстансу IPv4 acl-list-cache-dns-prober-v4.
- ACL_LIST_CACHE_PROBER_ADDRESS_V6 - инстанс кэширующего чёрного списка для IPv6 acl-list-cache-dns-prober-v6 из СЦОС.
- ACL_LIST_CACHE_PROBER_V6_USE_TLS - использовать ли TLS для подключения к инстансу IPv6 acl-list-cache-dns-prober-v6.
- DNS_REQUEST_CRON - строка-расписание в формате cron для запуска задачи получения IP адресов для доменных имен из файла конфигурации.
- SITE_NAME - имя площадки.

3.5. Настройка ddos-detector

Сервис ddos-detector анализирует подозрительный трафик (в рамках одного узла ТСПУ) на соответствие преднастроенным правилам, выявляет аномалии трафика (большое количество сессий на определённых IP адресах).

Порядок работы сервиса:

1. Через 1 минуту после запуска сервис забирает список IP адресов, в который включены IP адреса и URL, подлежащие отслеживанию на обнаружение DDoS-атак. Далее сервис получает список с IP адресами с частотой, заданной через переменную METRICS_BIND_ADDRESS.
2. Парсер обрабатывает список.
3. Валидатор проверяет полученный список адресов.
4. С частотой раз в 1 минуту запускается обработчик, который пробегает по БД Clickhouse и собирает метрики по каждому из IP и URL адресов.

В Prometheus передаются следующие метрики для IP адресов:

- открытые сессии,
- полуоткрытые сессии,
- количество переданных пакетов,
- суммарное количество байт в переданных пакетах,
- направление (Ingress, Egress),
- задействованные протоколы.
- количество сессий (для URL адресов).

Правила работы валидатора адресов:

- При изменении списка адресов старые метрики перестают высылаться и удаляются из Prometheus.
- Если сеть задана неправильно (с 0 маской, или 192.168.10.11/24), то такие адреса не принимаются.
- Список IP и URL ограничивается количеством 1000. Все что выше, отбрасывается.
- Дубликаты не добавляются.
- Перекрывающие друг друга IP диапазоны никаким образом не обрабатываются.
- Информация обо всех ошибках передается в лог.

Перед стартом сервиса в файле docker-compose.yml необходимо указать следующие обязательные переменные окружения:

- DB_HOST - IP или hostname, указывающие на базу данных ClickHouse.
- DB_PORT - порт для подключения к базе данных ClickHouse.
- ADDRESS_WITH_IPS_LIST - HTTP адрес, по которому сервис будет получать список IP адресов для слежения. Формат запроса и ответа описан в файле конфигурации ddos.proto.

- REQUESTS_PERIOD – период опроса.
- NODE_NAME - имя ноды, которое будет передаваться при запросе списка IP адресов.
- Необязательные переменные:
- METRICS_BIND_ADDRESS - IP:PORT, на котором сервис будет выдавать метрики.

Ниже приведён пример задания переменных окружения в файле docker-compose.yml для сервиса ddos-detector.

```
image: harbor.rdp.ru/tt/ddos-detector:v0.5.1
volumes:
- /etc/localtime:/etc/localtime:ro
restart: on-failure
environment:
- DB_HOST=clickhouse
- LOG_LEVEL=debug
- ADDRESS_WITH_IPS_LIST=127.0.0.1:51050
- REQUESTS_PERIOD=1
- NODE_NAME=playground
ports:
- 2117:2112
```

3.6. Настройка сервиса drop-log-reader

Сервис drop-log-reader получает журнал гистограммных и debug логов от устройств фильтрации, выполняет их парсинг и передает результаты в СУБД ClickHouse.

Для его корректной работы необходимо настроить следующие переменные окружения:

- DEBUG_HAS_SESSION_ID - если опция включена (значение true), то random log парсер будет пытаться распарсить SessionID в пакетах.
- USE_LIST_OF_PROTOCOLS - если опция включена (значение true), то сервис будет получать список протоколов от сервиса list-of-protocols СЦОС.
- LIST_OF_PROTOCOLS_USE_TLS - использовать или нет зашифрованное соединение для подключения к сервису list-of-protocols. При использовании самоподписанного сертификата необходимо будет добавить монтирование сертификата в секции volumes.
- LIST_OF_PROTOCOLS_ADDRESS - IP-адрес и TCP-порт инстанса list-of-protocols, запущенного в СЦОС.
- LIST_OF_PROTOCOLS_READING_INTERVAL - интервал опроса инстанса list-of-protocols, запущенного в СЦОС.

Для расширения списка протоколов необходимо внести изменения в config.yml и раскомментировать монтирование файла ./config.yml:/srv/config/config.yml, или использовать сервис СЦОС (определяется значением переменной USE_LIST_OF_PROTOCOLS).

Для хранения дампа списка протоколов при использовании функционала получения списка протоколов из сервиса list-of-protocols СЦОС необходимо раскомментировать монтирование папки - ./dump:/srv/dump

3.7. Настройка сервиса clickstream-log-reader

Сервис clickstream-log-reader получает clickstream логи (информация о HTTP/HTTPS сессиях) от устройств фильтрации, выполняет их парсинг и передает результаты в СУБД ClickHouse.

Для корректной работы сервиса необходимо настроить следующие переменные окружения:

- CLICKSTREAM_HAS_SESSION_ID - если опция включена (значение =true), то clickstream парсер будет пытаться распарсить SessionID в пакетах.
- CLICKSTERAM_HAS_FILTER_ID - если опция включена (значение =true), то clickstream парсер будет пытаться распарсить FilterID в пакетах.
- CLICKSTREAM_SKIP_FRONT_BYTES - количество байт, которое надо пропускать во всех clickstream пакетах.

Пример: CLICKSTREAM_SKIP_FRONT_BYTES=8, нужно пропускать первые 8 байт во всех clickstream пакетах.

3.8. Настройка сервиса proto-log-reader

Сервис proto-log-reader получает proto логи (информацию о распознанных протоколах) от устройств фильтрации, выполняет их парсинг и отправляет результаты в СУБД ClickHouse.

Для корректной работы сервиса необходимо настроить следующие переменные окружения в секции файла docker-compose.yml, относящейся к данному сервису:

- USE_LIST_OF_PROTOCOLS - если опция включена (значение `true`), то сервис будет получать список протоколов от сервиса list-of-protocols СЦОС.
- LIST_OF_PROTOCOLS_USE_TLS - использовать или нет зашифрованное соединение для подключения к list-of-protocols. При использовании

самоподписанного сертификата необходимо будет добавить монтирование сертификата в секции volumes.

- LIST_OF_PROTOCOLS_ADDRESS - IP-адрес и TCP-порт инстанса list-of-protocols, запущенного в СЦОС.
- LIST_OF_PROTOCOLS_READING_INTERVAL - интервал опроса инстанса list-of-protocols, запущенного в СЦОС.

Для расширения списка протоколов необходимо внести изменения в config.yml и раскомментировать монтирование файла ./config.yml:/srv/config/config.yml, или использовать сервис СЦОС (определяется значением переменной USE_LIST_OF_PROTOCOLS)..

Для хранения дампа списка протоколов при использовании функционала получения списка протоколов из сервиса list-of-protocols СЦОС необходимо раскомментировать монтирование папки - ./dump:/srv/dump

3.9. Настройка сервиса packets-routes-log-reader

Сервис packets-routes-log-reader получает данные по точкам маршрутов пользовательского трафика от устройств фильтрации, выполняет их парсинг и отправляет результаты в СУБД ClickHouse.

В секции docker-compose.yml, относящейся к сервису packets-routes-log-reader, необходимо настроить следующие переменные окружения:

- PACKETS_ROUTES_HOST - IP-адрес и TCP-порт инстанса packets-routes-log-reader, запущенного в СЦОС.
- PACKETS_ROUTES_USE_TLS - использовать или нет зашифрованное соединение для подключения к packets-routes-log-reader.
- SERVICE_CREATOR_NAME - имя сервиса (признак источника), породившего точку маршрута пакета. Значение по умолчанию - routes-log-reader. При необходимости следует раскомментировать и переопределить переменную.

3.10. Настройка сервиса dns-prober-creator

Сервис dns-prober-creator - это сервис подготовки пользовательских загружаемых списков IPv4 и IPv6 адресов, соотнесенных с доменными именами.

Сервис использует функциональность прокси-сервера устройства фильтрации трафика, настройка сервиса позволяет учитывать количество доступных фильтров и ограничения функциональности проксирования.

Для корректной работы сервиса в секции dns-prober-creator файла docker-compose.yml необходимо задать следующие переменные окружения:

- ACL_LIST_CACHE_PROBER_ADDRESS_V4 - инстанс кэширующего чёрного списка для IPv4 acl-list-cache-dns-prober-v4 из СЦОС.
- ACL_LIST_CACHE_PROBER_V4_USE_TLS - использовать ли TLS для подключения к инстансу IPv4 acl-list-cache-dns-prober-v4.
- ACL_LIST_CACHE_PROBER_ADDRESS_V6 - инстанс кэширующего чёрного списка для IPv6 acl-list-cache-dns-prober-v6 из СЦОС.
- ACL_LIST_CACHE_PROBER_V6_USE_TLS - использовать ли TLS для подключения к инстансу IPv6 acl-list-cache-dns-prober-v6.
- DNS_REQUEST_CRON - строка-расписание в формате cron для запуска задачи получения IP адресов для доменных имен из файла конфигурации.
- SITE_NAME - имя площадки.

3.11. Настройка сервиса log-proxy-reader

Сервис получения обработанных статистических данных с применением аналитических запросов на структурированных больших данных, подготовленных сервисом drop-log-reader в СУБД ClickHouse. Предоставляет API для чтения журналов сессий и блокировок.

Для отдельных площадок может потребоваться корректировка переменных окружения, отвечающих за максимальное количество обрабатываемых строк при выполнении запросов к базе данных.

По умолчанию все переменные имеют значение 300 000.

Если сервис выдаёт ошибки превышения лимита памяти, то значения необходимо уменьшить.

Если сервис выдаёт значения за очень маленький промежуток времени, то значение необходимо увеличить (в 2,3,5,10 раз).

Для корректной работы сервиса необходимо настроить следующие переменные окружения в секции docker-compose.yml, относящейся к данному сервису:

- ACCOUNTING_MAX_ROWS - максимальное количество обрабатываемых строк для запроса accounting.
- CLICKSTREAM_MAX_ROWS - максимальное количество обрабатываемых строк для запроса clickstream.

- PROTO_MAX_ROWS - максимальное количество обрабатываемых строк для запроса protolog.
- SHORT_MAX_ROWS - максимальное количество обрабатываемых строк для запроса shortlist.
- DEBUG_MAX_ROWS - максимальное количество обрабатываемых строк для запроса debug.

3.12. Настройка сервиса qoe-tracker

Сервис **QoE-tracker** загружает список отслеживаемых ресурсов. Для ресурсов из данного списка после обработки трафика сервисом формируются рассчитанные метрики QoE. Сервис QoE-tracker предоставляет в систему управления рассчитанные метрики QoE в формате Prometheus для последующего анализа и отображения в графической форме.

В секции файла **docker-compose.yml**, относящейся к сервису **qoe-tracker**, необходимо настроить следующие переменные окружения:

- ADDRESS_WITH_IPS_LIST - адрес сервиса со списком наблюдаемых ресурсов (IPv4, IPv6, Hostname).
- REQUESTS_PERIOD - период обновления списка ресурсов.
- UPDATE_ACCOUNTING_STAT_INTERVAL - период накопления данных для расчета параметров qoe.
- NODE_NAME - имя площадки, которое будет передаваться в сервис по адресу ADDRESS_WITH_IPS_LIST.
- CERT_PATH - путь и имя файла сертификата (ca-cert.pem), который использует сервер, находящийся по адресу ADDRESS_WITH_IPS_LIST. Указание данной переменной включает TLS при подключении к ADDRESS_WITH_IPS_LIST. Для того чтобы файл был доступен внутри docker контейнера сервиса, необходимо добавить соответствующую запись в секцию volumes в файле docker-compose.yml.
  Если переменная окружения не указана, то будет использоваться незащищённое (Insecure) подключение к gRPC серверу ADDRESS_WITH_IPS_LIST.

Для корректной работы сервиса **QoE-tracker** должны быть установлены следующие переменные окружения сервисов **accounting-log-reader** и **clickstream-log-reader** (см. таблицу ниже).

Таблица 4 – Переменные окружения сервисов accounting-log-reader и clickstream-log-reader

Переменная	**Значение**	**Описание**

PUSH_TO_EVENTS_QUEUE	true	Включает запись логов в очередь сообщений
EVENTS_QUEUE_URI	tcp://events-collector-queue:50055	Адрес сервиса очереди сообщений

3.13. Настройка сервиса events-collector-queue

Сервис events-collector-queue передает обработанные пакеты от accounting-log-reader, clickstream-log-reader и proto-log-reader сервису ddos-detector. Предоставляет интерфейс для приёма сообщения журналов и интерфейс подписки на очереди сообщений.

Для корректной работы сервиса в секции файла docker-compose.yml, относящейся к сервису events-collector-queue, необходимо настроить следующие переменные окружения:

- ENABLE_WRITE_DB - включение/выключение записи данных журналов в базу данных Clickhouse.
- ENABLE_NETFLOW_WRITE_DB - включение/выключение записи в базу данных Clickhouse пакетов netflow.

3.14. Заключительные действия

Выполнить действия:

1. Разместить файлы **mem.xml** и **config.yml** в одном каталоге вместе с **docker-compose.yml.**

2. Запустить контейнеры командой **docker-compose up -d**

3. Проверить, что сервисы стартовали, можно, просмотрев статус сервисов и логи командами:

```
docker-compose ps
docker-compose logs -f
```

Убедиться по выводу в наличии статуса здоровья у сервисов (healthy).

3.15. Особенности обновления ПО

При обновлении ПО запускается процедура миграции БД. На высокозагруженных узлах это может занять несколько часов.

Чтобы не дожидаться окончания миграции, рекомендуется:

– остановить все сервисы,
– удалить базу,

– запустить новую версию ПО на пустой БД.

При обновлении ПО необходимо выполнить изменение передаваемого формата логов ClickStream на EcoFilter. Должно быть включено использование бинарного формата ClickStream.

При обновлении ПО необходимо обратить внимание, что добавился файл acl.creator.config.yml. Этот файл должен присутствовать на всех площадках. В файле необходимо прописать имя площадки в переменную siteName, которая по умолчанию содержит значение UNDEFINED.

Адреса и порты кэширующих серверов acl-list уже прописаны в конфигурации.

4. НАСТРОЙКА МОНИТОРИНГА СЕРВИСОВ СПФС

Предусмотрена возможность комплексного мониторинга работы как ПО в целом, так и его отдельных сервисов. Данная возможность реализуется с помощью связки приложений Grafana и Prometheus, которые способны в режиме реального времени предоставлять в графическом и текстовом виде подробную информацию обо всех аспектах работы ПО.

Ниже описана процедура настройки взаимодействия приложений Grafana и Prometheus с основными сервисами СФПС.

4.1. Настройка сервиса acl-creator-black

Сервис **acl-creator-black** предоставляет статистику создания предварительных чёрных списков.

Файл дашборда **acl-creator.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса **acl-creator-black** содержится в файле **docker-compose.yml,** секция **services**. Сам файл **docker-compose.yml** находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса acl-creator на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: acl-creator
ip_type: v4
rule_type: white
log_type: random
playground: test

где:

- **rule_type** – может принимать значения **white** или **black**;
- **ip_type** – может принимать значения **v4** или **v6**;
- **log_type** – берётся из названия контейнера. Например, **tls** для hub.rdp.ru/acl-creator-black-tls-v4:v2.3.3;
- **playground** – название площадки, указанное в **docker-compose.yml.**

Пример секции дашборда сервиса acl-creator в docker-compose.yml

```
    image: harbor.rdp.ru/tt/acl-creator:v3.10.1
    volumes:
    - /etc/localtime:/etc/localtime:ro
    - ./acl.creator.config.yml:/srv/config/config.yaml
    restart: on-failure
    ports:
    - 2115:2112
```

4.2. Настройка drop-log-reader

Данный сервис предоставляет статистику приёма логов.

Файл дашборда **drop-log-reader.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса **drop-log-reader** содержится в файле **docker-compose.yml,** секция **services.** Сам файл **docker-compose.yml** также находится в архиве с релизом.

Для построения графиков сервиса drop-log-reader на основе файла дашборда необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: drop-log-reader

playground: test

где:

- **playground** – название площадки, указанное в **docker-compose.yml**.

Для расширения списка протоколов необходимо внести изменения в config.yml и удалить знаки комментариев в ./config.yml:/srv/config/config.yml.

```
image: harbor.rdp.ru/tt/drop-log-reader:v2.79.0
    volumes:
    - /etc/localtime:/etc/localtime:ro
    # Для расширения списка протоколов необходимо внести изменения
    #  в congig.yml и раскомментировать монтирование этого файла
    # - ./config.yml:/srv/config/config.yml
    restart: on-failure
    ports:
    - 5555:555/udp
    - 2112:2112
    environment:
    - READER_MODE=TG
    - *clickhouse
    - BUFFER_TIME_LIMIT_SECONDS=2
    - MAX_INPUT_PACKETS_PER_SECOND=50000
    - UDP_READER_MAX_BATCH_SIZE=5000
    - PARSERSCOUNT=1
    - GOGC=300
    - GOMAXPROCS=2
    # Если включена опция, то random log парсер будет пытаться
```

```
распарсить SessionID в пакетах
    - DEBUG_HAS_SESSION_ID=false
```

4.3. Настройка accounting-log-reader

Сервис **accounting-log-reader** - это сервис приема журнала accounting логов (информация об установленных сессиях) от устройств фильтрации, их парсинга и записи результатов в СУБД ClickHouse.

Файл дашборда **accounting-log-reader.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса **accounting-log-reader** содержится в файле **docker-compose.yml,** секция **services**. Файл **docker-compose.yml** также находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса **accounting-log-reader** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: accounting-log-reader

playground: test

где:

- **playground** – название площадки, указанное в **docker-compose.yml**.

Пример:

```
image: harbor.rdp.ru/tt/drop-log-reader:v2.82.0
    volumes:
    - /etc/localtime:/etc/localtime:ro
    restart: on-failure
    ports:
    - 5551:555/udp
    - 2122:2112
    environment:
    - READER_MODE=ACCOUNTING
    - NO_VLAN=1
    - *clickhouse
    - BUFFER_TIME_LIMIT_SECONDS=2
    - WRITERS_COUNT=8
    - MAX_INPUT_PACKETS_PER_SECOND=300000
    - WORKER_PACKETS_PER_SECOND=40000
    - UDP_READER_MAX_BATCH_SIZE=10000
    - GOGC=300
    - GOMAXPROCS=8
    healthcheck:
      test: curl --fail http://localhost:2112/healthz
      interval: 30s
      timeout: 30s
      retries: 3
```

```
      depends_on:
        clickhouse:
          condition: service_healthy
        drop-log-reader:
          condition: service_healthy
```

4.4. Настройка сервиса clickhouse

Сервис **clickhouse** предоставляет статистику приёма логов.

Файл дашборда **clickhouse-self-monitoring.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса clickhouse содержится в файле **docker-compose.yml**, секция **services**. Файл **docker-compose.yml** находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса **clickhouse** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: clickhouse

type: drop-log-reader

playground: test

где:

- playground – название площадки, указанное в docker-compose.yml.

Пример секции дашборда сервиса в docker-compose.yml:

```
   image: harbor.rdp.ru/orig/clickhouse/clickhouse-server:22.3.11.12-
alpine
    volumes:
    - /etc/localtime:/etc/localtime:ro
    - ./data:/var/lib/clickhouse
    - ./logs:/var/log/clickhouse-server
    - ./mem.xml:/etc/clickhouse-server/users.d/z_mem.xml
    - ./timeout.xml:/etc/clickhouse-server/users.d/z_timeout.xml
    - ./prometheus.xml:/etc/clickhouse-
server/config.d/z_prometheus.xml
    - ./server.xml:/etc/clickhouse-server/config.d/z_server.xml
    - ./del.xml:/etc/clickhouse-server/config.d/z_del.xml
    - ./log_disable.xml:/etc/clickhouse-
server/config.d/z_log_disable.xml
    ports:
    - 2121:2112
    restart: on-failure
    ulimits:
      nofile:
        soft: 262144
        hard: 262144
    healthcheck:
```

```
    test: wget --spider -S localhost:8123/ping
    interval: 30s
    timeout: 30s
    retries: 3
```

4.1. Настройка сервиса log-proxy-reader

Сервис **log-proxy-reader** предоставляет статистику передачи логов в ЦСУО.

Файл дашборда **log-proxy-reader.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса **log-proxy-reader** содержится в файле **docker-compose.yml,** секция **services**. Файл **docker-compose.yml** также находится в архиве с релизом.

Для построения графиков сервиса **log-proxy-reader** на основе файла дашборда необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: log-proxy-reader

playground: test

где:

- **playground** – название площадки, указанное в **docker-compose.yml**.

Пример секции дашборда сервиса log-proxy-reader в docker-compose.yml

```
image: harbor.rdp.ru/tt/log-proxy-reader:v1.1.4
volumes:
- /etc/localtime:/etc/localtime:ro
restart: on-failure
ports:
- 28080:8080
- 2123:2112
environment:
- DB_HOST=clickhouse
- NETFLOW_SRV_ADDR=netflow-log-reader:50052
- NO_BALANCER_NETFLOW=true
```

4.2. Настройка сервиса dns-prober-creator

Сервис **dns-prober-creator** отвечает за подготовку пользовательских загружаемых списков ip адресов, соотнесенных с доменными именами, с применением функционала socks proxy, настроенного на устройстве фильтрации.

Файл дашборда **dns-prober-creator.json** для приложения Grafana находится в архиве с релизом. Секция дашборда сервиса dns-prober-creator

содержится в файле **docker-compose.yml,** секция **services**. Файл **docker-compose.yml** также находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса dns-prober-creator на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:

service: dns-prober-creator

playground: test

где:

- **playground** – название площадки, указанное в **docker-compose.yml**.

Пример секции дашборда сервиса в docker-compose.yml

```
    image: harbor.rdp.ru/tt/dns-prober-creator:v0.6.1
    volumes:
    - /etc/localtime:/etc/localtime:ro
    - ./dns.prober.creator.config.yml:/srv/config/config.yaml
    restart: on-failure
    environment:
    - LOG_LEVEL=debug
      # Инстанс кэширующего чёрного списка для IPv4 acl-list-cache-
dns-prober-v4 из СЦОС
    -
ACL_LIST_CACHE_PROBER_ADDRESS_V4=list.cache.dns.prober.ipv4:30644 #
Заменить на актуальное
    - ACL_LIST_CACHE_PROBER_V4_USE_TLS=true
      # Инстанс кэширующего чёрного списка для IPv6 acl-list-cache-
dns-prober-v6 из СЦОС
    -
ACL_LIST_CACHE_PROBER_ADDRESS_V6=list.cache.dns.prober.ipv6:30664 #
Заменить на актуальное
    - ACL_LIST_CACHE_PROBER_V6_USE_TLS=true
      # Параметры запуска задачи "выяснения IP по Домену из конфига"
по расписанию.
    - DNS_REQUEST_CRON=0 */6 * * *
    - CONFIG_FILEPATH=/srv/config/config.yaml
      # Название площадки
    - SITE_NAME=playground # Заменить на актуальное
    - DEFAULT_ITEM_LIFETIME=24h
    ports:
    - 2118:2112
```

5. НАСТРОЙКА ШИФРОВАНИЯ СОЕДИНЕНИЯ С СЦОС

1. По умолчанию все запросы и ответы между сервисами СПФС и СЦОС передаются по незашифрованным соединениям. Однако в системе предусмотрена возможность использования TLS-шифрования. Данная возможность включается следующим образом:

- В файле конфигурации сервиса acl-creator (acl-creator/config/config.yaml) необходимо в секции acllist для ipv4 и ipv6 присвоить параметру secure значение true.
- В файле values.yml для сервиса acl-list присвоить параметру tls значение true.
- В файле values.yml для сервиса acl-list-cache присвоить параметру tls значение true.
- Перезапустить вышеуказанные сервисы, чтобы новые настройки вступили в силу.

При необходимости пользователи сами загружают сертификаты TLS в контейнеры соответствующих сервисов.

2. Для поддержки TLS-шифрования при взаимодействии сервиса dns-prober-creator с list-for-dns-prober (при условии, что для сервиса list-for-dns-prober в СЦОС используется конфигурация «сервер»), в файле конфигурации dns.prober.creator.config.yml нужно указать хост и порт сервиса list-for-dns-prober и присвоить параметру useTLS значение true:

```
# type: server
  # Хост и порт сервиса list-for-dns-prober из СЦОС
  # address: list-for-dns-prober:51002
  # useTLS: true

protocols: []
```

3. При использовании новой функциональности отправки маршрутных пакетов от сервиса routes-log-reader (СФПС) напрямую в сервис packets-routes (СЦОС) также предусмотрена возможность включения TLS-шифрования. Для этого во фрагменте **routes-log-reader** файла конфигурации **docker-compose.yaml** параметру **packets_routes_use_tls** необходимо присвоить значение **true**:

Фрагмент конфигурации **routes-log-reader** в **docker-compose.yaml**:

```
  routes-log-reader:
    environment:
    # Отправлять информацию о маркированных пакетах напрямую в сервис
packets-routes СЦОС
    - PACKETS_ROUTES_DIRECT=true
```

```
    # Использовать или нет зашифрованное соединение для подключения к
packets-routes
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - PACKETS_ROUTES_USE_TLS=true
    - PACKETS_ROUTES_STREAM_NUM=2
    # IP-адрес и TCP-порт инстанса packets-routes запущенного в СЦОС
    - PACKETS_ROUTES_HOST=packets-routes.scos:50051
```

4. При использовании новой функциональности централизованного получения сервисами acl-creator-black, drop-log-reader, proto-log-reader (СПФС) списка с протоколами из сервиса list-of-protocols (СЦОС) также предусмотрена возможность включения TLS-шифрования. Для этого в дополнительной секции listOfProtocols фрагмента acl-creator-black в файле конфигурации **acl.creator.config.yml** параметру secure необходимо присвоить значение **true**:

Пример секции конфигурации listOfProtocols:

```
listOfProtocols:
  host: list-of-protocols.local
  port: 8080
  secure: true
  readingInterval: 30m
  dumpPath: ./dump
```

Также для фрагментов drop-log-reader, proto-log-reader в файле docker-compose.yaml параметру list_of_protocols_use_tls необходимо присвоить значение true:

Фрагменты конфигурации сервисов drop-log-reader и proto-log-reader:

```
 drop-log-reader:
    volumes:
    # в config.yml и раскомментировать монтирование этого файла
    # или использовать сервис СЦОС (определяется переменной
USE_LIST_OF_PROTOCOLS)
    # - ./config.yml:/srv/config/config.yml
    # Для хранения дампа списка протоколов при использовании
функционала получения списка протоколов
    # из сервиса list-of-protocols СЦОС необходимо раскомментировать
монтирование этой папки
    # - ./dump:/srv/dump
    environment:
    # Брать список протоколов из сервиса list-of-protocols СЦОС
    - USE_LIST_OF_PROTOCOLS=true
    # Использовать или нет зашифрованное соединение для подключения к
list-of-protocols
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - LIST_OF_PROTOCOLS_USE_TLS=true
    # DOMAIN и TCP-порт или IP-адрес и TCP-порт инстанса list-of-
protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_ADDRESS=list-of-protocols.scos:8080
    # Интервал опроса инстанса list-of-protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_READING_INTERVAL=30m
```

```
  proto-log-reader:
    volumes:
    # в config.yml и раскомментировать монтирование этого файла
    # или использовать сервис СЦОС (определяется переменной
USE_LIST_OF_PROTOCOLS)
    # - ./config.yml:/srv/config/config.yml
    # Для хранения дампа списка протоколов при использовании
функционала получения списка протоколов
    # из сервиса list-of-protocols СЦОС необходимо раскомментировать
монтирование этой папки
    # - ./dump:/srv/dump
    environment:
    # Брать список протоколов из сервиса list-of-protocols СЦОС
    - USE_LIST_OF_PROTOCOLS=true
    # Использовать или нет зашифрованное соединение для подключения к
list-of-protocols
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - LIST_OF_PROTOCOLS_USE_TLS=true
    # DOMAIN и TCP-порт или IP-адрес и TCP-порт инстанса list-of-
protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_ADDRESS=list-of-protocols.scos:8080
    # Интервал опроса инстанса list-of-protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_READING_INTERVAL=30m
    # Местонахождение файла дампа для временного хранения списка
протоколов в отсутствие связи с сервисом list-of-protocols СЦОС
    - LIST_OF_PROTOCOLS_DUMP_PATH
```

6. ОПИСАНИЕ API

В системе реализован gRPC API, который позволяет работать с записями ACL (добавление, обновление, удаление), выполнять поиск записей по заданным критериям, просматривать историю добавления записей и журналы отладки, решать другие задачи. Разделы данной главы содержат отдельные описания работы с API для каждой задачи. Следует учитывать, что синтаксис запросов и ответов зависит от используемого gRPC-клиента (см. справочные материалы к gRPC-клиенту), поэтому даны только общие описания запросов, ответов и их параметров в виде таблиц.

API для работы с журналами сессий и блокировок реализуется через файл **log_proxy.proto**.

Таблица 5 – Методы API для работы с журналами сессий и блокировок

Метод	**Запрос**	**Ответ**	**Действие**
ListAccounting	RequestParams	AccountingResult (потоковый)	Вывод информации об установленных сессиях
ListClickStream	RequestParams	ClickStreamResult (потоковый)	Вывод информации о сессиях HTTP/HTTPS
ListProtocol	RequestParams	ProtocolResult (потоковый)	Вывод информации о распознанных протоколах
ListShortList	RequestParams	ShortListResult (потоковый)	Вывод журнала блокировок по единому реестру РКН и по IP-адресу/URL
ListDebugLog	RequestParams	DebugLogResult (потоковый)	Вывод журнала блокировок по сигнатурам (с тегом random)
ListTop100DNS	ListTop100DNSParams	DNS	Вывод Топ-100 самых популярных имён за последний час
ListTop10DNSServersRedirect	ListTop10DNSServersRedirectRequest	ListTop10DNSServersRedirectResponse (потоковый)	Вывод Топ-10 DNS серверов на перенаправление за последний час
ListIncorrectLogs	ListIncorrectLogsParams	ListIncorrectLogsResult	Запрос для извлечения некорректных логов
ListTopHalfOpenedSessions	ListTopHalfOpenedSessionsRequest	ListTopHalfOpenedSessionsResponse (потоковый)	Запрос Топ N IP адресов, с максимальным количеством полуоткрытых сессий за последние K минут
ListHalfOpenedSessionsWLimit	ListHalfOpenedSessionsWLimitRequest	ListHalfOpenedSessionsWLimitResponse (потоковый)	Запрос списка IP-адресов, с количеством полуоткрытых сессий, превышающих заданный порог
ListSessionsByIP	ListSessionsByIPRequest	AccountingResult (потоковый)	Запрос информации обо всех сессиях по IP-адресу абонента
ListSessionsByIPExtended	ListSessionsByIPRequest	ListSessionsByIPExtendedResponse (потоковый)	Запрос информации обо всех сессиях по IP-адресу абонента с

Метод	Запрос	Ответ	Действие
			указанием причины логирования записи
ListBalancerNetflow	ListBalancerNetflowRequest	ListBalancerNetflowResponse	Запрос информации, полученной через Netflow от балансера
ListTop100NTP	ListTop100NTPParams	ListTop100NTPResult	Топ 100 записей NTP
GetNTPStat	NTPStatRequest	NTPStatResponse	Запрос счетчиков NTP-запросов
ListFullSessionInfo	ListFullSessionInfoRequest	ListFullSessionInfoResponse (потоковый)	Запрос всей информации по сессии с различными параметрами

Примечание: Параметры запросов и ответов, названия которых указаны в графах **Запрос** и **Ответ** таблицы выше, приведены в соответствующих таблицах ниже.

Структуры для запросов

Таблица 6 – **RequestParams.** Параметры запроса

Поле	Тип данных	Описание
start_from	google.protobuf.Timestamp	Время начала выборки
end_time	google.protobuf.Timestamp	Время окончания выборки
filter_id	string	Уникальный идентификатор фильтра. IP или Серийный номер (зависит от того распарсено ли было поле и вида лога, который парсился)
parser_name	string	Имя парсера Доступные имена: AccountingLogParser, ClickStreamParser, DNSLogParser, ProtoLogParser, ShortlistParser, TGParser
error_contains	string	Текст ошибки содержит указанную подстроку

Таблица 7 – **ListIncorrectLogsParams.** Параметры для выборки некорректных логов

Поле	Тип данных	Описание
start_time	google.protobuf.Timestamp	Время начала выборки
end_time	google.protobuf.Timestamp	Время окончания выборки
filter_id	string	Уникальный идентификатор фильтра IP или Серийный номер
parser_name	string	Имя парсера. Доступные имена: AccountingLogParser, ClickStreamParser,

Поле	Тип данных	Описание
		DNSLogParser, ProtoLogParser, ShortlistParser, TGParser
error_contains	string	Текст ошибки содержит указанную подстроку
payload_starts_with_hex	string	Тело пакета содержит указанную подстроку, строка должна быть в виде base32 последовательности байт без пробелов
payload_starts_with_hex	string	Тело пакета начинается с указанной подстроки, строка должна быть в виде base32 последовательности байт без пробелов

Таблица 8 – **ListHalfOpenedSessionsWLimitRequest**. Параметры запроса максимального порога количества полуоткрытых сессий

Поле	Тип данных	Описание
max_halfopened_sessions	uint32	Максимальный порог количества полуоткрытых сессий, после которого IP адреса будут попадать в выборку
start_from	google.protobuf.Timestamp	Запросить список с этого момента времени
direction	DirectionCode	Направление сессии

Таблица 9 – **ListTopHalfOpenedSessionsRequest.** Параметры запроса

Поле	Тип данных	Описание
top_number	uint32	Количество IP в ответе с максимальным количеством полуоткрытых сессий
start_from	google.protobuf.Timestamp	Запросить список с этого момента времени
direction	DirectionCode	Направление сессии: входящий - пакет направлен в сторону абонента (LAN) из сети Интернет (WAN); исходящий – пакет направлен в сторону сети Интернет (WAN) от абонента (LAN)

Таблица 10 – **ListSessionsByIPRequest.** Выборка сессий по IP адресу абонента (ListSessionsByIP, ListSessionsByIPExtended)

Поле	Тип данных	Описание
ip	string	IP адрес (IPv4 или IPv6)

time_range	TimeRange	Опциональный диапазон выборки сессий за определённый промежуток времени выборка производится по времени начала сессии

Таблица 11 – **ListBalancerNetflowRequest.** Параметры получения Netflow записей, полученных от балансера (ListBalancerNetflowRequest)

Поле	**Тип данных**	**Описание**
start_from	google.protobuf.Timestamp	Запросить логи с этого момента времени
follow	bool	Ждать появления новых записей и отправлять их клиенту

Таблица 12 – **NTPStatRequest.** Запрос счетчиков кол-ва NTP-запросов

Поле	**Тип данных**	**Описание**
start	google.protobuf.Timestamp	Начало интервала выборки
end	google.protobuf.Timestamp	Конец интервала выборки

Таблица 13 – **TimeRange.** Диапазон выборки

Поле	**Тип данных**	**Описание**
start_time	google.protobuf.Timestamp	Время начала выборки
end_time	google.protobuf.Timestamp	Время окончания выборки

ListFullSessionInfoRequest. Параметры запроса

Запрос может выполняться не чаще 1 раза в 10 минут. Обязательно должно быть указано src_net или dst_net или sni_pattern. Ограничение по времени или портам является опциональным.

Условия выборки работают по принципу И если задано условие src_net и dst_net, то будут выбраны только те сессии, которые одновременно удовлетворяют каждому из этих условий.

Таблица 14 – Параметры запроса ListFullSessionInfoRequest

Поле	**Тип данных**	**Описание**
dst_net	string	Условие выборки по подсети назначения. Подсеть должна иметь валидный формат с точки зрения функции https://pkg.go.dev/net#ParseCIDR
dst_port_range	PortRange	Опциональное условие выборки по диапазону портов назначения
src_net	string	Условие выборки по подсети источника. Подсеть должна иметь валидный формат с точки зрения функции https://pkg.go.dev/net#ParseCIDR
src_port_range	PortRange	Опциональное условие выборки по диапазону портов источника

sni_pattern	string	Условие выборки по sni. sni_pattern должен быть задан в формате, подходящем для функции https://clickhouse.com/docs/en/sql-reference/functions/string-search-functions#match
session_start	TimeRange	Опциональная выборка по времени начала сессии
session_finished	TimeRange	Опциональная выборка по времени окончания сессии
session_in_db	TimeRange	Опциональная выборка по времени когда информация о сессии была обработана и попала в БД
session_merged_in_db	TimeRange	Опциональная выборка по времени когда вся информация о сессии была обработана и подготовлена для вычитывания

PortRange. Диапазон портов

PortRange предполагает, что искомый порт находится в диапазоне [lower;upper]. Диапазон портов работает включительно, т.е. если надо запросить порт 443, то указывается 443 в lower и upper, а если надо сделать запрос по 22 и 443 портам, то придётся делать отдельные запросы на каждый номер порта.

Таблица 15 – **PortRange**. Диапазон портов

Таблица 16 – **Поле**	**Тип данных**	**Описание**
lower	uint32	Начало диапазона портов
upper	uint32	Конец диапазона портов (upper >= lower)

Структуры для Ответов

Таблица 17 – **AccountingResult.** Информация об установленных сессиях

Поле	**Тип данных**	**Описание**
local_port	uint32	Порт клиента
remote_ip	string	IP-адрес удалённого ресурса в формате IPv4 или IPv6
remote_port	uint32	Порт удалённого ресурса
protocol	L4ProtocolCode	Протокол транспортного уровня: TCP или UDP
direction	DirectionCode	Направление сессии: входящая – сессия инициирована из сети Интернет (WAN) в сторону абонента (LAN); исходящая – сессия инициирована абонентом (LAN) в сторону сети Интернет (WAN)

Поле	Тип данных	Описание
start_time	google.protobuf.Timestamp	Время начала сессии
end_time	google.protobuf.Timestamp	Время завершения сессии
out_bytes	uint32	Количество переданных байтов
in_bytes	uint32	Количество полученных байтов
out_packets	uint32	Количество переданных пакетов
in_packets	uint32	Количество полученных пакетов
filter_id	string	Уникальный идентификатор фильтра
tcp_ip_fingerprint	TCPIPFingerprint	Отпечаток TCP/IP сессии
pkt_process_timestamp	google.protobuf.Timestamp	Время, когда пакет был принят в accounting-log-reader
in_tos	uint32	ingress tos
out_tos	uint32	egress tos
delta_synack_ack	uint32	RTT Syn Syn-Ack
ingress_retransmit_count	uint32	Количество входящих ретрансмитов
egress_retransmit_count	uint32	Количество исходящих ретрансмитов

Таблица 18 – **ClickStreamResult.** Информация о сессиях HTTP/HTTPS

Поле	Тип данных	Описание
start_time	google.protobuf.Timestamp	Время запроса
content	string	Для HTTP: содержимое HTTP запроса
sni	string	Для HTTPS: доменное имя
filter_id	string	Уникальный идентификатор фильтра
ja3_hash	string	Значение JA3_hash
ja3_str	string	Значение JA3_str
ja3s_hash	string	Значение JA3S_hash
ja3s_str	string	Значение JA3S_str

Таблица 19 – **ProtocolResult.** Информация о распознанных протоколах

Поле	Тип данных	Метка	Описание
local_ip	string		IP-адрес клиента в формате IPv4 или IPv6
local_port	uint32		Порт клиента

Поле	Тип данных	Метка	Описание
remote_ip	string		IP-адрес удалённого ресурса в формате IPv4 или IPv6
remote_port	uint32		Порт удалённого ресурса
start_time	google.protobuf.Timestamp		Время начала сессии
end_time	google.protobuf.Timestamp		Время окончания сессии
dpi_protocol_code	uint32	repeated	Список распознанных протоколов (список ID приведён в документации на EcoFilter)
filter_id	string		Уникальный идентификатор фильтра
fingerprint	uint32		Отпечаток сессии
behaviour	BehaviourCode		Причина, по которой данный протокол попал в логи
dpilist	uint32		идентификатор списка фильтрации, согласно которому была осуществлена блокировка, если данные не передаются с экофильтра, то будет -1
custom_lists_match_type	uint32		Битовая маска, указывающая по какому из пользовательских списков в сигнатуре произошло срабатывание (2 младших бита): - 1й бит - Black List Local (Черный список клиентов), - 2й бит - Black List Remote (Черный список удаленных ресурсов), - отсутствие выставленных битов значает, что срабатывание по пользовательским спискам не произошло.

Таблица 20 – **ShortListResult.** Блокировка по Единому реестру запрещённых ресурсов Роскомнадзора

Поле	Тип данных	Описание
protocol	DPIListProtocolCode	Блокированный протокол: HTTP, HTTPS или IP
url	string	URL для протокола HTTP
sni	string	SNI для протокола HTTPS
remote_ip	string	IP-адрес удалённого ресурса в формате IPv4 или IPv6

Поле	Тип данных	Описание
remote_port	uint32	Порт удалённого ресурса
local_ip	string	IP-адрес клиента в формате IPv4 или IPv6
local_port	uint32	Порт клиента
dpilist	uint32	Идентификатор списка фильтрации, согласно которому была произведена блокировка
start_time	google.protobuf.Timestamp	Время запроса
filter_id	string	Уникальный идентификатор фильтра
behaviour	BehaviourCode	Причина, по которой данная запись попала в логи

Таблица 21 – **DebugLogResult.** Информация о блокировках по сигнатурам

Поле	Тип данных	Описание
protocol	L4ProtocolCode	Протокол транспортного уровня: TCP или UDP
local_ip	string	IP-адрес клиента в формате IPv4 или IPv6
local_port	uint32	Порт клиента
remote_ip	string	IP-адрес удалённого ресурса в формате IPv4 или IPv6
remote_port	uint32	Порт удалённого ресурса
dpi_protocol_code	uint32	Тип распознанного протокола в соответствии со встроенной сигнатурой (список ID приведён в документации на EcoFilter)
direction	DirectionCode	Направление пакета: входящий – пакет направлен из сети Интернет (WAN) в сторону абонента (LAN); исходящий – пакет направлен от абонента (LAN) в сторону сети Интернет (WAN)
content	bytes	Содержимое распознанного Ethernet-пакета (до 256 байт)
pkt_time	google.protobuf.Timestamp	Время записи информации о пакете в БД
filter_id	string	Уникальный идентификатор фильтра

Таблица 22 – **DNS.** Структура для возврата статистики запросов по доменному имени

Поле	Тип данных	Описание
domain	string	Доменное имя
counter	uint32	Количество запросов

Таблица 23 – **ListTop10DNSServersRedirectResponse.** Структура для возврата статистики запросов по DNS серверам на перенаправление

Поле	Тип данных	Описание
redirect_server_ip	string	ip адрес DNS сервера
counter	uint32	Количество запросов на перенаправление

Таблица 24 – **ListIncorrectLogsResult.** Структура с результатами выборки некорректных логов

Поле	Тип данных	Описание
pkt_time	google.protobuf.Timestamp	Время, когда пакет был обработан
payload_hex	string	Представляет hex строку (base32) содержимого пакета
payload_length	uint32	Длина тела пакета
filter_id	string	Уникальный идентификатор фильтра, IP или серийный номер (зависит от того распарсено ли было поле и вида лога, который парсился)
error	string	Текст ошибки
parser_name	string	Имя парсера, который пытался разобрать пакет
reader_mode	string	Название режима в котором работал drop-log-reader
possible_paket_type	string	Возможный тип пакета
possible_paket_subtype	string	Возможный подтип пакета
possible_paket_l3_type	string	Возможный тип L3 для пакета
possible_paket_l4_type	string	Возможный тип L4 для пакета
possible_paket_version	uint32	Возможная версия пакета
possible_session_id	uint64	Возможный идентификатор сессии пакета
last_offset	uint32	Смещение последних прочитанных байт
last_bytes_order	string	Порядок, в котором пытались читать последние байты

Таблица 25 – **ListTopHalfOpenedSessionsResponse.** Ответ на запрос ListTopHalfOpenedSessions

Поле	Тип данных	Описание

ip	string	IP адрес
half_opened_sessions	uint32	Количество открытых сессий для IP адреса

Таблица 26 – **ListHalfOpenedSessionsWLimitResponse.** Ответ на запрос ListHalfOpenedSessionsWLimit

Поле	**Тип данных**	**Описание**
ip	string	IP адрес
half_opened_sessions	uint32	Количество открытых сессий для IP адреса

Таблица 27 – **ListSessionsByIPExtendedResponse.** Ответ на запрос ListSessionsByIPExtended

Поле	**Тип данных**	**Описание**
acct_data	AccountingResult	Информация об аккаунтинге
short_dpilist	uint32	Номер dpilist из таблицы shortlist_log, если в shortlist_log нет связанной записи с аккаунтингом, то будет -1
proto_dpilist	uint32	Номер dpilist из таблицы proto_log, если в proto_log нет свазанной записи с аккаунтингом, то будет -1
protocol_code	uint32	Код распознанного протокола (список ID см. в документации на EcoFILTER)
short_behaviour	BehaviourCode	Причина, по которой данная запись попала в shortlist логи
proto_behaviour	BehaviourCode	Причина, по которой данная запись попала в proto логи

Таблица 28 – **ListBalancerNetflowResponse.** Ответ на запрос информации, полученной через Netflow от балансера

Поле	**Тип данных**	**Метка**	**Описание**
pkt_time	google.protobuf.Timestamp		Время, когда пакет был получен СПФС
records	NetflowRecord	repeated	Netflow записи

Таблица 29 – **NetflowRecord.** Типы данных полей

Поле	**Тип данных**	**Описание (при наличии)**
bytes	uint64	
packets	uint64	
sampling_interval	uint32	

src_ip	string	
dst_ip	string	
src_port	uint32	
dst_port	uint32	
icmp_type	uint32	
tcp_flags	uint32	
ip_protocol	uint32	
ip_tos	uint32	
min_ttl	uint32	
max_ttl	uint32	
forwarding_status	uint32	
is_tricolor	bool	Флаг, указывающий, что этот трафик принадлежит Российскому сегменту Интернета

Таблица 30 – **ListTop100NTPResult.** Структура с параметрами NTP-запроса

Поле	**Тип данных**	**Описание**
client_ip	string	IP-адрес клиента
server_ip	string	IP-адрес сервера
origin	google.protobuf.Timestamp	Время клиента, когда запрос отправляется серверу
receive	google.protobuf.Timestamp	Время сервера, когда запрос приходит от клиента
transmit	google.protobuf.Timestamp	Время сервера, когда запрос отправляется клиенту
receive_spfs	google.protobuf.Timestamp	Время СПФС, когда запись попадает в clickhouse

Таблица 31 – **NTPStatResponse.** Счетчики кол-ва NTP-запросов с разницей во времени между запросом и ответом сервера, попадающей в соответствующий интервал

Поле	**Тип данных**	**Описание**
delta_less_10ms	uint32	Интервал 10 мс
delta_less_1000ms	uint32	Интервал 1000 мс
delta_less_100s	uint32	Интервал 100 мс
delta_less_10000s	uint32	Интервал 10000 мс
delta_more_10000s	uint32	Интервал более 10000 мс
count_total	uint32	Общее количество

Общие структуры

Таблица 32 – **DPIListProtocolCode.** Коды заблокированных протоколов

Обозначение	**Код**	**Описание**
DPI_LIST_PROTOCOL_CODE_UNKNOWN	0	Не определён
DPI_LIST_PROTOCOL_CODE_HTTP	1	HTTP
DPI_LIST_PROTOCOL_CODE_HTTPS	2	HTTPS
DPI_LIST_PROTOCOL_CODE_IP	3	IP

Таблица 33 – **DirectionCode.** Коды направлений сессий и пакетов

Обозначение	**Код**	**Описание**
DIRECTION_CODE_UNKNOWN	0	Не определено
DIRECTION_CODE_EGRESS	1	Исходящее (LAN ® WAN)
DIRECTION_CODE_INGRESS	2	Входящее (WAN ® LAN)

Таблица 34 – **L4ProtocolCode.** Коды протоколов транспортного уровня

Обозначение	**Код**	**Описание**
L4_PROTOCOL_CODE_UNKNOWN	0	Не определено
L4_PROTOCOL_CODE_TCP	1	TCP
L4_PROTOCOL_CODE_UDP	2	UDP

Таблица 35 – **BehaviourCode.** Коды причины, по которой данный протокол попал в логи

Обозначение	**Код**
BEHAVIOUR_CODE_UNKNOWN	0
BEHAVIOUR_CODE_BLOCK	1
BEHAVIOUR_CODE_DROP	2
BEHAVIOUR_CODE_REDIRECT	3
BEHAVIOUR_CODE_IGNORE	4
BEHAVIOUR_CODE_COLOR	5
BEHAVIOUR_CODE_PASS	6

Таблица 36 – **ListFullSessionInfoResponse**. Ответ на запрос

Поле	**Тип данных**	**Метка**	**Описание**
session_processed_timestamp	google.protobuf.Timestamp		Время, когда информация о сессии была собрана в полном объёме и записана в БД
accounting	Accounting		Информация по сессии

Поле	Тип данных	Метка	Описание
tcp_ip_fingerprint	TCPIPFingerprint		Отпечаток TCP/IP сессии
clickstream	Clickstream	repeated	
shortlist	Shortlist		
protolog	Protocol	repeated	
debug	Debug	repeated	

Таблица 37 – **Common.** Общая часть, которая встречается в каждом пакете

Поле	Тип данных	Описание
filter_id	string	Уникальный идентификатор фильтра, который отправил аккаунтинг сообщение
pkt_process_timestamp	google.protobuf.Timestamp	Время, когда пакет был записан в БД на СПФС

Таблица 38 – **Accounting.** Информация по сессии

Поле	Тип данных	Описание
common_part	Common	Общая часть, которая встречается в каждом пакете
local_ip	string	IP адрес клиента в формате IPv4 или IPv6
local_port	uint32	Порт клиента
remote_ip	string	IP адрес удаленного ресурса в формате IPv4 или IPv6
remote_port	uint32	Порт удаленного ресурса
protocol	L4ProtocolCode	Протокол TCP или UDP
direction	DirectionCode	Направление сессии: входящая – сессия инициирована из Интернет (WAN) в сторону абонента (LAN); исходящая – сессия инициирована абонентом (LAN) в сторону сети Интернет (WAN)
start_time	google.protobuf.Timestamp	Время начала сессии
end_time	google.protobuf.Timestamp	Время окончания сессии
out_bytes	uint32	Количество переданных байтов
in_bytes	uint32	Количество полученных байтов
out_packets	uint32	Количество переданных пакетов
in_packets	uint32	Количество полученных пакетов
in_tos	uint32	ingress tos
out_tos	uint32	egress tos

Поле	Тип данных	Описание
delta_syn_synack	uint32	RTT Syn Syn-Ack
ingress_retransmit_count	uint32	RTT Syn-Ack Ack
egress_retransmit_count	uint32	Количество исходящих ретрансмитов

Таблица 39 – **Clickstream.** Информация по сессиям

common_part	Common	Общая часть, которая встречается в каждом пакете
clickstream	oneof	Выбор варианта: - HTTP http - ClientHello client_hello - ServerHello server_hello - HTTPSConnection https_connection - HTTPResponse http_response

Таблица 40 – **Protocol**. Информация по протоколу

Поле	Тип данных	Описание
common_part	Common	Общая часть, которая встречается в каждом пакете
protocol_code	uint32	Распознанный протокол (список ID см. в документации на EcoFILTER)
fingerprint	uint32	fingerprint сессии
behaviour	BehaviourCode	Причина, по которой данный протокол попал в логи
dpilist	int32	Идентификатор списка фильтрации, согласно которому была осуществлена блокировка, если данные не передаются с экофильтра, то будет -1

Таблица 41 – **Shortlist**. Блокировка по Единому реестру запрещенных ресурсов Роскомнадзора

Поле	Тип данных	Описание
common_part	Common	Общая часть, которая встречается в каждом пакете
protocol	DPIListProtocolCode	Протокол блокировки (HTTP, HTTPS или IP)
url	string	URL для HTTP протокола
sni	string	SNI для HTTPS протокола

dpilist	int32	Идентификатор списка фильтрации, согласно которому была осуществлена блокировка
pkt_time	google.protobuf.Timestamp	Время события на экофильтре
behaviour	BehaviourCode	Причина, по которой данная запись попала в логи

Таблица 42 – **Debug**. Блокировка по сигнатурам

Поле	**Тип данных**	**Описание**
common_part	Common	Общая часть, которая встречается в каждом пакете
packet_seq_number	uint32	Порядковый номер пакета отправленных с EcoFILTER
dpi_protocol_code	uint32	Тип распознанного протокола в соответствии с встроенной сигнатурой (список ID см. в документации на EcoFILTER)
direction	DirectionCode	Направление пакета: входящий - пакет направлен в сторону абонента (LAN) из сети Интернет (WAN); исходящий – пакет направлен в сторону сети Интернет (WAN) от абонента (LAN)
content	bytes	Содержимое распознанного Ethernet-пакета (до 256 байт)

Описание скалярных типов данных, передаваемых в запросах и ответах, см. в п.10.9.

7. ОБНОВЛЕНИЯ СПФС

7.1. Автоматическая диагностика состояния сервисов

Функциональность автоматической диагностики состояния сервисов (СПФС) для быстрого выявления возможных неполадок в работе контейнеров (health-check) включает следующие компоненты:

- переменные окружения сервиса для настройки параметров выполнения самодиагностики;
- встроенный функционал автоматической самодиагностики состояния сервиса (готов к работе/есть проблемы);
- встроенная в docker-compose опция health-check для периодического опроса состояния сервиса, со следующими параметрами:
  - test — параметр определяет выполняемую команду,
  - interval — параметр определяет интервал выполнения проверки работоспособности,
  - timeout — таймаут исполнения команды проверки работоспособности, по истечении которого Docker вернет код завершения с ошибкой,
  - retries — параметр определяет количество последовательных сбоев health-check, необходимых для объявления контейнера неисправным (unhealthy);
- встроенная в docker compose опция depends_on для запуска контейнеров, с возможностью задания зависимостей и условий, при выполнении которых будет запускаться сервис.

Ниже приведены фрагменты конфигурации в соответствии с иерархией структуры конфигурации сервисов в docker-compose.yaml и с комментарием по каждой переменной окружения:

```
docker-compose.yaml (СПФС)
  drop-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz
      interval: 30s
      timeout: 30s
      retries: 3
```

```
    depends_on:
      clickhouse:
        condition: service_healthy

  accounting-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
   - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz
      interval: 30s
      timeout: 30s
     retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  clickstream-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz
      interval: 30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  shortlist-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
```

```
      timeout: 30s
     retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  proto-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  dns-log-reader:
    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  netflow-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Временной интервал, по истечении которого выполняется повторная
проверка возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3
```

```
    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  routes-log-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD=1m
    # Таймаут, по истечении которого выполняется повторная проверка
возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT=30s

    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy
      drop-log-reader:
        condition: service_healthy

  hrandom-creator:
    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy

  log-proxy-reader:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD"=1m
    # Таймаут, по истечении которого выполняется повторная проверка
возможности
    # подключения к ДБ в случае обнаружения её недоступности
    - RECONNECT_DB_TIMEOUT"=30s
    # Количество сбойных точек приема запроса на стороне сервиса, при
котором
    # сервис уходит в состояние "не здоров"
    - SERVICE_FAIL_LEVEL=2
    # Соотношение сбойных запросов к общему числу запросов при
превышении которого
    # точка приема запроса на стороне сервера считается сбойной
    - ENDPOINT_FAIL_LEVEL=0.7

    healthcheck:
      test: [ "CMD", "grpc_health_probe", "-addr=localhost:8080" ]
      interval: 30s
```

```
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy

  ddos-detector:
    environment:
    # Период внутренней проверки состояния здоровья сервиса
    - HEALTH_CHECK_PERIOD"=1m
    # Таймаут, по истечении которого выполняется повторная проверка
возможности
    # подключения к сервису events-collector-queue случае обнаружения
недоступности
    - CONNECT_QUEUE_TIMEOUT"=1s
    # Максимально допустимая длительность, в течение которой список
отслеживаемых ресурсов
    # признаётся валидным в отсутствии возможности его обновления
    - MAX_AGE_OF_TRACKED_RESOURCES=30m

    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

    depends_on:
      clickhouse:
        condition: service_healthy

  dns-prober-creator:
    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

  acl-creator-black:
    healthcheck:
      test: curl --fail http://localhost:2112/healthz      interval:
30s
      timeout: 30s
      retries: 3

  clickhouse:
    healthcheck:
      test: wget --spider -S localhost:8123/ping
      interval: 30s
      timeout: 30s
      retries: 3
    depends_on:
      clickhouse:
        condition: service_healthy
```

7.2. Централизованное получение сервисами СПФС списка с протоколами от СЦОС

Функциональность централизованного получения сервисами **acl-creator-black, drop-log-reader, proto-log-reader** (СПФС) списка с протоколами из сервиса **list-of-protocols** (СЦОС) предоставляет следующие возможности:

- централизованное получение сервисами acl-creator-black, drop-log-reader, proto-log-reader в составе СПФС актуального списка протоколов от нового сервиса list-of-protocols в СЦОС,
- централизованное получение сервисами drop-log-reader и proto-log-reader в составе СПФС актуального списка игнорируемых протоколов от нового сервиса list-of-protocols в СЦОС.

Использование списка игнорируемых протоколов позволяет исключать из обработки данные журналов **debug_log и proto_log** в соответствии со списком игнорируемых протоколов.

Ниже приведена дополнительная настраиваемая секция конфигурации listOfProtocols для **acl-creator-black** в **acl.creator.config.yml**:

Дополнительная секция конфигурации listOfProtocols для acl-creator-black в acl.creator.config.yml

```
listOfProtocols:
  host: list-of-protocols.local
  port: 8080
  secure: false
  readingInterval: 30m
  dumpPath: ./dump
```

Где:

host — DOMAIN или IP-адрес инстанса list-of-protocols, запущенного в СЦОС,

port — TCP-порт инстанса list-of-protocols, запущенного в СЦОС,

secure — использовать или нет зашифрованное соединение для подключения к **list-of-protocols**,

readingInterval — интервал опроса инстанса **list-of-protocols,** запущенного в СЦОС,

dumpPath — путь к дампу списка протоколов при использовании функционала получения списка протоколов.

Ниже приведены фрагменты конфигурации в соответствии с иерархией структуры конфигурации сервисов в **docker-compose.yaml** и с комментарием по каждой переменной окружения.

Фрагменты конфигурации

```
docker-compose.yaml (СПФС)
  acl-creator-black:
    volumes:
    # Для расширения списка протоколов необходимо внести изменения
    # в protocols.yaml и раскомментировать монтирование этого файла
    # или использовать сервис СЦОС (определяется переменной
USE_LIST_OF_PROTOCOLS)
    # - ./config/protocols.yaml:/srv/config/protocols.yaml
    environment:
    - PROTOCOLS_FILE=./config/protocols.yaml
    # Брать список протоколов из сервиса list-of-protocols СЦОС
    - USE_LIST_OF_PROTOCOLS=true

  drop-log-reader:
    volumes:
    # в config.yml и раскомментировать монтирование этого файла
    # или использовать сервис СЦОС (определяется переменной
USE_LIST_OF_PROTOCOLS)
    # - ./config.yml:/srv/config/config.yml
    # Для хранения дампа списка протоколов при использовании
функционала получения списка протоколов
    # из сервиса list-of-protocols СЦОС необходимо раскомментировать
монтирование этой папки
    # - ./dump:/srv/dump
    environment:
    # Брать список протоколов из сервиса list-of-protocols СЦОС
    - USE_LIST_OF_PROTOCOLS=true
    # Использовать или нет зашифрованное соединение для подключения к
list-of-protocols
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - LIST_OF_PROTOCOLS_USE_TLS=false
    # DOMAIN и TCP-порт или IP-адрес и TCP-порт инстанса list-of-
protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_ADDRESS=list-of-protocols.scos:8080
    # Интервал опроса инстанса list-of-protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_READING_INTERVAL=30m

  proto-log-reader:
    volumes:
    # в config.yml и раскомментировать монтирование этого файла
    # или использовать сервис СЦОС (определяется переменной
USE_LIST_OF_PROTOCOLS)
    # - ./config.yml:/srv/config/config.yml
    # Для хранения дампа списка протоколов при использовании
функционала получения списка протоколов
    # из сервиса list-of-protocols СЦОС необходимо раскомментировать
монтирование этой папки
    # - ./dump:/srv/dump
    environment:
```

```
    # Брать список протоколов из сервиса list-of-protocols СЦОС
    - USE_LIST_OF_PROTOCOLS=true
    # Использовать или нет зашифрованное соединение для подключения к
list-of-protocols
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - LIST_OF_PROTOCOLS_USE_TLS=false
    # DOMAIN и TCP-порт или IP-адрес и TCP-порт инстанса list-of-
protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_ADDRESS=list-of-protocols.scos:8080
    # Интервал опроса инстанса list-of-protocols запущенного в СЦОС
    - LIST_OF_PROTOCOLS_READING_INTERVAL=30m
    # Местонахождение файла дампа для временного хранения списка
протоколов в отсутствие связи с сервисом list-of-protocols СЦОС
    - LIST_OF_PROTOCOLS_DUMP_PATH
```

7.3. Получение сервисом ddos-detector обработанных пакетов через сервис events-collector-queue

Новый сервис **events-collector-queue** предоставляет интерфейс для приёма сообщения журналов и интерфейс подписки на очереди сообщений.

Основные особенности нового сервиса:

- приём и подписка на сообщения организованы с использованием высокопроизводительной асинхронной библиотеки обмена сообщениями - ZeroMQ,
- пакеты сообщений журналов передаются в бинарном формате данных с использованием библиотеки flatBuffers, предоставляющей средства для сериализации данных.

Получение сервисом **ddos-detector** обработанных пакетов от сервисов **accounting-log-reader, clickstream-log-reader и proto-log-reader** в обход СУБД ClickHouse позволяет формировать статистику по аномалиям трафика в условиях, приближенных к условиям реального времени.

Ниже приведены фрагменты конфигурации в соответствии с иерархией структуры конфигурации сервисов в **docker-compose.yaml** и с комментарием по каждой переменной окружения.

Фрагменты конфигурации

```
docker-compose.yaml (СПФС)
  events-collector-queue:
    image: harbor.rdp.ru/tt/events-collector-queue:latest
    volumes:
    - /etc/localtime:/etc/localtime:ro
    ports:
    - 2131:2112
    restart: on-failure
    <<: *cpuset1
```

```
  accounting-log-reader:
    environment:
    # Если включена опция, то сервис отправляет пакеты в формате
flatbuffers в сервис
    # events-collector-queue (очередь событий) в реальном режиме
времени
    - PUSH_TO_EVENTS_QUEUE=true
    # URI сервиса events-collector-queue в формате tcp://IP:PORT или
tcp://DOMAIN:PORT
    - EVENTS_QUEUE_URI=tcp://events-collector-queue:50055

  clickstream-log-reader:
    environment:
    # Если включена опция, то сервис отправляет пакеты в формате
flatbuffers в сервис
    # events-collector-queue (очередь событий) в реальном режиме
времени
    - PUSH_TO_EVENTS_QUEUE=true
    # URI сервиса events-collector-queue в формате tcp://IP:PORT или
tcp://DOMAIN:PORT
    - EVENTS_QUEUE_URI=tcp://events-collector-queue:50055

  proto-log-reader:
    environment:
    # Если включена опция, то сервис отправляет пакеты в формате
flatbuffers в сервис
    # events-collector-queue (очередь событий) в реальном режиме
времени
    - PUSH_TO_EVENTS_QUEUE=true
    # URI сервиса events-collector-queue в формате tcp://IP:PORT или
tcp://DOMAIN:PORT
    - EVENTS_QUEUE_URI=tcp://events-collector-queue:50055

  ddos-detector:
    environment:
    # Если включена опция, то сервис ddos-detector будет подключаться
к сервису
    # events-collector-queue для получения данных в реальном режиме
времени
    - USE_EVENTS_QUEUE=true
    # URI сервиса events-collector-queue в формате tcp://IP:PORT или
tcp://DOMAIN:PORT
    - EVENTS_QUEUE_URI=tcp://events-collector-queue:50056
    # Период обновления статистики по accounting
    - UPDATE_ACCOUNTING_STAT_INTERVAL=1m
    # Период обновления статистики по proto_log
    - UPDATE_PROTO_STAT_INTERVAL=1m
```

7.4. Взаимодействие между netflow-log-reader и log-proxy-reader через сервис посредник events-collector-queue

Функциональность взаимодействия между сервисами **netflow-log-reader** (СПФС) и **log-proxy-reader** (СПФС) через сервис-посредник **events-collector-queue** без использования СУБД ClickHouse включает следующие опции:

- возможность отправки сообщений журнала netflow сервисом **netflow-log-reader** в сервис-посредник **events-collector-queue**;
- получение обработанных сообщений журнала netflow сервисом **log-proxy-reader** от сервиса-посредника **events-collector-queue**;
- возможность отключения данного режима и возврат к традиционному способу получения обработанных сообщений журнала netflow через СУБД ClickHouse.

Фрагменты конфигурации:

```
docker-compose.yaml (СПФС)
x-events-collector-uri:
  environment:
  - &events-collector-push-uri EVENTS_QUEUE_URI=tcp://events-
collector-queue:50055
  - &events-collector-pull-uri EVENTS_QUEUE_URI=tcp://events-
collector-queue:50056

services:
  netflow-log-reader:
    environment:
    # url адрес сервиса events-collector-queue для отправки пакетов
    - *events-collector-push-uri
    # Если включена опция, то сервис не отправляет пакеты в очередь
    - DISABLE_WRITE=false

  log-proxy-reader:
    environment:
    # url адрес сервиса events-collector-queue для получения пакетов
    - *events-collector-pull-uri
    # Если включена опция, то возвращается ошибка, поскольку сервис
больше не принимает напрямую обработанные пакеты от сервиса netflow-
log-reader
    - NO_BALANCER_NETFLOW=false
```

7.5. Отправка маршрутных пакетов от сервиса routes-log-reader напрямую в сервис packets-routes

Новая функциональность предусматривает возможность отправки маршрутных пакетов от сервиса **packets-routes-log-reader** (СФПС) напрямую в сервис **packets-routes** (СЦОС).

Вносимое изменение подразумевает возможность отказа от использования сервиса **packets-routes-creator**. Сервис **packets-routes-creator** объявляется устаревшим и может быть исключён из конфигурации сервисов в **docker-compose.yaml.**

Ниже приведён фрагмент конфигурации **packets-routes-log-reader** в соответствии с иерархией структуры конфигурации сервисов в **docker-compose.yaml** и с комментарием по каждой новой переменной окружения.

Фрагмент конфигурации packets-routes-log-reader:

```
docker-compose.yaml
  routes-log-reader:
    environment:
    # Отправлять информацию о маркированных пакетах напрямую в сервис
packets-routes СЦОС
    - PACKETS_ROUTES_DIRECT=true
    # Использовать или нет зашифрованное соединение для подключения к
packets-routes
    # При использовании самоподписанного сертификата необходимо будет
добавить монтирование сертификата в секции volumes
    - PACKETS_ROUTES_USE_TLS=false
    - PACKETS_ROUTES_STREAM_NUM=2
    # IP-адрес и TCP-порт инстанса packets-routes запущенного в СЦОС
    - PACKETS_ROUTES_HOST=packets-routes.scos:50051
```

7.6. Регулирование частоты выполнения dns запросов в сервисе dns-prober-creator

Новый функционал позволяет регулировать временной период между запросами к DNS серверу для исключения блокировки доступа при выполнении процедуры разрешения доменных имён.

Ниже приведён фрагмент конфигурации dns-prober-creator в соответствии с иерархией структуры конфигурации сервисов в docker-compose.yaml.

```
  dns-prober-creator:
    environment:
    # Интервал между запросами к DNS серверу
    - REQUEST_PERIOD=100ms
```

7.7. Возможность пропуска байтов в начале clickstream пакета в сервисе clickstream-log-reader

Новый функционал позволяет регулировать сдвиг по байтам при парсинге данных в сообщении журнала clickstream.

Ниже приведён фрагмент конфигурации clickstream-log-reader в соответствии с иерархией структуры конфигурации сервисов в docker-compose.yaml.

```
  clickstream-log-reader:
    environment:
    # Пропускать первые 8 байт во всех clickstream пакетах
    - CLICKSTREAM_SKIP_FRONT_BYTES=8
```

7.8. Реализован перенос логики drop-log-reader в events-collector-queue

Ранее сервисы drop-log-reader, proto-log-reader, acl-creator получали данные по протоколу udp от устройств фильтрации, выполняли парсинг данных и записывали результаты в СУБД ClickHouse. Сервисы запрашивали информацию о протоколах в сервисе list-of-protocols (СЦОС). При недоступности сервиса list-of-protocols сервисы получали списки протоколов из кэша.

На настоящий момент сервисы drop-log-reader, proto-log-reader, acl-creator получают данные по протоколу udp от устройств фильтрации, но парсинг, валидацию информации они не выполняют и не записывают данные в БД. Сервисы drop-log-reader и остальные формируют flatbuffers сообщения и пишут их в очередь в events-collector-queue. Логика работы с сервисом list-of-protocols также перенесена в events-collector-queue.

Сервис events-collector-queue выполняет парсинг, валидацию информации и запись данных в другую очередь. На сообщения этой очереди подписан новый сервис events-db-writer. Сервис events-db-writer обрабатывает входящие логи, раскладывает по очередям, а затем пересылает их в формате flatbuffers в БД Clickhouse.

7.9. Сервис QoE-tracker

7.9.1. Взаимодействие сервисов СПФС

Работа сервиса QoE-tracker и его взаимодействие с остальными сервисами СПФС в рамках его назначения изображены на структурной схеме ниже.

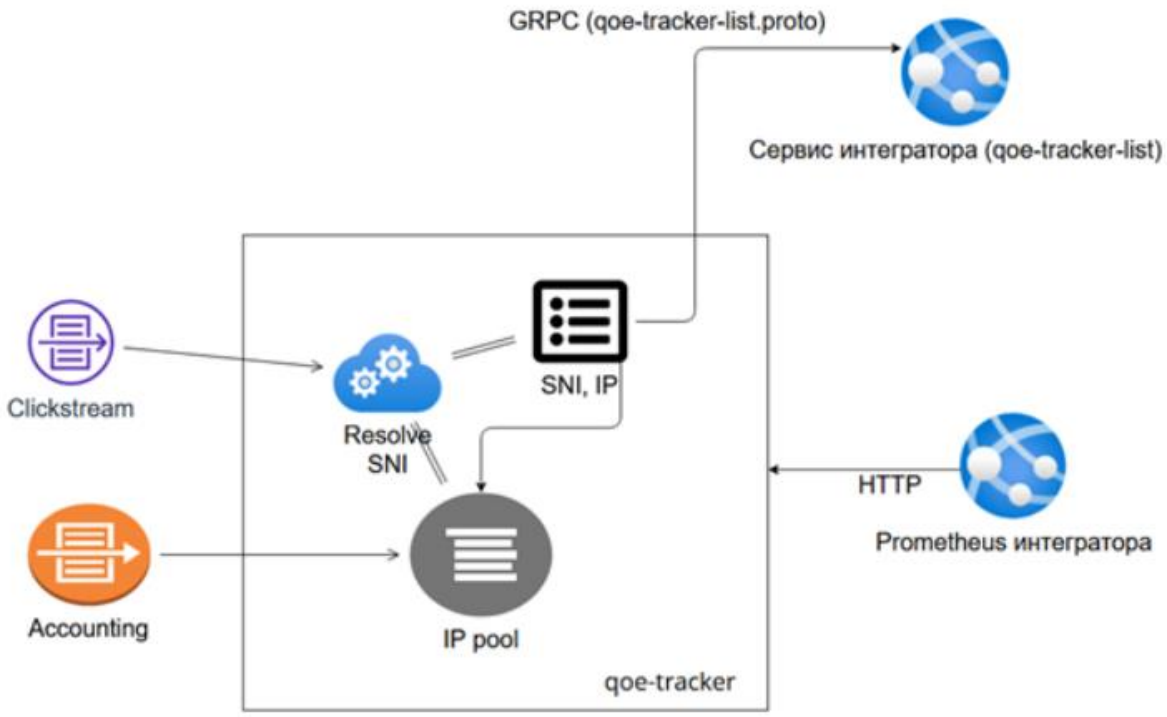

Рисунок 1 - Взаимодействие сервисов СПФС

1. Сервис QoE-tracker с заданной периодичностью, используя gRPC запросы (см. раздел "API получения списков"), получает список отслеживаемых ресурсов, включающих IP-адреса подсети в формате CIDR или доменные имена SNI. При этом принимается, что по умолчанию параметры подсети не более /24 (параметр определяется в переменных окружения, может быть изменен, см. раздел "Настройка сервиса QoE-tracker").

2. Сервис QoE-tracker, анализируя информацию от сервиса приема журнала clickstream логов о http/https сессиях, составляет таблицу соответствия адресов SNI и соответствующих IP адресов.

3. По информации от сервиса приема журнала accounting логов об установленных сессиях для наблюдаемой таблицы IP адресов и подсетей расчитывает значения метрик QoE, как среднее значение соответствующих параметров QoE за интервал опроса списков.

4. Рассчитанные значения параметров QoE сервис предоставляет для чтения через протокол HTTP в формате метрик Promethus. (подробнее см. раздел "Отправка метрик").

5. Правила вывода метрик:

– выводятся метрики, доступные на момент запроса - рассчитанные за предыдущий интервал накопления,

– по каждому ресурсу доступна статистика количества пакетов, использованных для расчета метрик.

Пример 1:

Список отслеживаемых ресурсов содержит адрес подсети 123.123.123.0/24, получены данные QoE для IP-адресов 123.123.123.1, 123.123.123.2, 123.123.123.122.

В этом случае метрика будет получена усреднением полученных данных для всех этих узлов: результат - 123.123.123.0/24.

Пример 2:

SNI соответствует множество IP адресов.

При расчете берётся среднее значение параметров QoE за минуту для всех IP, соответствующих этому SNI.

7.9.2. Настройка сервиса QoE-tracker

Для корректной работы сервиса QoE-tracker в файле **docker-compose.yml** необходимо указать следующие обязательные переменные окружения (см. таблицу ниже).

Таблица 43 – Переменные окружения

Переменная	**Значение по умолчанию**	**Описание**
ADDRESS_WITH_IPS_LIST		Адрес сервера со списком отслеживаемых ресурсов (IPv4, IPv6, Hostname)
CERT_PATH		Полный путь и имя файла корневого сертификата (например, /tmp/ca-cert.pem), который использует сервер, находящийся по адресу ADDRESS_WITH_IPS_LIST Указание данной переменной включает TLS при подключении к ADDRESS_WITH_IPS_LIST. Для того чтобы файл был доступен внутри docker контейнера сервиса, необходимо добавить соответствующую запись в секцию volumes в файле docker-compose.yml.
REQUESTS_PERIOD	1m	Период (в мин) для опроса ADDRESS_WITH_IPS_LIST (Ряд возможных значений: 30s, 1m, 1h и т.д.)

Переменная	Значение по умолчанию	Описание
NODE_NAME		Имя площадки, которое будет передаваться в сервис по адресу ADDRESS_WITH_IPS_LIST
UPDATE_ACCOUNTING_STAT_INTERVAL	1m	Период накопления данных для расчета параметров QoE для отслеживаемого списка ресурсов

Для корректной работы сервиса **QoE-tracker** также должны быть установлены следующие переменные окружения сервисов **accounting-log-reader** и **clickstream-log-reader** (см. таблицу ниже).

Таблица 44 – Переменные окружения сервисов **accounting-log-reader** и **clickstream-log-reader**

Переменная	Значение	Описание
PUSH_TO_EVENTS_QUEUE	true	Включает запись логов в очередь сообщений
EVENTS_QUEUE_URI	tcp://events-collector-queue:50055	Адрес сервиса очереди сообщений

7.9.3. API получения списков

API загрузки списков IP-адресов или SNI реализуется через файл **qoe-tracker-list.proto**. В таблицах ниже дано описание метода и формата запросов и ответов.

Метод GetIPsList

Таблица 2 - Метод API загрузки списков IP-адресов или SNI

Метод	Запрос	Ответ	Описание
GetIPsList	GetIPsListRequest	GetIPsListResponse (потоковый)	Получение списка IP-адресов для отслеживания

GetIPsListRequest

Таблица 3 - Параметры запроса GetIPsListRequest

Поле	Тип данных	Описание
node_name	string	Имя площадки

GetIPsListResponse

Таблица 4 - Параметры ответа GetIPsListResponse

Поле	Тип данных	Описание

addresses	Address	Список адресов, подлежащих слежению

Address – адрес для отслеживания

Таблица 5 - Параметры Address

Поле	**Тип данных**	**Описание**
Address oneof	address	IP адрес или url: - IP адрес с типом ip - или url с типом string

IPType - Тип IP адреса

Таблица 6 - Параметры IPType

Поле	**Тип данных**	**Описание**
IPType	enum	Тип IP адреса, возможные значения: IP_TYPE_UNSPECIFIED = 0 – неопределенный тип; IP_TYPE_IPV4 = 1 - IPv4; IP_TYPE_IPV6 = 2 – IPv6

IP - Представление IP-адреса для формирования записи

Таблица 7 – Параметры IP

Поле	**Тип данных**	**Описание**
type	IPType	Тип IP адреса в поле ip. Одно из значений в IPType
IP	string	IP адрес в формате: для IPv4: xxx.xxx.xxx.xxx для IPv6: xx:xx:xx:xx:xx:xx:xx:xx
mask	uint32	Маска для IPv4 или subnet prefix для IPv6

Типы данных

Таблица 8 – Типы данных

Тип данных	**Описание**
enum	Перечисление, тип данных, в котором каждое возможное значение определяется как символьная константа
uint32	Беззнаковое целое
string	Строка должна содержать текст в кодировке UTF-8 или 7-bit ASCII

7.9.4. Предоставление метрик

Сервис QoE-tracker предоставляет по запросу сервису интегратора следующие метрики (см. таблицу ниже).

Таблица 8 – Метрики QoE-tracker

Метрика	**Описание**
qoe_delta_syn_ack	RTT (время кругового пути между двумя сетевыми устройствами) между SYN/ACK и ACK в микросекундах
qoe_delta_syn_synack	RTT (время кругового пути между двумя сетевыми устройствами) между SYN и SYN/ACK в микросекундах
qoe_egress_retransmit_count	Количество исходящих ретрансмитов
qoe_ingress_retransmit_count	Количество входящих ретрансмитов
qoe_in_tos	Значение DSCP в поле ToS в заголовке входящих IP-пакетов, используется для маркировки входящего трафика
qoe_out_tos	Значение DSCP в поле ToS в заголовке исходящих IP-пакетов, используется для маркировки исходящего трафика
qoe_mq_received_events_total	Общее количество полученных пакетов в разрезе типа логов
qoe_hit_total	Общее количество обработанных пакетов в разрезе наблюдаемых ресурсов
qoe_mq_received_tracked_events_total	Общее количество обработанных пакетов в разрезе типа логов
qoe_mq_tracked_resources	Количество отслеживаемых ресурсов по типам: IPv4, IPv6, доменные имена SNI
qoe_mq_tracked_resources_age_seconds	Время с момента последнего обновления списка ресурсов

Полученные метрики обрабатываются программным обеспечением, установленным на сервере интегратора, которое предоставляет данные в графической форме.

На рисунке ниже приведен пример отображения метрик, передаваемых сервисом QoE-tracker, полученный с использованием программного обеспечения Grafana.

Рисунок 2 - Пример отображения QoE метрик ПО Grafana

8. УСТАНОВКА СЦОС

8.1. Требования к ПО

Для установки СЦОС потребуются следующие программные инструменты:

- Kubernetes версии не ниже 1.24;
- Helm версии не ниже 3;
- поддержка PV (Persistent Volume) и/или PVC (Persistent Volume Claim);
- поддержка режима ReadWrite для томов.

8.1.1. Описание параметров сервисов СЦОС и их конфигурируемых значений

В таблице ниже описаны параметры сервисов из состава СЦОС, настройка которых выполняется в файле конфигурации **values.yaml**.

Таблица 8 – Параметры сервисов СЦОС

Параметр	**Описание**	**Значение по умолчанию**
acl-creator-from-cache-black-v4.config.sourceAclList.host	Доменное имя хоста, на котором доступен кэширующий чёрный список IPv4	acllist.local
acl-creator-from-cache-black-v4.config.sourceAclList.port	TCP-порт, на котором доступен кэширующий чёрный список IPv4	8181
acl-creator-from-cache-black-v4.config.acllist.host	Доменное имя хоста, на котором доступен чёрный список IPv4	acllist.local
acl-creator-from-cache-black-v4.config.acllist.port	TCP-порт, на котором доступен чёрный список IPv4	8080
acl-creator-from-cache-black-v6.config.sourceAclList.host	Доменное имя хоста, на котором доступен кэширующий чёрный список IPv6	acllist.local
acl-creator-from-cache-black-v6.config.sourceAclList.port	TCP-порт, на котором доступен кэширующий чёрный список IPv6	8181
acl-creator-from-cache-black-v6.config.acllist.host	Доменное имя хоста, на котором доступен чёрный список IPv6	acllist.local
acl-creator-from-cache-black-v6.config.acllist.port	Доменное имя хоста, на котором доступен чёрный список IPv6	8080
acl-creator-from-cache-white-v4.config.sourceAclList.host	Доменное имя хоста, на котором доступен кэширующий белый список IPv4	acllist.local

Параметр	Описание	Значение по умолчанию
acl-creator-from-cache-white-v4.config.sourceAclList.port	TCP-порт, на котором доступен белый список IPv4	8181
acl-creator-from-cache-white-v4.config.acllist.host	Доменное имя белого списка IPv4	acllist.local
acl-creator-from-cache-white-v4.config.acllist.port	TCP-порт на котором доступен белый список IPv4	8080
acl-creator-from-cache-white-v6.config.sourceAclList.host	Доменное имя кэширующего белого списка IPv6	acllist.local
acl-creator-from-cache-white-v6.config.sourceAclList.port	TCP-порт, на котором доступен кэширующий белый список IPv6	8181
acl-creator-from-cache-white-v6.config.acllist.host	Доменное имя белого списка IPv6	acllist.local
acl-creator-from-cache-white-v6.config.acllist.port	TCP-порт, на котором доступен белый список IPv6	8080
rkn-creator.config.dump.username	Имя пользователя для скачивания списка РКН	undefined
rkn-creator.config.dump.password	Пароль пользователя для скачивания списка РКН	undefined
rkn-creator.config.acllist4.host	Доменное имя хоста, на котором доступен чёрный список РКН IPv4	acl-list-v4.local
rkn-creator.config.acllist4.port	TCP-порт, на котором доступен чёрный список РКН IPv4	30004
rkn-creator.config.acllist6.host	Доменное имя хоста, на котором доступен чёрный список РКН IPv6	acl-list-v6.local
rkn-creator.config.acllist6.port	TCP-порт, на котором доступен чёрный список РКН IPv6	30006
acl-differ-v4.web.ingress.hosts	Настройки Ingress-контроллера для доступа к спискам tls и hrandom с EcoFilter	[]
acl-differ-v4.web.ingress.annotations	Настройка аннотаций Ingress-контроллера для tls и hrandom с EcoFilter	{}
acl-differ-v4.whiteacllist.grpc	Доменное имя и порт для доступа к белому списку IPv4 в формате 'name:port'	list.main.white.ipv4:30641

Параметр	Описание	Значение по умолчанию
acl-differ-v4.blackacllist.grpc	Доменное имя и порт для доступа к чёрному списку IPv4 в формате 'name:port'	list.main.black.ipv4:30642
acl-differ-v6.web.ingress.hosts	Настройки Ingress-контроллера для доступа к спискам `tls` и `hrandom` с EcoFilter	[]
acl-differ-v6.web.ingress.annotations	Настройка аннотаций Ingress-контроллера для скачивания списков IPv6	{}
acl-differ-v6.whiteacllist.grpc	Доменное имя и порт для доступа к белому списку IPv6 в формате 'name:port'	list.main.white.ipv6:30661
acl-differ-v6.blackacllist.grpc	Доменное имя и порт для доступа к чёрному списку IPv6 в формате 'name:port'	list.main.black.ipv6:30662
acl-inverter-v4.web.ingress.hosts	Настройки Ingress-контроллера для доступа к спискам white с EcoFilter	[]
acl-inverter-v4.web.ingress.annotations	Настройка аннотаций для Ingress-контроллера для доступа к спискам white с EcoFilter	{}
acl-inverter-v4.whiteacllist.grpc	Доменное имя и порт любого списка IPv4 (по факту не используется) в формате name:port	list.main.white.ipv4:30641
acl-inverter-v4.blackacllist.grpc	Доменное имя и порт черного списка IPv4 (для инвертирования) в формате name:port	list.main.black.ipv4:30641
acl-inverter-v6.web.ingress.hosts	Настройки Ingress-контроллера для доступа к спискам white с EcoFilter	[]
acl-inverter-v6.web.ingress.annotations	Настройка аннотаций для Ingress-контроллера для доступа к спискам white с EcoFilter	{}
acl-inverter-v6.whiteacllist.grpc	Доменное имя и порт любого списка IPv6 в формате name:port (по факту не используется)	list.main.white.ipv6:30661
acl-inverter-v6.blackacllist.grpc	Доменное имя и порт белого списка IPv6 (для инвертирования) в формате name:port	list.main.white.ipv6:30661
acl-manager.config	Конфигурация менеджера по управлению записями во всех VRF	{предопределённый конфиг. см. ниже}
acl-list-black-v4.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]

Параметр	Описание	Значение по умолчанию
acl-list-black-v4.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-black-v4.mongodb.enabled	Настройки использования БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-black-v4.mongodb.auth.username	Имя пользователя для доступа к БД чёрного списка IPv4	пустое значение
acl-list-black-v4.mongodb.auth.password	Пароль для доступа к БД чёрного списка IPv4	пустое значение
acl-list-black-v4.mongodb.auth.rootPassword	Пароль для доступа к БД чёрного списка IPv4 с правами 'root'	пустое значение
acl-list-black-v4.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-black-v4.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-black-v4.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-black-v4.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-black-v4.externalDatabase.mongodbUsername	Имя пользователя для внешней БД чёрного списка IPv4	пустое значение
acl-list-black-v4.externalDatabase.mongodbPassword	Пароль для внешней БД чёрного списка IPv4	пустое значение
acl-list-black-v4.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение
acl-list-black-v4.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего просматривать текущее состояние чёрного списка	[]
acl-list-black-v4.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}
acl-list-black-v4.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/black/v4

Параметр	Описание	Значение по умолчанию
acl-list-black-v6.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-black-v6.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-black-v6.mongodb.enabled	Настройки использования БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-black-v6.mongodb.auth.username	Имя пользователя для доступа к БД чёрного списка IPv6	пустое значение
acl-list-black-v6.mongodb.auth.password	Пароль для доступа к БД чёрного списка IPv6	пустое значение
acl-list-black-v6.mongodb.auth.rootPassword	Пароль для доступа к БД чёрного списка IPv6 с правами 'root'	пустое значение
acl-list-black-v6.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-black-v6.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-black-v6.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-black-v6.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-black-v6.externalDatabase.mongodbUsername	Имя пользователя для внешней БД чёрного списка IPv6	пустое значение
acl-list-black-v6.externalDatabase.mongodbPassword	Пароль для БД внешней чёрного списка IPv6	пустое значение
acl-list-black-v6.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение
acl-list-black-v6.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего прасматривать текущее состояние чёрного списка	[]
acl-list-black-v6.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}

Параметр	Описание	Значение по умолчанию
acl-list-black-v6.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/black/v6
acl-list-white-v4.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-white-v4.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-white-v4.mongodb.enabled	Настройки использования БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-white-v4.mongodb.auth.username	Имя пользователя для доступа к БД белого списка IPv4	пустое значение
acl-list-white-v4.mongodb.auth.password	Пароль для доступа к БД белого списка IPv4	пустое значение
acl-list-white-v4.mongodb.auth.rootPassword	Пароль для доступа к БД белого списка IPv4 с правами 'root'	пустое значение
acl-list-white-v4.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-white-v4.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-white-v4.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-white-v6.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-white-v4.externalDatabase.mongodbUsername	Имя пользователя для внешней БД белого списка IPv4	пустое значение
acl-list-white-v4.externalDatabase.mongodbPassword	Пароль для БД внешней белого списка IPv4	пустое значение
acl-list-white-v4.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение
acl-list-white-v4.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего прасматривать текущее состояние белого списка	[]

Параметр	Описание	Значение по умолчанию
acl-list-white-v4.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}
acl-list-white-v4.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/white/v4
acl-list-white-v6.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-white-v6.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-white-v6.mongodb.enabled	Настройки использования БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-white-v6.mongodb.auth.username	Имя пользователя для БД белого списка IPv6	пустое значение
acl-list-white-v6.mongodb.auth.password	Пароль для БД белого списка IPv6	пустое значение
acl-list-white-v6.mongodb.auth.rootPassword	Пароль для root доступа к БД белого списка IPv6	пустое значение
acl-list-white-v6.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-white-v6.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-white-v6.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-white-v6.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-white-v6.externalDatabase.mongodbUsername	Имя пользователя для внешней БД белого списка IPv6	пустое значение
acl-list-white-v6.externalDatabase.mongodbPassword	Пароль для внешней БД белого списка IPv6	пустое значение
acl-list-white-v6.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение

Параметр	Описание	Значение по умолчанию
acl-list-white-v6.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего просматривать текущее состояние белого списка	[]
acl-list-white-v6.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}
acl-list-white-v6.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/white/v4
acl-list-ismon-v4.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-ismon-v4.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-ismon-v4.mongodb.enabled	Настройки использования БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-ismon-v4.mongodb.auth.username	Имя пользователя для БД синего списка IPv4	пустое значение
acl-list-ismon-v4.mongodb.auth.password	Пароль для БД синего списка IPv4	пустое значение
acl-list-ismon-v4.mongodb.auth.rootPassword	Пароль для root доступа к БД синего списка IPv4	пустое значение
acl-list-ismon-v4.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-ismon-v4.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-ismon-v4.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-ismon-v4.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-ismon-v4.externalDatabase.mongodbUsername	Имя пользователя для внешней БД синего списка IPv4	пустое значение
acl-list-ismon-v4.externalDatabase.mongodbPassword	Пароль для БД внешней синего списка IPv4	пустое значение

Параметр	Описание	Значение по умолчанию
acl-list-ismon-v4.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение
acl-list-ismon-v4.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего просматривать текущее состояние синего списка	[]
acl-list-ismon-v4.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}
acl-list-ismon-v4.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/ismon/v4
acl-list-ismon-v6.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-ismon-v6.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-ismon-v6.mongodb.enabled	Использование БД: true - запускать базу mongodb, false - использовать внешнюю базу	true
acl-list-ismon-v6.mongodb.auth.username	Имя пользователя для БД синего списка IPv6	пустое значение
acl-list-ismon-v6.mongodb.auth.password	Пароль для доступа к БД синего списка IPv6	пустое значение
acl-list-ismon-v6.mongodb.auth.rootPassword	Пароль для доступа к БД синего списка IPv6 с правами root	пустое значение
acl-list-ismon-v6.mongodb.persistence.enabled	Использовать PVC для постоянного хранилища	true
acl-list-ismon-v6.mongodb.persistence.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-ismon-v6.mongodb.persistence.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-ismon-v6.mongodb.persistence.size	Размер динамически выделяемого тома для постоянного хранилища	0Gi
acl-list-ismon-v6.externalDatabase.mongodbUsername	Имя пользователя для внешней БД синего списка IPv6	пустое значение

Параметр	Описание	Значение по умолчанию
acl-list-ismon-v6.externalDatabase.mongodbPassword	Пароль для внешней БД синего списка IPv6	пустое значение
acl-list-ismon-v6.externalDatabase.mongodbHost	Адрес внешней БД	пустое значение
acl-list-ismon-v6.ui.ingress.hosts	Настройки Ingress-контроллера для доступа к UI, позволяющего просматривать текущее состояние синего списка	[]
acl-list-ismon-v6.ui.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}
acl-list-ismon-v6.ui.grpc.proxy	Ссылка на web-grpc прокси сервер	http://cluster.domain.com:30273/acl/ismon/v4
acl-list-cache-black-v4.clickhouse.enabled	Использование БД: true - запускать базу clickhouse, false - использовать внешнюю базу	true
acl-list-cache-black-v4.clickhouse.storage.class	Имя класса для динамического выделения тома для постоянного хранилища	пустое значение
acl-list-cache-black-v4.clickhouse.storage.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-cache-black-v4.clickhouse.storage.size	Размер динамически выделяемого тома для постоянного хранилища IPv4	0Gi
acl-list-cache-black-v4.externalDatabase.clickhouseHost	Адрес БД при использовании внешней БД	пустое значение
acl-list-cache-black-v4.externalDatabase.clickhouseDatabase	Имя БД при использовании внешней БД	acllist
acl-list-cache-black-v4.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-cache-black-v4.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
acl-list-cache-black-v6.clickhouse.enabled	Настройки использования БД: true - запускать базу clickhouse, false - использовать внешнюю базу	true

Параметр	Описание	Значение по умолчанию
acl-list-cache-black-v6.clickhouse.storage.class	Имя класса для динамического выделения тома для постоянного хранилища IPv6	пустое значение
acl-list-cache-white-v6.clickhouse.storage.existingClaim	Имя существующего тома для постоянного хранилища	""
acl-list-cache-black-v6.clickhouse.storage.size	Размер динамически выделяемого тома для постоянного хранилища IPv6	0Gi
acl-list-cache-white-v6.externalDatabase.clickhouseHost	Адрес БД при использовании внешней БД	пустое значение
acl-list-cache-white-v6.externalDatabase.clickhouseDatabase	Имя БД при использовании внешней БД	acllist
acl-list-cache-white-v6.ingress.hosts	Настройки Ingress-контроллера для доступа по gRPC	[]
acl-list-cache-white-v6.ingress.annotations	Аннотации для Ingress-контроллера для доступа по gRPC	{}
envoy.templates.envoy.yaml	Конфигурация сервиса envoy	nil
acl-differ-web.ingress.hosts	Настройки Ingress-контроллера для загрузки списков с EcoFilter	[]
acl-differ-web.config	Конфигурация сервиса для настройки запросов и источников данных	nil
acl-differ-web.config.cacheLifetime	Периодичность формирования списка в формате "2h45m". Возможные единицы измерения "ns", "us", "ms", "s", "m", "h"	1m
rkn-resources-list.rkn.user	Логин для доступа к выгрузкам сайта vigruzki2.rkn.gov.ru	""
rkn-resources-list.rkn.pass	Пароль для доступа к выгрузкам сайта vigruzki2.rkn.gov.ru	""
rkn-resources-list.rkn.requestsPeriod	Частота опроса сайта с выгрузками (в минутах)	1
rkn-resources-list.ingress.hosts	Настройки Ingress-контроллера для скачивания списков	[]
rkn-resources-list.ingress.annotations	Настройка аннотаций для Ingress-контроллера	{}

8.2. Настройка конфигурации сервиса envoy

В приведённой ниже конфигурации **envoy.templates.envoy.yaml** нужно указать действительные хосты и порты, на которых развёрнуты сервисы acl-list, для того, чтобы UI мог с ними взаимодействовать.

Необходимо определить конфигурацию согласно шаблону ниже, раскоментировать и переопределить ip.address.of.cluster:

```
templates:
  envoy.yaml: |-
    admin:
      access_log_path: /dev/stdout
      address:
        socket_address:
          address: 0.0.0.0
          port_value: {{ .Values.ports.admin.containerPort }}

    static_resources:
      listeners:
      - name: listener_0
        address:
          socket_address:
            address: 0.0.0.0
            port_value: {{ .Values.ports.n0.containerPort }}
        filter_chains:
        - filters:
          - name: envoy.http_connection_manager
            config:
              codec_type: auto
              stat_prefix: ingress_http
              route_config:
                name: local_route
                virtual_hosts:
                - name: local_service
                  domains: ["*"]
                  routes:
                  - match: { prefix: "/acl/black/v4/" }
                    route:
                      cluster: acl_list_black_v4
                      max_grpc_timeout: 0s
                      prefix_rewrite: "/"
                  - match: { prefix: "/acl/white/v4/" }
                    route:
                      cluster: acl_list_white_v4
                      max_grpc_timeout: 0s
                      prefix_rewrite: "/"
                  - match: { prefix: "/acl/ismon/v4/" }
                    route:
                      cluster: acl_list_ismon_v4
                      max_grpc_timeout: 0s
                      prefix_rewrite: "/"
                  - match: { prefix: "/acl/black/v6/" }
                    route:
                      cluster: acl_list_black_v6
```

```
                        max_grpc_timeout: 0s
                        prefix_rewrite: "/"
                    - match: { prefix: "/acl/white/v6/" }
                      route:
                        cluster: acl_list_white_v6
                        max_grpc_timeout: 0s
                        prefix_rewrite: "/"
                    - match: { prefix: "/acl/ismon/v6/" }
                      route:
                        cluster: acl_list_ismon_v6
                        max_grpc_timeout: 0s
                        prefix_rewrite: "/"
                    cors:
                      allow_origin:
                      - "*"
                      allow_methods: GET, PUT, DELETE, POST, OPTIONS
                      allow_headers: keep-alive,user-agent,cache-
control,content-type,content-transfer-encoding,custom-header-1,x-
accept-content-transfer-encoding,x-accept-response-streaming,x-user-
agent,x-grpc-web,grpc-timeout
                      max_age: "1728000"
                      expose_headers: custom-header-1,grpc-
status,grpc-message
                http_filters:
                - name: envoy.grpc_web
                - name: envoy.cors
                - name: envoy.router
        clusters:
        - name: acl_list_black_v4
          connect_timeout: 0.25s
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30642 }}]
        - name: acl_list_white_v4
          connect_timeout: 0.25s
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30641 }}]
        - name: acl_list_ismon_v4
          connect_timeout: 0.25s
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30640 }}]
        - name: acl_list_black_v6
          connect_timeout: 0.25s
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30662 }}]
        - name: acl_list_white_v6
          connect_timeout: 0.25s
```

```
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30661 }}]
        - name: acl_list_ismon_v6
          connect_timeout: 0.25s
          type: logical_dns
          http2_protocol_options: {}
          lb_policy: round_robin
          hosts: [{ socket_address: { address: ip.address.of.cluster,
port_value: 30660 }}]
```

8.3. Настройка MongoDB

Со всеми настройками MongoDB можно ознакомиться в официальном репозитории по ссылке https://github.com/bitnami/charts/tree/master/bitnami/mongodb/

Минимально необходимые настройки указаны в списке параметров mongodbUsername, mongodbPassword, mongodbRootPassword, а также в параметрах хранилища persistence.

Параметры nameOverride, mongodbDatabase и metrics предопределены в данном чарте для всех экземпляров.

8.4. Настройка acl-list.ui

Для единого чёрного или белого списка IPv4/IPv6 должны быть определены секции ingress.

Ниже приведен шаблон секции. Необходимо заменить cluster.domain.ru на реальное доменное имя для доступа к UI.

Для корректной работы необходимо указать выделенный NodePort для envoy. В примере это - 30273, в реальности после развёртывания envoy необходимо обновить конфигурацию и указать действительное значение.

В путях необходимо указать black, white или ismon для чёрного, белого или синего списка соответственно.

В путях также должен быть определен тип IP: v4 или v6 для IPv4 и IPv6 соответственно.

Шаблон секции ingress

```
ui:
  ingress:
    hosts:
      - host: cluster.domain.ru
```

```
        paths: ['/acl/black/v4/(.*)']
    annotations:
      kubernetes.io/ingress.class: 'nginx'
      nginx.ingress.kubernetes.io/enable-rewrite-log: 'true'
      nginx.ingress.kubernetes.io/rewrite-target: '/$1'

  grpc:
    proxy: http://cluster.domain.ru:30273/acl/black/v4
```

8.5. Адреса черных списков для загрузки фильтром

Чёрные списки для скачивания фильтром доступны по следующим адресам:

- http://differ.v4.cluster.domain.com/tls - для IPv4 FakeTLS прокси серверов;
- http://differ.v4.cluster.domain.com/hrandom - для IPv4 MTProto прокси серверов;
- http://differ.v6.cluster.domain.com/tls - для IPv6 FakeTLS прокси серверов;
- http://differ.v6.cluster.domain.com/hrandom - для IPv6 MTProto прокси серверов;
- http://inverter.v4.cluster.domain.com/white - для белого IPv4 списка исключения из фильтрации;
- http://inverter.v6.cluster.domain.com/white - для белого IPv6 списка исключения из фильтрации;
- http://list.rkn.cluster.domain.com//api/v1/resources/urls - для списка из РКН.

8.6. Настройка acl-manager

В чарте предопределена следующая конфигурация acl-manager:

```
loop:
  # таймаут между итерациями обновления записей при успешном
выполнении обновления всех VRF
  timeout: 60s
  # таймаут между итерациями обновления записей при неудачном
выполнении обновления всех VRF
  err: 60s

# время в течение которого должны быть обновлены все VRF
# если acl-manager не успевает, то итерация будет прервана
iterationTimeout: 5m

# количество одновременно обновляемых VRF
parallel: 16

# источники записей
sources:
```

```
    # имя источника
  - name: acl-differ
    # тип источника: acl-differ, acl-list, web
    type: acl-differ
    # адрес источника записей IPv4
    v4: acl-differ-v4:30940
    # адрес источника записей IPv6
    v6: acl-differ-v6:30960

    # имя источника
  - name: acl-differ-inverter
    # тип источника: acl-differ, acl-list, web
    type: acl-differ
    # адрес источника записей IPv4
    v4: acl-differ-inverter-v4:30941
    # адрес источника записей IPv6
    v6: acl-differ-inverter-v6:30961

    # имя источника
  - name: acl-list-black
    # тип источника: acl-differ, acl-list, web
    type: acl-list
    # адрес источника записей IPv4
    v4: acl-list-black-v4:30642
    # адрес источника записей IPv6
    v6: acl-list-black-v6:30662

    # имя источника
  - name: rkn-port
    # тип источника: acl-differ, acl-list, web
    type: web
    # адрес источника записей IPv4
    v4: http://rkn-creator-api:8080/ports
    # адрес источника записей IPv6
    v6: http://rkn-creator-api:8080/ports

    # имя источника
  - name: acl-list-ismon
    # тип источника: acl-differ, acl-list, web
    type: acl-list
    # адрес источника записей IPv4
    v4: acl-list-ismon-v4:30640
    # адрес источника записей IPv6
    v6: acl-list-ismon-v6:30660

# список обслуживаемых VRF
vrfList:
    # настройки для VRF telegram_bb
    # название VRF
  - name: telegram_bb
    # настройки источника записей типа acl-differ для VRF
    aclDiffer:
      # имя источника записей
      source: acl-differ

      # список тегов для источника типа acl-differ
      tags:
        black:
```

```
        include:
          - hrandom
        exclude:
          - rkn

      white:
        include:
          - signature
          - pattern
          - hrandom_static
          - static
        exclude: []

  # настройки для VRF faketls_bb
- name: faketls_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - tls
        exclude:
          - rkn

      white:
        include:
          - tls_static
          - static
        exclude: []

  # настройки для VRF whatsapp_bb
- name: whatsapp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - whatsapp
        exclude:
          - rkn

      white:
        include:
          - whatsapp_static
          - static
        exclude: []

  # настройки для VRF viber_bb
- name: viber_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - viber
```

```
        exclude:
          - rkn

      white:
        include:
          - viber_static
          - static
        exclude: []

  # настройки для VRF whatsapp_voice_bb
- name: whatsapp_voice_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - whatsapp_voice
        exclude:
          - rkn

      white:
        include:
          - whatsapp_voice_static
          - static
        exclude: []

  # настройки для VRF viber_voice_bb
- name: viber_voice_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - viber_voice
        exclude:
          - rkn

      white:
        include:
          - viber_voice_static
          - static
        exclude: []

  # настройки для VRF utp_bb
- name: utp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - utp
        exclude:
          - rkn

      white:
```

```
        include:
          - utp_static
          - static
        exclude: []

  # настройки для VRF ipsec_bb
- name: ipsec_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - ipsec
        exclude:
          - rkn

      white:
        include:
          - ipsec_static
          - static
        exclude: []

  # настройки для VRF l2tp_bb
- name: l2tp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - l2tp
        exclude:
          - rkn

      white:
        include:
          - l2tp_static
          - static
        exclude: []

  # настройки для VRF pptp_bb
- name: pptp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - pptp
        exclude:
          - rkn

      white:
        include:
          - pptp_static
          - static
        exclude: []
```

```
  # настройки для VRF openvpn_udp_bb
- name: openvpn_udp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - openvpn_udp
        exclude:
          - rkn

      white:
        include:
          - openvpn_udp_static
          - static
        exclude: []

  # настройки для VRF openvpn_tcp_bb
- name: openvpn_tcp_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - openvpn_tcp
        exclude:
          - rkn

      white:
        include:
          - openvpn_tcp_static
          - static
        exclude: []

  # настройки для VRF vyprvpn_bb
- name: vyprvpn_bb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - vyprvpn
        exclude:
          - rkn

      white:
        include:
          - vyprvpn_static
          - static
        exclude: []

  # настройки для VRF operavpn_gb
- name: operavpn_gb
  aclDiffer:
```

```
    source: acl-differ

    tags:
      black:
        include:
          - operavpn
        exclude:
          - rkn

      white:
        include:
          - operavpn_static
          - static
        exclude: []

  # настройки для VRF youtube_gb
- name: youtube_gb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - youtube
        exclude:
          - rkn

      white:
        include:
          - youtube_static
          - static
        exclude: []

  # настройки для VRF facebook_gb
- name: facebook_gb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - facebook
        exclude:
          - rkn

      white:
        include:
          - facebook_static
          - static
        exclude: []

  # настройки для VRF instagram_gb
- name: instagram_gb
  aclDiffer:
    source: acl-differ

    tags:
      black:
```

```
        include:
          - instagram
        exclude:
          - rkn

      white:
        include:
          - instagram_static
          - static
        exclude: []

  # настройки для VRF quic_gb
- name: quic_gb
  aclDiffer:
    source: acl-differ

    tags:
      black:
        include:
          - quic
        exclude:
          - rkn

      white:
        include:
          - quic_static
          - static
        exclude: []

  # настройки для VRF rkn_ip_bb
- name: rkn_ip_bb
  # настройки для источника типа acl-list
  aclList:
    # имя источника записей
    source: acl-list-black

    # настройки тегов для запроса в acl-list
    tags:
      include:
        - rkn
      exclude: []

  # настройки для VRF rkn_port_gb
- name: rkn_port_gb
  # настройки для источника типа web
  web:
    # имя источника записей
    source: rkn-port

  # настройки для VRF ismon_cb
- name: ismon_cb
  aclList:
    source: acl-list-ismon

    tags:
      include:
        - ismon
      exclude: []
```

1. Для корректной работы сервиса в секции **acl-manager.config.sources** конфигурации **acl-manager**, приведенной выше, нужно заполнить настройки источников записей фильтрации.

Каждая запись секции **acl-manager.config.sources** содержит 4 поля:

- **name** - имя источника (произвольное имя, позволяющее отразить суть получаемых через него данных);
- **type** - тип источника записей;
- **v4** - IP:PORT источника записей для записей IPv4;
- **v6** - IP:PORT источника записей для записей IPv6;

Поддерживается три типа источника:

- **acl-differ** - сервис типа acl-differ;
- **acl-list** - сервис типа acl-list;
- **web** - особый случай представления портов, трафик которых требуется перенаправить на фильтры 2го эшелона.

Секцию необходимо настроить согласно настройкам развёрнутого инстанса.

Секция acl-manager.config.sources:

```
sources:
    # имя источника
  - name: acl-differ
    # тип источника: acl-differ, acl-list, web
    type: acl-differ
    # адрес источника записей IPv4
    v4: acl-differ-v4:30940
    # адрес источника записей IPv6
    v6: acl-differ-v6:30960

    # имя источника
  - name: acl-differ-inverter
    # тип источника: acl-differ, acl-list, web
    type: acl-differ
    # адрес источника записей IPv4
    v4: acl-differ-inverter-v4:30941
    # адрес источника записей IPv6
    v6: acl-differ-inverter-v6:30961

    # имя источника
  - name: acl-list-black
    # тип источника: acl-differ, acl-list, web
    type: acl-list
    # адрес источника записей IPv4
```

```
    v4: acl-list-black-v4:30642
    # адрес источника записей IPv6
    v6: acl-list-black-v6:30662

    # имя источника
  - name: rkn-port
    # тип источника: acl-differ, acl-list, web
    type: web
    # адрес источника записей IPv4
    v4: http://rkn-creator-api:8080/ports
    # адрес источника записей IPv6
    v6: http://rkn-creator-api:8080/ports

    # имя источника
  - name: acl-list-ismon
    # тип источника: acl-differ, acl-list, web
    type: acl-list
    # адрес источника записей IPv4
    v4: acl-list-ismon-v4:30640
    # адрес источника записей IPv6
    v6: acl-list-ismon-v6:30660
```

2. Для корректной работы сервиса в секции **acl-manager.config.vrfList** конфигурации **acl-manager**, приведенной выше, необходимо указать настройки каждой секции (**aclDiffer, aclList, web**).

Каждый элемент списка начинается с поля **name** (имя **VRF**).

В зависимости от типа источника (**acl-differ, acl-list, web**) указывается соответствующая секция (**aclDiffer, aclList, web**).

Секции **aclDiffer, aclList, web** содержат одно общее поле **source**, которое соответствует имени источника данных из секции **acl-manager.config.sources**.

Для секции **aclDiffer** необходимо указать набор тэгов **black и white**, как показано ниже.

Для секции **aclList** необходимо указать набор тэгов **include/exclude**, как показано ниже.

Секция **web** не содержит дополнительных настроек.

Секция acl-manager.config.vrfList:

```
vrfList:
    # настройки для VRF telegram_bb
    # название VRF
  - name: telegram_bb
    # настройки источника записей типа acl-differ для VRF
    aclDiffer:
      # имя источника записей
      source: acl-differ

      # список тегов для источника типа acl-differ
```

```
    tags:
      black:
        include:
          - hrandom
        exclude:
          - rkn

      white:
        include:
          - signature
          - pattern
          - hrandom_static
          - static
        exclude: []

  # настройки для VRF rkn_ip_bb
- name: rkn_ip_bb
  # настройки для источника типа acl-list
  aclList:
    # имя источника записей
    source: acl-list-black

    # настройки тегов для запроса в acl-list
    tags:
      include:
        - rkn
      exclude: []

  # настройки для VRF rkn_port_gb
- name: rkn_port_gb
  # настройки для источника типа web
  web:
    # имя источника записей
    source: rkn-port
```

8.7. Порядок установки СЦОС

Последовательность действий при установке:

1. Распаковать архив **core-services-release.tar** и перейти в каталог **core-services-release**.

2. Загрузить все образы контейнеров, выполнив на каждой ноде в кластере команду **docker load -i <имя образа>.tar** или команду для загрузки сразу всех образов **ls *.tar | xargs -n 1 docker load -i**.

3. Установить значения "по умолчанию" для конфигурируемых параметров файла **values.yaml** в соответствии с таблицей раздела "Значения параметров сервисов СЦОС".

4. В минимальной конфигурации (файл **values.yaml**) настроить следующие параметры:

– заменить доменные имена **cluster.domain.com** для доступа к сервисам снаружи;
– при использовании NodePort необходимо заменить внутренние доменные имена **cluster.local** для сервисов **acllist**;
– задать параметры хранилища для MongoDB и Clickhouse;
– указать имена пользователей и пароли для MongoDB и Clickhouse;
– указать логин и пароль учётной записи для скачивания Единого Реестра Запрещённых Ресурсов РКН.

5. Установить приложение командой:

```
helm install --namespace asbi --create-namespace echelon-core . -f
values.yaml
```

6. Проверить, что все сервисы были успешно запущены. Для этого отправить команды просмотра статуса сервисов и логов:

```
helm status -n asbi echelon-core
kubectl -n asbi get deployments
kubectl -n asbi get daemonsets
kubectl -n asbi get statefulsets
```

7. Проверить, что все компоненты системы успешно запущены (для проверки можно использовать документ «Контрольная карта»).

8.8. Настройка мониторинга сервисов СЦОС

Возможность комплексного мониторинга работы сервисов СЦОС реализована с помощью связки приложений Grafana и Prometheus, которые способны в режиме реального времени предоставлять в графическом и текстовом виде подробную информацию обо всех аспектах работы ПО.

Ниже описана процедура настройки взаимодействия приложений Grafana и Prometheus с основными сервисами СЦОС.

8.8.1. Настройка сервиса rkn-resources-list

Сервис rkn-resources-list формирует список URL по данным реестра РКН. Цель данного сервиса - заменить собой на фильтрах функционал скачивания списка запрещенных ресурсов и быть единым источником обработанного списка запрещённых ресурсов для устройств EcoFilter.

Сервис с заданной периодичностью обращается к Единому Реестру Запрещённых Ресурсов РКН (https://vigruzki2.rkn.gov.ru). Затем сервис формирует страничку, где на каждой отдельной строке расположен IP или URL запрещенного ресурса. Адрес, по которому можно получить список, имеет вид:

```
http://domain.name/api/v1/resources/urls,
```

где domain.name - конфигурируемое имя домена.

Перед стартом сервиса необходимо задать обязательные параметры, описание которых приведено в подразделе "Значения параметров сервисов СЦОС". Параметры задают в файле values.yaml.

Пример вывода сервиса:

:* (136.243.253.129)

:* (148.251.140.112)

http://203.13.32.134

https://203.13.32.134

http://2a06:9ac0:307:9318:b8bf:b6e:a76c:b139

https://2a06:9ac0:307:9318:b8bf:b6e:a76c:b139

8.8.2. Настройка сервиса rkn-creator

Сервис обновления информации из реестра РКН.

Файл дашборда rkn-creator.json для приложения Grafana находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса rkn-creator на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels

service: rkn-creator

rule_type: black

description: 'get ipv4 and ipv6 from dump.xml'

playground: test

где:

- rule_type – может принимать значения white или black;
- playground – название площадки или идентификатор ЦОД.

8.8.3. Настройка сервиса acl-list

Сервис для работы со списками.

Файл дашборда **acl-list.json** для приложения Grafana находится в архиве с релизом. Для того, чтобы приложение Grafana могло построить графики сервиса **rkn-creator** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:
service: acl-list
rule_type: black
ip_type: v4
db: mongodb
playground: test

где:

- **rule_type** – может принимать значения **white**, **black** или **ismon**;
- **ip_type** – может принимать значения **v4** или **v6**;
- **playground** – название площадки или идентификатор ЦОД.

8.8.4. Настройка сервиса acl-differ

Сервис для получения финального списка.

Файл дашборда **acl-differ.json** для приложения Grafana находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса **acl-differ** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:
service: acl-differ
ip_type: v4
playground: test

где:

- **ip_type** – может принимать значения **v4** или **v6**;
- **playground** – название площадки или идентификатор ЦОД.

8.8.5. Настройка сервиса acl-manager

Сервис управления записями в VRF.

Файл дашборда acl-manager.json для приложения Grafana находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса acl-manager на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:
service: acl-manager
ip_type: v4
type: block
playground: test

где:

- ip_type – может принимать значения v4 или v6;
- type – берётся из названия deployment. Например, block для scos-prod-acl-manager-rknip-v6.scos-prod;
- playground – название площадки или идентификатор ЦОД.

8.8.6. Настройка сервиса acl-creator-from-cache

Для того, чтобы приложение Grafana могло построить графики сервиса **acl-creator** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:
service: acl-creator
rule_type: black
ip_type: v4
tag: cache
playground: test

где:

- **rule_type** – может принимать значения white или black;
- **ip_type** – может принимать значения **v4** или **v6**;
- **playground** – название площадки или идентификатор ЦОД.

8.8.7. Настройка сервиса acl-list-cache

Сервис кэширующего списка.

Файл дашборда **acl-list-cache.json** для приложения Grafana находится в архиве с релизом.

Для того, чтобы приложение Grafana могло построить графики сервиса **acl-list-cache** на основе файла дашборда, необходимо, чтобы приложение Prometheus проставило следующие теги:

labels:
service: acl-list
rule_type: black
ip_type: v4
db: clickhouse
playground: test

где:

- **rule_type** – может принимать значения **white** или **black**;
- **ip_type** – может принимать значения **v4** или **v6**;
- **playground** – название площадки или идентификатор ЦОД.

9. ИЗМЕНЕНИЯ СЦОС

9.1. Настройка сервиса acl-differ-web для формирования необходимых итоговых списков фильтрации

Настройка формирования итоговых списков фильтрации, передаваемых на оборудование второго эшелона, выполняется в файле конфигурации **config.yaml** в секции **acl-differ-web**.

Файл конфигурации config.yaml:

```
cacheLifetime:

sources:
  black: []

  white: []

queries:
  name:
    source:
    params:
      black:
        include: []
        exclude: []
      white:
        include: []
        exclude: []
```

В файле конфигурации **config.yaml** (см. выше) переменная **cacheLifetime** задаёт периодичность формирования списков. Допустимые единицы измерения: ns, us, ms, s, m, h. Можно задавать в смешанном формате. Например, 1h30m.

В секции **sources** необходимо указать источники для загрузки чёрного (**black**) и белого (**white**) списков в формате **<IP-адрес или доменное имя сервера:номер порта>**. Параметры **black** и **white** могут принимать массив значений, т. е. можно указать несколько серверов.

В секции **queries** необходимо указать, какие списки следует загружать и какие записи должны быть включены в указанные списки. Описание параметров:

- **name** – вместо слова «**name»** необходимо указать имя загружаемого списка. Допустимые имена указаны в таблице ниже в столбцах "Список html" и "Отбеливающий список html". Для загрузки нескольких списков требуется создать в секции **queries** отдельную секцию **name** для каждого списка и задать в ней все необходимые параметры (см. пример конфигурации ниже).

- **source** – источник для загрузки. Допустимые значения: **black** и **white**, т. е. серверы, указанные в секции **sources**;
- **include** и **exclude** – эти параметры определяют, какие записи должны быть, соответственно, включены в загружаемый список и исключены из него. В этих параметрах необходимо указать имена тегов из таблицы ниже (столбцы "Чёрные/серые теги" и "Белые теги").

Возможные значения параметров конфигурации сервиса acl-differ-web приведены в таблице ниже.

Таблица 8 – Параметры конфигурации сервиса acl-differ-web

Чёрные / серые теги	**Белые теги**	**Сигнатура на фильтре**	**Список на ecohighway**	**Список html**	**Отбеливающий список html**	**Protocol ID**
	static	-	-	-	static_wh	
hrandom	signature, pattern, hrandom_static	telegram_ex	telegram_bb	telegram_bh	telegram_wh	0
tls	tls_static	block_cnt, tls_packet	faketls_bb	faketls_bh	faketls_wh	13
whatsapp	whatsapp_static	whatsapp_ex	whatsapp_bb	whatsapp_bh	whatsapp_wh	1
viber	viber_static	viber_ex	viber_bb	viber_bh	viber_wh	2
whatsapp_voice	whatsapp_voice_static	whatsapp_voice_ex	whatsapp_voice_bb	whatsapp_voice_bh	whatsapp_voice_wh	4
viber_voice	viber_voice_static	viber_voice_ex	viber_voice_bb	viber_voice_bh	viber_voice_wh	3
utp	utp_static	utp_ex	utp_bb	utp_bh	utp_wh	11
ipsec	ipsec_static	ipsec_ex	ipsec_bb	ipsec_bh	ipsec_wh	6
l2tp	l2tp_static	l2tp_ex	l2tp_bb	l2tp_bh	l2tp_wh	7
pptp	pptp_static	pptp_ex	pptp_bb	pptp_bh	pptp_wh	8
openvpn_udp	openvpn_udp_static	openvpn_udp_ex	openvpn_udp_bb	openvpn_udp_bh	openvpn_udp_wh	9
openvpn_tcp	openvpn_tcp_static	openvpn_tcp_ex	openvpn_tcp_bb	openvpn_tcp_bh	openvpn_tcp_wh	12
vyprvpn	vyprvpn_static	vyprvpn_ex	vyprvpn_bb	vyprvpn_bh	vyprvpn_wh	5

Чёрные / серые теги	**Белые теги**	**Сигнатура на фильтре**	**Список на ecohighway**	**Список html**	**Отбеливающий список html**	**Protocol ID**
operavpn	operavpn_static	operavpn_ex	operavpn_gb	-	operavpn_wh	14
youtube	youtube_static	youtube_ex	youtube_gb	-	youtube_wh	15
facebook	facebook_static	facebook_ex	facebook_gb	-	facebook_wh	16
instagram	instagram_static	instagram_ex	instagram_gb	-	instagram_wh	17
quic	quic_static	quic_list ( )	quic_gb	-	quic_wh	10
		-	rkn_port_gb	-	-	65000
rkn		-	rkn_ip_bb	-	-	65001
ismon		-	ismon_cb	-	-	65002

Обозначения

ex – сигнатура на фильтре

bb – чёрный/серый список для балансировщика

gb – серый список для балансировщика

cb – список клиентских подсетей для фильтрации на балансировщике

bh – чёрный список для фильтра

wh – отбеливающий список для фильтра

Формулы для наполнения списков

[Список на ecohighway] = [все чёрные теги] - [все белые теги] - static

[Список html] = [все чёрные теги] - [все белые теги] - static

[Отбеливающий html] = [все белые теги]

Списки на ecohighway загружаются в балансировщик, и уже на нём оператор выбирает, какие списки будут задействованы.

Списки html загружаются в фильтр. При загрузке указывается, для каких протоколов необходимо загрузить списки.

9.2. Сервис list-of-protocols

Секция конфигурации нового сервиса **list-of-protocols** в файле **values.yaml** helm чарта СЦОС:

```
values.yaml
  list-of-protocols:
    enabled: true

    postgresql:
      global:
        postgresql:
          auth:
            username: REPLACE_ME
            password: REPLACE_ME
            postgresPassword: REPLACE_ME
```

В сервисе list-of-protocols предусмотрены два API для управления списками протоколов и игнорируемых протоколов, описанные в файлах **protocols.proto** и **ignoreprotocols.proto** соответственно.

Для целей управления актуальным списком протоколов служат методы **Insert** и **Delete**, а для считывания списка протоколов сервисами СФПС — метод **List**:

```
// Список валидных протоколов
service ProtocolsService {
    rpc Insert(InsertRequest) returns (InsertResponse);
    rpc Delete(DeleteRequest) returns (DeleteResponse);
    rpc List(ListRequest) returns (stream ListResponse);
}

/*
 * Insert
 */
message InsertRequest {
  uint32 code = 1;
  string title = 2;
}

message InsertResponse {
}

/*
 * Delete
 */
message DeleteRequest {
  uint32 code = 1;
}

message DeleteResponse {
}

/*
 * List
 */
message ListRequest {
}

message ListResponse {
  uint32 code = 1;
```

```
  string title = 2;
  int64 created_at = 3;
}
```

Для целей управления актуальным списком игнорируемых протоколов служат методы Insert и Delete, а для считывания списка игнорируемых протоколов сервисами СФПС — метод List:

```
// Список игнорируемых протоколов
service IgnoreProtocolsService {
    rpc Insert(InsertRequest) returns (InsertResponse);
    rpc Delete(DeleteRequest) returns (DeleteResponse);
    rpc List(ListRequest) returns (stream ListResponse);
}

/*
 * Insert
 */
 message InsertRequest {
  uint32 code = 1;
}

message InsertResponse {
}

/*
 * Delete
 */
message DeleteRequest {
  uint32 code = 1;
}

message DeleteResponse {
}

/*
 * List
 */
message ListRequest {
}

message ListResponse {
  uint32 code = 1;
  int64 created_at = 2;
}
```

10. ОПИСАНИЕ API

В системе EcoDPIOS-DC реализован gRPC API, который позволяет работать с записями ACL (добавление, обновление, удаление), выполнять поиск записей по заданным критериям, просматривать историю добавления записей и журналы отладки, решать другие задачи. Разделы данной главы содержат отдельные описания работы с API для каждой задачи. Следует учитывать, что синтаксис запросов и ответов зависит от используемого gRPC-клиента (см. справочные материалы к gRPC-клиенту), поэтому даны только общие описания запросов, ответов и их параметров в виде таблиц. Для удобства навигации по разделам предусмотрены перекрёстные ссылки.

10.1. API для работы с записями ACL

API для добавления, обновления, удаления и поиска записей ACL реализуется через файл **protobuf/acl_list.proto**. В таблицах ниже дано описание доступных методов и формат запросов и ответов.

Таблица 45 – Методы работы с записями ACL

Метод	Запрос	Ответ	Действие
InsertOr Update	AclItem	AclListResult	Добавление или обновление одной записи
InsertOrUpd ateStream	AclItem (потоковый)	AclListResult	Добавление или обновление нескольких записей
Delete	AclItem	AclListResult	Удаление записи
DeleteByTag	DeleteByTagPar ams	AclListResult	Удаление записи по тегу
List	AclListSearchPar ams	AclItemWithTags (потоковый)	Поиск записей по заданным критериям
DeleteExpired Items	DeleteExpiredIte msParams	AclListResult	Удаление устаревших записей

AclItem

Таблица 46 – ACL-запись

Поле	**Тип данных**	**Метка**	**Описание**
IP	NetIP		IP-адрес
Port	uint32		TCP-порт, который необходимо фильтровать. '0' означает все порты

Поле	Тип данных	Метка	Описание
Tag	AclItemTag		Информация о теге
site	SiteInfo		Информация о площадке

DeleteByTagParams

Таблица 47 – Информация о теге

Поле	Тип данных	Метка	Описание
Tag	string		Информация о теге

AclItemTag

Таблица 48 – Тег, связанный с записью

Поле	Тип данных	Метка	Описание
Name	string		Имя тега
Description	string		Дополнительная информация
ExpiredAt	google.protobuf.Timestamp		Временная метка, после которой данный тег станет неактуальным

AclListSearchParams

Таблица 49 – Критерии поиска записей в ACL

Поле	Тип данных	Метка	Описание
Pagination	AclListSearchParams.PaginationParams		Параметры постраничного вывода
NoExpired	bool		Выводить только актуальные записи или все записи. True – выводить только актуальные записи, т. е. хотя бы у одного тега значение ExpiredAt должно быть больше, чем текущее время. False – выводить все записи
IP	string		IP-адрес или его часть. Можно искать по 8.8.8.8. Можно указать только начальную часть, например 8.8., тогда в этом случае будут найдены все адреса, начинающиеся с 8.8. Если указать для поиска 8.8, то будет подходить как 8.8.8.8, так и 8.8.80.8. Можно указать пустую строку, тогда IP-адрес не указывается
Port	uint32		TCP-порт. 0 означает все порты. При значении, отличном от 0, поиск будет выполнен по конкретному значению
IncludeTags	string	repeated	Запись должна содержать указанные теги. Если NoExpired = true, то хотя бы один из указанных тегов должен быть актуален

Поле	Тип данных	Метка	Описание
ExcludeTags	string	repeated	Запись не должна содержать указанные теги независимо от значения NoExpired

AclListSearchParams.PaginationParams

Таблица 50 – Параметры постраничного вывода. Основное назначение – для UI

Поле	Тип данных	Метка	Описание
PageNo	uint32		Номер страницы
PerPage	uint32		Количество элементов на одной странице

DeleteExpiredItemsParams

Таблица 51 – Параметры удаления устаревших записей

Поле	Тип данных	Метка	Описание
ItemAgeOverSpecified Hours	uint32		Записи должны быть старше указанного количества часов

Результаты выполнения запроса

AclListResult

Таблица 52 – Результат запроса на добавление, обновление или удаление записей

Поле	Тип данных	Метка	Описание
code	ResultCodes	enum	Код результата
message	string		Сообщение об ошибке

AclItemWithTags

Таблица 53 – ACL-запись со всеми тегами

Поле	Тип данных	Метка	Описание
IP	NetIP		IP-адрес
Port	uint32		TCP-порт, который необходимо фильтровать. '0' означает все порты
Tags	AclItemTag	repeated	Все теги, связанные с указанным IP-адресом и портом

AclListResult.ResultCodes

Таблица 54 – Коды результатов запроса на добавление, обновление или удаление записей

Результат	Код	Пояснение
OK	0	Успешно
FAIL	1	Ошибка

NetIP

Таблица 55 – Представление IP-адреса для формирования записи

Поле	Тип данных	Метка	Описание
IP	string		IP-адрес в формате IPv4 или IPv6
Mask	uint32		Маска, соответствующая адресу

AclItemTag

Таблица 56 – Тэг связанный с ACL-записью

Поле	Тип данных	Метка	Описание
Name	string		Имя тега
Description	string		Дополнительная информация
ExpiredAt	google.protobuf.Timestamp		Временная метка после которой данный тег потеряет актуальность

SiteInfo

Таблица 57 – Информация о площадке, с которой поступила запись

Поле	Тип данных	Метка	Описание
Name	string		Имя площадки

10.2. API для просмотра истории добавления записей в ACL

API для просмотра истории добавления записей в ACL реализуется через файл **history/history.proto**. В таблицах ниже дано описание доступных методов и формат запросов и ответов.

Таблица 58 – Методы API для просмотра истории добавления записей в ACL

Метод	Запрос	Ответ	Действие
IPHistory	IPHistoryParams	IPHistoryRecord (потоковый)	Вывод истории добавления записей по заданным параметрам
SiteTagStats	SiteTagStatsParams	SiteTagStatRecord (потоковый)	Вывод статистики добавления записей в разрезе тегов и площадок

IPHistoryParams

Таблица 59 – Параметры запроса истории добавления записей

Поле	Тип данных	Метка	Описание
ip	string		IP-адрес или его часть. Поиск выполняется по подстроке от начала к концу. Если указать 1.2.3. (с точкой в конце), то будут найдены все адреса, начинающиеся с 1.2.3. (например, 1.2.3.4, 1.2.3.40 и т. п.). Если указать 1.2.3 (без точки в конце), то, например, будет найден как адрес 1.2.3.4, так и адрес 1.2.30.4

Поле	Тип данных	Метка	Описание
timerange	TimeRange		Временной диапазон. Если не указан, то будут выведены результаты за всё время с момента начала работы системы

IPHistoryRecord

Таблица 60 – Содержимое ответа на запрос истории добавления записей.

Поле	Тип данных	Метка	Описание
ip_with_mask_and_port	string		IP-адрес в формате [ip/mask]:port
tags_history	TagHistoryRecord	repeated	Записи о тегах, связанных с данным IP-адресом

SiteTagStatRecord

Таблица 61 – Статистика добавления записей в разрезе тегов и площадок

Поле	Тип данных	Метка	Описание
insert_time	google.protobuf.Timestamp		Время добавления записи
site_name	string		Название площадки
tag_name	string		Имя тега
count	uint32		Количество добавленных записей

SiteTagStatsParams

Таблица 62 – Параметры выборки статистики добавления записей в разрезе тегов и площадок

Поле	Тип данных	Метка	Описание
timerange	TimeRange		Временной диапазон. Если не указан, то будут выведены результаты за всё время с момента начала работы системы

TagHistoryRecord

Таблица 63 – Информация о теге

Поле	Тип данных	Метка	Описание
site_name	string		Название площадки
tag_name	string		Имя тега у записи
expired_tag_at	google.protobuf.Timestamp		Время, до которого запись актуальна
insert_time	google.protobuf.Timestamp		Время добавления записи

TimeRange

Таблица 64 – Диапазон времени

Поле	Тип данных	Метка	Описание
from	google.protobuf.Timestamp		Начало диапазона
to	google.protobuf.Timestamp		Конец диапазона

10.3. API получения итоговых списков фильтрации

API для скачивания подготовленного списка устройствами фильтрации предоставляет сервис acl-differ-web. API получения итоговых списков фильтрации сервиса acl-differ-web представлено одним роутом, предназначенным для для использования оборудованием:

```
GET /api/v1/list
```

В параметрах GET запроса указывают набор списков, которые необходимо получить. Например:

```
/api/v1/list?queries[]=zoom_video_wh&queries[]=zoom_wh
```

Имена списков соответствуют именам, описанным в конфигурационном файле сервиса acl-differ-web.

В чарте инстанса СЦОС может быть указан префикс к API сервиса.

Пример полного адреса для роута:

```
echelon.ttraf.ru/dpilist/api/v1/list
```

В этом случае запрос будет иметь вид:

```
GET echelon.ttraf.ru/dpilist/api/v1/list?queries[]=
zoom_video_wh&queries[]=zoom_wh
```

10.4. API для получения списка данных по сигнатурам

API для получение списка данных по сигнатурам реализуется через файл **protocolsdata.proto**. В таблицах ниже дано описание метода и формат запроса и ответа.

Таблица 65 – Метод получения списка данных по сигнатурам

Метод	**Запрос**	**Ответ**	**Действие**
List	ListRequest	ListResponse (потоковый)	Получение списка IP/SNI по сигнатурам

ListResponse

Таблица 66 – Ответ на запрос метода List

Поле	**Тип данных**	**Метка**	**Описание**
protocol_code	uint32		Код сигнатуры
protocol_sub code	uint32		Подкод сигнатуры, если сигнатура не имеет подкодов, то 0, в остальных случаях больше 0

Поле	Тип данных	Метка	Описание
proto_data oneof			ip_data с типом IP или sni_data с типом SNI

IP

Таблица 67 – Сообщение, содержащее IPv4 или IPv6 адрес

Поле	Тип данных	Метка	Описание
ip	string		Сообщение, содержащее IPv4 или IPv6 адрес

SNI

Таблица 68 – Сообщение, содержащее SNI

Поле	Тип данных	Метка	Описание
sni	string		Сообщение, содержащее SNI

10.5. API получения данных о маршрутах пакетов трафика абонента

API для получения данных о маршрутах прохождения пакетов трафика абонента реализуется через файл **routeslist.proto**. В таблицах ниже дано описание метода и формат запроса и ответа.

Таблица 69 – Метод получения списка маршрутов

Метод	Запрос	Ответ	Действие
List	ListRequest	ListResponse (потоковый)	Получение списка маршрутов

Параметры запроса метода List

ListRequest

Таблица 70 – Запрос ListRequest

Поле	Тип данных	Метка	Описание
TimeRange	message		Запрос TimeRange
Ecofilter	message		Запрос Ecofilter
time_range	TimeRange		Диапазон времени для выборки маршрутов. Выборка производится по времени прохождения первого пакета в маршруте через экофильтр
filter oneof			Набор фильтров выборки маршрутов: - local_host с типом Host, - remote_host с типом Host, - ecofilter с типом Ecofilter

TimeRange

Таблица 71 – Запрос диапазона времени

Поле	Тип данных	Метка	Описание
start_time	uint32		Время начала выборки (unix timestamp)
end_time	uint32		Время окончания выборки (unix timestamp)

Ecofilter

Таблица 72 – Запрос идентификатора экофильтра

Поле	Тип данных	Метка	Описание
id	string		Уникальный идентификатор экофильтра

ListResponse

Таблица 73 – Ответ на запрос метода List

Поле	Тип данных	Метка	Описание
Route	message		Ответ Route
route	Route		Определение маршрута

Route

Таблица 74 – Определение Route

Поле	Тип данных	Метка	Описание
PathPoint	message		PathPoint
local_host	Host		Хост клиента
remote_host	Host		Хост удаленного ресурса
path_to_local_host	map<uint64, PathPoint>		Обратный маршрут. Ключ (время прохождения пакета через экофильтр (unix nano)) и его значение
path_to_remote_host	map<uint64, PathPoint>		Прямой маршрут. Ключ (время прохождения пакета через экофильтр (unix nano)) и его значение

PathPoint

Таблица 75 – Определение PathPoint

Поле	Тип данных	Метка	Описание
filter_id	string		Уникальный идентификатор экофильтра
marker_ttl	uint32		TTL маркера в точке маршрута пакета

Host

Таблица 76 – Определение Host

Поле	Тип данных	Метка	Описание
ip	string		IP в формате IPv4 или IPv6
port	uint32		Порт

10.6. API сервиса packets-routes для получения маршрутов напрямую от routes-log-reader

В сервис packets-routes добавлена возможность получения данных по маршрутам пакетов от routes-log-reader напрямую через отдельный API.

API для добавления данных по точке маршрута пакета Add реализуется через файл routesdata.proto.

Описание метода приведено в таблице ниже.

Таблица 77 – Описание метода

Метод	**Запрос**	**Ответ**	**Действие**
Add	AddRequest	AddResponse	Метод для добавления данных по точке маршрута пакета

Параметры запроса приведены в таблице ниже.

Таблица 78 – Параметры запроса

Поле	**Тип данных**	**Описание**
local_ip	string	IP адрес клиента в формате IPv4 или IPv6
local_port	uint32	Порт клиента (значение порта клиента в оригинальном пакете)
fake_local_port	uint32	Фэйковый порт клиента (значение сгенерированное фильтром для копии оригинального пакета)
remote_ip	string	IP адрес удаленного ресурса в формате IPv4 или IPv6
remote_port	uint32	Порт удаленного ресурса
marker	uint32	Маркер
marker_type	MarkerType	Тип маркера
filter_id	string	Уникальный идентификатор экофильтра
packet_unix_nano	uint64	Время, когда пакет был получен СПФС (время добавления записи в БД, unix nano)
marker_ttl	uint32	TTL маркера в точке маршрута пакета

Возможные значения MarkerType приведены в таблице ниже.

Таблица 79 – Возможные значения MarkerType

Поле	**Значение**	**Описание**
MARKER_TYPE_UNSPECIFIED	0	Неопределенный тип маркера

MARKER_TYPE_CLIENT_SYN	1	Подмена SYN пакета клиента (client->server)
MARKER_TYPE_SERVER_SYN_ACK	2	Подмена SYNACK пакета сервера (server->client)
MARKER_TYPE_PARENT_SYN	3	Оригинальный родительский SYN пакет (client->server)

10.7. API наполнения и просмотра списка доменов и соответствующих им id протоколов

API для наполнения и просмотра списка доменов и соответствующих им id протоколов реализуется через файл **list_for_dns_prober.proto**. В таблицах ниже дано описание доступных методов и формат запросов и ответов.

Таблица 80 – Методы наполнения и просмотра списка доменов и соответствующих им id протоколов

Метод	**Запрос**	**Ответ**	**Действие**
Insert	InsertRequest	InsertResponse	Наполнение списка доменов
Remove	RemoveRequest	RemoveResponse	Удаление домена
List	ListRequest	ListResponse (потоковый)	Вывод списка доменов и соответствующих id протоколов

Параметры запросов

InsertRequest

Таблица 81 – Запрос наполнения списка доменов

Поле	**Тип данных**	**Метка**	**Описание**
protocol	Protocol		Домен или код id протокола

RemoveRequest

Таблица 82 – Удаление домена

Поле	**Тип данных**	**Метка**	**Описание**
domain	string		Домен

Результаты выполнения запроса

ListResponse

Таблица 83 – Ответ на запрос ListRequest

Поле	**Тип данных**	**Метка**	**Описание**
protocol	Protocol		id протокола

Protocol

Таблица 84 – Параметры протокола

Поле	Тип данных	Метка	Описание
domain	string		Домен
code	uint32		Код id протокола
sub_code	uint32		Подкод id протокола

10.8. API для управления списками протоколов и игнорируемых протоколов

В сервисе **list-of-protocols** предусмотрены два API для управления списками протоколов и игнорируемых протоколов, описанные в файлах **protocols.proto** и **ignoreprotocols.proto** соответственно.

Таблица 85 – Методы работы с протоколами

Метод	**Запрос**	**Ответ**	**Действие**
Insert	InsertRequest	InsertResponse	Добавление протокола
Delete	DeleteRequest	DeleteResponse	Удаление протокола
List	ListRequest	ListResponse (потоковый)	Считывание списка протоколов

Примечание: Параметры запросов и ответов, названия которых указаны в графах **Запрос** и **Ответ** таблицы выше, приведены в соответствующих таблицах ниже.

Таблица 86 – InsertRequest

Поле	**Тип данных**	**Описание**
code	uint32	ID протокола
title	string	Доменное имя

DeleteRequest

Поле	Тип данных	Описание
code	uint32	ID протокола

Таблица 87 – ListResponse

Поле	Тип данных	Описание
code	uint32	ID протокола
title	string	Доменное имя
created_at	uint64	Создан

Таблица 88 – Методы работы с игнорируемыми протоколами:

Метод	Запрос	Ответ	Действие
Insert	InsertRequest	InsertResponse	Добавление игнорируемого протокола
Delete	DeleteRequest	DeleteResponse	Удаление игнорируемого протокола
List	ListRequest	ListResponse (потоковый)	Считывание списка игнорируемых протоколов

Таблица 89 – InsertRequest

Поле	Тип данных	Описание
code	uint32	ID протокола

Таблица 90 – DeleteRequest

Поле	Тип данных	Описание
code	uint32	ID протокола

Таблица 91 – ListResponse

Поле	Тип данных	Описание
code	uint32	ID протокола
created_at	uint64	Создан

10.9. Описание скалярных типов данных

В таблице (Таблица 92) дано описание скалярных типов данных, передаваемых в запросах и ответах для различных языков реализации API, в таблице (Таблица 93) – описание используемого типа даты и времени.

Таблица 92 – Описание скалярных типов данных

Тип данных protobuf	**Примечания**	**C++**	**Java**	**Python**	**Go**	**C#**	**PHP**	**Ruby**
double		double	double	float	float64	double	float	Float
float		float	float	float	float32	float	float	Float
int32	Использует кодировку переменной длины. Неэффективен для кодирования отрицательных чисел. Если в вашем случае поле может содержать как положительные, так и отрицательные значения, то следует использовать **sint32**.	int32	int	int	int32	int	integer	Bignum или Fixnum (в зависимости от конкретных требований)
int64	Использует кодировку переменной длины. Неэффективен для кодирования отрицательных чисел. Если в вашем случае поле может содержать как положительные, так и отрицательные значения, то следует использовать **sint64**.	int64	long	int/long	int64	long	integer/string	Bignum
uint32	Использует кодировку переменной длины.	uint32	int	int/long	uint32	uint	integer	Bignum или Fixnum (в зависимости от конкретных требований)
uint64	Использует кодировку переменной длины.	uint64	long	int/long	uint64	ulong	integer/string	Bignum или Fixnum (в зависимости от конкретных требований)
sint32	Знаковое целое число. Использует кодировку переменной длины. По сравнению с int32 более эффективен для кодирования отрицательных чисел.	int32	int	int	int32	int	integer	Bignum или Fixnum (в зависимости от конкретных требований)
sint64	Знаковое целое число. Использует кодировку переменной длины. По сравнению с int64 более эффективен для кодирования отрицательных чисел.	int64	long	int/long	int64	long	integer/string	Bignum
fixed32	Префикс длины – всегда 4 байта. Более эффективен, чем uint32, если значения обычно больше 2^28.	uint32	int	int	uint32	uint	integer	Bignum или Fixnum (в зависимости от конкретных требований)
fixed64	Префикс длины – всегда 8 байт. Более эффективен, чем uint64, если значения обычно больше 2^56.	uint64	long	int/long	uint64	ulong	integer/string	Bignum
sfixed32	Префикс длины – всегда 4 байта.	int32	int	int	int32	int	integer	Bignum или Fixnum (в зависимости от конкретных требований)
sfixed64	Префикс длины – всегда 8 байт.	int64	long	int/long	int64	long	integer/string	Bignum
bool		bool	boolean	boolean	bool	bool	boolean	TrueClass/FalseClass

Тип данных protobuf	Примечания	C++	Java	Python	Go	C#	PHP	Ruby
string	Строка должна содержать текст в кодировке UTF-8 или 7-bit ASCII.	string	String	str/unicode	string	string	string	String (UTF-8)
bytes	Может содержать любую произвольную последовательность байтов.	string	ByteString	str	[]byte	ByteString	string	String (ASCII-8BIT)

Таблица 93 – Тип даты и времени

Тип данных protobuf	Примечание	C++
google.protobuf.Timestamp		DateTime

11. УСТРАНЕНИЕ ТИПОВЫХ ПРОБЛЕМ

В этой главе рассмотрены типовые проблемы, которые могут возникнуть в процессе работы системы EcoDPIOS-DC, и даны указания по выявлению и устранению причин.

11.1. В коллектор не поступают записи журналов блокировок от оборудования фильтрации

За приём и обработку записей журналов блокировок от оборудования EcoFilter в коллекторе журналов отвечает сервис приёма первичных данных.

11.1.1. Описание проблемы

После настройки оборудования EcoFilter и запуска сервиса приёма первичных данных значения счётчиков для метрик сервиса не отвечают критериям исправной работы, а именно:

– значения метрик сервиса не приходят по запросу;

```
operator@operator-pc# curl 192.168.91.50:2113/metrics
curl: (7) Failed to connect to 192.168.91.50 port 2112: В соединении
отказано
```

– значения метрик сервиса равны нулю или не изменяются в течение продолжительного времени.

```
operator@operator-pc# curl -s 192.168.91.50:2112/metrics | grep -E
"reader_received_packets|writer_inserted_packets"
# HELP reader_received_packets The total number of received UDP
packets from EcoFilters
# TYPE reader_received_packets counter
reader_received_packets 0
# HELP writer_inserted_packets The total number of inserted packets in
DB
# TYPE writer_inserted_packets counter
writer_inserted_packets{query="insert_histogram_log"} 0
writer_inserted_packets{query="insert_incorrect_log"} 0
writer_inserted_packets{query="insert_random_log"} 0
```

Описание метрик

– **reader_received_packets** – метрика, счётчик которой показывает количество записей журналов блокировок, поступивших в коллектор от оборудования EcoFilter. Счётчик является накопительным.

При правильной настройке оборудования EcoFilter и правильной работе сервиса приёма первичных данных значение счётчика должно быть больше 0.

– **writer_inserted_packets** – набор метрик, счетчики которых показывают

количество записей, которые были распознаны коллектором и успешно отправлены в сервис хранения первичных данных. Для каждого типа распознанных записей предусмотрен отдельный счётчик метрики. Счётчики являются накопительными.

При правильной работе сервисов приёма и хранения первичных данных значения счётчиков должны быть больше 0.

11.1.2. Возможные причины неисправной работы

Причинами возникновения вышеописанной проблемы могут быть:

- неправильная настройка раздела **debug_logger** в конфигурации EcoFilter;
- проблема с сетевой связностью между EcoFilter и сервисом коллектора;
- несоответствующая версия прошивки EcoFilter;
- нерабочее состояние сервисов коллектора журналов блокировок и базы данных;
- неправильная конфигурация сервисов коллектора журналов;
- несоответствующая версия образа для контейнера с сервисом коллектора журналов.

11.1.3. Порядок выявления и устранения причин

1. Проверить настройки **debug_logger** в конфигурации EcoFilter. Убедиться, что отправка сообщений включена и указаны правильные значения параметра **protocols**, а также IP-адрес и порт коллектора. При необходимости внести изменения в конфигурацию оборудования EcoFilter.

2. Проверить актуальность версии прошивки. При необходимости обновить программное обеспечение оборудования EcoFilter.

```
2:# show version
EcoNAT generic v2.1 (C) Ecotelecom [RDP.RU Ltd.] 2013-2019. All rights
reserved.
Firmware version: 3.1.4.0.40.bp
S/N: 1C87764018C3
```

3. Проверить журнал событий для контейнера с сервисом коллектора журналов командой **`docker logs имя_сервиса`**

```
operator@operator-pc# docker logs collector
time="2020-04-27T16:18:49Z" level=info msg="Check for migrations"
func="drop-log-reader/db.(*ClickHouse).Migrate" file="/go/src/drop-
log-reader/db/clickhouse.go:89"
```

```
time="2020-04-27T16:18:49Z" level=info msg="No migrations. Current
version 13 is up to date" func="drop-log-
reader/db.(*ClickHouse).Migrate" file="/go/src/drop-log-
reader/db/clickhouse.go:100"
server listening [::]:555
```

или командой docker-compose logs имя_сервиса

```
docker-compose logs collector
Attaching to spfs_collector_1
collector_1         | time="2020-04-27T16:18:49Z" level=info
msg="Check for migrations" func="drop-log-
reader/db.(*ClickHouse).Migrate" file="/go/src/drop-log-
reader/db/clickhouse.go:89"
collector_1         | time="2020-04-27T16:18:49Z" level=info msg="No
migrations. Current version 13 is up to date" func="drop-log-
reader/db.(*ClickHouse).Migrate" file="/go/src/drop-log-
reader/db/clickhouse.go:100"
collector_1         | server listening [::]:555
```

В журнале событий контейнера с сервисом коллектора журналов не должно быть систематических ошибок.

Для получения дополнительной информации из журналов при неправильном поведении сервиса использовать команду **`docker logs имя_сервиса --details`**.

```
goroutine 1 [running]:
github.com/jmoiron/sqlx.MustConnect(...)
     /go/pkg/mod/github.com/jmoiron/sqlx@v1.2.0/sqlx.go:650
collector/db.(*ClickHouse).Open(0xc00013c400)
     /go/src/collector/db/clickhouse.go:57 +0x149
main.(*App).prepareDB(0xc0001182a0)
     /go/src/collector/main.go:92 +0x43
main.(*App).start(0xc0001182a0)
     /go/src/collector/main.go:68 +0x2f
main.main()
     /go/src/collector/main.go:46 +0x109
panic: dial tcp: lookup clickhouse-server on 127.0.0.11:53: no such
host
```

4. Проверить, что сервисы коллектора журналов и базы данных находятся в рабочем состоянии (STATUS -> Up). Для этого следует отправить команды **`docker ps`** и **`docker-compose ps`** (данную команду необходимо отправлять из директории, в которой находится файл **docker-compose.yaml**).

```
operator@operator-pc# docker ps
CONTAINER ID IMAGE        COMMAND          CREATED        STATUS      PORTS
NAMES
51250b4a6e30 clickhouse   "/entrypoint.sh" 9 minute ago   Up 9 minute
123/tcp,9000/tcp,   clickhouse
                                                                      9009/tcp
08c984509f21 collector:1.0 "/collector"    13 minutes ago Up 9 minute
192.168.91.50:30555 collector
                                                                      ->555/udp

192.168.91.50:2112
                                                                      ->2112/tcp
```

```
operator@operator-pc# docker-compose ps
         Name                    Command     State   Ports
-------------------------------------------------------------------------------
spfs_clickhouse_1         /entrypoint.sh    Up      8123/tcp, 9000/tcp, 9009/tcp
spfs_collector_1          /collector        UP      192.168.91.50:2112->2112/tcp,
                                                    192.168.91.50:3055>555/udp
```

5. Проверить параметры конфигурации сервиса коллектора журнала блокировок, применяемые при запуске сервиса, с помощью команды **`cat docker-compose.yml`** (данную команду необходимо отправлять из директории, в которой находится файл **docker-compose.yml**).

```
operator@operator-pc# cat docker-compose.yml
version: '3.7'
services:

  collector:
    image: collector:1.0
    environment:
      - LISTENIP=0.0.0.0
      - LISTENPORT=555
      - PARSERSCOUNT=4
      - CLICKHOUSEHOST=clickhouse
      - CLICKHOUSEPORT=9000
  clickhouse:
    image: clickhouse
```

Текущую конфигурацию в работающем контейнере с сервисом коллектора можно проверить командой **`docker inspect -f "{{json .Config.Env}}" имя_сервиса | jq`**.

```
operator@operator-pc# docker inspect -f "{{json .Config.Env}}"
collector | jq
[
  "LISTENIP=0.0.0.0",
  "LISTENPORT=555",
  "PARSERSCOUNT=4",
  "CLICKHOUSEHOST=clickhouse-server",
  "CLICKHOUSEPORT=9000",
  "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
]
```

В настройках сервиса коллектора журнала блокировок должны быть указаны правильные значения следующих параметров:

- **LISTENIP** и **LISTENPORT**. Эти параметры указывают, на каком внутреннем IP-адресе и внутреннем UDP-порту сервис принимает данные от EcoFilter. Достаточно значений по умолчанию: 0.0.0.0 и 555.
- **CLICKHOUSEHOST** и **CLICKHOUSEPORT**. Эти параметры указывают сервису коллектора журнала блокировок DNS-имя и порт сервиса базы данных. DNS-имя должно в точности совпадать с именем сервиса базы данных, указанным в файле **docker-compose.yaml**. В противном случае сервис коллектора журнала блокировок не сможет правильно работать.

При необходимости внести соответствующие правки в конфигурационный файл **docker-compose.yaml** и исправить поведение сервиса командой **`docker-compose up -d`**.

6. Проверить командами **`docker ps`** и **`docker-compose ps`**, что сервис коллектора журнала блокировок имеет правильную конфигурацию: IP-адрес и порт в разделе **ports** соответствуют настройкам раздела **debug_logger** в конфигурации оборудования EcoFilter:

```
operator@operator-pc# docker ps
CONTAINER ID IMAGE        COMMAND          CREATED        STATUS      PORTS
NAMES
51250b4a6e30 clickhouse   "/entrypoint.sh" 9 minute ago   Up 9 minute
123/tcp,9000/tcp,   clickhouse
                                                                       9009/tcp
08c984509f21 collector:1.0 "/collector"    13 minutes ago Up 9 minute
192.168.91.50:30555 collector
                                                                       ->555/udp

192.168.91.50:2112
                                                                       ->2112/tcp
operator@operator-pc# docker-compose ps
         Name                    Command     State   Ports
-------------------------------------------------------------------------------
spfs_clickhouse_1         /entrypoint.sh    Up      8123/tcp, 9000/tcp, 9009/tcp
spfs_collector_1          /collector        UP      192.168.91.50:2112->2112/tcp,
                                                    192.168.91.50:3055>555/udp
```

При необходимости внести соответствующие правки в конфигурационный файл **docker-compose.yaml** и обновить развёрнутый сервис командой **`docker-compose up -d`**.

7. Проверить, что сервис коллектора журнала блокировок проброшен на внешний порт сервера. Для этого следует отправить команду **`netstat -tulpn | grep -E "30555|2112"`**.

```
operator@operator-pc# netstat -tulpn  | grep -E  "30555|2112"
tcp   0   0 192.168.91.50:2112    0.0.0.0:*      LISTEN      25129/docker-proxy
udp   0   0 192.168.91.50:30555   0.0.0.0:*                  25116/docker-proxy
```

Проброс портов необходим для того, чтобы трафик, посылаемый EcoFilter, после получения сервером на UDP-порт (30555) мог быть перенаправлен в сервис коллектора журнала блокировок. Кроме основного порта для приёма трафика логов может быть проброшен TCP-порт (2112), используемый для доступа к метрикам сервиса коллектора журнала блокировок в целях диагностики. Для обслуживания этих портов должна использоваться служба **docker-proxy**.

8. Проверить версию образа сервиса командой docker inspect -f "{{json .Config.Image}}" имя_сервиса | jq.

```
docker inspect -f "{{json .Config.Image}}" collector | jq
"collector:1.0"
```

или командой docker ps | grep имя_сервиса:

```
operator@operator-pc# docker ps | grep collector
CONTAINER ID IMAGE      COMMAND         CREATED        STATUS    PORTS       NAMES
51250b4a6e30 clickhouse "/entrypoint.sh"9 minute ago Up 9 minute 123/tcp,  clickhouse
                                                                 9000/tcp,
9009/tcp
08c984509f21 collector:1.0 "/collector"    13 minutes ago Up 9 minute
192.168.91.50:30555 collector
                                                                        ->555/udp

192.168.91.50:2112
                                                                        ->2112/tcp
```

При необходимости следует скачать новый образ из репозитория (команды **`docker pull`** или **`docker-compose pull`**), внести соответствующие правки в конфигурационный файл **docker-compose.yaml** и обновить развёрнутые сервисы командой **`docker-compose up -d`**.

9. Проверить сетевую связность между сервисом коллектора журнала блокировок и оборудованием EcoFilter. Например, командой **`docker exec -it имя_сервиса sh`** зайти в оболочку сервиса коллектора журнала логов и проверить доступность управляющего интерфейса EcoFilter с помощью утилиты **ping**.

```
/srv # ping 10.86.4.101 -c 4
PING 10.86.4.101 (10.86.4.101): 56 data bytes
64 bytes from 10.86.4.101: seq=0 ttl=62 time=0.375 ms
64 bytes from 10.86.4.101: seq=1 ttl=62 time=0.438 ms
64 bytes from 10.86.4.101: seq=2 ttl=62 time=0.467 ms
64 bytes from 10.86.4.101: seq=3 ttl=62 time=0.486 ms

--- 10.86.4.101 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.375/0.441/0.486 ms
```

Устранить проблему сетевой связности при её наличии.

10. Проверить получение трафика логов от оборудования EcoFilter на сервер на системном уровне и на уровне сервиса коллектора журнала блокировок с использованием общедоступных системных утилит **tcpdump**, **iftop**, **nload**. Подключение к оболочке сервиса выполняется также с помощью команды **`docker exec -it имя_сервиса sh`**.

```
/ srv # tcpdump -ni eth0 udp port 30555 -c 10
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:22:25.217505 IP 10.86.4.192.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217510 IP 10.86.4.192.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217515 IP 10.86.4.192.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217543 IP 10.86.4.192.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217525 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217530 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217534 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217539 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217543 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
12:22:25.217548 IP 10.86.4.193.1088 > 192.168.91.50.30555: UDP, length 78
10 packets captured
```

```
11 packets received by filter
0 packets dropped by kernel
```

При необходимости откорректировать работу межсетевого экрана на системном уровне.

11.2. Не работают базовые сервисы генерации списка и сервисы промежуточного хранения данных

Базовые сервисы генерации списка СПФС отвечают за формирование записей из данных, полученных сервисом коллектора журнала блокировок. Сгенерированные записи должны быть отправлены в сервис хранения промежуточных данных, работающий в составе СЦОС.

В свою очередь, задачей сервиса хранения промежуточных данных в СЦОС является хранение записей, полученных от базовых сервисов генерации списков в составе СПФС.

11.2.1. Описание проблемы

После правильной настройки и запуска оборудования EcoFilter, сервисов коллектора журнала блокировок и базы данных, показатели метрик базовых сервисов генерации списка или сервисов хранения промежуточных данных не соответствуют критериям успешной работы, а именно:

– значения счётчиков для метрик базовых сервисов генерации списка не приходят по запросу:

```
operator@operator-pc# curl 192.168.91.50:2113/metrics
curl: (7) Failed to connect to 192.168.91.50 port 2113: В соединении
отказано
```

– значения счётчиков для метрик базовых сервисов генерации списка в течение продолжительного времени остаются равными нулю, однако при этом записи журнала от EcoFilter поступают в сервис коллектора журнала блокировок.

```
operator@operator-pc# operator@operator-pc# curl -s
192.168.91.50:2113/metrics|grep -E
"acl_creator_items_sended_via_grpc|acl_creator_items_from_source"
# HELP acl_creator_items_from_source Number of records returned from
source in last query
# TYPE acl_creator_items_from_source gauge
acl_creator_items_from_source 0
# HELP acl_creator_items_sended_via_grpc Number of records sended via
grpc
# TYPE acl_creator_items_sended_via_grpc gauge
acl_creator_items_sended_via_grpc 0
```

– значения счётчиков для метрик сервисов хранения промежуточных

данных не приходят по запросу:

```
operator@operator-pc# kubectl port-forward scos-acl-list-cache-black-
v4 --namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep
acl_list_items_in_last_list
Handling connection for 5000
E0429 15:45:55.670279 24686 portforward.go:400] an error occurred
forwarding 5000 -> 2112: error forwarding port 2112 to pod
d5f6b9c8c1e8f64646886f04bf696ab770ce45b4c7b7f117f4068d0c44a90ac5, uid
: exit status 1: 2020/04/29 12:45:55 socat[27411] E connect(5, AF=2
127.0.0.1:2112, 16): Connection refused
```

– значения счётчиков для метрик сервисов хранения промежуточных данных в течение длительного промежутка времени остаются равными нулю, однако при этом:

  - записи журнала от EcoFilter поступают в сервис коллектора журнала блокировок,
  - базовые сервисы генерации списка получают записи из сервиса базы данных.

```
operator@operator-pc# kubectl port-forward scos-acl-list-cache-black-
v4 --namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep
acl_list_items_in_last_list

# HELP acl_list_items_in_last_list Count items returned in last list
Query
# TYPE acl_list_items_in_last_list gauge
acl_list_items_in_last_list{exclude="",include="",ip="",no_expired="tr
ue",pagination_page_no="",pagination_per_page="",port="0"} 0
```

Описание метрик

– **acl_creator_items_from_source** (для базового сервиса генерации списка)

Метрика, счётчик которой показывает количество записей, попавших в базовый сервис генерации списка из сервиса базы данных. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при выполнении проверок базового сервиса генерации списка необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе базового сервиса генерации списка значение счётчика данной метрики в момент поступления записей из базы данных должно быть больше 0.

– **acl_creator_items_sended_via_grpc** (для базового сервиса генерации списка)

Метрика, счётчик которой показывает количество записей, отправленых из базового сервиса генерации списка в сервис хранения промежуточных данных. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при выполнении проверок базового сервиса генерации списка необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе базового сервиса генерации списка и сервиса хранения промежуточных данных значение счётчика данной метрики в момент отправки записей в сервис хранения промежуточных данных должно быть больше 0.

– **acl_list_items_in_last_list** (для сервиса хранения промежуточных данных)

Метрика, счётчик которой показывает количество записей, находящихся в сервисе хранения промежуточных данных. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при выполнении проверок сервиса хранения промежуточных данных необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса хранения промежуточных данных и базовых сервисов генерации списка значение счётчика данной метрики в момент поступления записей в сервис хранения промежуточных данных должно быть больше 0.

Базовые сервисы генерации списка и сервисы хранения промежуточных данных состоят из N экземпляров. Каждый экземпляр базового сервиса генерации списка генерирует определенные типы записей, а каждый экземпляр сервиса хранения промежуточных данных хранит определенные типы записей. При

обнаружении проблемы в работе этих сервисов необходимо выяснить, какие именно экземпляры работают неправильно.

11.2.2. Возможные причины неисправной работы

Возможными причинами являются:

- экземпляр базового сервиса генерации списка находится в нерабочем состоянии, и данные не передаются в сервис хранения промежуточных данных;
- экземпляр сервиса хранения промежуточных данных находится в нерабочем состоянии, и базовый сервис генерации списка не может передать ему на хранение сгенерированные записи;
- неправильная конфигурация экземпляра базового сервиса генерации списка или экземпляра сервиса хранения промежуточных данных;
- нарушение сетевой связности между сервисом базы данных и экземпляром базового сервиса генерации списка;
- нарушение сетевой связности между базовым сервисом генерации списка и сервисом хранения промежуточных данных;
- несоответствующая версия образа для контейнера с экземпляром базового сервиса генерации списка или экземпляром сервиса хранения промежуточных данных.

11.2.3. Порядок выявления и устранения причин

1. Проверить журнал событий для контейнера с экземпляром базового сервиса генерации списка командой `docker logs имя_сервиса.`

```
operator@operator-pc# docker logs spfs-gen-list-random
time="2020-04-28T14:39:01Z" level=info msg="config reload server
started" func=main.startConfigReloadServer file="/go/src/acl-
creator/main.go:70"
time="2020-04-28T14:39:01Z" level=info msg="Start processing"
func="acl-creator/creator.(*Creator).StartMainLoop" file="/go/src/acl-
creator/creator/creator.go:117"
time="2020-04-28T14:39:01Z" level=error msg="rpc error: code =
Unavailable desc = connection error: desc = \"transport: Error while
dialing dial tcp 192.168.160.4:8080: connect: connection refused\""
func="acl-creator/creator.(*Creator).sendItemsToACLList"
file="/go/src/acl-creator/creator/creator.go:183"
```

или командой docker-compose logs имя_сервиса

```
operator@operator-pc# docker-compose logs spfs-gen-list-random
Attaching to spfs-gen-list-random
gen-list-random_1   | time="2020-04-28T14:39:01Z" level=info
msg="config reload server started" func=main.startConfigReloadServer
file="/go/src/acl-creator/main.go:70"
```

```
gen-list-random_1   | time="2020-04-28T14:39:01Z" level=info
msg="Start processing" func="acl-
creator/creator.(*Creator).StartMainLoop" file="/go/src/acl-
creator/creator/creator.go:117"
gen-list-random_1   | time="2020-04-28T14:39:01Z" level=error msg="rpc
error: code = Unavailable desc = connection error: desc = \"transport:
Error while dialing dial tcp 192.168.160.4:8080: connect: connection
refused\"" func="acl-creator/creator.(*Creator).sendItemsToACLList"
file="/go/src/acl-creator/creator/creator.go:183"
```

В журнале событий контейнера с экземпляром базового сервиса генерации списка не должно быть систематических ошибок за последнее время.

Для получения дополнительной информации из журналов при неправильной работе сервиса использовать команду `docker logs имя_сервиса --details`.

```
operator@operator-pc# docker logs spfs-gen-list-random --details
time="2020-04-28T14:39:01Z" level=info msg="config reload server
started" func=main.startConfigReloadServer file="/go/src/acl-
creator/main.go:70"
 time="2020-04-28T14:39:01Z" level=info msg="Start processing"
func="acl-creator/creator.(*Creator).StartMainLoop" file="/go/src/acl-
creator/creator/creator.go:117"
 time="2020-04-28T14:39:01Z" level=error msg="rpc error: code =
Unavailable desc = connection error: desc = \"transport: Error while
dialing dial tcp 192.168.160.4:8080: connect: connection refused\""
func="acl-creator/creator.(*Creator).sendItemsToACLList"
file="/go/src/acl-creator/creator/creator.go:183
```

2. Проверить журнал событий для контейнера с экземпляром сервиса хранения промежуточных данных командой `kubectl logs имя_экземпляра_сервиса -n scos`.

```
operator@operator-pc# kubectl logs scos-acl-list-cache-black-v4 -n
scos
time="2020-04-23T14:13:25Z" level=info msg=Connected
func=main.getDBConnection file="/go/src/acl-list/main.go:87"
time="2020-04-23T14:13:25Z" level=info msg="Check for migrations"
func="acl-list/clickhouse.(*Connection).Migrate" file="/go/src/acl-
list/clickhouse/clickhouse.go:140"
time="2020-04-23T14:13:25Z" level=info msg="Migrate from 2 to 3 (steps
1)" func="acl-list/clickhouse.(*Connection).Migrate"
file="/go/src/acl-list/clickhouse/clickhouse.go:143"
time="2020-04-23T14:13:27Z" level=info msg=Migrated
func=main.getDBConnection file="/go/src/acl-list/main.go:90"
time="2020-04-23T14:13:27Z" level=error msg="not applicable method
CreateIndexes" func="acl-list/clickhouse.(*Writer).CreateIndexes"
file="/go/src/acl-list/clickhouse/writer.go:223"
time="2020-04-23T14:13:27Z" level=info msg="Indexes are created"
func=main.getDBConnection file="/go/src/acl-list/main.go:93"
time="2020-04-23T14:13:27Z" level=info msg="Start processing loop"
func=main.getDBConnection file="/go/src/acl-list/main.go:96"
```

В журнале событий контейнера с экземпляром сервиса промежуточного хранения данных не должно быть систематических ошибок за последнее время.

3. Проверить, что экземпляры базовых сервисов генерации списка находятся в рабочем состоянии (STATUS – Up). Для этого следует отправить команды **`docker ps`** и **`docker-compose ps`** (данную команду необходимо отправлять из директории, в которой находится файл **docker-compose.yaml**).

```
operator@operator-pc# docker ps
CONTAINER ID IMAGE       COMMAND      CREATED       STATUS       PORTS
NAMES
fc4e761bddba acl-creator "/gen-list"  22 hours ago  Up 22 hours
0.0.0.0:2113  spfs-gen-list-tls
                                                                 -
>2112/tcp
28d007a1d861 acl-creator "/gen-list"  22 hours ago  Up 22 hours
0.0.0.0:2114  spfs-gen-list-random
                                                                 -
>2112/tcp
50e693e88808 acl-creator "/gen-list"  22 hours ago  Up 22 hours
0.0.0.0:2115  spfs-gen-list-hrandom
                                                                 -
>2112/tcp
operator@operator-pc# docker-compose ps
         Name              Command            State          Ports
-------------------------------------------------------------------
-----------
spfs-gen-list-hrandom          /gen-list          Up
0.0.0.0:2115->2112/tcp
spfs-gen-list-random           /gen-list          Up
0.0.0.0:2114->2112/tcp
spfs-gen-list-tls              /gen-list          Up
0.0.0.0:2113->2112/tcp
```

4. Проверить, что экземпляр сервиса хранения промежуточных данных и его базы данных находятся рабочем состоянии (статус – Running). Для этого следует отправить команду `kubectl get pods --namespace=scos | grep имя_экземпляра_сервиса`.

```
operator@operator-pc# kubectl get pods --namespace=scos | grep list-
cache
NAME                            READY   STATUS    RESTARTS   AGE
scos-acl-list-cache-black-v4    1/1     Running   3          2d22h
scos-acl-list-cache-db-o        2/2     Running   0          2d22h
```

5. Проверить значения параметров конфигурации экземпляра базового сервиса генерации списка, применяемые при запуске экземпляра сервиса. Для этого необходимо отправить команду **`cat docker-compose.yaml`**. Данную команду необходимо отправлять из директории, в которой находится файл **docker-compose.yaml**.

```
operator@operator-pc# cat docker-compose.yml
version: '3.7'
```

```
services:

  gen-list-random:
    image: hub.scos.ru/acl-creator:latest
    environment:
      - ITEM_TAG=random
      - MASK_LEN=32
      - CLICKHOUSEHOST=clickhouse
      - CLICKHOUSEPORT=9000
      - ACLLISTHOST=list-cache
      - ACLLISTPORT=8080
    ports:
      - "2114:2112/tcp"
```

Сравнить полученную конфигурацию с текущей конфигурацией запущенного контейнера экземпляра базового сервиса генерации списка с помощью команды `docker inspect -f "{{json .Config.Env}}" имя_сервиса | jq` (для выполнения команды необходимо установить утилиту для обработки json из командной строки jq)

```
operator@operator-pc# docker inspect -f "{{json .Config.Env}}" gen-
list-random | jq
[
  "ITEM_TAG=random",
  "MASK_LEN=32",
  "CLICKHOUSEHOST=clickhouse",
  "CLICKHOUSEPORT=9000",
  "ACLLISTHOST=192.168.100.10",
  "ACLLISTPORT=30643",
  "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
]
```

В параметрах экземпляра сервиса коллектора журнала блокировок должны быть заданы правильные значения следующих параметров:

- **ITEM_TAG**. Этот параметр определяет тип записи, которую генерирует базовый сервис генерации списка. Каждый экземпляр сервиса должен генерировать записи для чёрного или белого списка. Экземпляры, генерирующие записи для черного списка, могут быть трёх типов: **random**, **hrandom** и **tls**;
- **MASK_LEN**. Этот параметр определяет длину генерируемого IP-адреса. Для IPv4 значение должно быть 32, для IPv6 – 128;
- **CLICKHOUSEHOST** и **CLICKHOUSEPORT**. Эти параметры указывают экземпляру базового сервиса генерации списка DNS-имя и номер порта сервиса базы данных. DNS-имя и номер порта должны совпадать с именем сервиса базы данных, указанным в файле **docker-compose.yaml**. При неверных значениях этих параметров сервис будет работать неправильно.
- **ACLLISTHOST** и **ACLLISTPORT**. Эти параметры указывают экземпляру базового сервиса генерации списка IP-адрес (или DNS-имя) и порт экземпляра сервиса хранения данных. IP-адрес (или DNS-имя) и номер

порта должны совпадать со значениями, присвоенными экземпляру сервиса хранения промежуточных данных. При неверных значениях этих параметров сервис будет работать неправильно.

При необходимости следует внести необходимые правки в конфигурационный файл **docker-compose.yaml** и обновить конфигурацию сервиса командой `docker-compose up --d`.

6. Проверить текущие параметры конфигурации экземпляра сервиса хранения промежуточных данных, применяемые при запуске экземпляра сервиса. Для этого следует отправить команду `kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].env"`

```
operator@operator-pc# kubectl get pod scos-acl-list-cache-black-v4 --
namespace=scos -o json | jq ".spec.containers[].env"
[
  {
    "name": "DBUSERNAME",
    "value": "CLICKHOUSEUSER"
  },
  {
    "name": "DBPASSWORD",
    "value": "CLICKHOUSEPASS"
  },
  {
    "name": "DBDATABASE",
    "value": "acllist"
  },
  {
    "name": "IP_TYPE",
    "value": "IPv4"
  },
  {
    "name": "DB_TYPE",
    "value": "CLICKHOUSE"
  },
  {
    "name": "DBHOST",
    "value": "scos-list-cache-db-native"
  }
]
```

В выводе параметров экземпляра сервиса хранения промежуточных данных должны быть указаны правильные значения следующих параметров:

- **DB_TYPE**. Этот параметр определяет тип базы данных, используемый для экземпляра сервиса хранения промежуточных данных. Должно быть задано значение **CLICKHOUSE**.
- **DBHOST**. Этот параметр определяет DNS-имя сервиса базы данных. DNS-имя сервиса базы данных можно узнать с помощью команды `kubectl get services --namespace=scos | grep list-cache`.

```
operator@operator-pc# kubectl get services --namespace=scos | grep list-cache
NAME                        TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
AGE
scos-list-cache-v4          NodePort    10.43.217.12    <none>
30643:30643/TCP    8d
scos-list-cache-db-http     ClusterIP   10.43.197.255   <none>        8123/TCP
8d
scos-list-cache-db-metrics  ClusterIP   10.43.86.109    <none>        9116/TCP
8d
scos-list-cache-db-native   ClusterIP   10.43.245.44    <none>        9000/TCP
8d
scos-list-cache-metrics     ClusterIP   10.43.126.189   <none>        2112/TCP
8d
```

Указанное имя должно совпадать с именем, указанным в выводе команды выше (с префиксом *-db-native). При неверном значении этого параметра сервис хранения промежуточных данных будет работать неправильно.

При необходимости следует задать правильные значения параметров экземпляра сервиса хранения промежуточных данных в файле с информацией о чарте (**chart.yaml**) и обновить сервис с помощью установщика пакетов **helm**, используя ключевое слово **`--upgrade`**.

7. Проверить сетевую связность между экземпляром базового сервиса генерации списка и сервисом базы данных. Например, командой **`docker exec -it имя_сервиса sh`** зайти в оболочку экземпляра базового сервиса генерации списка и проверить доступность сервиса базы данных с помощью утилиты **ping**.

```
operator@operator-pc# docker exec -it spfs-gen-list-random sh
/srv # ping -c4 spfs-clickhouse
PING spfs-clickhouse (192.168.160.6): 56 data bytes
64 bytes from 192.168.160.6: seq=0 ttl=64 time=0.044 ms
64 bytes from 192.168.160.6: seq=1 ttl=64 time=0.052 ms
64 bytes from 192.168.160.6: seq=2 ttl=64 time=0.051 ms
64 bytes from 192.168.160.6: seq=3 ttl=64 time=0.050 ms

--- spfs-clickhouse ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 0.044/0.049/0.052 ms
```

Проверить доступность сервиса базы данных по прослушивающему порту, указанному в файле **docker-compose.yaml**, с помощью общедоступных системных инструментов: **netcat**, **nmap**.

```
/srv # nmap spfs-clickhouse -p 9000
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 15:50 UTC
Nmap scan report for spfs-clickhouse (192.168.160.6)
Host is up (0.000071s latency).
rDNS record for 192.168.160.6: spfs-clickhouse.spfs_default

PORT     STATE SERVICE
9000/tcp open  cslistener
```

```
MAC Address: 02:42:C0:A8:A0:06 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.68 seconds
```

При отсутствии доступа к сервису базы данных по целевому порту необходимо устранить нарушение сетевой связности.

8. Проверить сетевую связность между базовым сервисом генерации списков и сервисом промежуточного хранения данных. Для получения данных из базового сервиса генерации списка сервис промежуточного хранения данных должен иметь тип сетевой службы **NodePort**.

Информацию об используемом типе службы можно получить с помощью команды `kubectl get services --namespace=scos | grep list-cache | grep NodePort`.

```
operator@operator-pc# kubectl get services --namespace=scos | grep
list-cache | grep NodePort
NAME                 TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)
AGE
scos-list-cache-v4   NodePort   10.43.217.12   <none>
30643:30643/TCP  8d
```

Эта же команда предоставляет информацию о номере TCP-порта для сервиса хранения промежуточных данных. Тип службы **NodePort** позволяет сервису промежуточных данных получать требуемые данные (предварительные списки) из-за пределов кластера kubernetes.

Следующий этап проверки сетевой связности – выявление IP-адреса экземпляра сервиса хранения промежуточных данных, который будет равен любому IP-адресу узла кластера kubernetes. Узнать IP-адреса всех узлов кластера можно с помощью команды `kubectl get nodes -o jsonpath='{ $.items[*].status.addresses[?(@.type=="InternalIP")].address }{"\n"}'`.

```
operator@operator-pc# kubectl get nodes -o jsonpath='{
$.items[*].status.addresses[?(@.type=="InternalIP")].address }{"\n"}'
192.168.100.10 192.168.100.11 192.168.100.12 192.168.100.13%
```

После получения информации о том, какой IP-адрес и TCP-порт используются экземпляром сервиса хранения промежуточных данных, выполняется проверка сетевой доступности со стороны экземпляра базового сервиса генерации списка с помощью общедоступных системных инструментов: **netcat**, **nmap**.

```
operator@operator-pc# docker exec -it spfs_gen-list-random_1 sh
/srv # nmap 192.168.100.10 -p 30643
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 16:52 UTC
Nmap scan report for rancher.ttraf.ru (192.168.100.10)
Host is up (0.00046s latency).
```

```
PORT      STATE SERVICE
30643/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

При отсутствии доступа к экземпляру сервиса хранения промежуточных данных по целевому порту следует устранить нарушение сетевой связности.

9. Проверить версию образа работающего экземпляра сервиса хранения промежуточных данных с помощью команды **`kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].image"`**.

```
operator@operator-pc# kubectl get pod scos-acl-list-cache-black-v4 --
namespace=scos -o json | jq ".spec.containers[].image"
"acl-list:v2.6.0"
```

Проверить в файле с информацией о чарте (**chart.yaml**) соответствие версии образа для экземпляра сервиса промежуточного хранения данных, при необходимости изменить версию образа и обновить экземпляр сервиса с помощью установщика пакетов **helm**, используя ключевое слово **--upgrade.**

10. Проверить версию образа экземпляра сервиса командой `docker inspect -f "{{json .Config.Image}}" имя_сервиса | jq`:

```
docker inspect -f "{{json .Config.Image}}" spfs-gen-list-random | jq
"acl-creator:latest"
```

или **docker ps | grep имя_сервиса:**

```
operator@operator-pc# docker ps | grep spfs-gen-list-random
CONTAINER ID IMAGE       COMMAND     CREATED      STATUS      PORTS          NAMES
28d007a1d861 acl-creator "/gen-list" 22 hours ago Up 22 hours 0.0.0.0:2114 spfs-gen-
list-random
                                                              ->2112/tcp
```

При необходимости следует скачать новый образ из репозитория (команда **`docker pull`** или **`docker-compose pull`**), внести соответствующие правки в конфигурационный файл **docker-compose.yaml** и обновить развёрнутые сервисы командой **`docker-compose up -d`**.

11.3. Не работают сервисы создания основных списков или сервисы хранения основных списков

Сервисы создания основных списков отвечают за формирование основных списков из данных, хранящихся в сервисе хранения промежуточных данных. В свою очередь, сервисы хранения основных списков обеспечивают доступ к основным спискам для возможности их дальнейшей обработки другими сервисами в составе СЦОС.

11.3.1. Описание проблемы

После правильной настройки оборудования фильтрации и начала отправки журналов пользовательских блокировок, настройки и запуска сервисов коллектора журнала блокировок и базы данных базовые сервисы генерации списков и сервисы хранения промежуточных данных работают правильно, но значения счётчиков для сервисов создания основных списков или сервисов хранения основных списков не соответствуют критериям исправной работы, а именно:

– значения счётчиков для метрик сервисов создания основных списков не приходят по запросу:

```
operator@operator-pc# kubectl port-forward scos-acl-creator-from-
cache-black-v4 --namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"acl_creator_items_sended_via_grpc|acl_creator_items_from_source"
Handling connection for 5000
E0429 15:45:55.670279 24686 portforward.go:400] an error occurred
forwarding 5000 -> 2112: error forwarding port 2112 to pod
d5f6b9c8c1e8f64646886f04bf696ab770ce45b4c7b7f117f4068d0c44a90ac5, uid
: exit status 1: 2020/04/29 12:45:55 socat[27411] E connect(5, AF=2
127.0.0.1:2112, 16): Connection refused
```

– значения счётчиков для метрик сервисов создания основных списков в течение продолжительного времени равны нулю, однако при этом записи от EcoFilter успешно поступают в сервис коллектора журнала блокировок, базовые сервисы генерации списков успешно получают записи от сервиса базы данных и успешно отправляют сгенерированные записи в сервис хранения промежуточных данных.

```
operator@operator-pc# kubectl port-forward scos-acl-creator-from-
cache-black-v4 --namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"acl_creator_items_sended_via_grpc|acl_creator_items_from_source"

# HELP acl_creator_items_from_source Number of records returned from
source in last query
# TYPE acl_creator_items_from_source gauge
acl_creator_items_from_source 0
# HELP acl_creator_items_sended_via_grpc Number of records sended via
grpc
# TYPE acl_creator_items_sended_via_grpc gauge
acl_creator_items_sended_via_grpc 0
```

– значения счётчиков для метрик сервисов хранения основных списков не приходят по запросу:

```
operator@operator-pc# kubectl port-forward scos-acl-list-black-v4 --
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep
acl_list_items_in_last_list
Handling connection for 5000
E0429 15:45:55.670279 24686 portforward.go:400] an error occurred
forwarding 5000 -> 2112: error forwarding port 2112 to pod
d5f6b9c8c1e8f64646886f04bf696ab770ce45b4c7b7f117f4068d0c44a90ac5, uid
: exit status 1: 2020/04/29 12:45:55 socat[27411] E connect(5, AF=2
127.0.0.1:2112, 16): Connection refused
```

– значения счётчиков для метрик сервисов хранения основных списков в течение продолжительного времени остаются равными нулю, однако при этом записи журналов EcoFilter поступают в сервис коллектора журнала блокировок, базовые сервисы генерации списков получают записи из сервиса базы данных и передают сгенерированные записи в сервис хранения промежуточных данных, а сервисы создания основных списков формируют записи для сервисов хранения основных списков.

```
operator@operator-pc# kubectl port-forward scos-acl-list-black-v4 --
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep
acl_list_items_in_last_list

acl_list_items_in_last_list{exclude="",include="rkn,static",ip="",no_e
xpired="true",pagination_page_no="",pagination_per_page="",port="0"} 0
acl_list_items_in_last_list{exclude="rkn,static",include="tls,hrandom"
,ip="",no_expired="true",pagination_page_no="",pagination_per_page="",
port="0"} 0
acl_list_items_in_last_list{exclude="static,rknip",include="tls",ip=""
,no_expired="true",pagination_page_no="",pagination_per_page="",port="
0"} 0
acl_list_items_in_last_list{exclude="static,rknip,tls",include="hrando
m",ip="",no_expired="true",pagination_page_no="",pagination_per_page="
",port="0"} 0
```

Описание метрик:

– **acl_creator_items_from_source** (для сервиса создания основных списков)

Метрика, счётчик которой показывает количество записей, поступивших в сервис создания основных списков из сервиса хранения промежуточных списков. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при выполнении проверок сервиса

создания основных списков необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе базового сервиса генерации списка значение счётчика данной метрики в момент получения записей от сервиса хранения промежуточных списков должно быть больше 0.

– **acl_creator_items_sended_via_grpc** (для сервиса создания основных списков)

Метрика, счётчик которой показывает количество записей, отправленных сервисом создания основных списков в сервис хранения основных списков. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при выполнении проверок сервиса создания основных списков необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса создания основных списков и сервиса хранения основных списков значение счётчика данной метрики в момент отправки записей в сервис хранения основных списков должно быть больше 0.

– **acl_list_items_in_last_list** (для сервиса хранения основных данных)

Метрика, счётчик которой показывает количество записей в сервисе хранения основных данных.

При правильной работе сервиса хранения основных данных и сервиса создания основных списков значение счётчика данной метрики должно быть больше 0.

Сервисы создания основных списков и сервисы хранения основных списков состоят из N экземпляров. Каждый экземпляр сервиса создания основных списков генерирует определённые типы записей, а каждый экземпляр сервиса хранения основных данных хранит определённые типы записей (чёрный список для IPv4, белый список для IPv4, чёрный список для IPv6 и белый список для IPv6). При обнаружении проблемы в работе этих сервисов необходимо выяснить, какие именно экземпляры работают неправильно.

11.3.2. Возможные причины неисправной работы

Возможными причинами являются:

- экземпляр сервиса создания основных списков находится в нерабочем состоянии, и данные не передаются в сервис хранения основных списков;
- экземпляр сервиса хранения основных списков находится в нерабочем состоянии, и сервис создания основных списков не может передать ему на хранение сгенерированные записи;
- неправильная конфигурация экземпляра сервиса создания основных списков или экземпляра сервиса хранения основных списков;
- нарушение сетевой связности между сервисом хранения промежуточных данных и экземпляром сервиса создания основных списков;
- нарушение сетевой связности между экземпляром сервиса создания основных списков и сервисом хранения основных списков;
- несоответствующая версия образа для контейнера с экземпляром сервиса создания основных списков или экземпляром сервиса хранения основных списков.

11.3.3. Порядок выявления и устранения причин

1. Проверить журнал событий для контейнера с экземпляром сервиса создания основных списков командой **`kubectl logs имя_экземпляра_сервиса -n scos`**.

```
operator@operator-pc# kubectl logs  scos-acl-creator-from-cache-black-v4 --namespace=scos
time="2020-04-17T15:30:19Z" level=info msg="no config reload server" func=main.startConfigReloadServer file="/go/src/acl-creator/main.go:72"
time="2020-04-17T15:30:19Z" level=info msg="Start processing" func="acl-creator/creator.(*Creator).StartMainLoop" file="/go/src/acl-creator/creator/creator.go:117"
time="2020-04-20T09:50:18Z" level=error msg="rpc error: code = Unavailable desc = connection error: desc = \"transport: Error while dialing dial tcp 10.210.9.250:30643: connect: connection refused\"" func="acl-creator/acllist.(*Source).GetItemsChannel.func1" file="/go/src/acl-creator/acllist/datasource.go:100"
```

В журнале событий контейнера с экземпляром сервиса создания основных списков за последнее время не должно быть систематических ошибок.

2. Проверить журнал событий для контейнера с экземпляром сервиса хранения основных списков командой `kubectl logs имя_экземпляра_сервиса -n scos -c имя основного_контейнера_для_экземпляра_сервиса`.

```
operator@operator-pc# kubectl logs scos-acl-list-black-v4 --namespace=scos -c acl-list-black-v4
time="2020-04-23T14:13:25Z" level=info msg=Connected func=main.getDBConnection file="/go/src/acl-list/main.go:87"
```

```
time="2020-04-23T14:13:25Z" level=info msg="No migrations" func="acl-
list/mongodb.(*Connection).Migrate" file="/go/src/acl-
list/mongodb/mongodb.go:110"
time="2020-04-23T14:13:25Z" level=info msg=Migrated
func=main.getDBConnection file="/go/src/acl-list/main.go:90"
time="2020-04-23T14:13:25Z" level=error msg="index created:
tags_name_idx" func="acl-
list/mongodb.(*Connection).createTagNamesIndex" file="/go/src/acl-
list/mongodb/mongodb.go:598"
time="2020-04-23T14:13:25Z" level=info msg="index created:
tags_expiredat_idx" func="acl-
list/mongodb.(*Connection).createTagExpiredIndex" file="/go/src/acl-
list/mongodb/mongodb.go:621"
time="2020-04-23T14:13:25Z" level=error msg="index creation failed:
(IndexKeySpecsConflict) Index must have unique name.The existing
index: { v: 2, key: { tags.name: 1, tags.expiredat: 1 }, name:
\"tags_name_expiredat_idx\", ns: \"blacklist.acllist\" } has the same
name as the requested index: { v: 2, key: { tags.expiredat: 1,
tags.name: 1 }, name: \"tags_name_expiredat_idx\", ns:
\"blacklist.acllist\" }" func="acl-
list/mongodb.(*Connection).createTagNameAndExpiredIndex"
file="/go/src/acl-list/mongodb/mongodb.go:643"
time="2020-04-23T14:13:25Z" level=error msg="index creation failed:
(IndexKeySpecsConflict) Index must have unique name.The existing
index: { v: 2, unique: true, key: { mask: 1, port: 1, ip: 1 }, name:
\"ip_mask_port_idx\", ns: \"blacklist.acllist\" } has the same name as
the requested index: { v: 2, unique: true, key: { ip: 1, mask: 1,
port: 1 }, name: \"ip_mask_port_idx\", ns: \"blacklist.acllist\" }"
func="acl-list/mongodb.(*Connection).createFindOneItemIndex"
file="/go/src/acl-list/mongodb/mongodb.go:668"
time="2020-04-23T14:13:25Z" level=info msg="Indexes are created"
func=main.getDBConnection file="/go/src/acl-list/main.go:93"
time="2020-04-23T14:13:25Z" level=info msg="Start processing loop"
func=main.getDBConnection file="/go/src/acl-list/main.go:96"
time="2020-04-23T14:13:25Z" level=info msg="Start processing items
inserting" func="acl-list/mongodb.(*Connection).processInsertItems"
file="/go/src/acl-list/mongodb/mongodb.go:136"
```

В журнале событий контейнера с экземпляром сервиса хранения основных списков за последнее время не должно быть систематических ошибок.

3. Проверить, что экземпляр сервиса создания основных списков находится в рабочем состоянии (STATUS – Running). Для этого следует отправить команду **`kubectl get pods -n scos | grep имя_экземпляра_сервиса`**.

```
operator@operator-pc# kubectl get pods -n scos |grep scos-acl-creator-
from-cache-black-v4
NAME                                  READY   STATUS    RESTARTS
AGE
scos-acl-creator-from-cache-black-v4   1/1     Running   0
12d
```

4. Проверить, что экземпляр сервиса хранения основных списков и экземпляр его базы данных находятся в рабочем состоянии (STATUS

– Running). Для этого следует отправить команду **kubectl get pods --namespace=scos | grep имя_экземпляра_сервиса**.

```
operator@operator-pc# kubectl get pods --namespace=scos | grep scos-
acl-list-black-v4

NAME                                READY   STATUS    RESTARTS   AGE
scos-acl-list-black-v4     2/2     Running   0          6d4h
scos-acl-list-black-v4-db  2/2     Running   0          8d
```

5. Проверить текущие параметры конфигурации экземпляра сервиса создания основных списков с помощью команды **kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].env".**

```
operator@operator-pc# kubectl get pod scos-acl-creator-from-cache-
black-v4 --namespace=scos -o json | jq ".spec.containers[].env"
[
  {
    "name": "ACLLISTHOST",
    "value": "server.scos.ru"
  },
  {
    "name": "ACLLISTPORT",
    "value": "30642"
  },
  {
    "name": "SOURCE_TYPE",
    "value": "ACL_LIST"
  },
  {
    "name": "ACL_LIST_SOURCE_HOST",
    "value": "server.scos.ru "
  },
  {
    "name": "ACL_LIST_SOURCE_PORT",
    "value": "30643"
  }
]
```

В конфигурации экземпляра сервиса создания основных списков должны быть указаны правильные значения следующих параметров:

– **ACLLISTHOST** и **ACLLISTPORT**. Эти параметры определяют сетевую связность с сервисом хранения основных списков. Значение **ACLLISTHOST** должно содержать правильное DNS-имя или IP-адрес сервиса хранения основных списков, а значение **ACLLISTPORT** – правильный номер порта сервиса хранения основных списков. Правильные значения этих параметров можно узнать следующим способом:

Шаг 1. Узнать номер порта и тип сетевой службы для экземпляра сервиса хранения основных списков с помощью команды

kubectl get services --namespace=scos | grep scos-acl-list-black-v4.

```
operator@operator-pc# kubectl get services --namespace=scos | grep
scos-acl-list-black-v4
NAME                        TYPE        CLUSTER-IP    EXTERNAL-IP
PORT(S)            AGE
scos-acl-list-black-v4    NodePort    10.43.117.56  <none>
30642:30642/TCP    9d
scos-acl-list-black-v4-db ClusterIP   10.43.36.248  <none>
27017/TCP,9216/TCP 9d
```

Для экземпляра сервиса хранения основных списков используется тип сетевой службы **NodePort** и TCP-порт 30642. Сетевая служба **NodePort** обеспечивает возможность доступа к экземпляру сервиса из-за пределов кластера.

Шаг 2. Узнать IP-адрес экземпляра сервиса хранения основных списков, который будет равен любому IP-адресу узла кластера. Узнать IP-адреса всех узлов кластера можно командой **kubectl get nodes -o jsonpath='{ $.items[*].status.addresses[?(@.type=="InternalIP")].address }'**.

```
operator@operator-pc# kubectl get nodes -o jsonpath='{
$.items[*].status.addresses[?(@.type=="InternalIP")].address }'
192.168.100.10 192.168.100.11 192.168.100.12 192.168.100.13%
Вместо IP-адреса можно использовать DNS-имя одного из узлов кластера.
nslookup 192.168.100.10
Server:    192.168.100.2
Address:   192.168.100.2#53

10.100.168.192.in-addr.arpa     name = server.scos.ru.
```

В приведённом выше примере правильными значениями параметров **ACLLISTHOST** и **ACLLISTPORT** экземпляра сервиса создания основных списков являются, соответственно, **server.scos.ru** (или **192.168.100.10**) и **30642**.

– **ACL_LIST_SOURCE_HOST** и **ACL_LIST_SOURCE_PORT**. Эти параметры определяют сетевую связность с сервисом хранения промежуточных данных. Значение **ACL_LIST_SOURCE_HOST** должно содержать правильное DNS-имя или IP-адрес сервиса хранения промежуточных данных, а значение **ACL_LIST_SOURCE_PORT** – правильный номер порта сервиса хранения промежуточных данных. Правильные значения этих параметров можно узнать следующим способом:

Шаг 1. Узнать номер порта и тип сетевой службы для экземпляра сервиса хранения промежуточных данных с помощью команды **`kubectl get services --namespace=scos | grep scos-acl-list-cache-black-v4`**.

```
operator@operator-pc# kubectl get services -n scos | grep scos-acl-
list-cache-black-v4
NAME                              TYPE      CLUSTER-IP    EXTERNAL-IP
PORT(S)         AGE
scos-acl-list-cache-black-v4    NodePort  10.43.217.12  <none>
30643:30643/TCP     9d
scos-acl-list-cache-black-v4-db ClusterIP 10.43.245.44  <none>
9000/TCP            9d
```

Для экземпляра сервиса хранения промежуточных данных используется тип сетевой службы **NodePort** и TCP-порт 30643. Сетевая служба **NodePort** обеспечивает возможность доступа к экземпляру сервиса из-за пределов кластера.

Шаг 2. Узнать IP-адрес экземпляра сервиса хранения промежуточных данных, который будет равен любому IP-адресу узла кластера. Узнать IP-адреса всех узлов кластера можно командой **`kubectl get nodes -o jsonpath='{ $.items[*].status.addresses[?(@.type=="InternalIP")].address }'.`**

```
operator@operator-pc# kubectl get nodes -o jsonpath='{
$.items[*].status.addresses[?(@.type=="InternalIP")].address }'
192.168.100.10 192.168.100.11 192.168.100.12 192.168.100.13%
Вместо IP-адреса можно использовать DNS-имя одного из узлов кластера.
nslookup 192.168.100.10
Server:    192.168.100.2
Address:   192.168.100.2#53

10.100.168.192.in-addr.arpa     name = server.scos.ru.
```

В приведённом выше примере правильными значениями параметров **ACLLISTHOST** и **ACLLISTPORT** экземпляра сервиса создания основных списков являются, соответственно, **server.scos.ru** (или **192.168.100.10**) и **30643**.

– **SOURCE_TYPE.** Значение данного параметра должно быть по умолчанию **ACL_LIST**.

При необходимости следует задать правильные значения параметров экземпляра сервиса создания основных списков в файле со значениями чарта (**values.yaml**) и обновить сервис с помощью установщика пакетов **helm**, используя ключевое слово **`--upgrade`**.

6. Проверить текущие параметры конфигурации экземпляра сервиса хранения основных списков командой **`kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].env"`**.

```
operator@operator-pc# kubectl get pod scos-acl-list-black-v4 --
namespace=scos -o json | jq ".spec.containers[].env"
[
  {
    "name": "DBUSERNAME",
    "value": "list"
  },
  {
    "name": "DBPASSWORD",
    "value": "mongodb-password"
  },
  {
    "name": "DBDATABASE",
    "value": "blacklist"
  },
  {
    "name": "IP_TYPE",
    "value": "IPv4"
  },
  {
    "name": "DB_TYPE",
    "value": "MONGODB"
  },
  {
    "name": "DBHOST",
    "value": "scos-acl-list-black-v4-db"
  }
]
[
  {
    "name": "GRPC_PROXY",
    "value": "http://server.scos.ru:30273/acl/black/v4"
  }
]
```

В конфигурации экземпляра сервиса хранения основных списков должны быть заданы правильные значения следующих параметров:

- **DBUSERNAME.** Параметр определяет имя пользователя для доступа к базе данных экземпляра сервиса хранения основных списков. Значение данного параметра должно соответствовать значению в конфигурации экземпляра базы данных для данного экземпляра сервиса хранения основных списков;
- **DBPASSWORD.** Параметр определяет пароль для доступа к базе данных экземпляра сервиса хранения основных списков. Значение данного параметра должно соответствовать значению в конфигурации экземпляра базы данных для данного экземпляра сервиса хранения основных списков;
- **DBDATABASE.** Параметр определяет имя базы данных экземпляра сервиса хранения основных списков. Значение данного параметра должно соответствовать значению в конфигурации экземпляра базы данных для данного экземпляра сервиса хранения основных списков;

– **IP_TYPE.** Параметр определяет тип адресов, данные о которых хранятся в экземпляре хранения основных списков. Параметр может принимать два значения: IPv4 и IPv6;
– **DB_TYPE.** Параметр определяет тип базы данных (документо-ориентированная СУБД). Значение по умолчанию должно быть **MONGODB**;
– **DBHOST.** Параметр определяет сетевую связность между экземпляром сервиса хранения основных списков и его экземпляром базы данных. Значение данного параметра должно содержать правильное DNS-имя сервиса базы данных для данного экземпляра сервиса. Правильное значение данного параметра можно узнать с помощью команды **`kubectl get services --namespace=scos | grep scos-acl-list-black-v4`**.

```
operator@operator-pc# kubectl get services --namespace=scos | grep
scos-acl-list-black-v4-db
NAME                        TYPE        CLUSTER-IP    EXTERNAL-IP
PORT(S)             AGE
scos-acl-list-black-v4-db  ClusterIP  10.43.36.248  <none>
27017/TCP,9216/TCP  9d
```

При необходимости следует задать правильные значения параметров экземпляра сервиса хранения основных списков в файле со значениями чарта (**values.yaml**) и обновить сервис с помощью установщика пакетов **helm**, используя ключевое слово **`--upgrade`**.

7. Проверить сетевую связность между сервисом хранения промежуточных данных и сервисом создания основных списков. Для этого необходимо зайти в оболочку экземпляра сервиса создания основных списков с помощью команды **`kubectl exec -it scos-acl-creator-from-cache-black-v4 --namespace=scos  sh`**

```
operator@operator-pc# kubectl exec -ti scos-acl-creator-from-cache-
black --namespace=scos  sh
/srv #
```

и проверить доступность экземпляра сервиса хранения промежуточных данных по прослушивающему порту с помощью общедоступных системных инструментов: **nmap**, **netcat**.

```
/srv # nmap server.scos.ru -p 30643
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 20:06 UTC
Nmap scan report for server.scos.ru (192.168.100.10)
Host is up (0.00025s latency).
rDNS record for 192.168.100.2: scos-dns-unbound.scos-
dns.svc.cluster.local

PORT      STATE SERVICE
```

```
30643/tcp open  unknown
Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

При необходимости следует устранить нарушение сетевой связности между экземпляром сервиса создания основных списков и экземпляром сервиса хранения промежуточных данных.

8. Проверить сетевую связность между экземпляром сервиса создания основных списков и экземпляром сервиса хранения основных списков. Для этого необходимо зайти в оболочку экземпляра сервиса создания основных списков с помощью команды **`kubectl exec -it scos-acl-creator-from-cache-black-v4 --namespace=scos sh`**

```
operator@operator-pc# kubectl exec -ti scos-acl-creator-from-cache-
black --namespace=scos  sh
/srv #
```

и проверить доступность экземпляра сервиса хранения основных списков по прослушивающему порту с помощью общедоступных системных инструментов: **nmap**, **netcat**.

```
operator@operator-pc# kubectl exec -ti scos-acl-creator-from-cache-
black  --namespace=scos sh
```

```
/srv # nmap server.scos.ru -p 30642
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 20:06 UTC
Nmap scan report for server.scos.ru (192.168.100.10)
Host is up (0.00025s latency).
rDNS record for 192.168.100.2: scos-dns-unbound.scos-
dns.svc.cluster.local

PORT      STATE SERVICE
30642/tcp open  unknown
Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

При необходимости следует устранить нарушение сетевой связности между экземпляром сервиса создания основных списков и экземпляром сервиса хранения основных списков.

9. Проверить версию образа работающего экземпляра сервиса создания основных списков командой `kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].image"`.

```
operator@operator-pc# kubectl get pod scos-acl-creator-from-cache-
black-v4--namespace=scos -o json | jq ".spec.containers[].image"
"hub.scos.ru/acl-creator:v2.1.0"
```

Следует проверить в файле с информацией о чарте (**chart.yaml**) соответствие версии образа для экземпляра сервиса создания основных списков, при необходимости изменить версию образа и обновить экземпляр сервиса с помощью установщика пакетов **helm**, используя ключевое слово **`--upgrade`**.

10. Проверить версию образа работающего экземпляра сервиса хранения основных списков командой kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].image".

```
operator@operator-pc# kubectl get pod scos-acl-list-black-v4  --
namespace=scos -o json | jq ".spec.containers[].image"
"hub.scos.ru/acl-list:v2.6.0"
```

Следует проверить в файле с информацией о чарте (**chart.yaml**) соответствие версии образа для экземпляра сервиса хранения основных списков, при необходимости изменить версию образа и обновить экземпляр сервиса с помощью установщика пакетов **helm**, используя ключевое слово **`--upgrade`**.

11.4. Не работают сервисы сравнения списков и сервисы выгрузки списков на узлы фильтрации

Сервисы сравнения списков отвечают за сопоставление записей в основных чёрных и белых списках, передачу результирующих списков сервисам выгрузки и предоставление доступа к чёрным спискам оборудованию первого уровня фильтрации (EcoFilter). Сервисы выгрузки списков обеспечивают отправку актуальных серых и чёрных списков фильтрации на оборудование балансировки трафика в составе второго уровня фильтрации (EcoHighway).

11.4.1. Описание проблемы

После правильной настройки оборудования фильтрации и начала отправки журналов пользовательских блокировок, настройки и запуска сервисов коллектора журналов блокировок и базы данных основные сервисы генерации списков и сервисы хранения промежуточных данных работают правильно, сервисы создания основных списков и сервисы хранения основных списков также работают правильно, но значения счётчиков метрик сервисов сравнения списков и/или сервисов доставки списков не соответствуют критериям успешной работы, а именно:

– значения метрик сервисов сравнения списков не приходят по запросу:

```
operator@operator-pc# kubectl port-forward scos-acl-differ-v4 --
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"acl_differ_black_list_len|acl_differ_white_list_len|acl_differ_diffed
_list_len"

Handling connection for 5000
E0429 15:45:55.670279 24686 portforward.go:400] an error occurred
forwarding 5000 -> 2112: error forwarding port 2112 to pod
d5f6b9c8c1e8f64646886f04bf696ab770ce45b4c7b7f117f4068d0c44a90ac5, uid
: exit status 1: 2020/04/29 12:45:55 socat[27411] E connect(5, AF=2
127.0.0.1:2112, 16): Connection refused
```

– значения метрик сервисов сравнения списков продолжительное время остаются равными нулю, однако при этом:

  - записи журналов от EcoFilter поступают в сервис коллектора журнала блокировок;
  - базовые сервисы генерации списков получают записи от сервиса базы данных и передают сгенерированные записи в сервис хранения промежуточных данных;
  - сервисы создания основных списков отправляют записи в сервисы хранения основных списков.

```
operator@operator-pc# kubectl port-forward scos-acl-differ-v4--
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"acl_differ_black_list_len|acl_differ_white_list_len|acl_differ_diffed
_list_len"

# HELP acl_differ_black_list_len Number of records received from black
list.
# TYPE acl_differ_black_list_len gauge
acl_differ_black_list_len{black_exclude="rkn,static",black_include="tl
s,hrandom",white_exclude="",white_include="static"} 0
acl_differ_black_list_len{black_exclude="rkn,static,tls,hrandom",black
_include="random",white_exclude="tag3,tag4",white_include="tag1,tag2"}
0
acl_differ_black_list_len{black_exclude="static,rknip",black_include="
tls",white_exclude="",white_include="static"} 0
acl_differ_black_list_len{black_exclude="static,rknip,tls",black_inclu
de="hrandom",white_exclude="",white_include="static"} 0
# HELP acl_differ_diffed_list_len Number of records sended to
destination.
# TYPE acl_differ_diffed_list_len gauge
acl_differ_diffed_list_len{black_exclude="rkn,static",black_include="t
ls,hrandom",white_exclude="",white_include="static"} 0
acl_differ_diffed_list_len{black_exclude="rkn,static,tls,hrandom",blac
k_include="random",white_exclude="tag3,tag4",white_include="tag1,tag2"
} 0
```

```
acl_differ_diffed_list_len{black_exclude="static,rknip",black_include=
"tls",white_exclude="",white_include="static"} 0
acl_differ_diffed_list_len{black_exclude="static,rknip,tls",black_incl
ude="hrandom",white_exclude="",white_include="static"} 0
# HELP acl_differ_white_list_len Number of records received from white
list.
# TYPE acl_differ_white_list_len gauge
acl_differ_white_list_len{black_exclude="rkn,static",black_include="tl
s,hrandom",white_exclude="",white_include="static"} 0
acl_differ_white_list_len{black_exclude="rkn,static,tls,hrandom",black
_include="random",white_exclude="tag3,tag4",white_include="tag1,tag2"}
0
acl_differ_white_list_len{black_exclude="static,rknip",black_include="
tls",white_exclude="",white_include="static"} 0
acl_differ_white_list_len{black_exclude="static,rknip,tls",black_inclu
de="hrandom",white_exclude="",white_include="static"} 0
```

– значения метрик сервисов доставки списков на узлы фильтрации не приходят по запросу:

```
operator@operator-pc# kubectl port-forward scos-acl-manager --
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112

operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"success|timeout|fail|tls_ex_bb|acl_manager_redis_collect_table_second
s|acl_manager_redis_table_size|acl_manager_source_fetch_seconds|acl_ma
nager_source_received_records|acl_manager_time_from_start_of_update_se
conds|acl_manager_vrf_update_time_seconds"

E0429 15:45:55.670279 24686 portforward.go:400] an error occurred
forwarding 5000 -> 2112: error forwarding port 2112 to pod
d5f6b9c8c1e8f64646886f04bf696ab770ce45b4c7b7f117f4068d0c44a90ac5, uid
: exit status 1: 2020/04/29 12:45:55 socat[27411] E connect(5, AF=2
127.0.0.1:2112, 16): Connection refused
```

– значения метрик сервисов доставки списков продолжительное время остаются равными нулю, однако при этом:

  - записи журналов от EcoFilter поступают в сервис коллектора журнала блокировок;
  - базовые сервисы генерации списков получают записи от сервиса базы данных и передают сгенерированные записи в сервис хранения промежуточных данных;
  - сервисы создания основных списков отправляют записи в сервисы хранения основных списков;
  - сервисы сравнения основных списков формируют итоговые списки для сервисов доставки списков.

```
operator@operator-pc# kubectl port-forward scos-acl-manager --
namespace=scos 5000:2112
Forwarding from 127.0.0.1:5000 -> 2112
Forwarding from [::1]:5000 -> 2112
```

```
operator@operator-pc# curl -s 127.0.0.1:5000/metrics | grep -E
"acl_manager_source_received_records
"success|timeout|fail|tls_ex_bb|acl_manager_redis_collect_table_second
s|acl_manager_redis_table_size|acl_manager_source_fetch_seconds|acl_ma
nager_source_received_records|acl_manager_time_from_start_of_update_se
conds|acl_manager_vrf_update_time_seconds"

acl_manager_iterations_total{result="fail",storage="redis"} 0
acl_manager_iterations_total{result="success",storage="redis"} 812
acl_manager_iterations_total{result="timeout",storage="redis"} 0
acl_manager_redis_collect_table_seconds{vrf="block"} 0.000502846
acl_manager_redis_collect_table_seconds{vrf="telegram_tls_ex_bb"}
0.002702649
acl_manager_redis_table_size{vrf="block"} 0
acl_manager_redis_table_size{vrf="telegram_tls_ex_bb"} 0
acl_manager_source_fetch_seconds{ip_type="v4",storage="redis",vrf="blo
ck"} 0
acl_manager_source_fetch_seconds{ip_type="v4",storage="redis",vrf="tel
egram_tls_ex_bb"} 0
acl_manager_source_fetch_seconds{ip_type="v6",storage="redis",vrf="blo
ck"} 0
acl_manager_source_fetch_seconds{ip_type="v6",storage="redis",vrf="tel
egram_tls_ex_bb"} 0
acl_manager_source_received_records{ip_type="v4",storage="redis",vrf="
block"} 0
acl_manager_source_received_records{ip_type="v4",storage="redis",vrf="
telegram_tls_ex_bb"} 0
acl_manager_source_received_records{ip_type="v6",storage="redis",vrf="
block"} 0
acl_manager_source_received_records{ip_type="v6",storage="redis",vrf="
telegram_tls_ex_bb"} 0
acl_manager_time_from_start_of_update_seconds{storage="redis",vrf="blo
ck"} 0
acl_manager_time_from_start_of_update_seconds{storage="redis",vrf="tel
egram_tls_ex_bb"} 0
acl_manager_vrf_update_time_seconds{storage="redis",vrf="block"}
0.086317706
acl_manager_vrf_update_time_seconds{storage="redis",vrf="telegram_tls_
ex_bb"} 0.113159704
```

Описание метрик

– **acl_differ_black_list_len** (для сервиса сравнения списков)

Метрика, счётчик которой показывает количество записей, поступивших в сервис сравнения списков от сервиса хранения основных (чёрных) списков. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при проведении проверок сервиса сравнения списков необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса хранения основных (чёрных) списков и сервиса доставки списков на узлы фильтрации значение счётчика этой метрики в

момент получения записей от сервиса хранения основных списков должно быть больше 0.

– **acl_differ_white_list_len** (для сервиса сравнения списков)

Метрика, счётчик которой показывает количество записей, поступивших в сервис сравнения списков от сервиса хранения основных (белых) списков. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при проведении проверок сервиса сравнения списков необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса хранения основных (белых) списков и сервиса доставки списков на узлы фильтрации значение счётчика этой метрики в момент получения записей от сервиса хранения основных списков должно быть больше 0.

– **acl_differ_diffed_list_len** (для сервиса сравнения списков)

Метрика, счётчик которой показывает количество записей, успешно обработанных сервисом сравнения списков и успешно отправленных сервису доставки списков на узлы фильтрации. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при проведении проверок сервиса сравнения списков необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса хранения основных (чёрных и белых) списков и сервиса доставки списков значение счётчика этой метрики в момент получения записей от сервиса хранения основных списков должно быть больше 0.

– **acl_manager_source_received_records** (для сервиса доставки списков)

Метрика, счётчик которой показывает количество записей в отдельных таблицах vrf, полученных сервисом доставки списков от сервисов сравнения или хранения списков. Тип данного счётчика – gauge, и он может принимать в разные моменты времени как нулевое, так и ненулевое значение, поэтому при проведении проверок сервиса доставки списков на узлы фильтрации необходимо отправить несколько запросов этой метрики за короткий интервал времени.

При правильной работе сервиса сравнения или хранения списков и сервиса доставки списков значение счётчика этой метрики в момент получения записей от сервиса сравнения или хранения списков должно быть больше 0.

Сервисы сравнения или сервисы хранения основных списков и сервисы доставки списков состоят из N экземпляров. Каждый экземпляр сервиса сравнения или хранения основных списков генерирует результат сравнения чёрного и белого списка в соответствии с версией протокола IP-адресов (IPv4 или IPv6). Экземпляры сервисов сравнения или хранения основных списков также предоставляют доступ к чёрным спискам для оборудования первого уровня фильтрации.

Каждый экземпляр сервиса доставки списков запрашивает и получает от сервиса сравнения или хранения основных списков набор записей определённого типа и транслирует их в сервис acl_manager для распространения на узлах фильтрации второго уровня.

При выявлении проблемы в работе сервисов сравнения или хранения основных списков или сервисов доставки списков необходимо определить, какие именно экземпляры работают неправильно.

11.4.2. Возможные причины неисправной работы

Возможные причины неисправной работы:

- экземпляр сервиса сравнения или хранения списков находится в нерабочем состоянии, и данные не передаются в сервис доставки списков;
- неправильная конфигурация экземпляра сервиса сравнения или хранения списков, экземпляра сервиса доставки списков;
- нарушение сетевой связности между экземпляром сервиса сравнения или хранения списков и экземпляром сервиса доставки списков;
- нарушение сетевой связности между экземпляром сервиса доставки списков и сервисом acl_manager;
- несоответствующая версия образа для контейнера с экземпляром сервиса сравнения или хранения списков или для контейнера с экземпляром сервиса доставки списков.

11.4.3. Порядок выявления и устранения причин

1.Проверить журнал событий для контейнера с экземпляром сервиса сравнения списков командой kubectl logs имя_экземпляра_сервиса -n scos -c имя основного_контейнера_для_экземпляра_сервиса.

```
operator@operator-pc# kubectl logs scos-acl-differ-v4  -n scos -c acl-
differ-v4
tion refused\"" func="acl-differ/api.(*GRPCServer).List"
file="/go/src/acl-differ/api/grpc.go:139"
time="2020-04-20T16:06:07Z" level=error msg="can't get blacklist, err
rpc error: code = Unavailable desc = all SubConns are in
TransientFailure, latest connection error: connection error: desc =
\"transport: Error while dialing dial tcp 10.210.9.250:30642: connect:
connec
tion refused\"" func="acl-differ/api.(*GRPCServer).List"
file="/go/src/acl-differ/api/grpc.go:139"
time="2020-04-20T16:06:12Z" level=error msg="can't get blacklist, err
rpc error: code = Unavailable desc = all SubConns are in
TransientFailure, latest connection error: connection error: desc =
\"transport: Error while dialing dial tcp 10.210.9.250:30642: connect:
connec
tion refused\"" func="acl-differ/api.(*GRPCServer).List"
file="/go/src/acl-differ/api/grpc.go:139"
time="2020-04-21T05:00:48Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:01:18Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:01:49Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:02:19Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:02:49Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:03:19Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:03:37Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
time="2020-04-21T05:03:50Z" level=info msg="&{178.47.103.241 ffffffff
[{29673 29673}]} &{178.47.96.0 ffffe000 [{1 65535}]} []" func=acl-
differ/differ.SubtractItemLists file="/go/src/acl-
differ/differ/differ.go:160"
```

В журнале событий контейнера с экземпляром сервиса сравнения списков за последнее время не должно быть систематически повторяющихся ошибок.

2. Проверить журнал событий для контейнера с экземпляром сервиса доставки списков командой `kubectl logs имя_экземпляра_сервиса -n scos`.

```
operator@operator-pc# kubectl logs scos-acl-manager -n scos
/app/internal/log/log.go:42"
time="2022-12-20T18:54:01Z" level=info msg="options which can be set
via env: \n  METRICS_BIND_ADDRESS string\n    \tIP:PORT to bind
metrics HTTP socket (default \":2112\")\n  LOG_LEVEL string\n    \tlog
level: trace, debug, info, warn, error, fatal, panic (default
\"info\")\n  GRPC_BIND_ADDRESS string\n    \tIP:PORT to bind to
(default \":9000\")\n  CONFIG_PATH string\n    \tpath to the config
file for MULTIPLE mode (default \"config/config.yml\")\n
REDIS_ADDRESS string\n    \tRedis DB address for write operations\n
REDIS_READONLY_ADDRESS string\n    \tRedis DB address for read
operations\n  REDIS_DB_INDEX uint\n    \tRedis DB Index (default
\"1\")\n  REDIS_KEY_DELIMETER string\n    \tDelimiter of key parts in
Redis DB (default \";\")"
time="2022-12-20T18:54:02Z" level=info msg="Watch config in /config"
func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/config.(*Config).StartConfigReloader"
file="/opt/app/internal/config/config.go:209"
time="2022-12-20T18:54:02Z" level=info msg="Connecting to \"redis-acl-
manager-rw:6379\", db index 1." func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/writer.(*Writer).Open"
file="/opt/app/internal/redis/writer/writer.go:45"
time="2022-12-20T18:54:02Z" level=info msg="Connected to redis
\"redis-acl-manager-rw:6379\"." func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/writer.(*Writer).Open"
file="/opt/app/internal/redis/writer/writer.go:59"
time="2022-12-20T18:54:02Z" level=info msg="Connecting to \"redis-acl-
manager-ro:6379\", db index 1." func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/reader.(*Reader).Open"
file="/opt/app/internal/redis/reader/reader.go:39"
time="2022-12-20T18:54:02Z" level=info msg="Connected to redis
\"redis-acl-manager-ro:6379\"." func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/reader.(*Reader).Open"
file="/opt/app/internal/redis/reader/reader.go:53"
time="2022-12-20T18:54:02Z" level=info msg="Redis pipeline created."
func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/writer.(*Writer).AutoFlushPipeline"
file="/opt/app/internal/redis/writer/writer.go:73"
time="2022-12-20T18:54:02Z" level=info msg="Redis pipeline created."
func="gitlab.rdp.ru/tt/acl-
manager/v2/internal/redis/writer.(*Writer).AutoFlushPipeline"
file="/opt/app/internal/redis/writer/writer.go:73"
[GIN-debug] [WARNING] Creating an Engine instance with the Logger and
Recovery middleware already attached.

[GIN-debug] [WARNING] Running in "debug" mode. Switch to "release"
mode in production.
 - using env:     export GIN_MODE=release
 - using code:    gin.SetMode(gin.ReleaseMode)
```

```
[GIN-debug] GET    /api/v1/list              --> gitlab.rdp.ru/tt/acl-
manager/v2/internal/api/rest.(*Web).listHandler-fm (3 handlers)
[GIN-debug] POST   /api/v1/reload            --> gitlab.rdp.ru/tt/acl-
manager/v2/internal/api/rest.(*Web).reloadConfig-fm (3 handlers)
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="Start VRF processing"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
```

```
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
opened" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
```

```
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream opened"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=block
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=redirect
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=appleappstore_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=viber_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: stream closed"
func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=telegram_ex_bb
time="2022-12-20T18:54:03Z" level=info msg="acl-differ: connection
closed" func=gitlab.rdp.ru/tt/acl-manager/v2/internal/log.Info
file="/opt/app/internal/log/log.go:42" idx=0 vrf=whatsapp_ex_bb
```

В журнале событий контейнера с экземпляром сервиса доставки списков за последнее время не должно быть систематически повторяющихся ошибок.

3. Проверить текущие параметры конфигурации экземпляра сервиса сравнения списков командой kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].env"

```
operator@operator-pc# kubectl get pod scos-acl-differ-v4  --
namespace=scos -o json | jq ".spec.containers[].env"
[
  {
    "name": "BLACKLISTRPC",
    "value": "server.scos.ru:30642"
  },
  {
    "name": "WHITELISTRPC",
    "value": " server.scos.ru:30641"
  }
]
[
  {
    "name": "ACL_DIFFER_ADDR",
    "value": "localhost:9090"
  },
  {
    "name": "LISTS",
    "value": "/lists/lists.yaml"
  }
]
```

В конфигурации экземпляра сервиса сравнения списков должны быть заданы правильные значения следующих параметров:

– **BLACKLISTGRPC** и **WHITELISTGRPC**. Данные параметры определяют сетевую связность с сервисами хранения основных списков. Значение **BLACKLISTGRPC** должно содержать правильное DNS-имя (или IP-адрес) и номер порта экземпляра сервиса хранения основного чёрного списка, а значение **WHITELISTGRPC** – правильное DNS-имя (или IP-адрес) и номер порта экземпляра сервиса хранения основного белого списка.

Правильные значения этих параметров можно узнать следующим способом:

Шаг 1. Узнать номер порта и тип сетевой службы для экземпляра сервиса хранения основного черного списка командой

```
kubectl get services --namespace=scos | grep scos-acl-list-black-v4.
```

```
operator@operator-pc# kubectl get services --namespace=scos | grep scos-acl-
list-black-v4
NAME                      TYPE     CLUSTER-IP  EXTERNAL-IP PORT(S)          AGE
scos-acl-list-black-v4    NodePort 10.43.17.16 <none>      30642:30642/TCP  9d
scos-acl-list-black-v4-db ClusterIP 10.43.36.28 <none> 27017/TCP,9216/TCP  9d
```

Экземпляр сервиса хранения основных списков использует тип сетевой службы **NodePort** и TCP-порт 30642. Сетевая служба типа **NodePort** обеспечивает возможность доступа к экземпляру сервиса из-за пределов кластера.

Шаг 2. Узнать IP-адрес экземпляра сервиса хранения основных списков, который будет равен любому IP-адресу узла кластера. Узнать IP-адреса всех узлов кластера можно с помощью команды `kubectl get nodes -o jsonpath='{ $.items[*].status.addresses[?(@.type=="InternalIP")].address }{"\n"}'`.

```
operator@operator-pc# kubectl get nodes -o jsonpath='{
$.items[*].status.addresses[?(@.type=="InternalIP")].address }{"\n"}'
192.168.100.10 192.168.100.11 192.168.100.12 192.168.100.13
```

Вместо IP-адреса можно использовать DNS-имя одного из узлов кластера.

```
nslookup 192.168.100.10
Server:         192.168.100.2
Address:   192.168.100.2#53

10.100.168.192.in-addr.arpa     name = server.scos.ru.
```

Шаг 3. Повторить шаги 1 и 2 для получения правильного значения для основного сервиса хранения белого списка.

В приведённом выше примере правильными значениями параметров **BLACKLISTGRPC** и **WHITELISTGRPC** экземпляра сервиса сравнения списков являются:

- server.scos.ru (или 192.168.100.10) и 30642 для параметра BLACKLISTGRPC;
- server.scos.ru (или 192.168.100.10) и 30641 для параметра WHITELISTGRPC.
  - **ACL_DIFFER_ADDR.** Значение данного параметра должно быть по умолчанию **localhost:9090**.
  - **LISTS.** Значение данного параметра должно быть по умолчанию **/lists/lists.yaml**.

При необходимости следует задать правильные значения параметров экземпляра сервиса сравнения списков в файле со значениями чарта (**values.yaml**) и обновить сервис с помощью установщика пакетов **helm**, используя ключевое слово `--upgrade`.

4. Проверить текущие параметры конфигурации экземпляра сервиса доставки списков командой kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[].env".

```
operator@operator-pc# kubectl get pod scos-acl-manager --
namespace=scos -o json | jq ".spec.containers[].env"
[
  {
    "name": "WORKING_MODE",
    "value": "MULTIPLE"
  },
  {
    "name": "CONFIG_PATH",
    "value": "/config/config.yml"
  },
  {
    "name": "LOG_LEVEL",
    "value": "info"
  },
  {
    "name": "REDIS_ADDRESS",
    "value": "acl-manager-redis-master:6379"
  },
  {
    "name": "REDIS_DB_INDEX",
    "value": "1"
  },
  {
    "name": "REDIS_READONLY_ADDRESS",
    "value": "acl-manager-redis-replicas:6379"
  },
  {
    "name": "GRPC_BIND_ADDRESS",
    "value": ":50051"
  }
]
```

В конфигурации экземпляра сервиса доставки списков должны быть заданы правильные значения следующих параметров:

- **WORKING_MODE**. Режим работы сервиса доставки списков. Значение «Multiple» указывает, что данный экземпляр сервиса доставляет множество списков;
- **CONFIG_PATH**. Путь к конфигурационным параметрам для работы сервиса доставки в режиме «Multiple».
- **LOG_LEVEL**. Уровень логирования (значения: trace, debug, info, warn, error, fatal, panic);
- **REDIS_ADDRESS**. Указывает адрес, куда сервис доставки кэширует записи, полученные от сервисов хранения и сравнения списков;
- **REDIS_DB_INDEX**. Индекс БД для кэша;
- **REDIS_READONLY_ADDRESS**. Указывает, откуда сервис доставки читает кэшированные записи, ранее полученные от сервисов хранения и сравнения списков.
- **GRPC_BIND_ADDRESS**. Определяет порт, на который сервис доставки отправляет записи по протоколу GRPC.

5. Проверить сетевую связность между сервисом сравнения хранения списков. Для этого необходимо:

– зайти в оболочку экземпляра сервиса сравнения списков с помощью команды **kubectl exec -it scos-acl-differ-v4 --namespace=scos sh**

```
operator@operator-pc# kubectl exec -it scos-acl-differ-v4  --
namespace=scos  sh
/srv #
```

– проверить доступность экземпляра сервиса хранения чёрных списков по прослушиваемому порту с помощью общедоступных системных инструментов: **nmap**, **netcat**:

```
/srv # nmap server.scos.ru -p 30642
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 20:06 UTC
Nmap scan report for server.scos.ru (192.168.100.10)
Host is up (0.00025s latency).
rDNS record for 192.168.100.2: scos-dns-unbound.scos-
dns.svc.cluster.local

PORT      STATE SERVICE
30642/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

– проверить доступность экземпляра сервиса хранения белых списков по прослушиваемому порту с помощью общедоступных системных инструментов (**nmap**, **netcat**):

```
/srv # nmap server.scos.ru -p 30641

Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 20:06 UTC
Nmap scan report for server.scos.ru (192.168.100.10)
Host is up (0.00025s latency).
rDNS record for 192.168.100.2: scos-dns-unbound.scos-
dns.svc.cluster.local

PORT      STATE SERVICE
30641/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

При необходимости устранить нарушение сетевой связности между экземпляром сервиса сравнения списков и экземплярами сервисов хранения списков.

6. Проверить сетевую связность между сервисом доставки списков и сервисом сравнения списков. Для этого необходимо:

– зайти в оболочку экземпляра сервиса доставки списков на узлы фильтрации с помощью команды **kubectl exec -it scos-acl-**

manager --namespace=scos sh

```
operator@operator-pc# kubectl exec -it scos-acl-manager  --
namespace=scos  sh
/srv #
```

– проверить доступность экземпляра сервиса сравнения списков по прослушиваемому порту с помощью общедоступных системных инструментов (**nmap**, **netcat**):

```
/srv # nmap server.scos.ru -p 30940
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-29 20:06 UTC
Nmap scan report for server.scos.ru (192.168.100.10)
Host is up (0.00025s latency).
rDNS record for 192.168.100.2: scos-dns-unbound.scos-
dns.svc.cluster.local

PORT      STATE SERVICE
30940/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

При необходимости устранить нарушение сетевой связности между экземпляром сервиса доставки списков и сервисом сравнения списков.

7. Проверить версию образа работающего экземпляра сервиса сравнения списков командой **kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[0].image"**.

```
operator@operator-pc# kubectl get pod scos-acl-differ-v4 --
namespace=scos -o json | jq ".spec.containers[0].image"
"hub.scos.ru/acl-differ:v3.0.1"
```

Проверить в файле с информацией о чарте (**chart.yaml**) соответствие версии образа для экземпляра сервиса сравнения списков, при необходимости изменить версию образа и обновить экземпляр сервиса с помощью установщика пакетов **helm**, используя ключевое слово **--upgrade**.

8. Проверить версию образа работающего экземпляра сервиса доставки списков командой kubectl get pod имя_экземпляра_сервиса --namespace=scos -o json | jq ".spec.containers[0].image".

```
operator@operator-pc# kubectl get pod scos-acl-manager --
namespace=scos -o json | jq ".spec.containers[0].image"
"hub.scos.ru/acl-manager:v2.4.2"
```

Проверить в файле с информацией о чарте (**chart.yaml**) соответствие версии образа для экземпляра сервиса доставки списков, при необходимости изменить версию образа и обновить экземпляр сервиса с помощью установщика пакетов **helm**, используя ключевое слово **--upgrade**.

ПРИЛОЖЕНИЕ А. СХЕМА ОСНОВНЫХ КОМПОНЕНТОВ СИСТЕМЫ С ПО ECODPIOS-DC

Схема основных компонентов системы сбора и анализа статистических данных с ПО EcoDPIOS-DC приведена на рисунке ниже (см. Рисунок А.1).

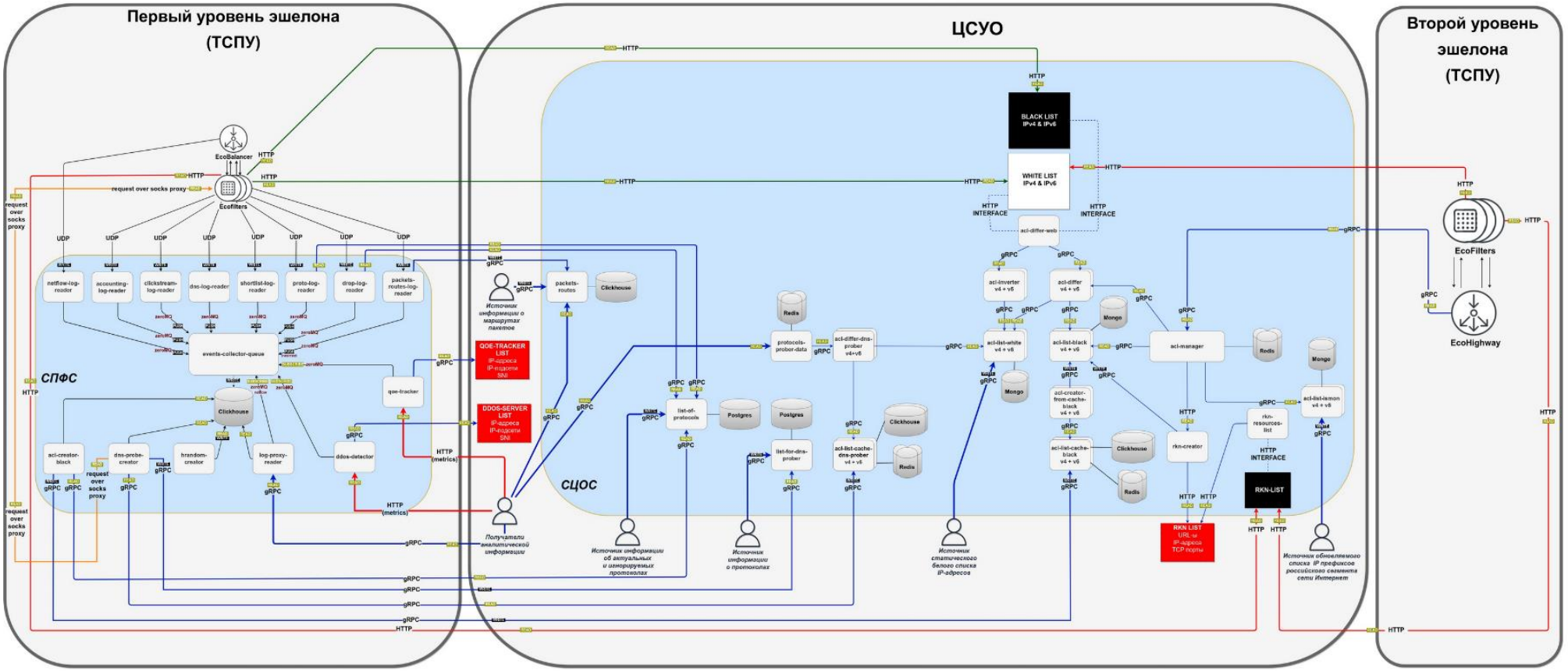

Рисунок А. 1 - Схема основных компонентов системы сбора и анализа статистических данных с ПО EcoDPIOS-DC