ЦЕНТРАЛЬНЫЙ ПУЛЬТ

Руководство администратора

v.3.11.85 - 21.12.2022

РОССИНКО

Оглавление

1 Введение

Центральный Пульт - это платформа для визуализации и мониторинга работы сети, оборудования, приложений и служб.

1.1 Описание документа

Настоящий документ является руководством для администрирования автоматизированной системы "Центральный Пульт" и предназначен для конкретизации задач и функций должностных лиц организации (предприятия, фирмы), планирующих и осуществляющих сбор, хранение, передачу и анализ данных по объектам мониторинга с применением системы.

В документе приведены основные функции администратора, архитектура системы и её модулей, алгоритм создания учётных записей, порядок установки прав доступа пользователей и другие сведения, необходимые для управления АС "Центральный Пульт".

1.2 Назначение системы

Система предназначена для визуализации и мониторинга различных объектов. Центральный Пульт нацелен на упрощение сбора данных, ускорение их анализа, визуализации результатов и беспрерывного хранения.

Автоматизации подвергаются следующие функциональные возможности процесса мониторинга:

- процесс обработки данных;
- хранение оригинальных значений;
- обеспечение анализа информации;
- управление объектами мониторинга;
- уведомление пользователей о состояниях объектов;
- исправление аварийных ситуаций;
- преобразование данных в компактный вид;
- экспорт данных;
- удаление устаревших данных.

2 Системные требования

Система может быть установлена на выделенных аппаратных или виртуальных мощностях.

Для стабильной работы клиента системы необходим web-браузер Google Chrome версии не ниже 58.0.

2.1 Системные требования к серверу

Для работы сервера системы требуется следующая конфигурация:

- 64-bit OS;
- CPU - 4 cores;
- RAM - 8 GB;
- HDD - 72 GB.

Гарантируется корректная работа инсталляционного скрипта сервера на следующих ОС:

- Ubuntu Linux 20.04;
- Ubuntu Linux 18.04.

Установка системы также возможна на операционные системы:

- РЕД ОС,
- Red Hat Enterprise Linux,
- Astra Linux,
- Debian.

"Центральный Пульт" может быть запущен и на других дистрибутивах Linux. Свяжитесь с нами welcome@saymon.info, если вы не нашли нужную вам операционную систему среди перечисленных.

Серверная часть системы может быть поставлена в виде готового образа виртуальной машины или Docker-контейнера. Ссылки можно получить в службе технической поддержки по адресу welcome@saymon.info.

2.2 Системные требования к агенту

Агенты обладают кросс-платформенной совместимостью и могут быть установлены на различные операционные системы, например:

- Ubuntu Linux;
- Red Hat Enterprise Linux / CentOS Linux;
- Raspberry Pi;
- Mac OS X;
- Windows.

Требованием к операционным системам является поддержка Java SE 6, 7 и 8.

Рекомендуемая конфигурация для работы агентов системы:

- OS with Java 6/7/8 support;
- CPU — 2 GHz single core;
- RAM — 1 GB;
- HDD — OS + 2 GB

2.3 Сетевые доступы

Порты по умолчанию, необходимые для корректной работы Центрального Пульта (часть 1 из 2):

Порт	Протокол	Сервис	Инициатор соединения	Место назначения	Комментарий
22	TCP	SSH	Сотрудники технической поддержки	Все серверы платформы	Удалённый доступ к серверам для настройки и оказания технической поддержки.
80	TCP	HTTP	Пользователи Центрального Пульта	Серверы с сервисами Центрального Пульта	Доступ к web-интерфейсу платформы. Рекомендуется перенаправление на 443 порт.
162	UDP	SNMP Trap	Оборудование, отправляющее SNMP Trap	Серверы с агентами	Получение SNMP Trap от оборудования. По умолчанию на каждом сервере с сервисами Центрального Пульта установлен агент, который получает SNMP Trap'ы.
443	TCP	HTTPS	Пользователи Центрального Пульта	Серверы с сервисами Центрального Пульта	Доступ к web-интерфейсу Центрального Пульта с шифрованием трафика.
3001	TCP	Wetty	Пользователи Центрального Пульта	Оборудование в инфраструктуре заказчика	Доступ к терминалам устройств через SSH и Telnet.
4242	TCP	OpenTSDB	Серверы с сервисами Центрального Пульта	Серверы с OpenTSDB	Доступ к базе данных хранения временных рядов для построения графиков. Необходим при инсталляции OpenTSDB на отдельных серверах.
6379	TCP	Redis	Серверы с сервисами Центрального Пульта	Серверы с Redis	Доступ к кэшу данных. Необходим при инсталляции Redis на отдельных серверах.

Порты по умолчанию, необходимые для корректной работы Центрального Пульта (часть 2 из 2):

Порт	Протокол	Сервис	Инициатор соединения	Место назначения	Комментарий
8090	TCP	SAYMON REST API	Пользователи, интеграционные модули и расширения	Серверы с сервисами Центрального Пульта	Взаимодействие с системой и данными через REST API.
8091	TCP	SAYMON Notification Actor	Пользователи Центрального Пульта	Серверы с сервисами Центрального Пульта	Необходим для быстрого обновления данных в web-интерфейсе и мобильных клиентах через WebSocket.
8092	TCP	SAYMON Resource Server	Пользователи, интеграционные модули и расширения	Серверы с сервисами Центрального Пульта	Отображение в web-интерфейсе приложенных к объектам документов и фоновых изображений.
9092	TCP	Kafka	Серверы с сервисами Центрального Пульта	Серверы с Kafka	Доступ к шине данных. Необходим при инсталляции системы на нескольких серверах и при использовании агента для сбора данных.
27017	TCP	MongoDB	Серверы с сервисами Центрального Пульта	Серверы с MongoDB	Доступ к базе данных основной информации об объектах и пользователях. Необходим при инсталляции MongoDB на отдельных серверах.

3 Принципы функционирования системы

Программное обеспечение платформы "Центральный Пульт" имеет открытые API-интерфейсы, которые обеспечивают информационную совместимость системы и возможность интеграции с другими автоматизированными системами.

3.1 Архитектура системы

Платформа имеет клиент-серверную архитектуру и включает в себя три основных уровня (Рис. 3.1.1):

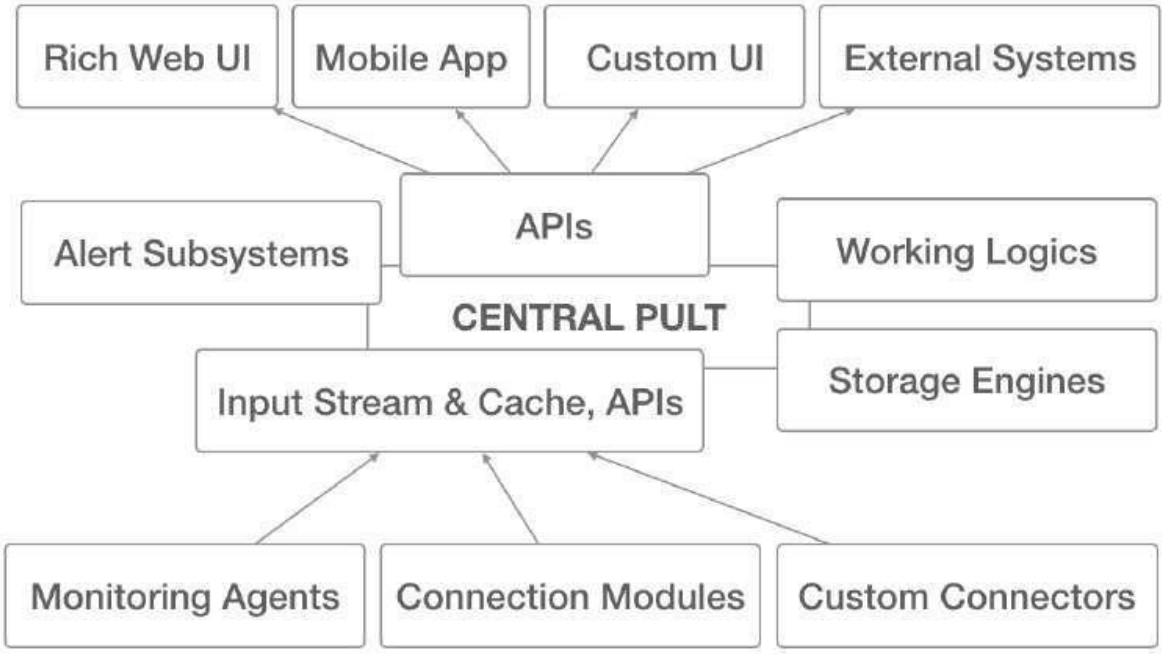


Рис. 3.1.1. *Общая архитектура платформы "Центральный Пульт"*

Нижний уровень предназначен для сбора данных и осуществления управления над полученной информацией, которая периодически отправляется в Input Stream & Cache для дальнейшей обработки и анализа.

Данный слой включает в себя три элемента:

- Monitoring Agents - агенты системы, собирающие с узлов мониторинга информацию;
- Connection Modules - готовое и переиспользуемое решение для сбора информации без агента;
- Custom Connectors - разработанные для клиента интерфейсы, осуществляющие мониторинг объектов без агента.

На втором уровне хранится и обрабатывается полученная от агентов и прочих интерфейсов информация, а затем передаётся клиентам в требуемом для каждого из решений виде.

Второй слой подразделяется на:

- Working Logics (бизнес-логики) - совокупность правил, принципов, зависимостей поведения объектов, на основе которых обрабатываются поступившие с нижнего слоя данные и сохраняются в Storage Engines для анализа ситуаций в настоящем и прошлом и построения математически обоснованных прогнозов в будущем.
- Storage Engines - отвечает за хранение данных. Storage Engines реализовано в виде:
  - SQL - формирует запросы, описывающиеся, какую информацию из Storage Engines необходимо получить, пути решения определяются автоматически;
  - noSQL - обеспечивает гибкость и согласованность системы, благодаря гарантированному завершению запроса;
  - TimeSeries - специализированный компонент управления базой данных временных рядов, что позволяет хранить данные с высокой скважностью.
- Alert Subsystems - каналы уведомлений, по которым осуществляется информирование пользователей о смене состояний объектов, разрыве соединений и других нестандартных ситуациях. При соответствующей настройке Центральный Пульт может:
  - отправлять email-уведомления,
  - запускать программы или скрипты с параметрами,
  - отправлять сообщения в Telegram,
  - показывать визуальные уведомления в браузере, сопровождающиеся звуком,
  - создавать задачи в JIRA,
  - запускать операции,
  - отправлять SMS,
  - совершать голосовые вызовы.
- APIs - относятся к категории REpresentational State Transfer (REST), что позволяет выполнять RESTful-операции на добавление, чтение, изменение и удаление информации для облачной учётной записи или инсталляции на сервере.
- Input Stream & Cache - хранилище данных, поступивших в сервер платформы с нижнего уровня, и их хранение. Cache реализован в виде сетевого журналируемого хранилища Redis. Взаимодействие между агентами и сервером платформы осуществляется при помощи Kafka:
  - собирает данные с ниже располагающегося слоя,
  - хранит данные у себя в распределённом хранилище по топикам,
  - передаёт данные серверу по запросу.

Верхний уровень отвечает за визуализацию полученных и обработанных данных, а также осуществление операций над ними конечным пользователем. В качестве средства интеграции приложений используются открытые API-интерфейсы:

- Rich Web UI - web-интерфейс платформы, является основным средством работы с системой для конечного пользователя, где при наличии определённых прав возможны изменения как всей структура, так и отдельного объекта.
- Mobile App - мобильные приложения для операционных систем Android и iOS.
- Custom UI - кастомизированные интерфейсы, созданные под специальную бизнес-задачу или проект. Функциональные возможности позволяют вносить изменения - добавлять, удалять и редактировать объекты – аналогично Rich Web UI. Web-интерфейс имеет уникальное отображение, согласно заявленным требованиям.
- External Systems - внешние системы для отображения или сбора данных, полученных средствами мониторинга платформы "Центральный Пульт".

3.2 Описание функционирования системы и её частей

С помощью API в web-интерфейсе системы возможно реализовать большинство операций, например:

- получение списка объектов;
- получение текущего статуса объектов и истории их состояний;
- запись данных в объекты без использования агентов;
- изменение свойств объектов;
- работа с авариями;
- создание классов с добавлением проверок.

Система состоит из следующих логических подсистем:

- Server - централизованный сервер, на котором хранится и анализируется информация, полученная от агентов, а затем отдаётся клиенту.
- СУБД (MongoDB, OpenTSDB) - совокупность программных средств, предназначенных для создания, использования и управления базами данных.
- Agent - множество агентов системы, установленных на узлах инфраструктуры и собирающих информацию по ним. Полученные данные периодически отправляются в кэш и затем анализируются сервером.
- Клиент - тонкий web-клиент системы и клиенты для мобильных платформ Android и iOS.

4 Задачи администратора

Глава содержит информацию об основных задачах, возникающих в процессе администрирования платформы.

4.1 Настройка параметров работы системы

Раздел настройки - один из самых важных разделов, предназначенный для администратора системы. Этот раздел наполнен советами о том, как настроить Центральный Пульт для мониторинга вашей среды, начиная настройкой сервера для получения необходимой информации и заканчивая просмотром данных, настройкой оповещений и удалённых команд, выполняемых в случае возникновения проблем.

4.1.1 Установка агента

Перед началом работы с платформой "Центральный Пульт" необходимо выполнить следующие действия:

1. Получить актуальную версию агента одним из способов:
   - зайти на сайт платформы на страницу "Загрузки" saymon.info/support_ru/downloads_en/ и скачать подходящую под ОС сборку агента;
   - перенести с CD-ROM, USB-накопителя или другого носителя, на котором поставляется платформа "Центральный Пульт", подходящую под пользовательскую ОС сборку агента на компьютер, сервер или устройство, на котором планируется осуществлять сбор данных.
2. Установить и настроить агента согласно дальнейшим инструкциям.

⚠ *При установке агента на Linux в системе автоматически создаётся пользователь **saymon**, от имени которого будет работать агент.*

Если в системе уже есть пользователь с таким логином, работа агента будет проходить от его имени.

4.1.1.1 Автоинсталляция

Чтобы приступить к началу работы и настройке мониторинга, необходимо выполнить следующие действия:

1. Открыть web-интерфейс платформы "Центральный Пульт".
2. Ввести логин и пароль учётной записи с правами на управление объектами, свойствами и документами.

> *Назначение прав пользователей описано в подразделе 4.7.4 "Назначение пользователям прав доступа" на стр. 115.*

3. Создать объект-агент, для этого нажать кнопку 

на панели инструментов, ввести имя объекта и выбрать класс **Saymon Agent**.
4. Навести курсор на созданный объект и нажать на появившуюся иконку - Настройки агента (Рис. 4.1.1.1.1):

Рис. 4.1.1.1.1. *Мониторинговый агент*

5. В появившемся окне скопировать ссылку из строки "Команда для установки агента" (Рис. 4.1.1.1.2):

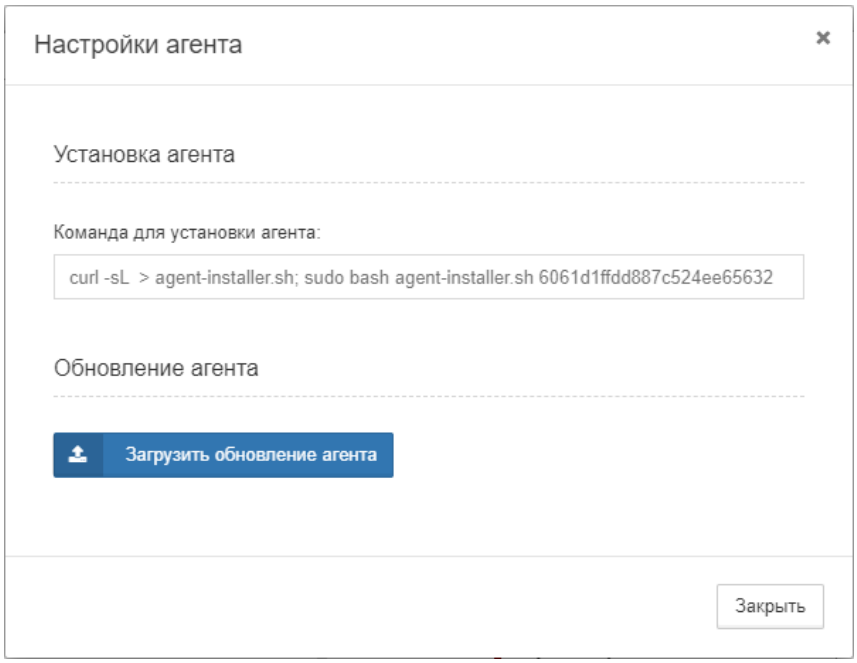


Рис. 4.1.1.1.2. *Команда для установки агента*

6. Выполнить эту команду в терминале на необходимом сервере.

> *Номер, который дал сервер - последние 24 символа команды - совпадает с ID агента.*

В течение 30 секунд агент скачивается и устанавливается. Если установка выполнена корректно, состояние агента в web-интерфейсе будет отражено зелёным цветом: 

.

4.1.1.2 Linux

Инструкция применима к операционным системам с менеджерами служб "systemd", "upstart" и "init.d", например:

- Ubuntu Linux;
- Red Hat Enterprise Linux;
- CentOS Linux;
- Debian 8 "Jessie" (для пользователей Raspberry Pi).

Если вы используете другого менеджера, пишите нам на care@saymon.info.

> ⚠ *При установке агента в системе автоматически создаётся пользователь **saymon**, от имени которого будет работать агент.*
>
> *Если в системе уже есть пользователь с таким логином, работа агента будет проходить от его имени.*

Для установки агента на Linux необходимо:

На хосте без доступа в Интернет:

1. Скачать sh-скрипт установщика агента для Linux 64bit или Linux 32bit.
2. Перенести скрипт на нужный хост в домашнюю директорию пользователя.
3. Сделать скрипт исполняемым:

   64bit:

   ```
   sudo chmod +x saymon-agent-rl-linux-x64-jre-installer.sh
   ```

   32bit:

   ```
   sudo chmod +x saymon-agent-rl-linux-i586-jre-installer.sh
   ```

4. Запустить скрипт:

   64bit:

   ```
   sudo ./saymon-agent-rl-linux-x64-jre-installer.sh
   ```

   32bit:

   ```
   sudo ./saymon-agent-rl-linux-i586-jre-installer.sh
   ```

5. Выполнить дальнейшие инструкции в терминале.

На хосте с доступом в Интернет:

1. Выполнить однострочник:

   64bit:

   ```
   curl https://saymon.info/downloads/saymon-agent-rl-linux-x64-jre-installer.sh –o saymon-agent-instal.sh ; chmod +x saymon-agent-instal.sh ; sudo ./saymon-agent-instal.sh
   ```

   32bit:

   ```
   curl https://saymon.info/downloads/saymon-agent-rl-linux-i586-jre-installer.sh –o saymon-agent-instal.sh ; chmod +x saymon-agent-instal.sh ; sudo ./saymon-agent-instal.sh
   ```

2. Выполнить дальнейшие инструкции в терминале.

Конфигурация агента выполняется в файле `/opt/saymon-agent/conf/agent.properties`.

4.1.1.3 Mac OS X

Для установки агента на Mac OS X необходимо:

1. Создать системного пользователя "Saymon agent", из-под которого будет запускаться агент.
2. Скачать архив Mac OS X 64bit и распаковать его в папку `/opt`.
3. Отредактировать файл конфигурации агента `saymon-agent/conf/agent.properties`.
4. Создать папку для хранения log-файлов:
   ```
   sudo mkdir saymon-agent/log && sudo chown -R saymon:staff saymon-agent
   ```
5. При необходимости включить централизованное логирование агента через SYSLOG в файле `saymon-agent/conf/logback.xml`.
6. Сделать файл агента исполняемым:
   ```
   sudo chmod +x saymon-agent/saymon-agent.sh
   ```
7. Запустить агента:
   ```
   cd saymon-agent && sudo -u saymon ./saymon-agent.sh
   ```

4.1.1.4 Windows

Для установки агента на Windows необходимо:

1. Скачать инсталлятор для Windows 64bit или Windows 32bit.
2. Запустить скачанный файл от имени администратора.
3. Выбрать папку установки (например, `C:\Program Files (x86)\SAYMON Agent`) и нажать кнопку **Install**.
4. После завершения установки нажать кнопку **Close**.
5. В папке установки агента из шага 3 отредактировать файл конфигурации агента `...\conf\agent.properties`.
6. Запустить службу **SAYMON Agent**.

4.1.1.5 Wiren Board 6

Wiren Board 6 - это универсальный контроллер для автоматизации с открытым ПО на базе Linux. Предназначен для домашней и промышленной автоматизации и мониторинга: опроса датчиков и счетчиков, использования в качестве УСПД, в системах АСКУЭ, для замены ПЛК, а также в системах "умного дома".

Для установки агента на контроллер автоматизации Wiren Board 6 необходимо:

1. Скачать архив с агентом.
2. Распаковать архив в папку `/opt/saymon-agent`.
3. Скачать архив с JDK (потребуется регистрация на сайте).
4. Распаковать архив в папку `temp`, оттуда полностью скопировать директорию `jre` в папку `/opt/saymon-agent`.
5. Выполнить следующие команды:

```
useradd -M -r -s /bin/false -K MAIL_DIR=/dev/null saymon
chown -R saymon:saymon /opt/saymon-agent
cp /opt/saymon-agent/systemd/* /etc/systemd/system
systemctl enable saymon-agent
service saymon-agent start
```

6. При необходимости отредактировать файл конфигурации агента `/opt/saymon-agent/conf/agent.properties`.

Благодарим за оказанную помощь пользователя svdu с форума компании Wiren Board.

4.1.2 Конфигурация агента

Конфигурация агента выполняется в файле **.../saymon-agent/conf/agent.properties**.

Для применения изменений необходимо перезапустить службу **saymon-agent**.

Общие параметры конфигурационного файла агента (часть 1 из 2):

Параметр	Описание
agent.discoveryEnabled	Включает автоматический поиск агентом сетевых устройств; • false - автоматический поиск выключен, • true - автоматический поиск включен. По умолчанию - **false** - автоматический поиск выключен. *Задать родительский объект для обнаруженных устройств можно с помощью параметра **discovery_parent_id** в секции **Server** конфигурационного файла сервера* **/etc/saymon/saymon-server.conf**.
agent.id	Уникальный идентификатор объекта класса **Saymon Agent** в web-интерфейсе системы. *При указании **agent.id=0** на сервере будет создан новый объект класса **Saymon Agent**. По умолчанию агенты создаются в корневом объекте, задать родительский объект для создаваемых агентов можно с помощью параметра **agent_registration_parent_id** в секции **Server** конфигурационного файла сервера* **/etc/saymon/saymon-server.conf**. *ID созданного объекта будет передан агенту. На хосте с агентом будет создан файл* **.../saymon-agent/storage/registration.json**, *где будет записан данный идентификатор.* *Чтобы заново инициировать процедуру получения ID, к примеру, если необходимо подключить агента к другому серверу, достаточно удалить файл* **registration.json** *и перезапустить агента.*
agent. optimizedThreadManagement	Включает режим оптимизации использовани потоков агентом; • false - режим оптимизации выключен, • true - режим оптимизации включен. По умолчанию - **false** - режим оптимизации выключен.

Общие параметры конфигурационного файла агента (часть 2 из 2):

Параметр	Описание
agent.scriptsEnabled	Включает выполнение агентом скриптов с указанным текстом; • false - режим оптимизации выключен, • true - режим оптимизации включен. По умолчанию - **true** - выполнение скриптов включено.
agent.snmpTrapEnabled	Позволяет использовать агента в качестве получателя трапов; • false - получение трапов агентом выключено, • true - получение трапов агентом включено. По умолчанию - **false** - получение трапов выключено.
agent.snmpTrapListenPort	Порт для получения трапов. По умолчанию - **1162**.
agent. snmpTrapReceiverThreadPoolSize	Количество одновременных потоков для получения трапов. По умолчанию - **4**.
metrics. buffer.in-memory.capacity	Включает буфер для сохранения данных от агента при отсутствии связи между агентом и сервером. Значение параметра - количество наборов данных от агента (результатов проверок), которые будут храниться в буфере до восстановления связи с сервером. После восстановления связи все данные из буфера пересылаются на сервер, и буфер очищается. При переполнении буфера очередные наборы данных заменяют наиболее старые из имеющихся в буфере. По умолчанию буфер выключен. *Функционал поддерживается агентами версии 4.5.72 и выше.*
server.host	Адрес сервера Kafka. По умолчанию - **127.0.0.1**.
server.port	Порт Kafka, по которому осуществляется подключение агентов к серверу. По умолчанию - **9092**.

4.1.2.1 Подключение к Kafka Cluster

При использовании Kafka Cluster необходимо также использовать дополнительные параметры подключения, начинающиеся на **kafka.**

Все возможные параметры представлены в документации по Kafka.

Пример:

```
...
kafka.bootstrap.servers=localhost:9092
kafka.security.protocol=SASL_SSL
kafka.ssl.truststore.location=/opt/kafka/kafka.client.truststore.jks
kafka.ssl.truststore.password=saymon
kafka.sasl.mechanism=PLAIN
kafka.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule
required \
username="user" \
password="password"
...
```

kafka.bootstrap.servers - список пар хост:порт, использующихся для установки первоначального соединения с кластером Kafka.

⚠ *При использовании Kafka Cluster поле **server.host** не должно быть пустым, там необходимо указать адрес одного из **kafka.bootstrap.servers**.*

Параметры конфигурационного файла агента для авторизации и шифрования:

Параметр	Описание
kafka.sasl.jaas.config	Параметры контекста входа в систему для SASL-соединений в формате, используемом конфигурационными файлами JAAS.
kafka.sasl.mechanism	Механизм SASL, используемый для клиентских подключений.
kafka.security.protocol	Протокол, используемый для взаимодействия с брокерами. Возможные значения: • PLAINTEXT, • SSL, • SASL_PLAINTEXT, • SASL_SSL.
kafka.ssl.truststore.location	Расположение файла **truststore**.
kafka.ssl.truststore.password	Пароль для файла **truststore**. Если пароль не задан, сконфигурированный файл также будет использоваться, но без проверки целостности.

4.1.3 Конфигурация сервера

Конфигурация сервера системы выполняется в файле `/etc/saymon/saymon-server.conf`.

Для применения изменений необходимо перезапустить службу **saymon-server**:

```
sudo service saymon-server restart
```

Параметры конфигурационного файла сервера (часть 1 из 21):

Секция/Параметр	Описание

Cache	В этой секции задаются настройки сервера Redis.
	`"cache": {`
	`  "host": "127.0.0.1",`
	`  "port": 6379`
	`}`

cache.auth_pass	Пароль для доступа к серверу Redis.
cache.host	Адрес сервера Redis. По умолчанию - **"127.0.0.1"**.
cache.port	Порт сервера Redis. По умолчанию - **6379**.

cache.cluster	В этой подсекции задаются настройки при использовании Redis Cluster.
	`"cache": {`
	`  "cluster": {`
	`  "nodes": [`
	`  {`
	`  "port": 7000,`
	`  "host": "127.0.0.1"`
	`  },`
	`  {`
	`  "port": 7001,`
	`  "host": "127.0.0.1"`
	`  }`
	`  ],`
	`  "options": {}`
	`  }`
	`}`
	*Если задействована эта подсекция, система использует указанные здесь настройки подключения, параметры **cache.host** и **cache.port** игнорируются.*

cache.cluster.options	Дополнительные параметры подключения к кластеру.
cache.cluster.nodes	Серверы кластера, к которым необходимо подключиться.
cache.cluster.nodes.host	Адрес сервера кластера.
cache.cluster.nodes.port	Порт сервера кластера.

Параметры конфигурационного файла сервера (часть 2 из 21):

Секция/Параметр	Описание

Custom_scripts	В этой секции задаются настройки пользовательских скриптов, срабатывающих при создании и удалении объектов.
	`"custom_scripts": {`
	`"entity_triggers_path": "/opt/saymon-extensions/entity-triggers-path"`
	`}`

custom_scripts. entity_triggers_path	Полный путь до папки с файлами скриптов.

DB	В этом разделе задаются настройки подключения к базе данных для хранения объектной модели. В качестве базы данных используется MongoDB.
	`"db" : {`
	`"mongodb": {`
	`"url": "mongodb://localhost:27017/saymon?w=1"`
	`}`
	`}`

db.mongodb	Данные для подключения к MongoDB.
db.mongodb.url	Адрес сервера MongoDB и дополнительные параметры подключения. По умолчанию - **"mongodb://localhost:27017/saymon?w=1"**. *При использовании кластера из нескольких хостов их адреса указываются через запятую в произвольном порядке:* *"url": "mongodb://host01:27017,host02:27017,host03:27017/saymon?w=1"* Все возможные дополнительные параметры URL представлены в документации по MongoDB.

Elasticsearch	В этой секции задаются настройки для работы с Elasticsearch.
	`"elasticsearch": {`
	`"node": "http://127.0.0.1:9200",`
	`"auth": {`
	`"username": "elastic",`
	`"password": "ypezA1ZoVv46fJh8Abj4"`
	`},`
	`"index_pattern": "myindex*",`
	`"request_timeout": 30000,`
	`"hits_per_request": 1000`
	`}`

elasticsearch. hits_per_request	Максимальное число записей, запрашиваемых в одном обращении к elasticsearch. По умолчанию - **1000**. *При повышении значения параметра снижается число запросов к Elasticsearch, но увеличивается расход памяти и время блокировки потока на сервере Центрального Пульта при обработке ответа.*

Параметры конфигурационного файла сервера (часть 3 из 21):

Секция/Параметр	Описание
elasticsearch. index_pattern	Индекс Elasticsearch, в котором происходит поиск.
elasticsearch.node	IP-адрес инстанса Elasticsearch.
elasticsearch. request_timeout	Время ожидания ответа от сервера elasticsearch (в миллисекундах). По умолчанию - **30000** - 30 секунд.
elasticsearch. auth	Авторизационные данные пользователя elasticsearch.
elasticsearch.auth. password	Пароль пользователя elasticsearch.
elasticsearch.auth. username	Логин пользователя elasticsearch.

Kafka	В этой секции задаются настройки подключения к брокеру Kafka.
	`"kafka" : {`
	`  "host" : "localhost",`
	`  "port" : 9092,`
	`  "requestTimeout" : 30000,`
	`  "fetch_latest" : false,`
	`  "logLevel" : 2`
	`}`
	Помимо перечисленных здесь параметров, возможно также использовать дополнительные параметры подключения.

kafka.brokers	Пул адресов для подключения при использовании Kafka Cluster.
	`"kafka" : {`
	`  ...`
	`  "brokers": [`
	`  "192.168.1.10:9092",`
	`  "192.168.1.11:9092"`
	`  ],`
	`  ...`
	`}`
	При указании данного параметра система игнорирует параметры **kafka.host** и **kafka.port**.
	При использовании Kafka Cluster необходимо дополнительно настроить конфигурационный файл агента. Подробнее в подразделе 4.1.2 "Конфигурация агента" на стр. 25 и части 4.1.2.1 "Подключение к Kafka Cluster" на стр. 27.

Параметры конфигурационного файла сервера (часть 4 из 21):

Секция/Параметр	Описание
kafka.fetch_latest	Данный параметр определяет правило обработки сообщений, поступивших в Kafka при выключенной системе: • false - обработать накопленные данные, • true - игнорировать накопленные данные. По умолчанию - **false**.
kafka.host	Адрес брокера Kafka. По умолчанию - **"localhost"**.
kafka.logLevel	Уровень логирования Kafka; 0 - NOTHING, 1 - ERROR, 2 - WARN, 4 - INFO, 5 - DEBUG. По умолчанию - **2** - WARN.
kafka.port	Порт брокера Kafka. По умолчанию - **9092**.
kafka. requestTimeout	Время на ожидание ответа от брокера (в миллисекундах). По умолчанию - **30000** - 30 секунд.

kafka.ssl	Настройки шифрования. Ключ и сертификат можно указывать как путь к файлу:
	`"kafka" : {`
	`  ...`
	`  "ssl": {`
	`  "certFile": "/opt/kafka/for-saymon-server/cert.pem",`
	`  "keyFile": "/opt/kafka/for-saymon-server/key.pem"`
	`  },`
	`  ...`
	`}`

kafka.sasl	Настройки аутентификации. Параметры зависят от механизма аутентификации, используемом в Kafka.
	Пример настроек для механизма PLAIN/SCRAM:
	`"kafka" : {`
	`...`
	`  "sasl": {`
	`  "mechanism": "plain",`
	`  "username": "user",`
	`  "password": "password"`
	`  },`
	`...`
	`}`

Параметры конфигурационного файла сервера (часть 5 из 21):

Секция/Параметр	Описание
kafka.sasl. mechanism	Механизм аутентификации.
Keycloak	В этом разделе задаются параметры для взаимодействия с сервером Keycloak для авторизации пользователей. **"keycloak": {** **"realm_certificate": "<данные из keycloak_realm_certificate>",** **"realm_certificate_file": "/etc/saymon/keycloak_realm_certificate",** **"config_file": "/etc/saymon/keycloak_backend.json"** **}** *Параметры в примере перечислены в порядке понижения приоритета при одновременном указании. Для работы с Keycloak достаточно указать один из параметров.*
keycloak.config_file	Путь к файлу конфигурации, сгенерированному сервером Keycloak для Backend-клиента.
keycloak. realm_certificate	Данные сертификата (Realm Certificate), которым сервер Keycloak подписывает пользовательские токены.
keycloak. realm_certificate_file	Путь к файлу с Keycloak Realm Certificate.
LDAP	В этой секции задаются параметры внешнего LDAP-сервера для авторизации. **"ldap" : {** **"url" : "ldaps://192.168.1.1:636",** **"suffix" : "dc=example,dc=com",** **"login" : "cn=admin,dc=example,dc=com",** **"pass" : "root",** **"allow_self_signed" : true,** **"create_user_for_existing_group_only": true,** **"import_non_existing_groups": false,** **"update_existing_groups": false** **}**
ldap.allow_ self_signed	Разрешает соединение по LDAPs с сервером, на котором установлен самоподписанный SSL-сертификат; • false - соединение запрещено, • true - соединение разрешено. По умолчанию - **false** - соединение запрещено.

Параметры конфигурационного файла сервера (часть 6 из 21):

Секция/Параметр	Описание
ldap.create_user_for_existing_group_only	Ограничивает авторизацию новых LDAP-пользователей только теми пользователями, у которых имя группы на LDAP-сервере совпадает с именем существующей (созданной ранее) группы в Центральном Пульте: • false - разрешена авторизация всех новых LDAP-пользователей, • true - разрешена авторизация только тех новых LDAP-пользователей, для которых заготовлены группы в Центральном пульте. По умолчанию - **false** - разрешена авторизация всех новых LDAP-пользователей. *На LDAP-пользователей, у которых уже есть учётная запись, применение данного параметра не распространяется.*
ldap.group_name_attribute	Атрибут, где хранятся имена групп пользователей. Указывается, если необходимо использовать не стандартный атрибут. Стандартный атрибут - "**cn**".
ldap.import_non_existing_groups	Разрешает автоматическое создание групп, в которых пользователь состоит на LDAP-сервере, но которых нет в Центральном Пульте: • false - при авторизации LDAP-пользователя в Центральном Пульте не создаются новые группы; • true - при авторизации LDAP-пользователя создаются все его группы в Центральном Пульте, пользователь автоматически добавляется в свои группы. По умолчанию - **false** - при авторизации LDAP-пользователей в Центральном Пульте не создаются новые группы.
ldap.login	Логин администратора LDAP.
ldap.login_attribute_name	Атрибут, где хранятся логины пользователей для входа в систему. Указывается, если необходимо использовать не стандартный атрибут. Стандартный атрибут - "**name**".
ldap.pass	Пароль администратора LDAP.
ldap.suffix	Корневой элемент (как правило, доменное имя организации).

Параметры конфигурационного файла сервера (часть 7 из 21):

Секция/Параметр	Описание
ldap.update_existing_groups	Разрешает обновлять группы пользователей, созданные в Центральном Пульте, до LDAP-групп при импорте групп пользователя из LDAP и совпадении имен групп: • false - при входе LDAP-пользователей обновления групп не происходит. • true - при входе LDAP-пользователя его группы в Центральном Пульте, совпадающие с группами в LDAP, обновляются до LDAP-групп. По умолчанию - **false** - при входе LDAP-пользователей обновления групп не происходит. Для обновленной группы отключается возможность смены имени и добавляется источник - LDAP.
ldap.url	Адрес LDAP-сервера. Поддерживаются протоколы **LDAP** и **LDAPs**.
Monitoring	В этой секции задаются настройки пользовательских сенсоров мониторинга. **"monitoring": { "custom_tasks_path": "/opt/saymon-agent/custom_tasks" }**
monitoring.custom_tasks_path	Полный путь до папки с файлами пользовательских проверок.
MQTT	В этой секции задаются настройки подключения к MQTT-брокеру. **"mqtt" : { "broker" : "mqtt://username:password@localhost:1883" }**
mqtt.broker	Адрес и порт брокера. По умолчанию - **"mqtt://localhost:1883"**. *Для аутентификации по имени пользователя и паролю нужно указать пользовательские данные перед адресом сервера.*
OpenTSDB	В этой секции задаются параметры доступа к OpenTSDB.
openTsdb.enabled	Запись исторических данных в OpenTSDB; • false - запись выключена, • true - запись включена. По умолчанию - **true** - запись включена.
openTsdb.host	Адрес хоста с OpenTSDB. По умолчанию - **"localhost"**.

Параметры конфигурационного файла сервера (часть 8 из 21):

Секция/Параметр	Описание
openTsdb.port	Порт OpenTSDB. По умолчанию - **4242**.
openTsdb.timeout	Время на ожидание ответа от OpenTSDB (в миллисекундах). По умолчанию - **15000** - 15 секунд.
Push_ notification	В этой секции задаются параметры push-уведомлений в мобильном приложении. Для работы с ними используется Firebase Cloud Messaging (FCM).
push_notification. disabled	Выключает уведомления; • true - уведомления выключены, • false - уведомления включены. По умолчанию - **true** - уведомления выключены.
push_notification. key_path	Путь к ключу авторизации сервера Центрального Пульта на сервере Firebase. По умолчанию - **"/etc/saymon/saymon-mobile-firebase-adminsdk.json"**.
push_notification. on_state_change	Включает отправку уведомления при изменении состояний на случай другого источника уведомлений - MQTT-сообщений. • true - отправка включена, • false - отправка выключена. По умолчанию - **true** - отправка включена.
push_notification. timeout	Время, через которое каждому пользователю отправляется уведомление (в миллисекундах). По умолчанию - **0** - задержки нет.
push_notification.url	URL, полученный пользователем от Firebase, для принятия содержимого уведомлений с сервера. По умолчанию - **"https://saymon-mobile.firebaseio.com"**.
Resource_server	В этой секции задаются параметры, связанные с хранением файлов, загруженных в Центральный Пульт.
resource_server.debug	Включает debug-режим для логирования в файл **/var/log/saymon/saymon-server.log**; • false - режим выключен, • true - режим включен. По умолчанию - **false** - режим выключен.

Параметры конфигурационного файла сервера (часть 9 из 21):

Секция/Параметр	Описание
resource_server. file_storage_dir	Путь к директории для хранения документов, прикрепляемых к объектам. По умолчанию - **"/var/saymon/resources"**.
resource_server. ip_address	Адрес Resource-сервера. По умолчанию - **"127.0.0.1"**.
resource_server.port	Порт Resource-сервера. По умолчанию - **8092**.
Rest_server	В этой секции задаются параметры REST-сервера.
rest_server.base_url	Путь к API. По умолчанию - **"/api"**.
rest_server. colorize_log	Включает цветную раскраску лога; • false - раскраска выключена, • true - раскраска включена. По умолчанию - **false** - раскраска выключена.
rest_server.debug	Включает debug-режим для логирования в файл **/var/log/saymon/saymon-server.log**; • false - режим выключен, • true - режим включен. По умолчанию - **false** - режим выключен.
rest_server.document_ download_url	URL к файлам, сохраненным в **$document_storage_dir**. По умолчанию - **"http://localhost/node/api/docs".**
rest_server.ip_address	Адрес хоста для запуска REST-сервера. По умолчанию - **"127.0.0.1"**.
rest_server. keepAliveTimeout	Время ожидания сервером следующего запроса до разрыва текущего соединения (в миллисекундах). По умолчанию - **5000** - 5 секунд.
rest_server. num_workers	Число процессов для загрузки данных. По умолчанию - **1**.
rest_server.port	Порт REST-сервера. По умолчанию - **8090**.
rest_server.public_url	Адрес для доступа к web-интерфейсу из уведомлений.

Параметры конфигурационного файла сервера (часть 10 из 21):

Секция/Параметр	Описание
rest_server.session.ttl	Время, по истечении которого происходит автоматическое завершение сессии при неактивности пользователя (в миллисекундах). По умолчанию - **3600000** - 1 час.
rest_server. snmp_mib_url	Адрес сервера трансляции SNMP OID в текстовый формат.
rest_server. update_download_url	Путь к файлу для обновления агента. По умолчанию - **"http://localhost/node/api/agents/update"**.
rest_server.limits	В этой подсекции задаются ограничения на число запросов к API. **"limits" : [** **{** **"__comment": "администратора не ограничиваем",** **"user": "5048c849d7b6e40593dfee71"** **},** **{** **"__comment": "запросы на states и classes не ограничиваем",** **"uri": ["/states", "/classes"]** **},** **{** **"__comment": "ограничиваем запросы на объекты",** **"uri": ["/objects"],** **"rate": "100/minute"** **},** **{** **"__comment": "ограничиваем авторизовавшихся пользователей с 127.0.0.1",** **"user": true,** **"ip": '127.0.0.1',** **"rate": "120/minute"** **},** **{** **"__comment": "ограничения для неавторизованных пользователей",** **"user": false,** **"rate": "10/minute"** **}** **]** По умолчанию ограничения не накладываются. *Для каждого запроса поиск правил осуществляется сверху вниз. Применяется только одно правило.*

Параметры конфигурационного файла сервера (часть 11 из 21):

Секция/Параметр	Описание

rest_server.limits.group	Группы пользователей, для которых устанавливается ограничение;
	• string|string[] - ID группы или групп, к которым применяется правило.
	Примеры:
	`"limits": [ { "group": "60e562277936944ebf158282", ... } ]`
	`"limits": [ { " group ": [ "60e562277936944ebf158282", "616406e1a3db5948cd815438" ], ... } ]`

rest_server.limits.ip	IP-адреса, для которых устанавливается ограничение:
	• true - свой лимит для каждого IP-адреса;
	• string|string[] - IP-адрес или список адресов, к которым применяется правило.
	Примеры:
	`"limits": [ { "ip": true, ... } ]`
	`"limits": [ { "ip": "127.0.0.1", ... } ]`
	`"limits": [ { "ip": [ "127.0.0.1", "192.168.1.2" ], ... } ]`

Параметры конфигурационного файла сервера (часть 12 из 21):

Секция/Параметр	Описание
rest_server.limits.rate	Количество запросов в единицу времени. Если не указано, то ограничение не накладывается. Задаётся в формате **количество_запросов/единица_измерения_времени**. Возможные единицы измерения времени: **y** - год, **Q** - квартал, **M** - месяц, **w** - неделя, **d** - день, **h** - час, **m**/**minute** - минута, **s** - секунда, **ms** - миллисекунда. Пример: `"limits": [ { "rate": 100/minute, ... } ]`

rest_server.limits.uri	REST-методы, к которым устанавливается ограничение;
	• true - свой лимит для каждого метода;
	• string|string[] - метод или список методов, к которым применяется правило.
	В качестве пути могут быть заданы любые API без приставки /node/api, например, **"/time"**.
	Примеры:
	`"limits": [ { "uri": true, ... } ]`
	`"limits": [ { "uri": "/ping", ... } ]`
	`"limits": [ { "uri": [ "/ping", "/node/api/objects/:id/stat" ], ... } ]`

Параметры конфигурационного файла сервера (часть 13 из 21):

Секция/Параметр	Описание

rest_server.limits.user	Пользователи, для которых устанавливается ограничение;
	• true - свой лимит для каждого авторизованного пользователя;
	• false - лимит по IP-адресу для каждого авторизованного пользователя;
	• string|string[] - ID пользователя или пользователей, к которым применяется правило.
	Примеры:
	`"limits": [ { "user": true, ... } ]`
	`"limits": [ { "user": false, ... } ]`
	`"limits": [ { "user": "6048c849d7b6e40593dfee71", ... } ]`
	`"limits": [ { "user": [ "57ff6853fa6db3a63d16d07b", "6048c849d7b6e40593dfee71" ], ... } ]`

Server	В этой секции задаются общие параметры сервера.
server.agent_ registration_parent_id	ID объекта, в котором появляются агенты при саморегистрации. По умолчанию - "**1**".
server. analytics_enabled	Включает аналитику значений метрик в процесс обработки данных; • false - аналитика выключена, • true - аналитика включена. По умолчанию - **false** - аналитика выключена.

Параметры конфигурационного файла сервера (часть 14 из 21):

Секция/Параметр	Описание
server. analytics_processes	Количество логических ядер, выделяемых для обработки аналитики временных рядов. По умолчанию используются все доступные ядра.
server.colorize_log	Включает цветную раскраску лога; • false - раскраска выключена, • true - раскраска включена. По умолчанию - **false** - раскраска выключена.
server. comet_ping_interval	Временной интервал между отправками comet-сервером сообщений (в миллисекундах). По умолчанию - **5000** - 5 секунд.
server. comet_ping_timeout	Время ожидания сообщения от comet-сервера (в миллисекундах). По умолчанию - **12000** - 12 секунд.
server.comet_port	Порт для соединения. По умолчанию - **8091**.
server.comet_secure	Включает SSL-соединение; • false - соединение выключено, • true - соединение включено. По умолчанию - **false** - соединение выключено.
server. comet_ssl_certificate	Путь к сертификату.
server.comet_ssl_key	Путь к ключу.
server.conditional_ incidents_enabled	Включает функционал генерации аварий; • false - функционал выключен, • true - функционал включен. По умолчанию - **false** - функционал выключен.
server.conditional_ incidents_script	Путь до скрипта, выполняющегося при возникновении аварии. *Подробности описаны в подразделе 4.1.10 "Передача аварии в сторонние сервисы" на стр. 66.*
server.debug	Включает debug-режим для логирования в файл **/var/log/saymon/saymon-server.log**; • false - режим выключен, • true - режим включен. По умолчанию - **false** - режим выключен.

Параметры конфигурационного файла сервера (часть 15 из 21):

Секция/Параметр	Описание
server. default_result_timeout	Время, через которое срабатывает условие **Нет данных от объекта** с момента создания объекта или получения последних данных (в миллисекундах). По умолчанию - **120000** - 2 минуты.
server.default_state_id	Состояние объекта по умолчанию. По умолчанию - **7** - NO DATA.
server. discovery_parent_id	ID объекта, в котором появляются найденные агентами сетевые устройства. По умолчанию - **"1"**.
server. event_log_max_bytes	Размер записей консоли в mongoDB, при достижении которого происходит ротация данных (в байтах). По умолчанию - **"1 G"** - 1 гигабайт.
server.extension_path	Путь к директории с серверными расширениями.
server.history_ temporary_storage_ period	Интервал времени для буферизации метрик, по истечении которого все данные из Redis записываются в OpenTSDB (в миллисекундах). Применяется, только если **history_update_period** равен **0**.
server.history_update_ period	Интервал записи исторических данных (в миллисекундах). 0 - немедленная запись пришедших значений. По умолчанию - **60000** - 1 минута.
server. history_write_length	Ограничение количества точек в одном запросе к OpenTSDB. При указании этого параметра данные в базу записываются по частям. По умолчанию ограничения нет, запись данных в OpenTSDB происходит без разбиения на части. *Функционал полезен при превышении размера записываемых за один раз данных. Рекомендуемое значение в таком случае - 100000.*
server. incident_timeout	Время перехода аварии из списка активных аварий в список исторических аварий после погашения аварии (в миллисекундах). По умолчанию - **120000** - 2 минуты.
server.notification_ buffering_period	Период ожидания для сбора сообщений о смене состояний объектов и отправки группового уведомления (в миллисекундах). По умолчанию - **0** - буферизация отключена. *Подробности об уведомлениях описаны в разделе 4.4 "Настройка уведомлений" на стр. 77.*

Параметры конфигурационного файла сервера (часть 16 из 21):

Секция/Параметр	Описание
server. retain_expired_stat	Включает хранение последних полученных данных после их устаревания; • false - хранение выключено, • true - хранение включено. По умолчанию - **false** - хранение выключено.
server.script_ trigger_timeout	Максимальное время выполнения триггера (в миллисекундах). По умолчанию - **30000** - 30 секунд.
server.self_object_id	ID объекта, используемого для самомониторинга.
server.sms_script	Путь до скрипта, отправляющего SMS-уведомления.
server. sql_history_enabled	Включает запись исторических данных в MySQL; • false - запись выключена, • true - запись включена. По умолчанию - **false** - запись выключена.
server.stat_local_ timestamp_field_ name	Имя поля, где передается время, с которым нужно сохранять данные в OpenTSDB. По умолчанию - **"localTimestamp"**.
server. stat_scan_period	Период проверки актуальности пришедших данных (в миллисекундах). По умолчанию - **3000** - 3 секунды.
server. use_stat_timestamp	Сохраняет временную метку (поле **timestamp**) из данных агента; • false - значение поля **timestamp** из данных агента заменяется на временную метку сервера, • true - используется значение поля **timestamp** из данных агента, оно не перезаписывается временем сервера. По умолчанию - **false** - временная метка из данных агента перезаписывается временем сервера.
server.voice_call_script	Путь до скрипта, осуществляющего голосовой вызов.

Параметры конфигурационного файла сервера (часть 17 из 21):

Секция/Параметр	Описание

server.email	В этой подсекции задаются параметры доступа к почтовому серверу:
	`"email" : {`
	`"disabled" : false,`
	`"fields" : {`
	`"from" : "saymon@saas.saymon.info"`
	`},`
	`"max_json_length": 1000,`
	`"transport" : {`
	`"auth" {`
	`"user" : "saymon@saas.saymon.info",`
	`"pass" : "P@ssw0rd"`
	`},`
	`"host" : "smtp.gmail.com",`
	`"port" : 465,`
	`"secure" : true`
	`},`
	`}`

server.email.disabled	Выключает отправку почтовых уведомлений; • true - отправка выключена, • false - отправка включена. По умолчанию - **true** - отправка выключена.
server.email. max_json_length	Ограничение размера письма с уведомлением (в символах). По умолчанию - **1000** - 1000 символов.
server.email.fields	Данные об отправителе уведомлений.
server.email.fields.from	Почтовый адрес отправителя.
server.email.transport	Данные почтового сервера.
server.email.transport.host	Адрес почтового сервера.
server.email.transport.port	Порт почтового сервера.
server.email.transport. secure	Включает использование TLS при подключении к серверу; • false - использоване TLS выключено, • true - использоване TLS включено. Значение по умолчанию зависит от порта.
server.email.transport.service	Встроенный в коннектор набор служб. При наличии задаёт **host**, **port**, **secure** автоматически. По умолчаниию - **"Gmail"**.
server.email.transport.auth	Данные для аутентификации пользователя.

Параметры конфигурационного файла сервера (часть 18 из 21):

Секция/Параметр	Описание
server.email.transport.auth.pass	Пароль пользователя.
server.email.transport.auth.user	Логин пользователя.
server.user	В этой подсекции задаются параметры пользователей: **"user" : {** **"auth_enabled": "true",** **"new_user_access": "all",** **"lang_default" : "ru",** **"template": {** **"permissions": ["manage-objects", "view-section-stat"],** **"objectPermissions": {** **"include": [],** **"exclude": ["5fb643ddf277b96c8401119b",** **"5f8dc28407e86603bfe281f8"]** **}** **},** **"usersRoot": "5800d9aaac7bf0f90d3d520e"** **}**
server.user.auth_enabled	Включает самостоятельную регистрацию для пользователей; • false - регистрация выключена, • true - регистрация включена. По умолчанию - **false** - регистрация выключена.
server.user.lang_default	Язык пользователей по умолчанию; • "en" - английский, • "it" - итальянский, • "ru" - русский. По умолчанию - **"en"** - английский.
server.user.new_user_access	Права доступа к объектам для нового пользователя; • "all" - есть доступ ко всем объектам, • "not" - нет доступа ни к одному объекту. По умолчанию - **"all"** - доступ ко всем объектам.
server.user.usersRoot	Идентификатор корневого объекта для создаваемых новым пользователем объектов. По умолчанию - **"1"**.
server.user.template	Шаблон прав нового пользователя. Данные параметры применяются только для пользователей, регистрирующихся самостоятельно.

Параметры конфигурационного файла сервера (часть 19 из 21):

Секция/Параметр	Описание
server.user.template.permissions	Список прав на операции, доступных пользователю по умолчанию. *Список всех возможных прав доступен в Приложении А на стр.166.*
server.user.template.objectPermissions	Права пользователя на доступ к объектам.
server.user.template.objectPermissions.include	Список идентификаторов объектов, к которым пользователю по умолчанию доступ разрешён.
server.user.template.objectPermissions.exclude	Список идентификаторов объектов, к которым пользователю по умолчанию доступ запрещён.

SNMP	В этой секции задаются параметры обработки SNMP-трапов.
	`"snmp": {`
	`  "filter": "/users/admin/tmp/snmp-filter.js"`
	`}`

snmp.filter	Путь к файлу с функцией фильтрации входящих SNMP Trap.
	Если функция возвращает значение **false**, сообщение отбрасывается.
	Пример функции фильтрации:
	`module.exports = function(message, rawMessage) {`
	`  rawMessage = String(rawMessage);`
	`  // console.log(message, rawMessage);`
	`  let filters = [`
	`  'HLR Call Processing Message',`
	`  'AIR Rejected - No EPS Subscription',`
	`  'GPRS SIBB Processing Exception',`
	`  'AuC Call Processing Message',`
	`  'Received notification from CLIM',`
	`  'AuC Event Log Clean Up',`
	`  'SOFT 0006'`
	`  ];`
	
	`  return !filters.some((filter) => rawMessage.includes(filter));`
	`}`
	Приведённый выше пример блокирует трапы, которые содержат любую строку, указанную в массиве *filters*.

Параметры конфигурационного файла сервера (часть 20 из 21):

Секция/Параметр	Описание

Zabbix	В этой секции задаются параметры для импорта данных из системы Zabbix.
	`"zabbix" : [{`
	`"url" : "http://192.168.1.215/zabbix/api_jsonrpc.php",`
	`"user" : "saymon",`
	`"password" : "saymon_user_password",`
	`"parent_id" : "58b586d5c3a2f96642e25537",`
	`"debug": true,`
	`"polling_period": 30000,`
	`"classes" : {`
	`"4" : "CPU",`
	`"619503be0ffb595aebe22222" : "Memory|General"`
	`}`
	`}]`
	Подробности описаны в подразделе 4.5.1 "Интеграция с Zabbix" на стр. 83.

zabbix.classes	Массив соответствий идентификаторов классов и имён обнаруженных объектов. Объектам с указанными именами будут при создании присвоены соответствующие классы.
zabbix.debug	Включает вывод параметров и результатов запросов в лог; • false - вывод выключен, • true - вывод включен. По умолчанию - **false** - вывод выключен.
zabbix.parent_id	ID объекта, используемого в качестве родителя для объектов, импортируемых из Zabbix.
zabbix.password	Пароль пользователя Zabbix с правами на чтение хостов или групп хостов, данные с которых необходимо импортировать.
zabbix.polling_period	Период обновления данных (в миллисекундах). По умолчанию - **120000** - 2 минуты.
zabbix.url	Адрес файла `api_jsonrpc.php` требуемой инсталляции Zabbix.
zabbix.user	Имя пользователя Zabbix с правами на чтение хостов или групп хостов, данные с которых необходимо импортировать.

Параметры конфигурационного файла сервера (часть 21 из 21):

Секция/Параметр	Описание

zabbix.
request_options
	В этой подсекции можно задать изменения опций для запросов хостов, триггеров и данных.
	`"request_options": {`
	`  "hosts": {`
	`  "filter": {`
	`  ...`
	`  }`
	`  },`
	`  "triggers": {`
	`  "filter": {`
	`  ...`
	`  }`
	`  },`
	`  "data": {`
	`  "filter": {`
	`  ...`
	`  }`
	`  }`
	`}`

zabbix.request_ options.hosts	Параметры для запроса хостов.
zabbix.request_options. hosts.filter	Поля фильтров для запроса хостов. Все возможные поля фильтров hosts представлены в документации по Zabbix API.
zabbix.request_options. triggers	Параметры для запроса триггеров.
zabbix.request_options. triggers.filter	Поля фильтров для запроса триггеров. Все возможные поля фильтров triggers представлены в документации по Zabbix API.
zabbix.request_options. data	Параметры для запроса данных.
zabbix.request_options. data.filter	Поля фильтров для запроса данных. Все возможные поля фильтров data представлены в документации по Zabbix API.

Управление некоторыми настройками доступно в web-интерфейсе системы:

1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".
2. Перейти в раздел "Опции конфигурации".

Изменённые параметры в интерфейсе приоритетнее настроек в конфигурационном файле.

Установленные значения параметров автоматически сохраняются в MongoDB, конфигурационный файл не перезаписывается.

Раздел "Опции конфигурации" окна конфигурации системы состоит из подразделов:

1. "Общие настройки" соответствуют параметрам секции **Server** (Рис. 4.1.3.1):

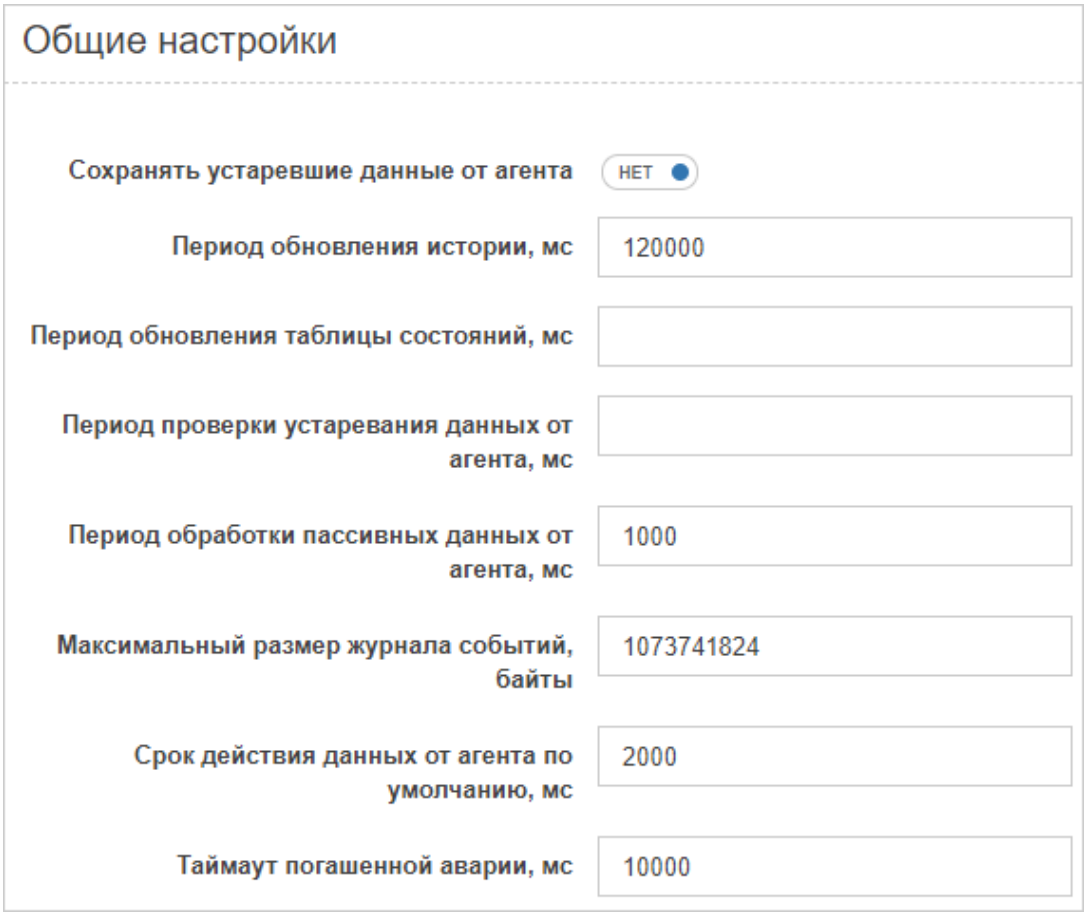


Рис. 4.1.3.1. *Общие настройки*

2. "Настройки электронной почты" соответствуют параметрам подсекции **Server.Email** (Рис. 4.1.3.2):

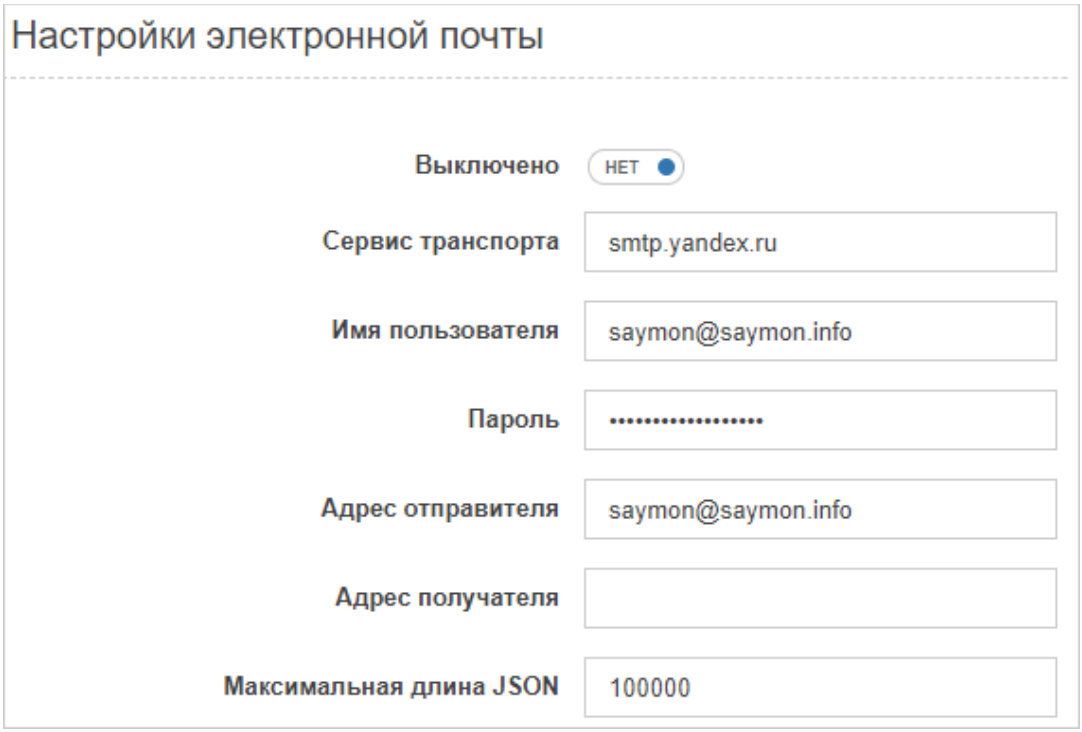


Рис. 4.1.3.2. *Настройки электронной почты*

3. "Настройки пользователей" соответствуют параметрам подсекции **Server.User** (Рис. 4.1.3.3):

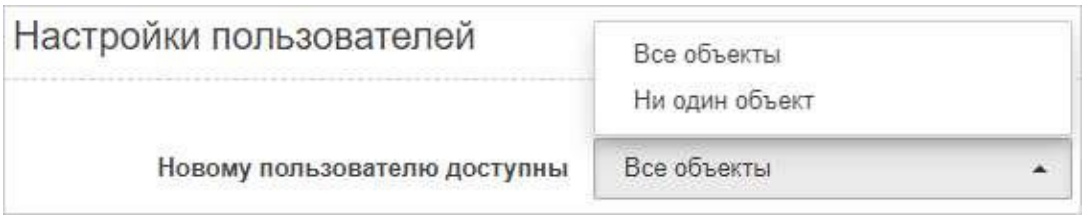


Рис. 4.1.3.3. *Настройки пользователей*

Просмотреть актуальные настройки сервера возможно при помощи REST API метода:

```
GET /node/api/configuration
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/configuration

curl -X GET $url -u $login:$password
```

4.1.4 Настройка элементов web-интерфейса

Некоторые элементы web-интерфейса можно настроить в конфигурационном файле клиента `/usr/local/saymon/client/client-config.js`.

В некоторых случаях файл `client-config.js` может находиться в другой директории, путь к которой может быть найдет в конфигурационном файле nginx или командой:

```
sudo find / -name "client-config.js"
```

Параметры конфигурационного файла клиента (часть 1 из 10):

Секция/Параметр	Описание
agentInstallerUrl	Ссылка для скачивания установочного скрипта для агента.
authEnabled	Включает самостоятельную регистрацию для пользователей; • false - регистрация выключена, • true - регистрация включена. По умолчанию - **false** - регистрация выключена.
Authentication	В этой секции можно настроить методы аутентификации пользователей. `authentication: { availableMethods: ["native", "keycloak"] methodByDefault: "keycloak" services: { keycloak: { configUri: "keycloak_frontend.json", legacySupport: false } } }`
authentication. availableMethods	Список доступных для пользователей методов аутентификации: • **"native"** - вход по логину и паролю непосредственно на сервере Центрального Пульта; • **"keycloak"** - вход с использованием внешнего Keycloak-сервера. По умолчанию используется только "native".
authentication. methodByDefault	Метод аутентификации, предлагаемый пользователю при отображении формы аутентификации. По умолчанию - **"native"**.
authentication.services	Секция с настройками сервисов аутентификации.
authentication.services. keycloak	Секция с настройками аутентификации через Keycloak.

Параметры конфигурационного файла клиента (часть 2 из 10):

Секция/Параметр	Описание
authentication.services.keycloak.configUri	Название файла настроек соединения, сгенерированного Keycloak-сервером для Frontend-клиента. *Файл необходимо разместить на сервере Центрального Пульта в каталоге с конфигурационным файлом клиента* **client-config.js**.
authentication.services.keycloak.legacySupport	Включает поддержку работы с Keycloak-сервером версии ниже 19; • false - поддержка работы с Keycloak-сервером версии ниже 19 выключена; • true - поддержка работы с Keycloak-сервером версии ниже 19 включена. По умолчанию - **false** - работа с Keycloak-сервером версии ниже 19 не поддерживается.
canEditObjectStyle	Включает возможность вручную настраивать CSS-стили для объектов и связей. При включении опции в разделе **Параметры** появляется подраздел **Стили**; • false - настройка стилей выключена, • true - настройка стилей включена. По умолчанию - **true** - настройка стилей включена.
collapseSections	Задаёт способ первоначального отображения секций и графиков; • false – секции закреплены, графики раскрыты, • true – секции откреплены, графики свёрнуты. По умолчанию - **true** - секции откреплены, графики свёрнуты.
Comet	Наряду с REST API, Центральный Пульт использует модель Comet для обмена данными между клиентом и сервером. **comet: {** **port: 8091,** **url: [** **"https://cpult.ru/ws",** **"http://10.78.5.11:8092"** **]** **},** **...**
comet.port	Номер порта, по которому клиент подключается к Comet-серверу. По умолчанию - **8091**.

Параметры конфигурационного файла клиента (часть 3 из 10):

Секция/Параметр	Описание
comet.url	Адреса для подключения клиентов к Comet-серверу из разных подсетей. *Параметр **comet.port** игнорируется при использовании параметра **comet.url**.*
DefaultClassImage	Позволяет задать стандартное изображение и его размер для объектов определённого класса. **defaultClassImage: {** **'29': {img: '/images/cloud.svg', dim: [100, 75] }** **},** **...**
defaultClassImage.'<number>'	Идентификатор класса, например, **29**.
defaultClassImage.'<number>'.img	Путь до изображения относительно папки, где находится конфигурационный файл.
defaultClassImage.'<number>'.dim	Размер изображения (в пикселях, [ширина, высота]).
DefaultEmbeddableLink	Позволяет встраивать виджеты с информацией об объектах на сторонние ресурсы. **defaultEmbeddableLink: {** **url: "http://saymon.info/",** **name: "SAYMON"** **},** **...**
defaultEmbeddableLink.name	Имя виджета.
defaultEmbeddableLink.url	URL, на который ведёт виджет.
defaultPieChartColors	Цвета секторов по умолчанию для круговой диаграммы. **defaultPieChartColors: [** **'#a90329',** **'#3276b1',** **'#009F3D',** **'#9868B8',** **'#F2C001',** **'#FF742C',** **'#90574E',** **'#EB77BE',** **'#FF9796',** **'#AAC6E5',** **'#8BDE90',** **'#FDB5D0'** **],** **...**

Параметры конфигурационного файла клиента (часть 4 из 10):

Секция/Параметр	Описание
disabledSections	Позволяет удалить секции, которые показываются при подробном отображении объекта в верхней части экрана. Список всех доступных секций: • audit-log (История изменений), • config-log (Изменения конфигурации), • documents (Документы), • entity-incident-conditions (Условия генерации аварий), • entity-settings (Параметры), • entity-state-conditions (Условия перехода состояний), • history-graph (Графики), • monitoring (Мониторинг), • operations (Операции), • properties (Свойства), • stat (Данные), • stat-rules (Правила формирования данных), • state-history (История состояний), • state-triggers (Действия при смене состояний), • widgets (Виджеты).
Documentation	В этой секции можно настроить ссылки на различную документацию для Центрального Пульта. **documentation: {** **main: {** **root: 'https://wiki.saymon.info/display/SAYMONEN',** **ru: 'https://wiki.saymon.info'** **},** **notificationTemplates: {** **root: 'https://wiki.saymon.info/display/SAYMONEN/ Notification+templates',** **ru: 'https://wiki.saymon.info/pages/ viewpage.action?pageId=45385140'** **}** **},** **...**
documentation.main	Ссылки на пользовательскую документацию.
documentation.main.root	Ссылка на англоязычную документацию.
documentation.main.ru	Ссылка на русскоязычную документацию.

Параметры конфигурационного файла клиента (часть 5 из 10):

Секция/Параметр	Описание
documentation. notificationTemplates	Ссылки на шаблоны оповещений.
documentation. notificationTemplates.root	Ссылка на англоязычные шаблоны оповещений.
documentation. notificationTemplates.ru	Ссылка на русскоязычные шаблоны оповещений.
enableAnalytics	Включает функционал модуля аналитики; • false - функционал выключен, • true - функционал включен. По умолчанию - **false** - функционал выключен.
enableConditionalIncidents	Включает функционал генерации аварий; • false - функционал выключен, • true - функционал включен. По умолчанию - **false** - функционал выключен.
enableSmsTrigger	Включает использование SMS-уведомлений; • false - SMS-уведомления выключены, • true - SMS-уведомления включены. По умолчанию - **false** - SMS-уведомления выключены.
enableSnmpTranslate	Включает трансляцию SNMP OID в текстовый формат; • false - трансляция выключена, • true - трансляция включена. По умолчанию - **false** - трансляция выключена.
enableVoiceCallTrigger	Включает использование голосовых вызовов; • false - голосовые вызовы выключены, • true - голосовые вызовы включены. По умолчанию - **false** - голосовые вызовы выключены.
ForceEmptyStandardView	При выборе объекта, в том числе не содержащего вложенных объектов, Центральный Пульт по умолчанию показывает для него подробную информацию. В параметре задаётся список идентификаторов классов, при выборе пустых объектов которых Центральный Пульт переходит в стандартный вид. По умолчанию: • 1 (Root), • 3 (Host), • 13 (Node), • 43 (Dashboard).

Параметры конфигурационного файла клиента (часть 6 из 10):

Секция/Параметр	Описание
GeoMap	Центральный Пульт позволяет привязывать объекты к их геоположению и просматривать их на карте. **geoMap: {** **serverName: openstreetmap.org,** **initialPosition: [30.341306, 59.930089],** **initialZoom: 8** **},** **...**
geoMap.initialPosition	Начальная позиция на карте в формате [долгота, широта]. По умолчанию - Россия, Санкт-Петербург.
geoMap.initialZoom	Начальный масштаб карты. По умолчанию - **8**.
geoMap.serverName	Имя сервера, отвечающего за отображения карты. По умолчанию - openstreetmap.org. Также можно использовать локальный OSM-сервер.
Grid	В этой секции задаются настройки сетки для выравнивания объектов в стандартном виде. **grid: {** **dim: 20,** **color: "rgba(128, 128, 128, 0.3)",** **border: 4** **},** **...**
grid.border	Максимальное значение отступа границы объекта от границы сетки (в пикселях). По умолчанию - **4**.
grid.color	Цвет сетки в формате RGBA. По умолчанию - **"rgba(128, 128, 128, 0.3)"**.
grid.dim	Размер сетки (в пикселях). По умолчанию - **20**.

Параметры конфигурационного файла клиента (часть 7 из 10):

Секция/Параметр	Описание
GridView	В этой секции задаются параметры отображения объектов в виде сетки. **gridView: {** **cols: 6,** **colWidth: 240,** **rowHeight: 90,** **gutterSize: 5** **},** **...**
gridView.cols	Количество объектов в строке сетки. При **gridView.cols > 0** ширина объектов меняется автоматически. При **gridView.cols = 0** число объектов в строке сетки подстраивается под ширину экрана. По умолчанию - **6**.
gridView.colWidth	Ширина объектов при **gridView.cols = 0** (в пикселях). По умолчанию - **240**.
gridView.gutterSize	Величина отступа между объектами (в пикселях). По умолчанию - **5**.
gridView.rowHeight	Высота объектов (в пикселях). По умолчанию - **90**.
hideConnectionStatusNotifications	Скрывает оповещения о статусе соединения с Comet-сервером; • false - оповещения выводятся в web-интерфейс, • true - оповещения выводятся в консоль. По умолчанию - **false** - оповещения о статусе соединения с Comet-сервером выводятся в web-интерфейс.
hideResponseErrorNotifications	Скрывает оповещения об ошибках, возникающих при отправке автоматических запросов на сервер; • false - оповещения выводятся в web-интерфейс, • true - оповещения скрыты. По умолчанию - **false** - оповещения выводятся в web-интерфейс. *Ошибки, возникающие при участии пользователя, например сохранение каких-либо данных в формах, будут выводиться в web-интерфейс в любом случае.*

Параметры конфигурационного файла клиента (часть 8 из 10):

Секция/Параметр	Описание

History	В этой секции задаются параметры, отвечающие за отображение данных на графиках, а также активируются некоторые дополнительные элементы интерфейса.
	`history: {`
	`  slider: false,`
	`  downsampling: [`
	`  [[1, 'hours'], '15s-avg'],`
	`  [[1, 'days'], '5m-avg'],`
	`  [[1, 'weeks'], '1h-avg'],`
	`  ['1d-avg']`
	`  ],`
	`  modelChangeAnnotations: false`
	`},`
	`...`

history.downsampling	Параметры уменьшения размера выборки данных - downsampling.
history. modelChangeAnnotations	Включает смену аннотаций на графиках; • false - смена аннотаций выключена, • true - смена аннотаций включена. По умолчанию - **false** - смена аннотаций выключена.
history.slider	Включает ползунок, который используется для задания интервала времени отображения данных; • false - ползунок выключен, • true - ползунок включен. По умолчанию - **false** - ползунок выключен.

Languages	В этой секции происходит управление предустановленными языками web-интерфейса.
	`languages: {`
	`  en: { short: 'us', long: 'English' },`
	`  ru: { short: 'ru', long: 'Русский' },`
	`  it: { short: 'it', long: 'Italiano' }`
	`},`
	`...`

languages.<id>	Идентификатор языка. Доступны три языка: • en - английский, • it - итальянский, • ru - русский.
languages.<id>.long	Полное название языка. Это название отображается в меню текущего пользователя Центрального Пульта при выборе языка интерфейса.

Параметры конфигурационного файла клиента (часть 9 из 10):

Секция/Параметр	Описание
languages.<id>.short	Идентификатор иконки флага. Иконка отображается в меню текущего пользователя Центрального Пульта при выборе языка интерфейса.  - es,  - gb,  - it,  - ru,  - us.
millisecondMonitoring	Включает обновление данных каждую миллисекунду; • false - обновление выключено, • true - обновление включено. По умолчанию - **false** - обновление выключено.
Monitoring	В этой секции настраивается скрытие стандартных мониторинговых сенсоров из интерфейса. **monitoring: {** **standardTasks: [‘ping’, ‘snmpGet’]** **},** **...** *Механизм скрытия стандартных сенсоров описан в подразделе 4.3.1 "Скрытие неиспользуемых сенсоров" на стр. 74.*
monitoring. standardTasks	Массив идентификаторов отображаемых стандартных сенсоров.
pollInterval	Интервал между запросами данных с REST-сервера, если соединение между клиентом и Comet-сервером недоступно (в миллисекундах). По умолчанию - **5000** - 5 секунд.
pollIntervalSocket	Интервал между запросами данных с Comet-сервера, если соединение между клиентом и Comet-сервером доступно (в миллисекундах). По умолчанию - **60000** - 1 минута.
title	Заголовок для вкладок, а также имя, отображаемое в левом верхнем углу web-интерфейса. По умолчанию - **SAYMON**.

Параметры конфигурационного файла клиента (часть 10 из 10):

Секция/Параметр	Описание
Tree	В этом разделе задаются настройки для дерева объектов. **tree: {** **filterSubmit: true** **},** **...**
tree.filterSubmit	Включает кнопку применения фильтра и отключает его автоматическое применение при вводе символов; • false - кнопка выключена, • true - кнопка включена. По умолчанию - **false** - кнопка выключена.
themes	Этот параметр отвечает за настройку тем. По умолчанию используется тема **Light**. Также доступна тема **Dark**. CSS-файлы с темами находятся в директории **$SAYMON_ROOT/target/client/css**. **$SAYMON_ROOT** - директория, где установлен Центральный Пульт. **themes: {** **default: { name: 'Light' },** **dark: { name: 'Dark' }** **},** **...**
useNaturalSort	Включает сортировку объектов с помощью библиотеки javascript-natural-sort: • false - объекты сортируются с помощью нативных функций браузера, • true - объекты сортируются с помощью библиотеки javascript-natural-sort. По умолчанию - **false** - объекты сортируются с помощью нативных функций браузера. *При установленном useNaturalSort = true возможно замедление работы web-интерфейса при большом количестве объектов в системе (от 100 000).*

4.1.5 Служба "saymon-server"

Узнать состояние, запустить, перезапустить и остановить службу сервера можно следующими командами соответственно:

```
sudo service saymon-server status
sudo service saymon-server start
sudo service saymon-server restart
sudo service saymon-server stop
```

4.1.6 Просмотр websocket-нотификаций

Центральный пульт использует гибкий механизм оповещений, который позволяет пользователю оперативно реагировать на возникающие ситуации.

Чтобы просмотреть пришедшие websocket-уведомления, необходимо:

1. Запустить Chrome.
2. Авторизоваться на сервере Центрального Пульта.
3. Перейти по ссылке https://<your_server>/incidents.html?debug=comet
4. Открыть в Chrome Инструменты разработчика.
5. Открыть вкладку Консоль (Console) в Инструментах разработчика.
6. В качестве примера нажать правой кнопкой мыши на строке с аварией и выбрать пункт "Подтвердить".

В консоли отобразится websocket-нотификация.

4.1.7 Увеличение количества обработчиков SNMP-Trap

Для поддержания бесперебойного наблюдения за объектами и оповещения администратора Центральный Пульт использует SNMP-Trap.

Если на сервер поступает большое количество SNMP-Trap, то возможно увеличить количество их обработчиков. Для этого необходимо:

1. Открыть в текстовом редакторе файл настроек акторов:
   ```
   sudo nano /usr/local/saymon/backend/server/actors.json
   ```
2. Добавить следующую секцию:
   ```
   "snmpTrapMessageHandlerActor": {
      "mode": "forked",
      "clusterSize": 3,
      "onCrash": "respawn"
   }
   ```
3. Перезапустить сервер:
   ```
   sudo service saymon-server restart
   ```

Для проверки корректности выполненной операции необходимо выполнить команду:

```
ps ax | grep node
```

В выводе должно отображаться число процессов **SnmpTrapMessageHandlerActor**, которое было указано в параметре **clusterSize** выше (в этом примере - 3):

```
5793 ? Rsl 10:23 /usr/bin/nodejs --
harmony /usr/local/saymon/backend/server/saymon-server.js

5857 ? Sl 3:18 /opt/nodejs/bin/node --
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js RestServerActor

5862 ? Sl 0:37 /opt/nodejs/bin/node --
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js
ResourceServerActor

5867 ? Sl 1:43 /opt/nodejs/bin/node --
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js
HistoryWriterActor

5872 ? Sl 0:45 /opt/nodejs/bin/node –
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js
SnmpTrapMessageHandlerActor

5877 ? Sl 0:42 /opt/nodejs/bin/node --
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js
SnmpTrapMessageHandlerActor

5882 ? Sl 0:42 /opt/nodejs/bin/node --
harmony /usr/local/saymon/backend/server/actors/forked-actor-worker.js
SnmpTrapMessageHandlerActor
```

4.1.8 Сброс системы к заводским настройкам

Администратор может в любой момент сбросить настройки Центрального Пульта, чтобы восстановить настройки по умолчанию.

Для этого необходимо выполнить следующий скрипт:

```
#!/bin/bash
#
#Script deletes instance-specific data from MongoDB, Redis and OpenTSDB.
# Let's stop SAYMON Server and begin.
service saymon-server stop

mongo saymon --eval 'db.dropDatabase()'

echo flushall | redis-cli

docker stop opentsdb
docker rm opentsdb
docker run -d -p 127.0.0.1:4242:4242 --restart=always --name=opentsdb
rossinno/opentsdb

# Purges logs also.
rm /var/log/saymon/* /var/log/nginx/saymon*

# Forward to new monitoring adventures!
service saymon-server start
```

4.1.9 Включение функционала генерации аварий

Функционал генерации аварий опционален и выключен по умолчанию. Для его активации необходимо:

1. В секции **Server** конфигурационного файла сервера `/etc/saymon/saymon-server.conf` установить параметру **conditional_incidents_enabled** значение **true**:

```
{
   ...
   "server" : {
      ...
      " conditional_incidents_enabled ": true,
      },
   ...
}
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

2. В конфигурационном файле клиента `/usr/local/saymon/client/client-config.js` установить параметру **enableConditionallncidents** значение **true**:

```
return {
   ...
   enableConditionallncidents : true,
   ...
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

4.1.10 Передача аварий в сторонние сервисы

Для передачи аварийных сообщений в системы верхнего уровня необходимо выполнить следующие действия:

1. Включить функционал генерации аварий в секции **Server** конфигурационного файла сервера `/etc/saymon/saymon-server.conf` и там же указать путь к скрипту, который будет отправлять данные:

```
"server" : {
   "conditional_incidents_enabled" : true,
   "conditional_incidents_script": "/home/saymon/scripts/send_incident.sh",
   ...
}
```

2. Добавить в MongoDB триггеры по авариям для классов или отдельных объектов:

```
mongo saymon

db.incidentTriggers.insert({ "ownerId" : 1042, "ownerType" : 4, "body" : [ { "type" : "conditional_incidents_script", "payload" : { }, "states" : [ 1,2,4] } ] })
```

   где

   - ownerId - идентификатор сущности, к которой добавляется триггер;
   - ownerType - тип сущности, для которой добавляется триггер (4 - класс, 1 - объект);
   - states - уровень критичности аварий, при которых запускается триггер.

   В указанном выше примере скрипт триггера будет вызываться при возникновении аварии любой критичности у объектов класса 1042.

   В скрипт передается два аргумента:
   - $1 - тело аварии;
   - $2 - свойства объекта, к которому относится авария.

4.1.11 Перевод SNMP OID в текстовый формат

По умолчанию в результатах проверки SNMP Trap OID отображается в числовом формате. Для перевода OID в "человекочитаемый" текстовый формат необходимо:

1. В секции **rest_server** конфигурационного файла сервера `/etc/saymon/saymon-server.conf` задать адрес сервера для трансляции OID в текстовый формат в параметре **snmp_mib_url**:

```
{
   ...
   "rest_server" : {
      ...
      "snmp_mib_url": "http://192.168.1.189:5550",
      },
   ...
}
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

2. В конфигурационном файле клиента `/usr/local/saymon/client/client-config.js` установить параметру **enableSnmpTranslate** значение **true**:

```
return {
   ...
   enableSnmpTranslate : true,
   ...
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

4.1.12 Выполнение скриптов при создании и удалении объектов

При создании или удалении объектов того или иного класса система способна выполнять скрипты и передавать им аргументы, настраиваемые через web-интерфейс.

Для применения изменений достаточно обновить страницу в браузере.

Чтобы настроить скрипты, необходимо:

1. Задать каталог для хранения скриптов:

1.1. Открыть конфигурационный файл сервера `/etc/saymon/saymon-server.conf` и добавить в него секцию **custom_scripts**:

```
{
    ...
    "custom_scripts ": {
    "entity_triggers_path": " /opt/saymon-extensions/entity-triggers-path"
    },
    ...
}
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

1.2. В параметре **entity_triggers_path** указать полный путь до каталога.

2. Поместить исполняемый файл в указанный каталог.

> *Исполняемые файлы скриптов могут располагаться в любом количестве подпапок. Поиск производится рекурсивно.*

3. Задать настройки скрипта в конфигурационном файле.

   Параметры скрипта определяются в JSON-файле, именуемом как **имя_исполняемого_файла + .conf**. Располагаться этот файл должен в том же каталоге, что и исполняемый файл.

   Пример:

```
/opt/saymon-extensions/entity-triggers-path/myscript.sh
/opt/saymon-extensions/entity-triggers-path/myscript.sh.conf
```

> *Возможные поля файла настроек параметров и пример конфигурационного файла представлены в Приложении Б на стр. 171.*

4.1.13 Автоматическое обнаружение объектов

Агент сканирует локальную подсеть и отправляет на сервер информацию обо всех IP-адресах, доступных для TCP-соединения либо доступных по PING. Для обнаруженных хостов объекты класса **Host** с соответствующими адресами создаются в Центральном Пульте и отображаются на карте инфраструктуры. Внутри каждого из этих объектов автоматически создаётся объект класса **Ping**, в котором включается мониторинг с помощью агента, выполнившего автообнаружение, и сенсора **Ping**.

IP-адрес обнаруженного объекта автоматически записывается в свойство **IP**, что можно использовать для перехода по этому адресу в браузере или доступа к объекту по SSH или Telnet через контекстное меню.

Чтобы настроить автообнаружение объектов, необходимо:

1. В конфигурационном файле агента `.../saymon-agent/conf/agent.properties` установить параметру **agent.discoveryEnabled** значение **true**:

```
agent.discoveryEnabled=true
```

> *Подробную информацию о файле* `.../saymon-agent/conf/agent.properties` *см. в подразделе 4.1.2 "Конфигурация агента" на стр. 25.*

2. Указать ID родительского объект для обнаруживаемых объектов в параметре **server.discovery_parent_id** конфигурационного файла сервера `/etc/saymon/saymon-server.conf`:

> ⚠ *По умолчанию обнаруженные объекты помещаются в корневой объект.*

```
...
    "server": {
        ...
        "discovery_parent_id": "632aae8b1a687b43378c0657",
        ...
    },
...
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

3. Перезапустить службы saymon-agent и saymon-server:

```
sudo service saymon-agent restart
sudo service saymon-server restart
```

Заключительные действия:

Не требуются.

4.2 Модуль аналитики

По умолчанию модуль аналитики, используемый для прогнозирования и определения всплесков, не включен в ISO-образ Центрального Пульта. Чтобы установить модуль аналитики, необходимо:

1. Скачать скрипт analytics-create-and-run.sh и разместить его в файловой системе на виртуальной машине, где установлен ISO-образ Центрального Пульта.
2. Запустить скрипт командой

```
sudo ANALYTICS_BRANCH=x.y.z ./analytics-create-and-run.sh
```

> ⚠ *x.y.z - версия модуля аналитики, совместимая с вашей версией Центрального Пульта.*

Проверить, что контейнер **saymon-analytics** запустился и работает, можно с помощью команды

```
docker ps –a
```

4.2.1 Подключение модуля аналитики к Центральному Пульту

Чтобы подключить модуль аналитики, потребуется внести изменения в конфигурацию Центрального Пульта:

1. В секции **Server** конфигурационного файла сервера `/etc/saymon/saymon-server.conf` добавить или изменить следующие параметры:

```
{
   ...
   "server" : {
      ...
      "analytics_enabled": true,
      "analytics_processes": 2 # количество логических ядер для аналитики
   },
   ...
}
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

2. В конфигурационном файле клиента `/usr/local/saymon/client/client-config.js` установить параметру **enableAnalytics** значение **true**:

```
return {
   ...
   enableConditionallncidents : true,
   ...
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

3. Перезапустить сервер Центрального Пульта:

```
sudo service saymon-server restart
```

4.2.2 Мониторинг работы модуля аналитики

Чтобы мониторить работу модуля аналитики можно использовать логи контейнера **saymon-analytics**. Для этого необходимо:

1. Подключиться к контейнеру командой

```
docker exec -ti saymon-analytics bash
```

2. Открыть файл логов модуля командой

```
tail -f /opt/analytics/analytics.log
```

В логах модуля содержится информация о:

- настройках, с которыми запущен модуль (слушающий сокет и количество задействованных в пуле обработчика процессов),
- обрабатываемых метриках,
- результатах обработки метрик,
- ошибках, возникающих в процессе работы модуля.

4.3 Управление сенсорами мониторинга

Системой предусмотрена возможность скрывать в интерфейсе неиспользуемые мониторинговые сенсоры, а также настраивать пользовательские сенсоры и внедрять их в интерфейс.

4.3.1 Скрытие неиспользуемых сенсоров

Для того, чтобы неиспользуемые сенсоры не отображались в пользовательском интерфейсе, необходимо:

1. Открыть конфигурационный файл клиента `/usr/local/saymon/client/client-config.js` и добавить в него секцию **monitoring**:

```
return {
    ...
    monitoring: {
        standardTasks: ['ping', 'snmpGet']
    },
    ...
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

2. В переменной **monitoring.standardTasks** указать идентификаторы сенсоров, отображаемых в web-интерфейсе.

> *Идентификаторы сенсоров чувствительны к регистру символов.*

Список идентификаторов стандартных сенсоров:

- binaryProtocol (бинарный протокол),
- configFile (конфигурационный файл / директория),
- databaseQuery (запрос в базу данных),
- exec (выполнение программы / скрипта),
- external (внешняя проверка - записанные через API данные),
- ftp (FTP-сенсор),
- httpRequest (HTTP-запрос),
- jmx (jmx-сенсор),
- localNetworkPort (локальный порт),
- mqtt (mqtt-сенсор),
- ping (пинг-сенсор),
- processInfo (процесс по имени),
- remoteNetworkPort (удалённый порт),
- snmpGet (SNMP Get-сенсор),
- snmpTrap (SNMP Trap-сенсор),
- wmi (WMI-сенсор).

4.3.2 Настройка пользовательских сенсоров

В качестве сенсора можно использовать собственноручно написанный скрипт и передавать ему в качестве аргументов значения параметров через удобный web-интерфейс.

Для применения изменений достаточно обновить страницу в браузере.

Чтобы сделать пользовательский скрипт мониторинговым сенсором, необходимо:

1. Задать каталог для хранения пользовательских сенсоров:
   1.1. Открыть конфигурационный файл сервера `/etc/saymon/saymon-server.conf` и добавить в него секцию **monitoring**:

```
{
    ...
    "monitoring": {
    "custom_tasks_path": "/opt/saymon-agent/custom_tasks"
    },
    ...
}
```

   > *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

   1.2. В параметре **monitoring.custom_tasks_path** указать полный путь до каталога.

2. Поместить исполняемый файл в указанный каталог.

   > *Исполняемые файлы пользовательских сенсоров могут располагаться в любом количестве подпапок. Поиск производится рекурсивно.*

3. Проверить идентификатор пользовательского сенсора. Идентификатор сенсора создаётся автоматически и эквивалентен имени исполняемого файла с относительным путём (если есть). Например: **utils/network/ping.**

   > *Идентификатор пользовательского сенсора не должен совпадать с идентификатором стандартного, иначе пользовательский сенсор будет проигнорирован.*

   > *С перечнем идентификаторов стандартных сенсоров можно ознакомиться* *в подразделе 4.3.1 "Скрытие неиспользуемых сенсоров" на стр. 74.*

4. Задать настройки сенсора в конфигурационном файле.

   Параметры сенсора определяются в JSON-файле, именуемом как **имя_исполняемого_файла + .conf**. Располагаться этот файл должен в том же каталоге, что и исполняемый файл.

   Пример:

```
/opt/saymon-agent/custom_tasks/mytask.sh
/opt/saymon-agent/custom_tasks/mytask.sh.conf
```

   Возможные поля файла настроек параметров и пример конфигурационного файла представлены в Приложении Б на стр. 171.

Пользовательские сенсоры выполняются агентами локально, поэтому необходимо скопировать исполняемые файлы в одну и ту же папку на все хосты, на которых предполагается их запускать. Конфигурационные файлы достаточно разместить на хостах с сервером системы.

4.4 Настройка уведомлений

При переходе объектов в определенные состояния система может:

- отправлять email-уведомления,
- автоматически запускать программу или скрипт с параметрами,
- отправлять сообщения в Telegram,
- показывать визуальное уведомление в браузере, сопровождающееся звуком,
- создавать задачи в JIRA,
- запускать операции,
- отправлять SMS,
- совершать голосовые вызовы.

Уведомления настраиваются в виде подробной информации об объекте в секции "Действия при смене состояния" (Рис. 4.4.1):

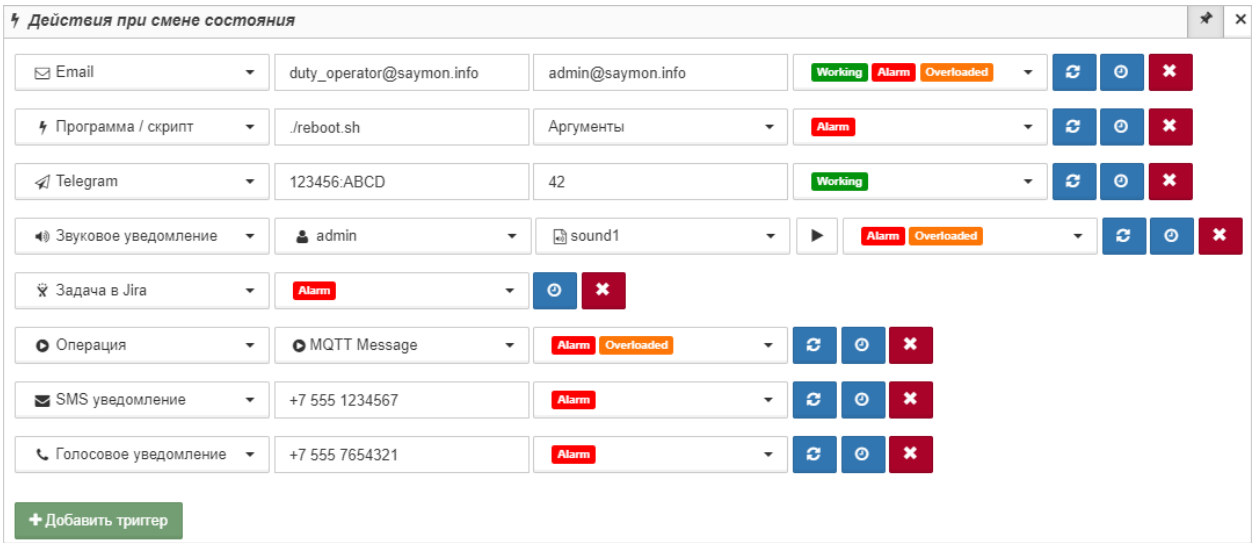


Рис. 4.4.1. *Действия при смене состояния*

4.4.1 Активация функционала отправки SMS и голосовых вызовов

Для активации функционала необходимо выполнить описанные ниже действия:

1. В файле конфигурации клиента `/usr/local/saymon/client/client-config.js` установить параметрам **enableSmsTrigger** и **enableVoiceCallTrigger** значение **true**:

```
return {
    ...
    enableVoiceCallTrigger : true,
    enableSmsTrigger : true,
    ...
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

2. В секции **Server** конфигурационного файла сервера `/etc/saymon/saymon-server.conf` задать путь до скриптов, осуществляющих отправку SMS-уведомлений и голосовые вызовы:

```
{
    ...
    "server" : {
        ...
        "sms_script" : "путь до скрипта, отправляющего sms-уведомления",
        "voice_call_script" : "путь до скрипта, осуществляющего голосовые
        вызовы",
        ...
    },

    ...
}
```

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

3. Для применения изменений перезапустить службу "saymon-server":

```
sudo service saymon-server restart
```

4.4.2 Отправка почтовых уведомлений

Для настройки отправки почтовых уведомлений требуется настроить необходимые параметры доступа к почтовому серверу в подсекции **Email** секции **Server** файла `/etc/saymon/saymon-server.conf`.

> *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе* *4.1.3 "Конфигурация сервера" на стр. 28.*

Пример настройки подсекции **Email**:

```
"email" : {
   "disabled" : false,
   "fields" : {
      "from" : "saymon@saas.saymon.info"
      },
   "max_json_length": 1000,
   "transport" : {
      "auth" {
         "user" : "saymon@saas.saymon.info",
         "pass" : "P@ssw0rd"
      },
      "host" : "smtp.gmail.com",
      "port" : 465,
      "secure" : true
   },
}
```

4.4.3 Настройка уведомлений в Telegram

Для настройки отправки уведомлений с помощью Telegram необходимо:

1. Задать бота, от которого будут поступать уведомления:

1.1. Создать нового бота:

1.1.1. Найти в приложении Telegram контакт **BotFather**.

1.1.2. Отправить ему сообщение **/newbot**.

1.1.3. Задать боту отображаемое имя (**name**, позже возможно изменить).

1.1.4. Задать боту уникальное имя (**username,** изменить будет невозможно).

1.1.5. Скопировать токен бота вида
210979209:AAFfT2mt3oW4EK1gYqE_d3OjAJSIRLSrAL.

1.1.6. Отправить контакту **BotFather** сообщение **/setprivacy**.

1.1.7. Выбрать созданного бота по его **username**.

1.1.8. Выбрать опцию **Disable**.

1.2. Использовать существующего бота:

1.2.1. Найти в приложении Telegram контакт **BotFather**.

1.2.2. Отправить ему сообщение **/mybots**.

1.2.3. Выбрать нужного бота.

1.2.4. Выбрать опцию **API Token**.

1.2.5. Скопировать токен бота вида
210979209:AAFfT2mt3oW4EK1gYqE_d3OjAJSIRLSrAL.

2. Настроить канал, чат или группу:

2.1. Создать приватный канал (рекомендуется):

2.1.1. В приложении Telegram создать новый канал.

2.1.2. Открыть настройки канала, добавить бота в список администраторов.

2.1.3. Отправить сообщение в канал.

2.1.4. Перейти по ссылке в любом web-браузере, вставив в неё токен своего бота (без пробелов и знаков <>):
https://api.telegram.org/bot<токен_бота>/getUpdates

2.1.5. Найти текст со словами **chat** и **id**, например
..."channel_post":{"message_id":4,"chat":{"id":-1001156346945,"title":"SAYMON"...
Здесь **-1001156346945** - искомый ID канала.

2.2. Создать чат:

2.2.1. Отправить боту любое сообщение.

2.2.2. Перейти по ссылке в любом web-браузере, вставив в неё токен своего бота (без пробелов и знаков <>):
https://api.telegram.org/bot<токен_бота>/getUpdates

2.2.3. Найти текст со словами **chat** и **id**, например
... ":"K"},"chat":{"id":121399918,"first_ ...
Здесь **121399918** - искомый ID чата.

2.3. Настроить группу:

2.3.1. Добавить бота в группу.

2.3.2. Отправить боту в группу любое сообщение, начав его со знака @.

2.3.3. Перейти по ссылке в любом web-браузере, вставив в неё токен своего бота (без пробелов и знаков <>):

```
https://api.telegram.org/bot<токен_бота>/getUpdates
```

2.2.4. Найти текст со словами **chat** и **id**, например

```
... ":"K"},"chat":{"id":-209194473,"first_ ...
```

Здесь **-209194473** - искомый ID группы.

3. В настройках Telegram-уведомлений ввести токен (ID) бота и ID канала/чата/группы в соответствующие поля.

4.5 Интеграция со сторонними системами

Система предусматривает возможность импортировать данные с других мониторинговых платформ и экспортировать метрики в платформы для визуализации данных.

4.5.1 Интеграция с Zabbix

Для импорта данных из Zabbix необходимо:

1. Проверить доступ от сервера Центрального Пульта до сервера Zabbix, выполнив команду (в одну строку), заменив **http://192.168.1.215** на адрес файла `api_jsonrpc.php` вашей инсталляции Zabbix:

   ```
   curl -H "Content-Type: application/json" -X POST -d
   '{"jsonrpc":"2.0","method":"apiinfo.version","id":1,"auth":null,"params"
   {}}' http://192.168.1.215/zabbix/api_jsonrpc.php
   ```

   При наличии доступа сервер Zabbix ответит похожим сообщением:

   ```
   {"jsonrpc":"2.0","result":"3.2.1","id":1}
   ```

2. Добавить в Zabbix пользователя с правами на чтение необходимых хостов или групп хостов.
3. На сервере системы создать объект, в который будут импортироваться данные о хостах из Zabbix, и запомнить его ID.
4. Открыть конфигурационный файл сервера `/etc/saymon/saymon-server.conf` и добавить в него секцию **zabbix**:

   ```
   "zabbix": [
   {
      "url" : "http://192.168.1.215/zabbix/api_jsonrpc.php",
      "user" : "saymon",
      "password" : "saymon_user_password",
      "parent_id" : "58b586d5c3a2f96642e25537"
   },
   ...
   ]
   ```

   где:

   - **url** - адрес файла `api_jsonrpc.php` вашей инсталляции Zabbix, который был использован на шаге 1, например,
     `http://192.168.1.215/zabbix/api_jsonrpc.php;`
   - **user** - имя пользователя, добавленного в Zabbix на шаге 2;
   - **password** - пароль пользователя, добавленного в Zabbix на шаге 2;
   - **parent_id** - ID объекта, созданного на шаге 3.

   > *Подробную информацию о файле* `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

5. Для каждого дополнительного сервера Zabbix добавить новый JSON-документ с его данными в массив:

```
"zabbix" : [
   {
      "url" : "http://192.168.1.215/zabbix/api_jsonrpc.php",
      "user" : "saymon",
      "password" : "saymon_user_password",
      "parent_id" : "58b586d5c3a2f96642e12345"
   },
   {
      "url" : "http://192.168.1.180:8020/zabbix/api_jsonrpc.php",
      "user" : "saymon",
      "password" : "saymon_user_password",
   }
]
```

Для применения изменений необходимо перезапустить службу **saymon-server**:

```
sudo service saymon-server restart
```

По умолчанию данные на сервере системы обновляются каждые 2 минуты. Для изменения периода обновления данных необходимо добавить в конфигурационный файл сервера `/etc/saymon/saymon-server.conf` параметр **polling_period** со значением периода обновления данных в миллисекундах:

```
"zabbix" : [
   {
      "url" : "http://192.168.1.215/zabbix/api_jsonrpc.php",
      "user" : "saymon",
      "password" : "saymon_user_password",
      "parent_id" : "58b586d5c3a2f96642e12345"
      "polling_period": 30000
   }
]
```

Возможно изменять опции для запросов

- хостов (hosts),
- триггеров (triggers),
- данных (data).

Все возможные поля фильтров представлены в документации по Zabbix API:

- *hosts*,
- *triggers*,
- *data*.

```
"zabbix" : [{
  ...
     "request_options": {
        "hosts": {
           "filter":  {
              ...
           }
        },
        "triggers": {
           "filter": {
              ...
           }
        },
        "data": {
           "filter": {
              ...
           }
        }
     }
...
}]
```

Чтобы убрать фильтр из опции запроса, необходимо указать параметру **filter** значение **null**.

Пример (убрать поле **filter** из опций запроса триггеров Zabbix):

```
"zabbix" : [{
...
  "triggers": {
     "filter": null
  }
...
}]
```

Обнаруженным объектам по умолчанию присваивается класс **Info**. Чтобы присвоить новым объектам другие классы, необходимо задать массив **zabbix.classes** и указать в нём соответствия идентификаторов классов и имён объектов:

```
"zabbix" : [{
...
      "classes" : {
        "4" : "CPU",
        "619503be0ffb595aebe22222" : "Memory|General"
      },
...
]
```

Порядок применения правил привязки не гарантируется.

*Если назначить двум разным классам одинаковые регулярные выражения или использовать выражение **".*"** (любой текст), то нет гарантии, что сначала будет применено правило, указанное выше в списке.*

Для каждого Zabbix-подключения массив настраивается индивидуально. Регулярное выражение по имени объекта нечувствительно к регистру символов.

При необходимости связать класс с несколькими именами необходимо указать все имена в одном регулярном выражении с использованием оператора ИЛИ, в противном случае применится только одно соответствие.

Пример:
"619503be0ffb595aebe22222": "Memory|General"

Чтобы выводить параметры и результаты запросов в лог, необходимо задать параметр **zabbix.debug** со значением **true**:

```
"zabbix" : [{
  ...
    "debug": true,
  ...
}]
```

4.5.2 Интеграция с Grafana

> ⚠ *Все действия, связанные с интеграцией, осуществляются в интерфейсе Grafana.*

Для экспорта метрик в Grafana необходимо:

1. Авторизоваться в Grafana.
2. С главной страницы рабочего пространства перейти в **Configurations - Data Sources** через боковое меню и добавить data source (источник данных) нажатием на соответствующую кнопку (Рис. 4.5.2.1):

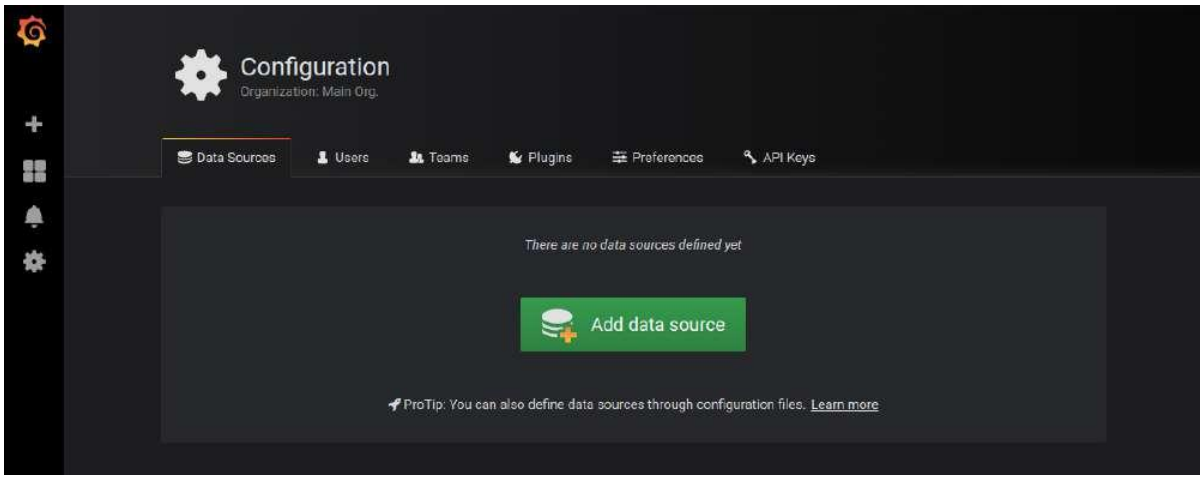


Рис. 4.5.2.1. *Добавление источника данных в Grafana*

3. В открывшемся окне типов источников данных выбрать **SAYMON** (Рис. 4.5.2.2):

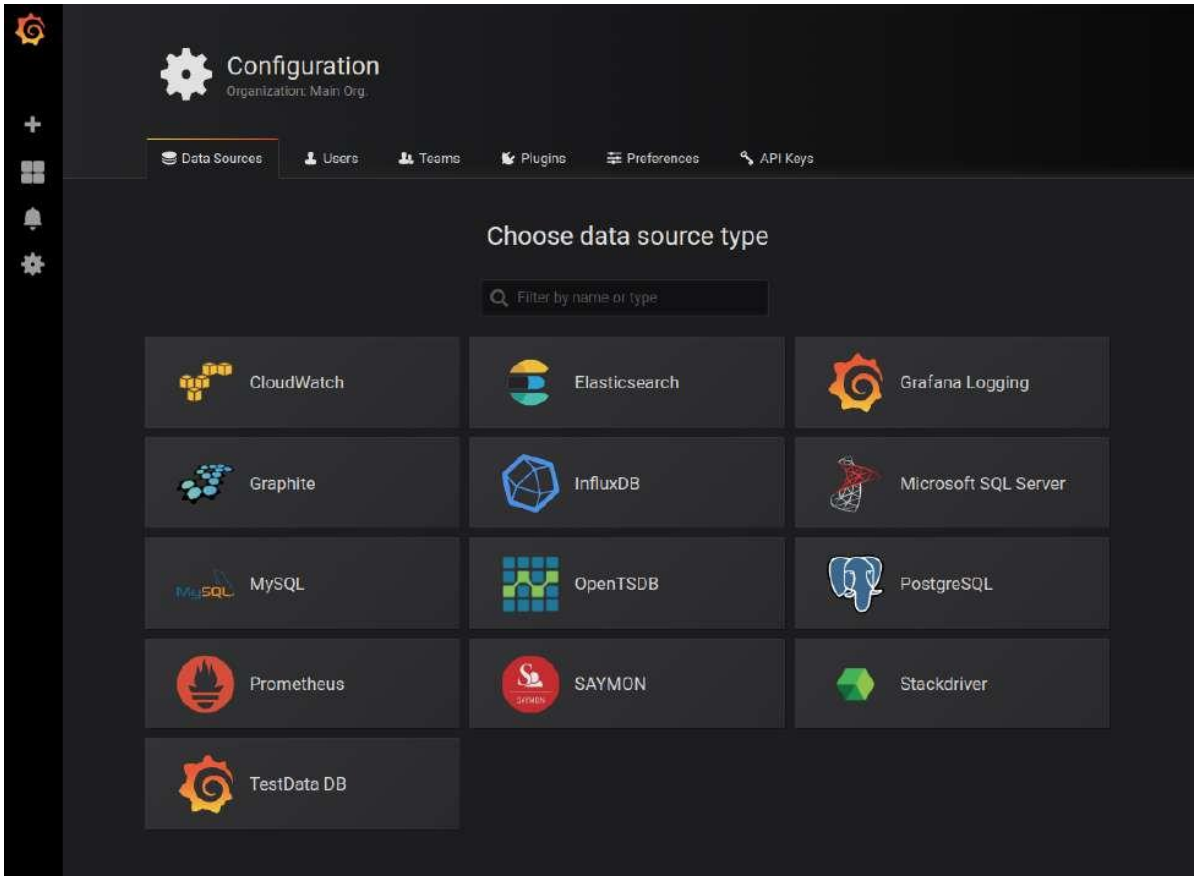


Рис. 4.5.2.2. *Выбор источника данных в Grafana*

4. В окне настроек необходимо указать (Рис. 4.5.2.3):
    - имя источника данных;
    - URL-адрес, откуда будут собираться данные;
    - режим доступа, как будут обрабатываться запросы к источнику данных: **server** или **browser**;
    - имена cookies-файлов, которые должны быть перенаправлены в источник данных;
    - тип аутентификации;
    - логин и пароль, используемые в Центральном Пульте.

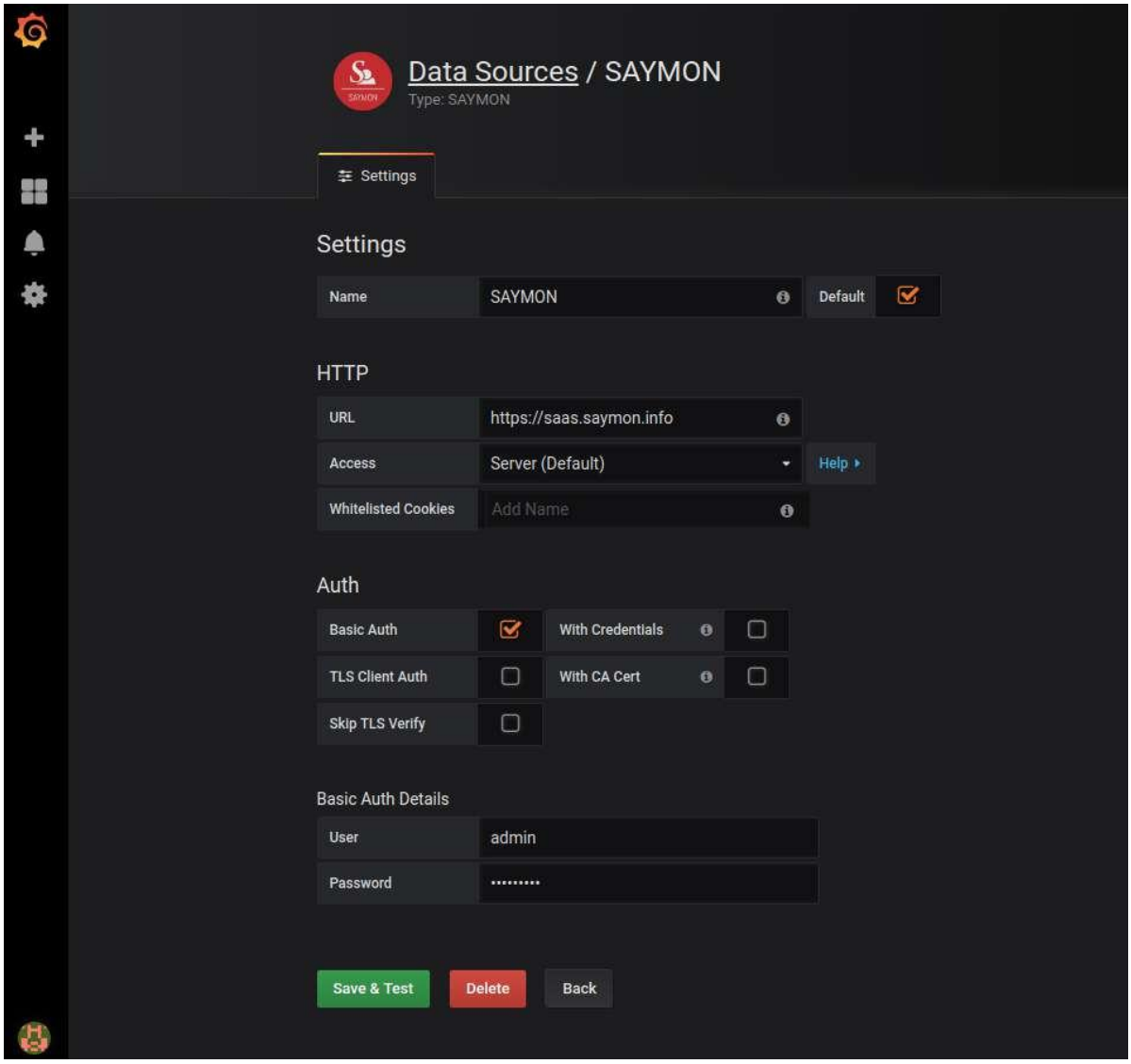


Рис. 4.5.2.3. *Настройка источника данных в Grafana*

5. Через боковое меню перейти в окно управления дашбордами (**dashboards** - **manage**) и приступить к созданию нового дашборда нажатием соответствующей кнопки (Рис. 4.5.2.4):

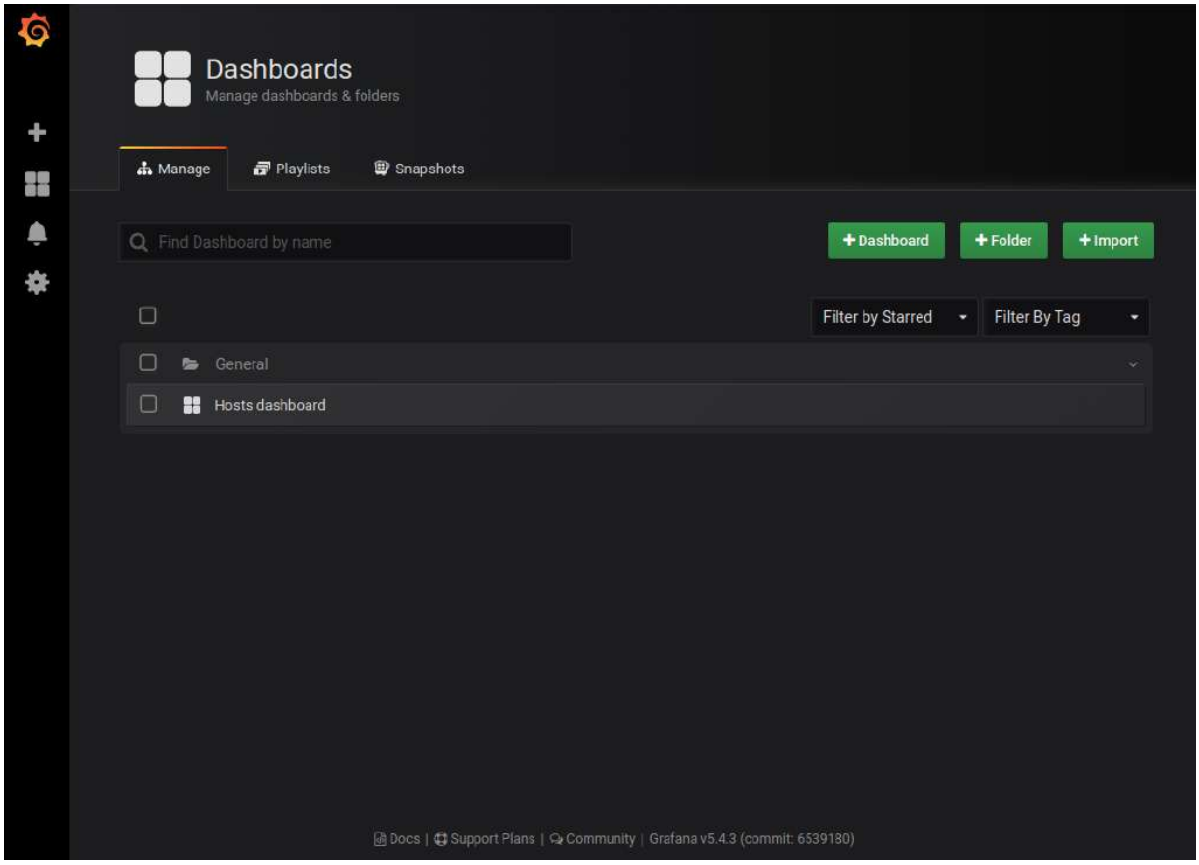


Рис. 4.5.2.4. *Создание нового дашборда в Grafana*

6. Выбрать тип отображения дашборда. Например, график (Рис. 4.5.2.5):

Рис. 4.5.2.5. *Выбор типа отображения дашборда в Grafana*

7. Вызвать контекстное меню нажатием на стрелку рядом с заголовком панели и перейти в режим редактирования.

8. В окне редактирования перейти во вкладку **Metrics** и добавить объекты, к которым у пользователя, под чьей учётной записью совершён вход, есть доступ в Центральном Пульте (Рис. 4.5.2.6). Указать:
    - имя или ID объекта;
    - метрику, которую необходимо отобразить на графике;
    - отображать ли путь до объекта.

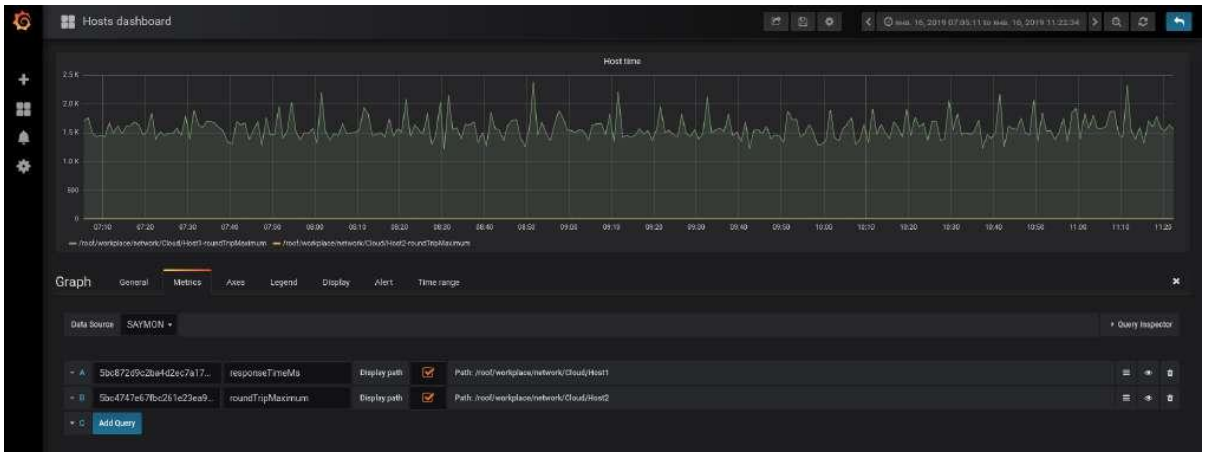


Рис. 4.5.2.6. *Отображение дашборда в Grafana*

9. Настроить временной период, за который данные отображаются на графике, и интервал их обновления.

4.5.3 Интеграция с Keycloak

Для пользователей Центрального Пульта предусмотрена возможность авторизации посредством сервера Keycloak.

Клиенты в Keycloak - это параметры соединения с Keycloak-сервером для внешних приложений, которые авторизуют пользователей.

Главной особенностью Backend-клиента является наличие собственной аутентификации: такой клиент получает конфиденциальный доступ по секретному ключу.

Главной особенностью Frontend-клиента является отсутствие собственной аутентификации: данный клиент получает публичный доступ.

Для настройки взаимодействия между сервером Центрального Пульта и внешним сервером Keycloak необходимо получить хотя бы один из параметров:

- Realm Certificate, которым сервер Keycloak подписывает пользовательские токены,
- файл настроек соединения для Backend-клиента.

Для возможности авторизации пользователей посредством Keycloak через web-интерфейс Центрального Пульта необходимо получить файл настроек соединения Frontend-клиента.

⚠ *Настройка клиентов и получение параметров осуществляются в консоли администратора Keycloak.*

4.5.3.1 Настройка сервера

Для получения Realm Certificate необходимо:

1. Выбрать требуемую область (Realm) из списка (Рис. 4.5.3.1.1; здесь **SAYMON**):

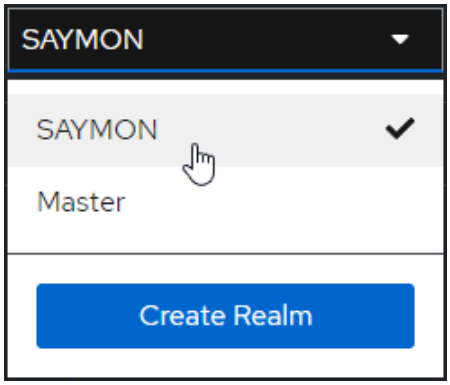


Рис. 4.5.3.1.1. *Выбор области Keycloak*

2. В секции **Realm settings** на вкладке **Tokens** уточнить алгоритм подписи токенов в поле **Default Signature Algorithm** (Рис. 4.5.3.1.2; здесь **RS256**):

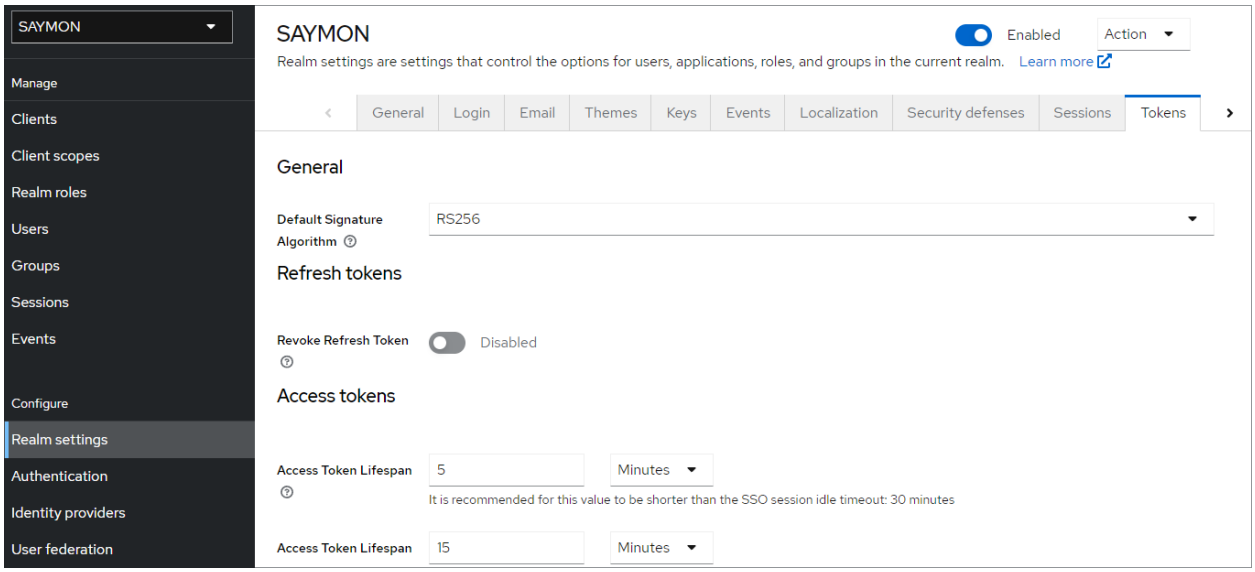


Рис. 4.5.3.1.2. *Проверка алгоритма подписи токенов*

3. На вкладке **Keys** найти сертификат для данного алгоритма (Рис. 4.5.3.1.3):

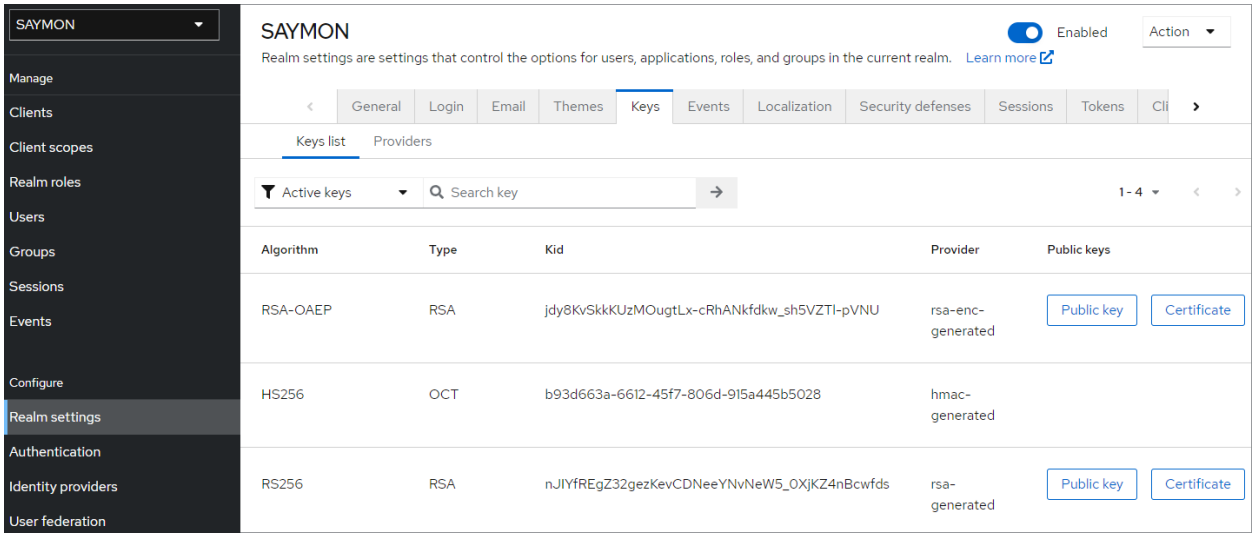


Рис. 4.5.3.1.3. *Получение сертификата*

4. Кнопка Certificate в соответствующей строке открывает искомый сертификат (Рис. 4.5.3.1.4):

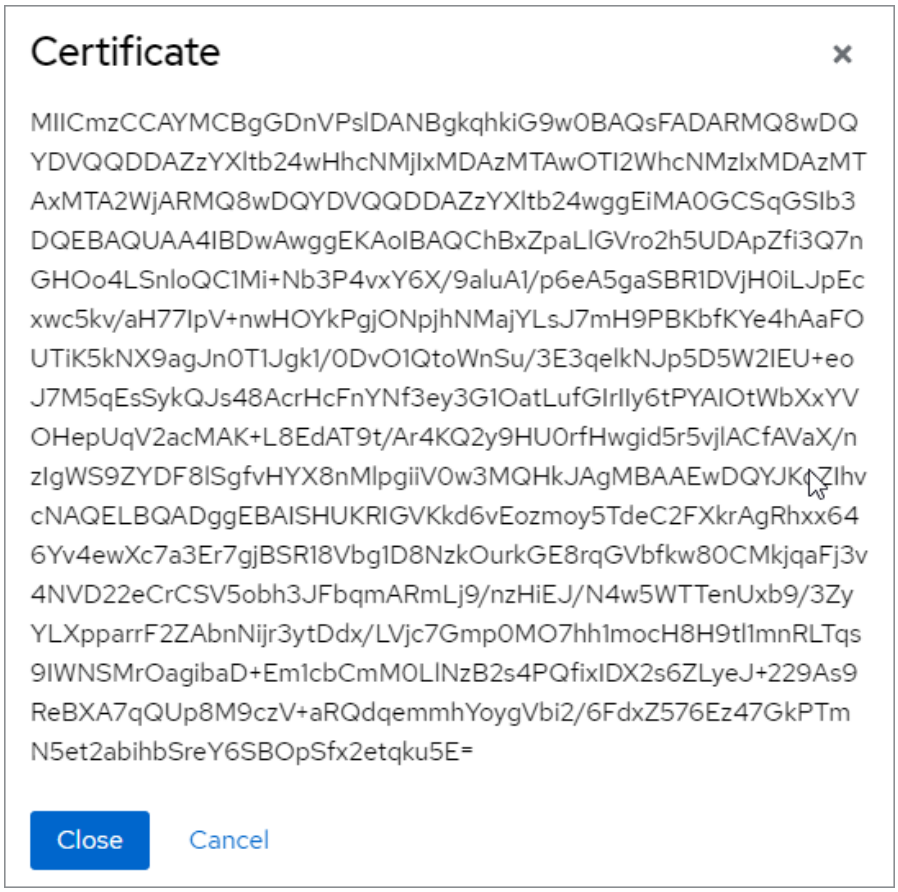


Рис. 4.5.3.1.4. *Пример данных сертификата*

> *Данные рекомендуется сохранить в файл* ***keycloak_realm_certificate****.*

Для настройки Backend-клиента необходимо:

1. Выбрать требуемую область (Realm) из списка (Рис. 4.5.3.1.1 на стр. 92).
2. В секции **Clients** нажать кнопку Create client (Рис. 4.5.3.1.5):

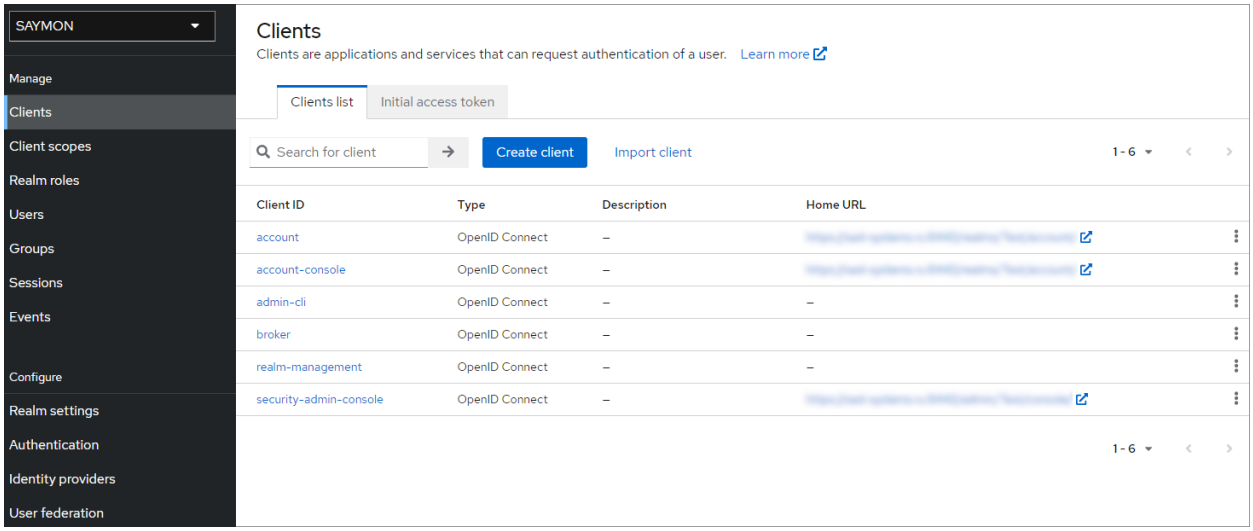


Рис. 4.5.3.1.5. *Создание клиента*

3. Ввести имя Backend-клиента в поле **Client ID** (Рис. 4.5.3.1.6):

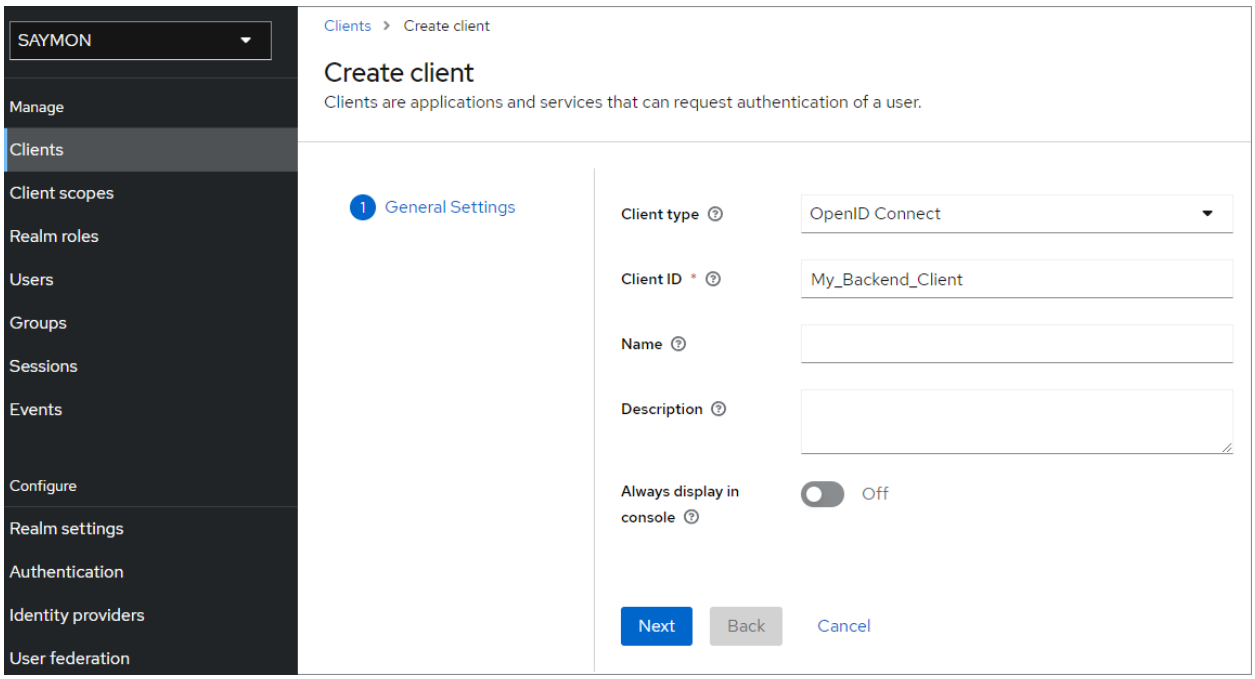


Рис. 4.5.3.1.6. *Ввод имени Backend-клиента*

и нажать кнопку Next.

4. Слайдер **Client authentication** установить в положение **On** (Рис. 4.5.3.1.7)**:**

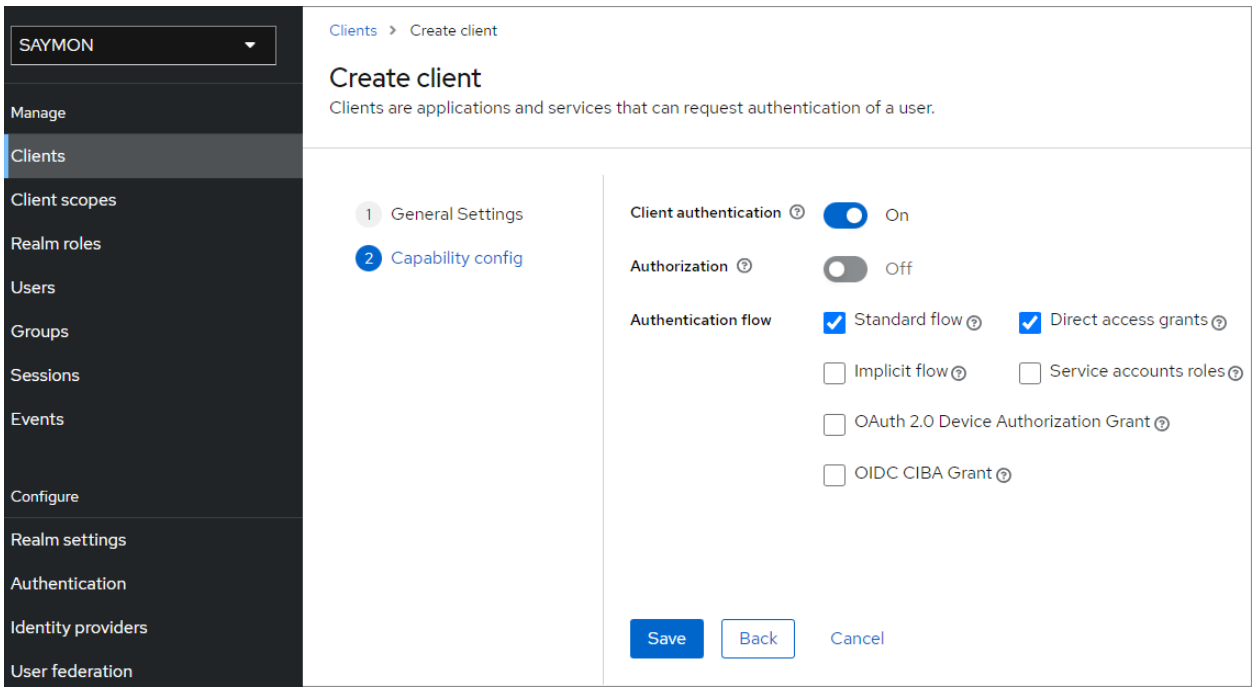


Рис. 4.5.3.1.7. *Включение клиентской аутентификации*

и нажать кнопку Save.

5. В списке **Action** выбрать **Download adapter config** (Рис. 4.5.3.1.8):

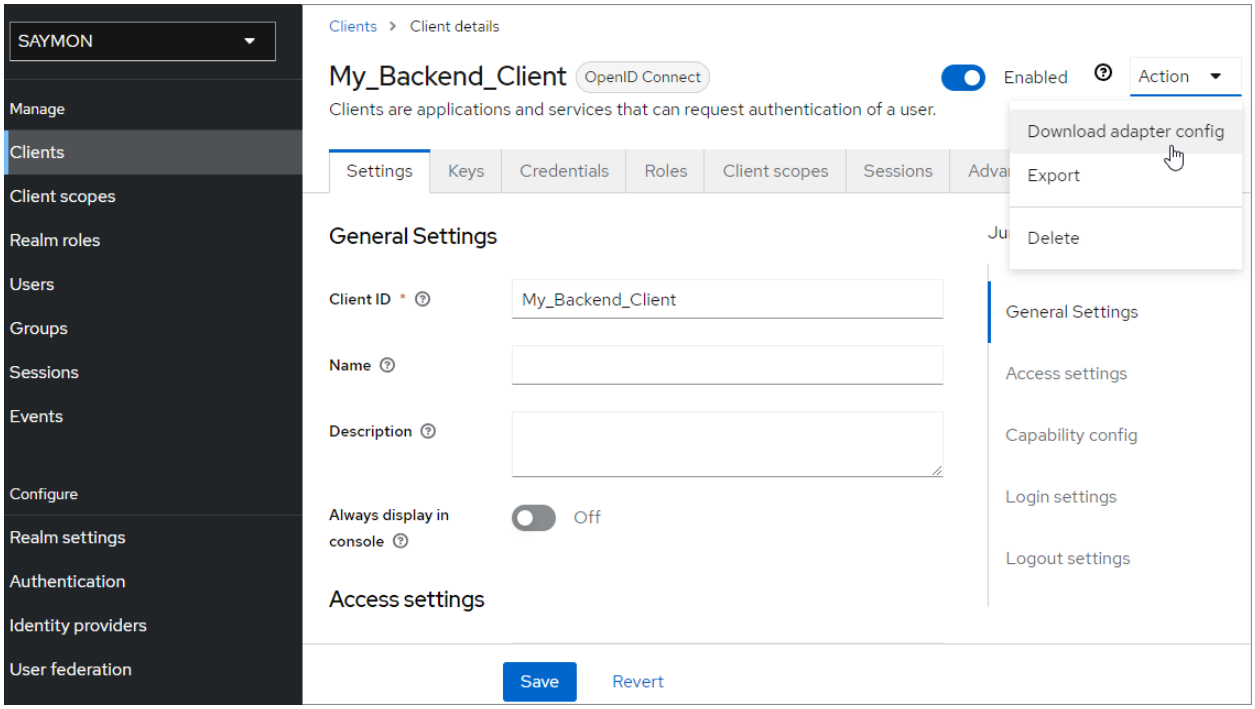


Рис. 4.5.3.1.8. *Получение файла конфигурации Backend-клиента*

6. Нажать кнопку Download, чтобы получить файл настроек соединения для Backend-клиента в формате JSON (Рис. 4.5.3.1.9):

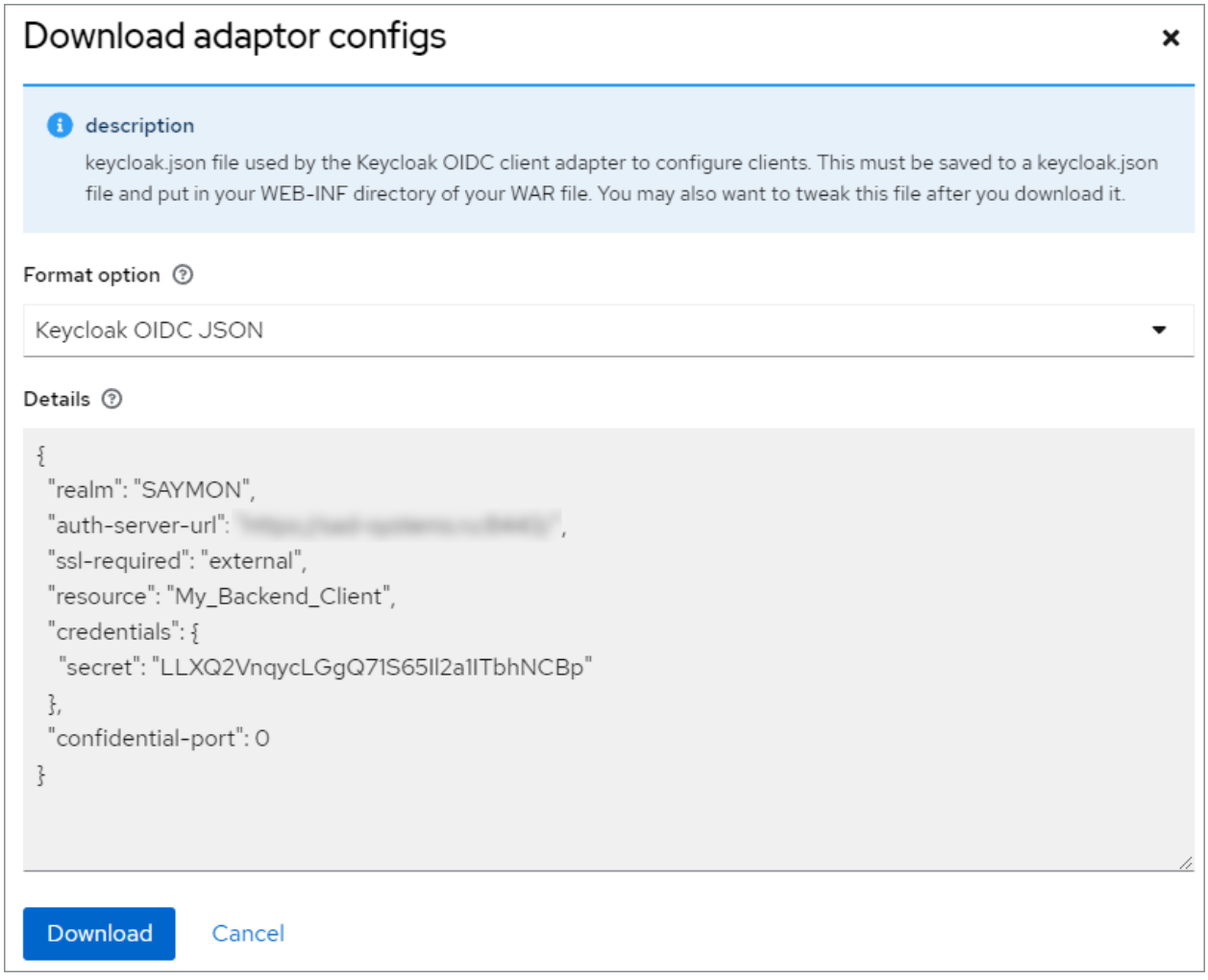


Рис. 4.5.3.1.9. *Пример данных файла конфигурации Backend-клиента*

Для взаимодействия с сервером Keycloak на сервере Центрального Пульта необходимо:

1. Добавить раздел **keycloak** в конфигурационный файл сервера `/etc/saymon/saymon-server.conf`:

```
"keycloak": {
   "realm_certificate": "<данные keycloak_realm_certificate>",
   "realm_certificate_file": "/etc/saymon/keycloak_realm_certificate",
   "config_file": "/etc/saymon/keycloak_backend.json"
}
```

Подробную информацию о файле `/etc/saymon/saymon-server.conf` *см. в подразделе 4. 1.3 "Конфигурация сервера" на стр. 28.*

где:

- **realm_certificate** - данные Keycloak Realm Certificate;
- **realm_certificate_file** - путь к файлу с Keycloak Realm Certificate;
- **config_file** - путь к файлу конфигурации, сгенерированному Keycloak-сервером (для backend-клиента).

Параметры перечислены в порядке понижения приоритета при одновременном указании. Для работы с Keycloak достаточно указать один из параметров.

2. Перезапустить службу saymon-server:

```
sudo service saymon-server restart
```

4.5.3.2 Настройка web-интерфейса

Для настройки Frontend-клиента необходимо:

1. Выбрать требуемую область (Realm) из списка (Рис. 4.5.3.1.1 на стр. 92).
2. В секции **Clients** нажать кнопку Create client (Рис. 4.5.3.1.5 на стр. 93).
3. Ввести имя Frontend-клиента в поле **Client ID** (Рис. 4.5.3.2.1):

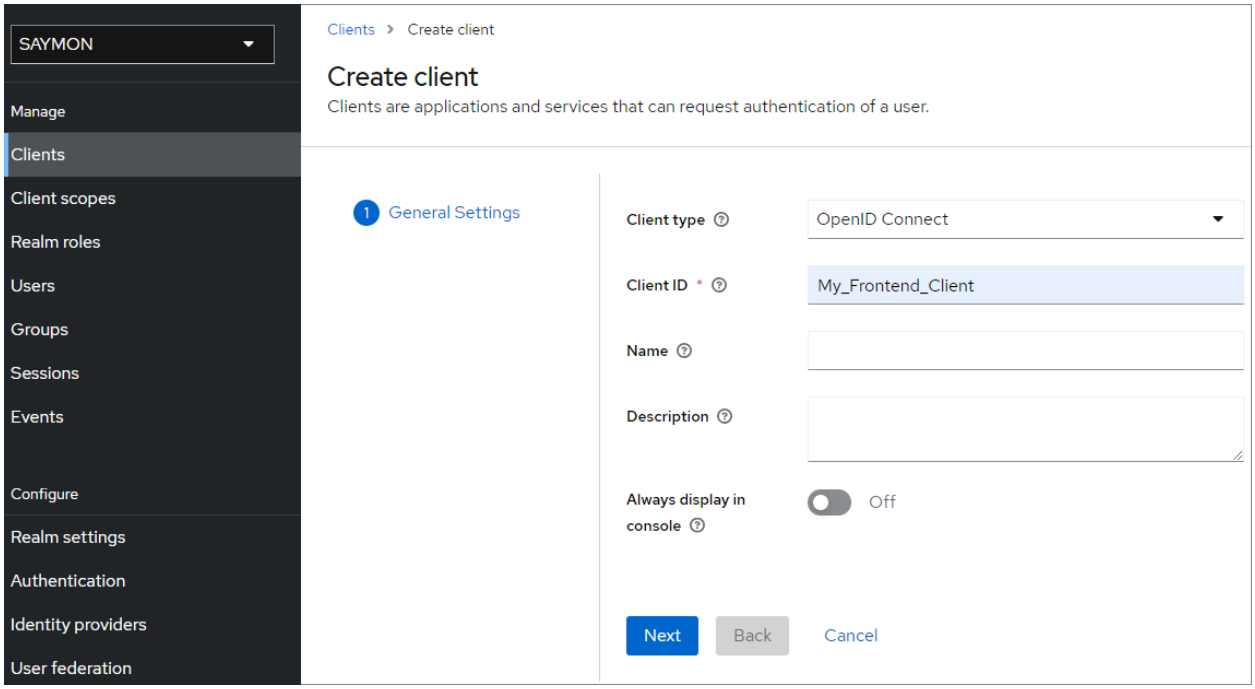


Рис. 4.5.3.2.1. *Ввод имени Frontend-клиента*

и нажать кнопку Next.

4. Слайдер **Client authentication** установить в положение **Off** (Рис. 4.5.3.2.2):

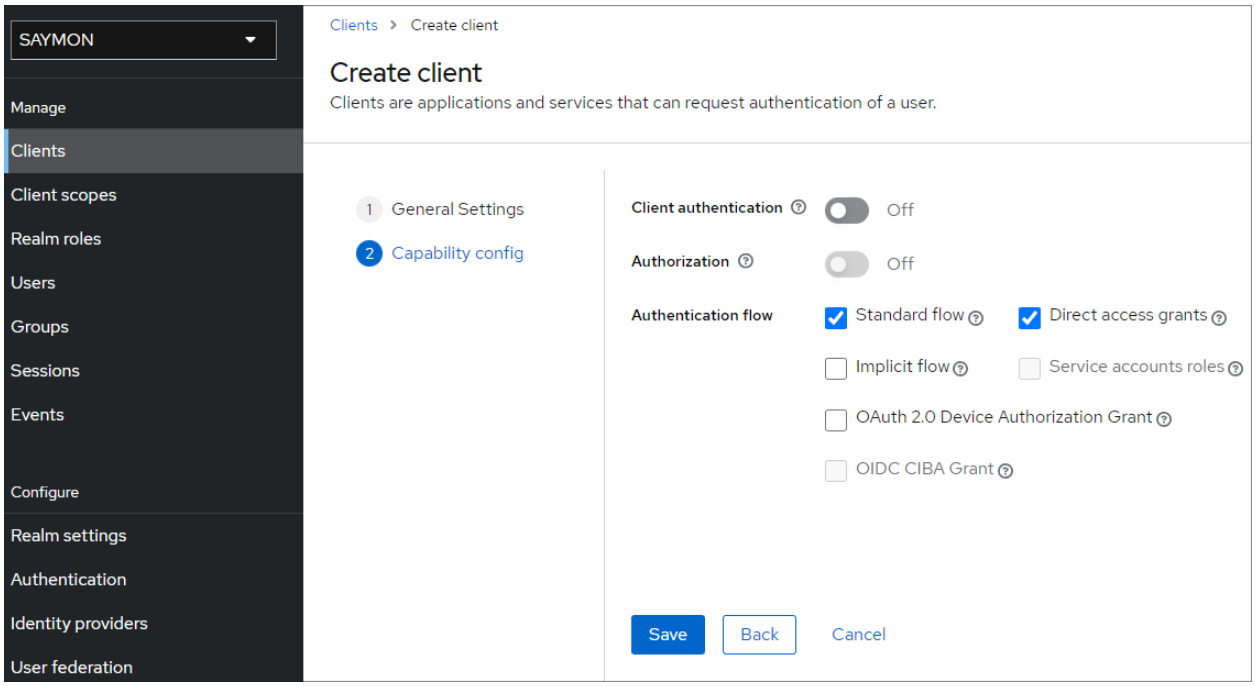


Рис. 4.5.3.2.2. *Отключение клиентской аутентификации*

и нажать кнопку Save.

5. На вкладке **Settings** в разделе **Access settings** указать корректные URI-адреса, с которых будет осуществляться доступ к Keycloak-серверу, и на которые будут перенаправляться пользователи после успешной аутентификации и выхода (Рис. 4.5.3.2.3):

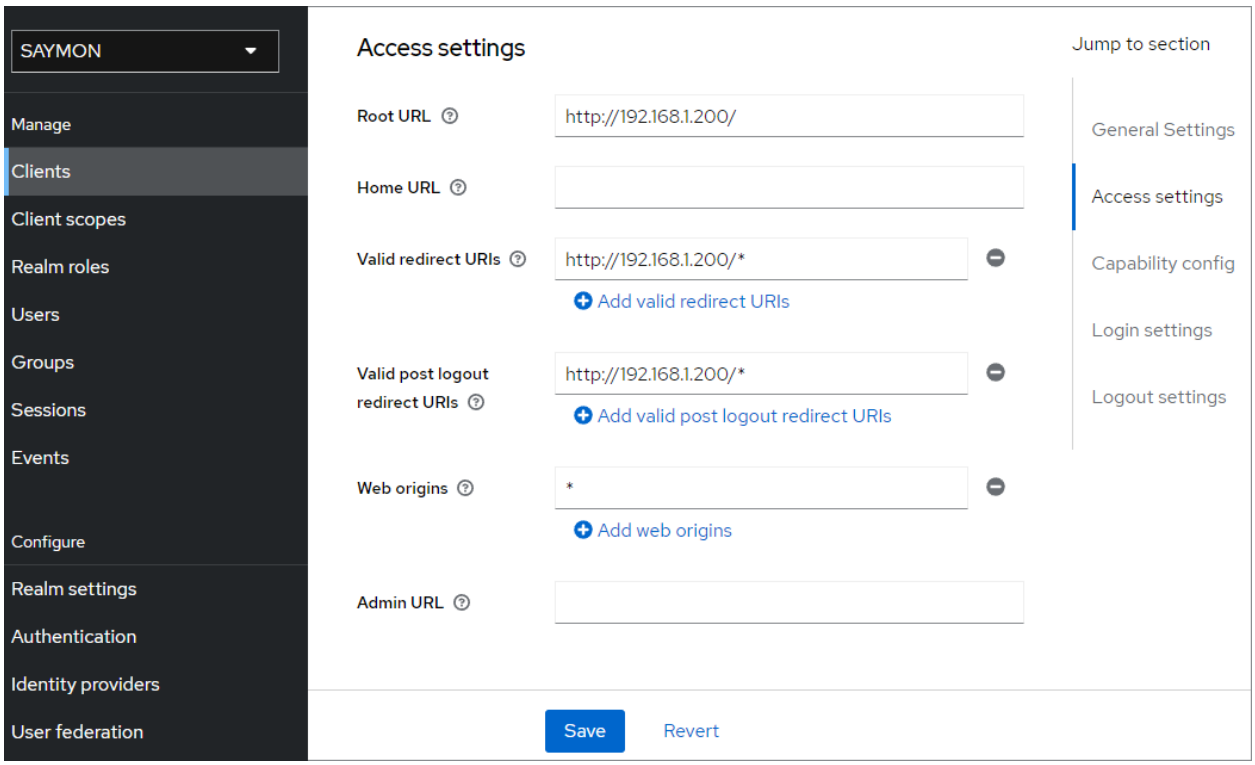


Рис. 4.5.3.2.3. *Отключение клиентской аутентификации*

и нажать кнопку Save

6. В списке **Action** выбрать **Download adapter config** (Рис. 4.5.3.2.4):

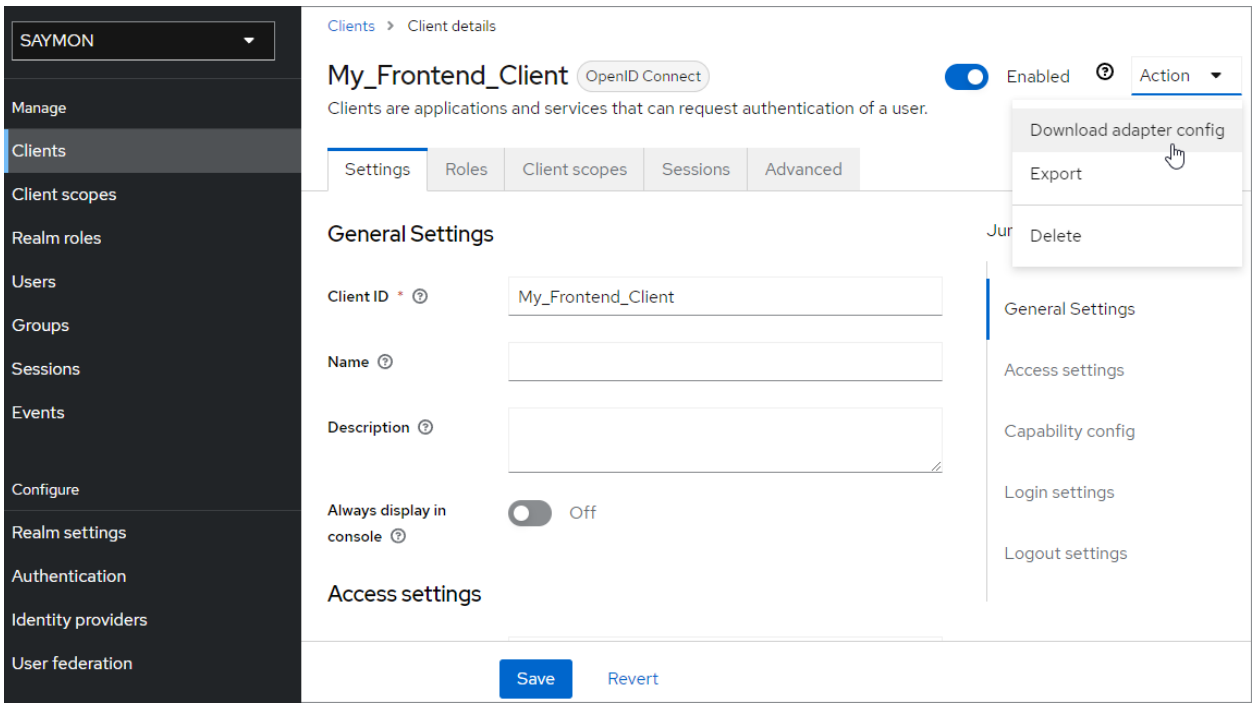


Рис. 4.5.3.2.4. *Отключение клиентской аутентификации*

7. Нажать кнопку Download, чтобы получить файл настроек соединения для Frontend-клиента в формате JSON (Рис. 4.5.3.2.5):

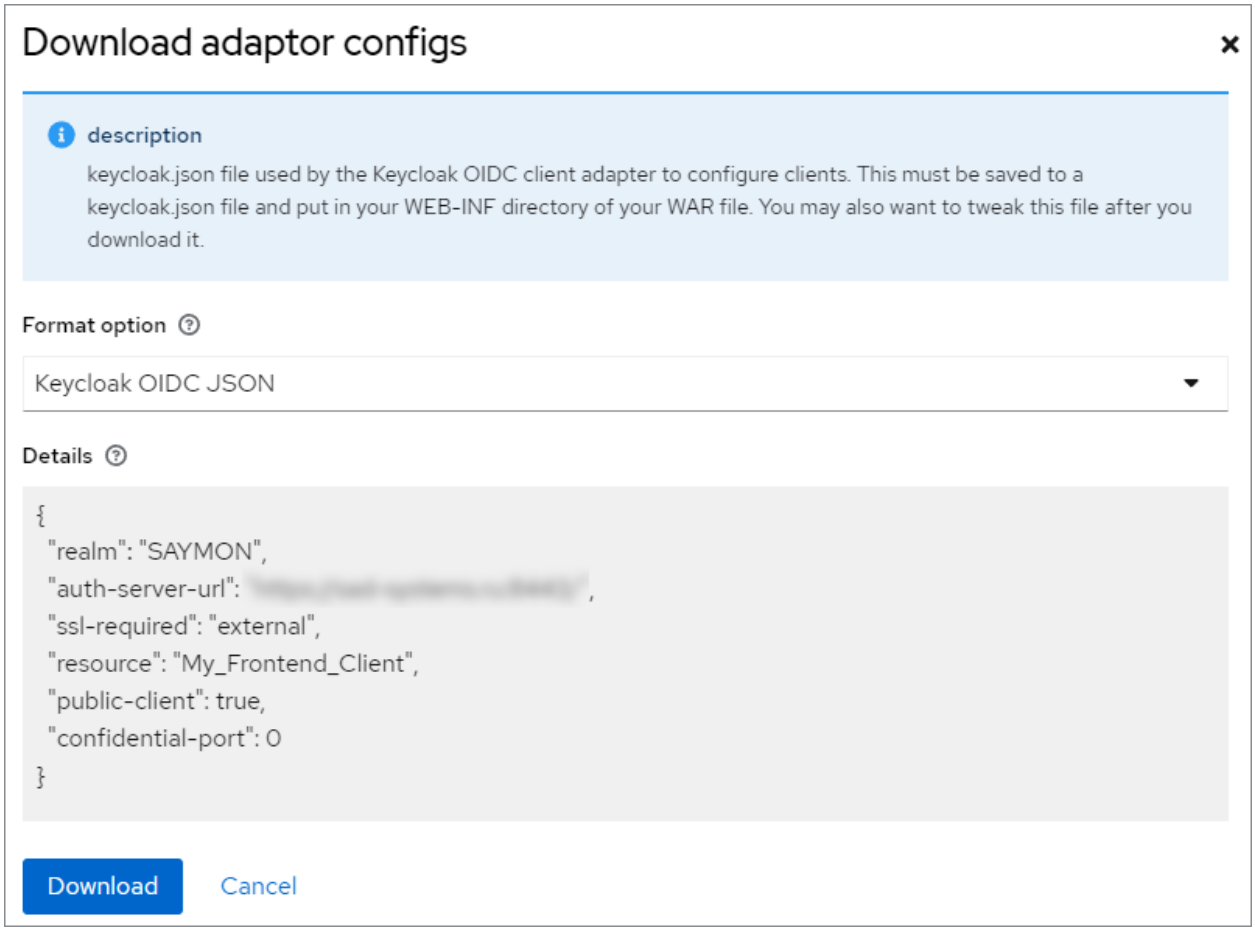


Рис. 4.5.3.2.5. *Отключение клиентской аутентификации*

Для активации формы авторизации через Keycloak в web-интерфейсе системы необходимо:

1. Добавить на сервере Центрального Пульта раздел **authentication** в конфигурационный файл клиента `/usr/local/saymon/client/client-config.js`

```
authentication: {
    availableMethods: ['native', 'keycloak'],
    methodByDefault: 'keycloak',
    services: {
        keycloak: {
            configUri: 'keycloak_frontend.json'
        }
    }
}
```

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

где:

- **availableMethods** - список доступных для пользователей методов аутентификации;
- **methodByDefault** - метод аутентификации, который будет предложен пользователю сразу при отображении аутентификационной формы;
- **configUri** - файл настроек соединения с Keycloak-сервером (для frontend-клиента).

> *Файл необходимо поместить в каталог с конфигурационным файлом клиента.*

2. Обновить страницу Центрального Пульта в web-браузере.

4.6 Управление логированием

Для хранения и дальнейшего анализа изменений в системной работе агента Центральный Пульт использует логирование.

Логи агента хранятся в папках:

- Linux, Wiren Board 6 - **/var/log/saymon**;
- Mac OS X - **/opt/saymon-agent/log**;
- Windows - **папка_установки_агента\log**.

Также в журналы записывается информация о следующих событиях:

- дата и время удачных и неудачных попыток входа пользователей в систему;
- причина и время выхода пользователей из системы;
- дата, время и инициатор выполнения операций;
- факт использования прав администратора;
- факт доступа пользователей к основным конфигурационным данным платформы;
- запуск и остановка сервисов аудита действий пользователя.

Все записи имеют временной штамп, признак инициатора (ID пользователя или процесса) и признак сессии, если событие было инициировано пользователем.

4.6.1 Конфигурация log-файлов

Конфигурация log-файлов агента осуществляется в файле **/opt/saymon-agent/conf/logback-upstart.xml**.

Секция настройки debug-режима:

```
-<appender name="FILE-DEBUG"
class="ch.qos.logback.core.rolling.RollingFileAppender">
   <file>/var/log/saymon/saymon-agent.debug.log</file>
      -<rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <!-- Daily rollover -->
      -<fileNamePattern>
      /var/log/saymon/saymon-agent.debug.%d{yyyy-MM-dd}.log.gz
      </fileNamePattern>
      <!-- Keep 10 days' worth of history -->
      <maxHistory>10</maxHistory>
      </rollingPolicy>
   -<encoder>
      -<pattern>
      %d{dd.MM.yyyy HH:mm:ss.SSS} [%-15thread] %-5level %logger{36} -
      %msg%n
      </pattern>
   </encoder>
</appender>
```

где:

- <file>/var/log/saymon/saymon-agent.debug.log</file> – размещение log-файла;
- <fileNamePattern>... .gz</fileNamePattern> – указание на архивацию файлов в формат .gz;
- <maxHistory>10</maxHistory> – длительность хранения файлов в днях.

Для отключения debug-режима закомментировать соответствующую строку в секции ROOT следующим образом:

```
<!--<appender-ref ref="FILE-DEBUG"/>-->
```

Настройки и структура секции базового логирования аналогична секции настройки debug-режима:

```
<appender name="FILE-INFO"
class="ch.qos.logback.core.rolling.RollingFileAppender">
...
</appender>
```

4.6.2 Конфигурация ротации log-файлов

Конфигурация ротации log-файлов сервера выполняется в файле **/etc/logrotate.d/saymon**.

Параметры файла конфигурации ротации log-файлов:

Параметр	Описание
/var/log/saymon/saymon-rest-server.log /var/log/saymon/saymon-server.log	Размещение log-файлов.
daily	Ежедневная ротация.
missingok	Продолжать ротацию без ошибки, если отсутствует один из файлов.
rotate N	Длительность хранения файлов в днях.
compress	Архивация файлов в формат .gzip.
notifempty	Не производить ротацию лога, если он пуст.
copytruncate	Писать лог в один файл, урезая его после каждого шага ротации.

4.6.3 Просмотр информации о Журнале событий

Информацию о содержании журнала событий возможно просмотреть с помощью REST API метода:

GET /node/api/event-log/info

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/event-log/info

curl -X GET $url -u $login:$password
```

4.6.4 Назначение ответственного за событие

При возникновении критической ситуации возможно установить ответственного за неё пользователя двумя способами:

1. Через Web UI:

   1.1. В панели режимов отображения открыть Журнал событий кнопкой  .

   1.2. Нажать правой кнопкой мыши на требуемое событие, затем выбрать пункт "Назначить ответственного".

   1.3. Выбрать ответственного за событие из выпадающего списка.

   Имя ответственного пользователя отобразится в соответствующем столбце (Рис. 4.6.4.1):

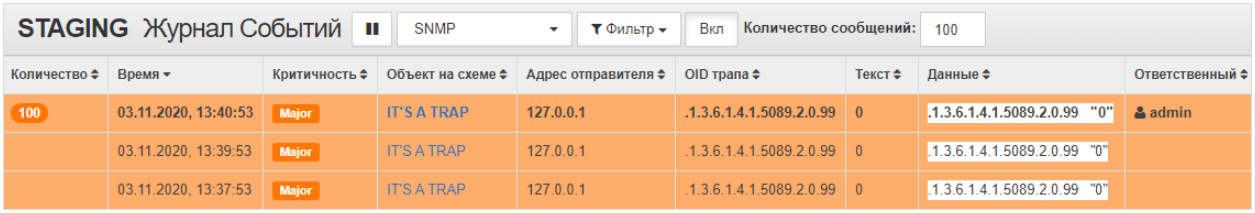


Рис. 4.6.4.1. *Журнал событий*

2. REST API методом:

PATCH /node/api/event-log/:id/assignee

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
record_id=<...>
url=https://$saymon_hostname/node/api/event-log/$record_id/assignee

curl -X PATCH $url -u $login:$password -H "Content-Type: application/json" \
--data '{"userId": "..."}'
```

4.6.5 Установка ограничения для логирования

Центральный Пульт позволяет установить максимальный объём хранилища двумя способами:

1. Через Web UI:

1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

1.2. Перейти в раздел "Журнал событий" (Рис. 4.6.5.1):

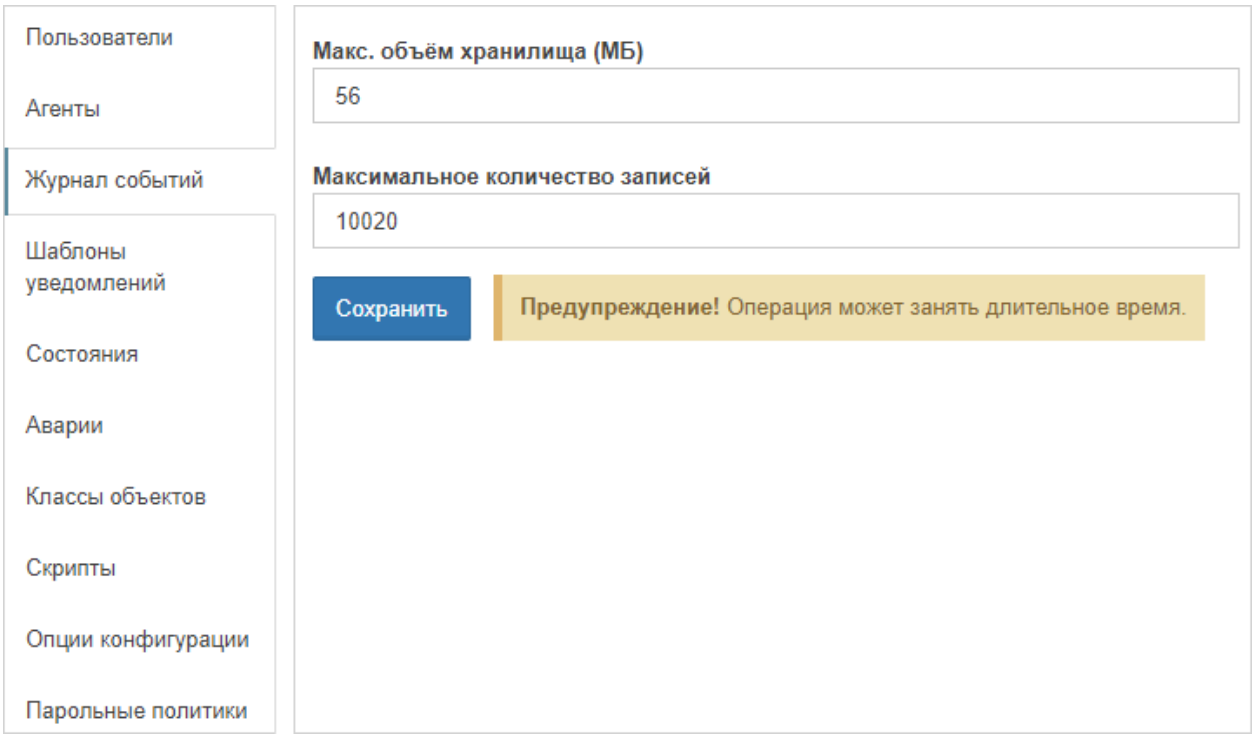


Рис. 4.6.5.1. *Журнал событий*

1.3. Заполнить поля требуемыми значениями.

1.4. Сохранить изменения.

2. Через REST API метод:

PUT /node/api/event-log/limits

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/event-log/limits

curl -X PUT $url -u $login:$password -H "Content-Type: application/json" \
--data '{"maxBytes": 1024, "maxRecords": 100}
```

4.6.6 Удаление всех SNMP-Trap'ов из Журнала событий

Для очистки Журнала событий необходимо:

1. Выполнить в терминале следующие команды:

```
mongo saymon
>db.eventLog.drop()
```

2. После сброса проверить создание новой коллекции командой:

```
>db.eventLog.stats()
```

3. Если ответ выглядит следующим образом, необходимо создать новую коллекцию:

```
{ "ok" : 0, "errmsg" : "ns not found" }
```

   - создание новой коллекции с лимитом по объёму в байтах:

     ```
     >db.createCollection("eventLog", {capped:true, size: 100000000})
     ```

   - создание новой коллекции без лимита:

     ```
     >db.createCollection("eventLog")
     ```

4. Повторно проверить создание коллекции командой:

```
>db.eventLog.stats()
```

Ответ должен выглядеть примерно следующим образом:

```
{
"ns" : "saymon.eventLog",
"count" : 0,
"size" : 0,
"storageSize" : 100003840,
"numExtents" : 1,
"nindexes" : 1,
"lastExtentSize" : 100003840,
"paddingFactor" : 1,
"systemFlags" : 1,
"userFlags" : 0,
"totalIndexSize" : 8176,
"indexSizes" : {
"_id_" : 8176
},
"capped" : true,
"max" : NumberLong("9223372036854775807"),
"ok" : 1
}
```

4.6.7 Удаление логов

Удаление логов осуществляется следующей командой в консоли:

```
sudo rm -rf /var/log/upstart/*
```

Также возможно удалить логи с помощью REST API:

```
DELETE /node/api/event-log/:id
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
record_id=<...>
url=https://$saymon_hostname/node/api/event-log/$record_id

curl -X DELETE $url -u $login:$password
```

4.7 Управление учётными записями пользователей

Раздел содержит информацию об администрировании пользовательских учётных записей.

Группы пользователей позволяют упростить настройку и редактирование прав пользователей. При необходимости настроить нескольким пользователям идентичные

- права на операции,
- доступы к объектам,
- права на просмотр событий в Журнале событий,

достаточно создать группу пользователей, настроить ей необходимые права и добавить в эту группу нужных пользователей. В дальнейшем возможно редактировать права группы. Пользователь может принадлежать к неограниченному числу групп.

Права пользователя и его групп суммируются. Чтобы пользователь мог выполнять те или иные действия в системе, нужно, чтобы соответствующие права были либо у самого пользователя, либо у какой-то из его групп.

4.7.1 Авторизация пользователей через LDAP

Для авторизации пользователей с помощью сервера службы каталогов (Active Directory, OpenLDAP, и т.д.) необходимо добавить секцию LDAP в конфигурационный файл сервера **/etc/saymon/saymon-server.conf**.

> *Подробную информацию о файле* **/etc/saymon/saymon-server.conf** *см. в подразделе* *4.1.3 "Конфигурация сервера" на стр. 28*.

При первом входе пользователя, аутентифицированного через LDAP, для него автоматически создаётся учётная запись в системе. У таких пользователей на вкладке "Общие" есть поле "Источник" со значением "LDAP", в списке пользователей они отмечаются иконкой  .

> *Если логин LDAP-пользователя совпадает с логином другого пользователя, уже зарегистрированного в системе, LDAP-пользователь авторизоваться не сможет.*

Пользователям из LDAP, у которых ещё нет учётной записи в Центральном Пульте, можно ограничить вход в систему, подготовив группы для нужных пользователей. Чтобы учётная запись создавалась только для пользователей, у которых имя группы на LDAP-сервере совпадает с именем существующей (созданной ранее) группы в Центральном Пульте, в конфигурационном файле сервера необходимо указать параметр **ldap.create_user_for_existing_group_only** со значением **true**:

```
"ldap" : {
   ...
      " create_user_for_existing_group_only": true,
   ...
}
```

При использовании протокола LDAP во время авторизации пользователя в системе могут быть автоматически созданы группы с именами, эквивалетными именам групп из LDAP, которые отсутствуют в Центральном Пульте. Авторизовавшийся пользователь автоматически добавляется в импортированные группы. В конфигурационном файле сервера для этого необходимо указать параметр **ldap.import_non_existing_groups** со значением **true**:

```
"ldap" : {
   ...
      "import_non_existing_groups": true,
   ...
}
```

LDAP-группы - группы, созданные при авторизации LDAP-пользователей - обозначаются в списке иконкой  . Для таких групп недоступна смена имени, в качестве источника указан LDAP.

Группы, созданные в Центральном Пульте, имена которых совпадают с именами групп из LDAP, можно обновить до LDAP-групп. В конфигурационном файле сервера для этого необходимо указать параметр **ldap.update_existing_groups** со значением **true**:

```
"ldap" : {
   ...
      "update_existing_groups": true,
   ...
}
```

В этом случае при авторизации LDAP-пользователя те группы в Центральном Пульте, для которых есть одноимённые группы среди групп пользователя в LDAP, будут обновлены до LDAP-групп.

⚠ *Если пользователь был исключён из группы в Центральном Пульте - аналога своей группы в LDAP, при следующем входе в систему пользователь автоматически будет возвращён в эту группу.*

4.7.2 Авторизация пользователей через Keycloak

Для авторизации пользователей с помощью внешнего сервера Keycloak необходимо:

- выполнить настройку клиентов и получить необходимые параметры в консоли администратора Keycloak;

  Описание необходимых действий в консоли администратора Keycloak приведено в подразделе 4.5.3 "Интеграция с Keycloak" на стр. 91.

- добавить секцию **Keycloak** в конфигурационный файл сервера `/etc/saymon/saymon-server.conf`;

  Подробную информацию о файле `/etc/saymon/saymon-server.conf` *см. в подразделе 4.1.3 "Конфигурация сервера" на стр. 28.*

При первом входе пользователя, аутентифицированного через Keycloak, для него автоматически создаётся учётная запись в системе. У таких пользователей на вкладке "Общие" есть поле "Источник" со значением " Keycloak", в списке пользователей они отмечаются иконкой  .

Если логин Keycloak-*пользователя совпадает с логином другого пользователя, уже зарегистрированного в системе,* Keycloak-*пользователь авторизоваться не сможет.*

4.7.3 Создание учётной записи

При создании пользователей и групп в именах и паролях допускаются любые символы, кроме символа @.

Создание учётных записей пользователей осуществляется двумя способами:

1. Через Web UI:

1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

1.2. Перейти в раздел "Журнал событий" (Рис. 4.7.2.1):

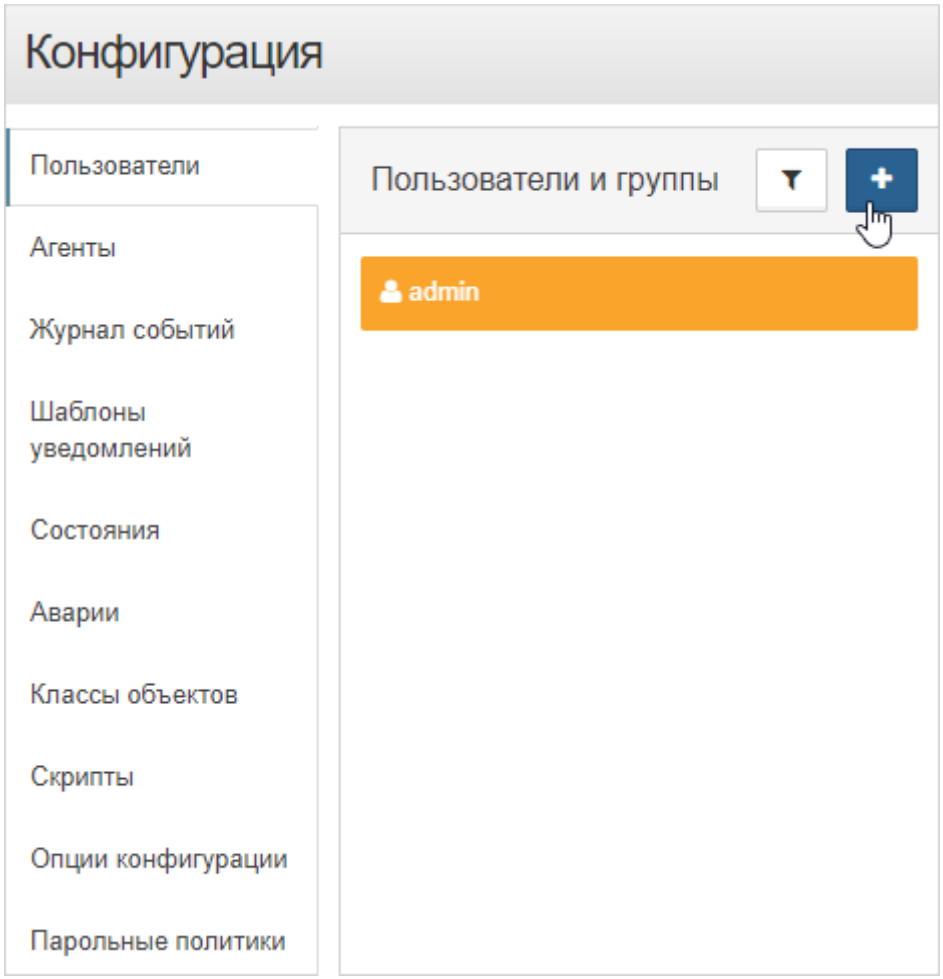


Рис. 4.7.2.1. *Журнал событий*

1.3. Нажать кнопку  и выбрать добавление пользователя.

1.4. Ввести логин пользователя, пароль и подтверждение.

1.5. Нажать кнопку  .

2. Через REST API:

POST /node/api/users

Пример (bash)

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/users

curl -X POST $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "login": "Bob",
   "password": "qwerty",
   "permissions": [
      "manage-objects",
      "manage-links"
   ]
}
EOF
```

4.7.4 Назначение пользователям прав доступа

> *Список операций с подробным описанием содержится в Приложении А на стр. 166.*

Настройка прав пользователей осуществляется двумя способами:

1. Через Web UI:

1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

1.2. Перейти в раздел "Пользователи".

1.3. Выбрать нужного пользователя или группу из списка.

1.4. Открыть вкладку "Права на операции" (Рис. 4.7.3.1):

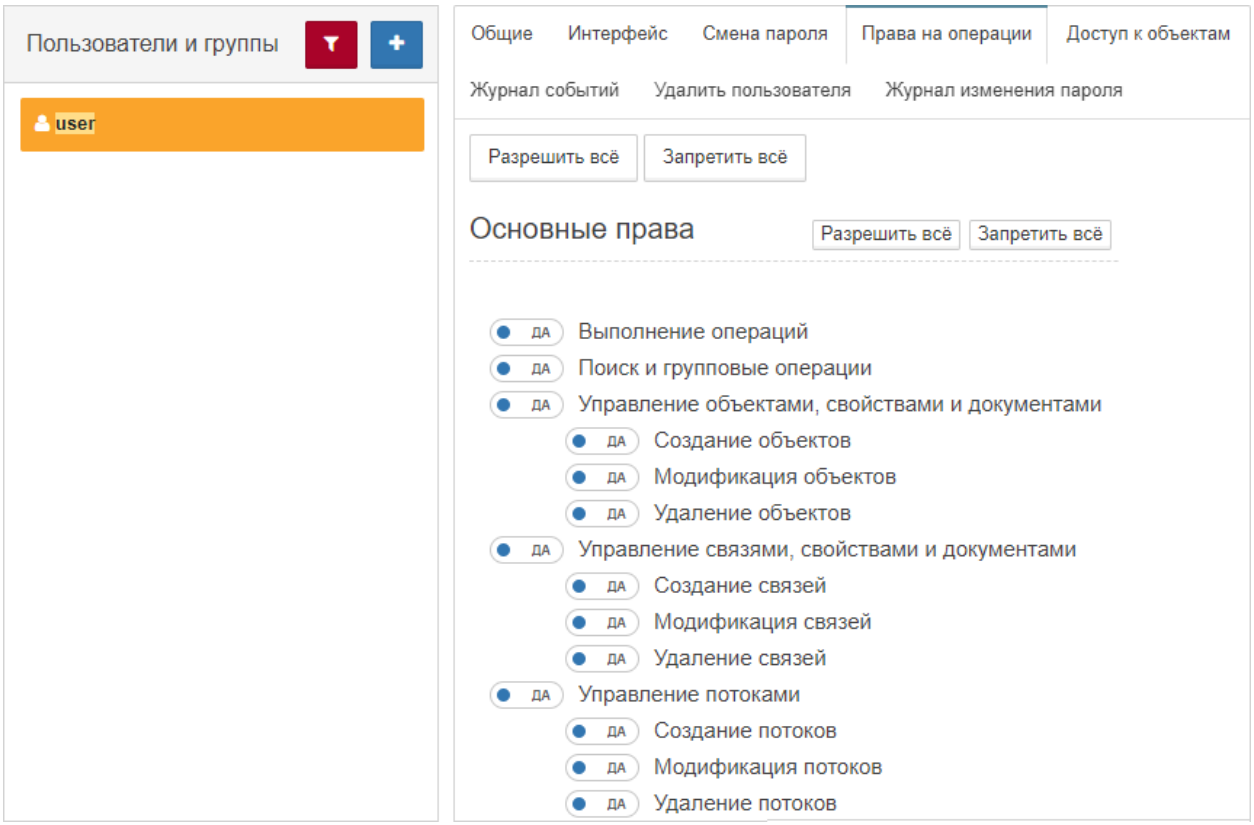


Рис. 4.7.3.1. *Права на операции*

1.5. Отредактировать права пользователя или группы.

> *Если пользователь принадлежит к группам, права пользователя и его групп суммируются.*

2. Через REST API:

```
PATCH /node/api/users/:id
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
user_id=<...>
url=https://$saymon_hostname/node/api/users/$user_id

curl -X PATCH $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "login": "Bob",
   "password": "qwerty",
   "permissions": [
      "manage-objects",
      "manage-links"
   ]
}
EOF
```

4.7.5 Назначение парольных политик

Настройка парольных политик осуществляется через Web UI.

1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".
2. Перейти в раздел "Парольные политики" (Рис. 4.7.4.1):

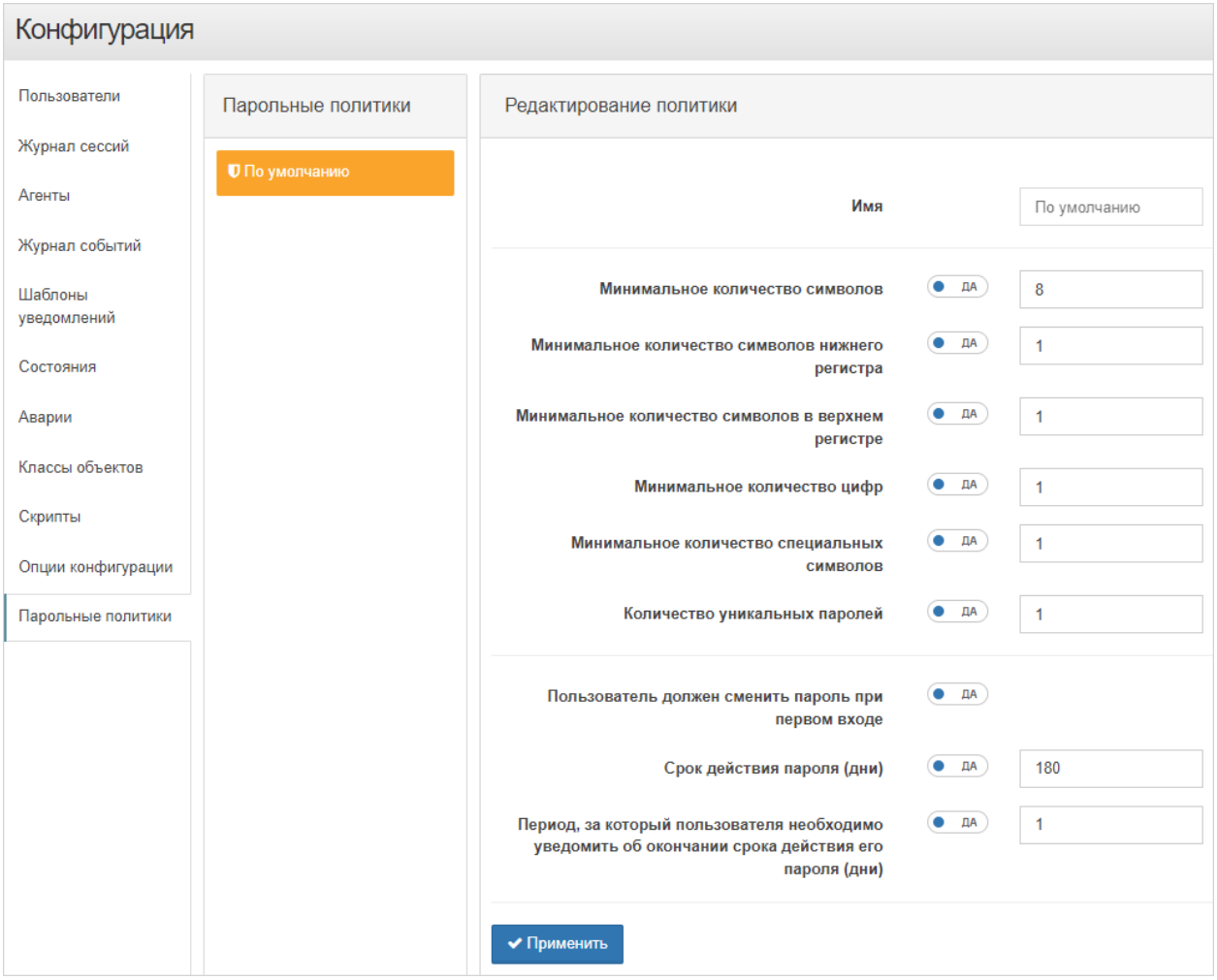
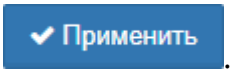


Рис. 4.7.4.1. *Парольные политики*

3. Установить необходимые требования к паролям пользователей.
4. Нажать кнопку 

.

4.7.6 Изменение пароля от учётной записи

Смена пароля пользователей осуществляется двумя способами:

1. Через Web UI:

1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

1.2. Перейти в раздел "Пользователи".

1.3. Выбрать нужного пользователя из списка.

1.4. Открыть вкладку "Смена пароля" (Рис. 4.7.5.1):

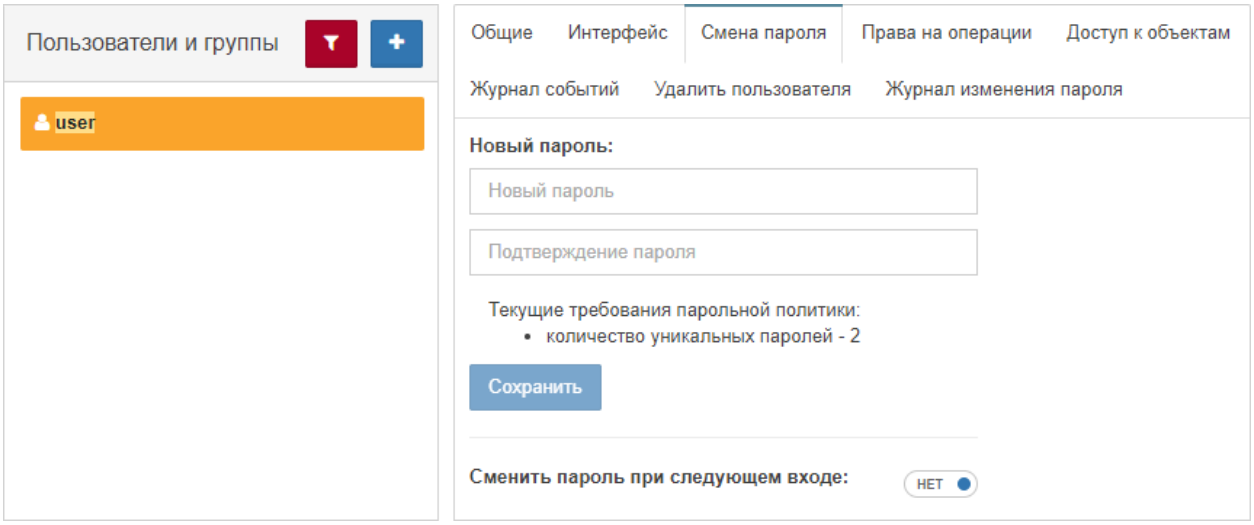


Рис. 4.7.5.1. *Смена пароля пользователя*

1.5. Ввести новый пароль пользователя и подтверждение.

> ⚠ *При смене собственного пароля необходимо также ввести текущий пароль.*

1.6. Нажать кнопку Сохранить.

2. Через REST API:

PUT /node/api/users/:id/password

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
user_id=<...>
url=https://$saymon_hostname/node/api/users/$user_id/password

curl -X PUT $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "currentPassword": "qwerty",
   "newPassword": "qwerty_qwerty"
}
EOF
```

4.7.7 Настройка доступа к объектам

Настройка доступа к объектам осуществляется через Web UI.

1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".
2. Перейти в раздел "Пользователи".
3. Выбрать нужного пользователя или группу из списка.
4. Открыть вкладку "Доступ к объектам" (Рис. 4.7.6.1):

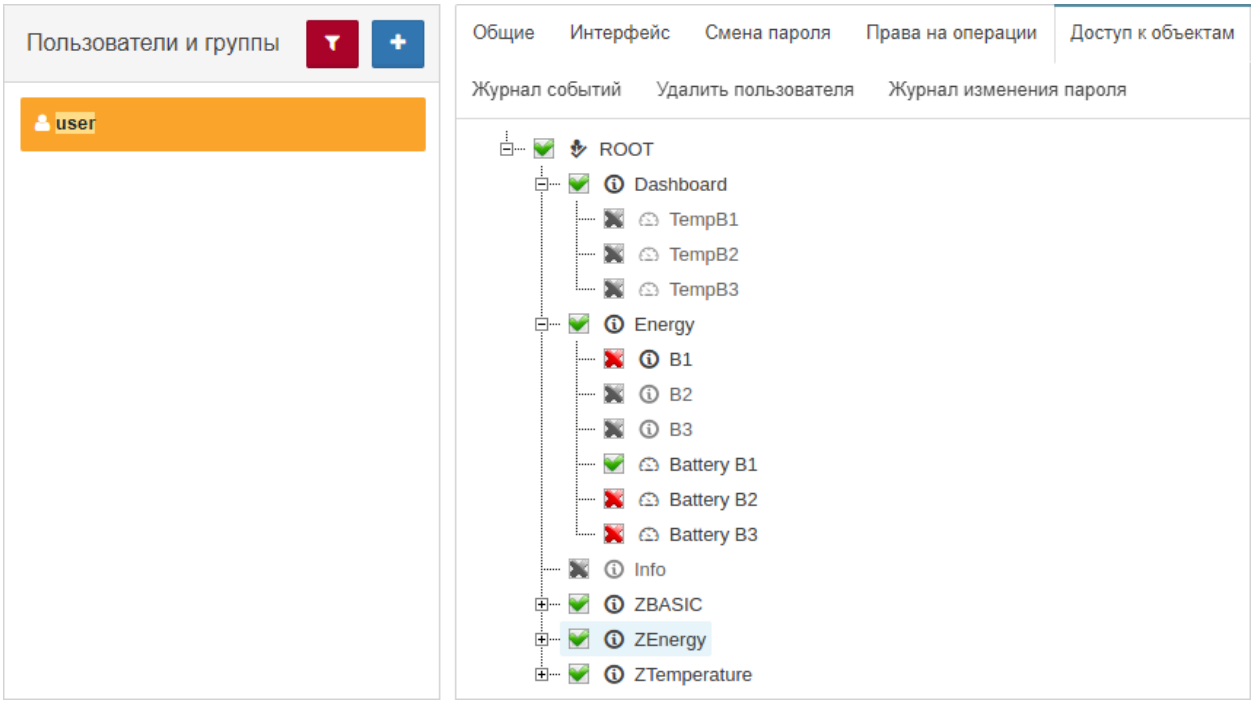


Рис. 4.7.6.1. *Доступ к объектам*

5. Отметить объекты, к которым необходимо предоставить/заблокировать доступ.

> ⚠ *Объекты, недоступные для группы, становятся недоступными для всех членов группы.*

4.7.8 Блокировка учётной записи

Блокировка учётных записей пользователей осуществляется двумя способами:

1. Через Web UI:

1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

1.2. Перейти в раздел "Пользователи".

1.3. Выбрать нужного пользователя из списка.

1.4. Открыть вкладку "Общие" (Рис. 4.7.7.1):

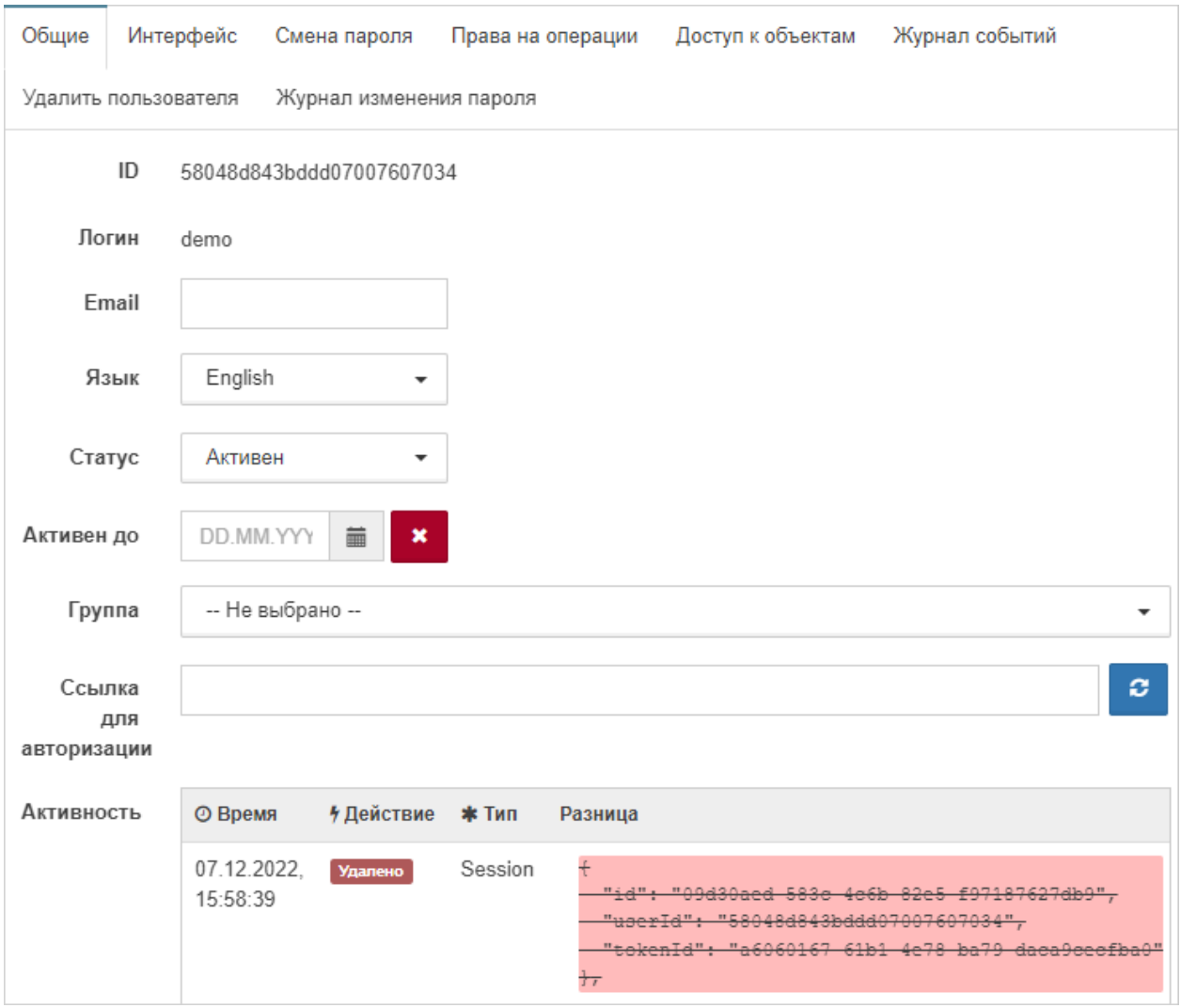


Рис. 4.7.7.1. *Общие параметры пользователя*

1.5. В выпадающем списке "Статус" выбрать "Заблокирован".

В поле **Активен до** можно задать дату и время блокировки учётной записи. При наступлении указанного времени статус учётной записи будет автоматически изменён на **Заблокирован**.

> ⚠ *В случае если указанное время меньше текущего, статус учётной записи немедленно изменяется на **Заблокирован**.*

2. Через REST API:

```
PATCH /node/api/users/:id
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
user_id=<...>
url=https://$saymon_hostname/node/api/users/$user_id/password

curl -X PATCH $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "status": "3",
}
EOF
```

4.7.9 Удаление учётной записи

Удаление учётных записей осуществляется двумя способами:

1. Через Web UI:

   1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".

   1.2. Перейти в раздел "Пользователи".

   1.3. Выбрать нужного пользователя из списка.

   1.4. Открыть вкладку "Удалить пользователя" (Рис. 4.7.8.1):

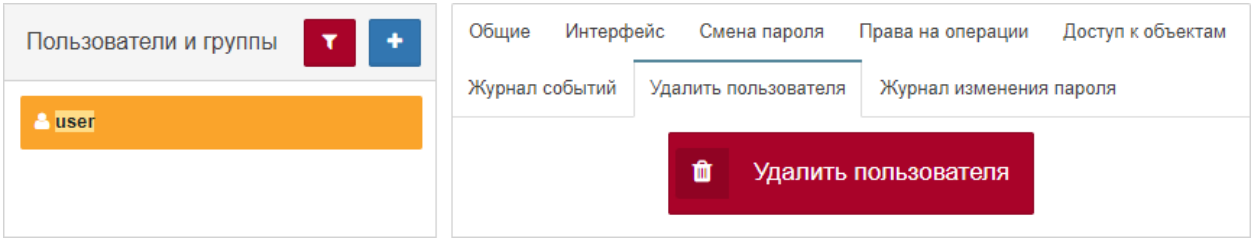


Рис. 4.7.8.1. *Удаление пользователя*

   1.5. Нажать кнопку удаления пользователя.

   1.6. Подтвердить удаление во всплывающем окне.

2. Через REST API:

```
DELETE /node/api/users/:id
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
user_id=<...>
url=http://$saymon_hostname/node/api/users/$user_id

curl -X DELETE $url -u $login:$password
```

> ⚠ *После удаления учётной записи пользователя LDAP/Keycloak с сервера системы пользователь по-прежнему сможет войти в систему под своими учётными данными, авторизовавшись через LDAP/Keycloak. Учётная запись при этом будет создана заново с правами по умолчанию; права, полученные через группы, останутся прежними.*
>
> *Чтобы заблокировать такому пользователю доступ к данным, достаточно сменить его статус на* ***Заблокирован****.*

4.8 Работа с объектами и связями

Вся управляемая инфраструктура в системе представлена в виде объектов и связей между ними.

Все метрики, характеризующие текущее состояние платформы, связываются с отдельными объектами мониторинга. Контроль загрузки и наличия свободных ресурсов осуществляется стандартными средствами платформы.

4.8.1 Создание объекта

Создать объект можно двумя способами:

1. Через Web UI:

   1.1. Нажать кнопку 

на панели инструментов.

   1.2. В окне "Новый объект" ввести имя объекта и выбрать класс объекта (Рис. 4.8.1.1):

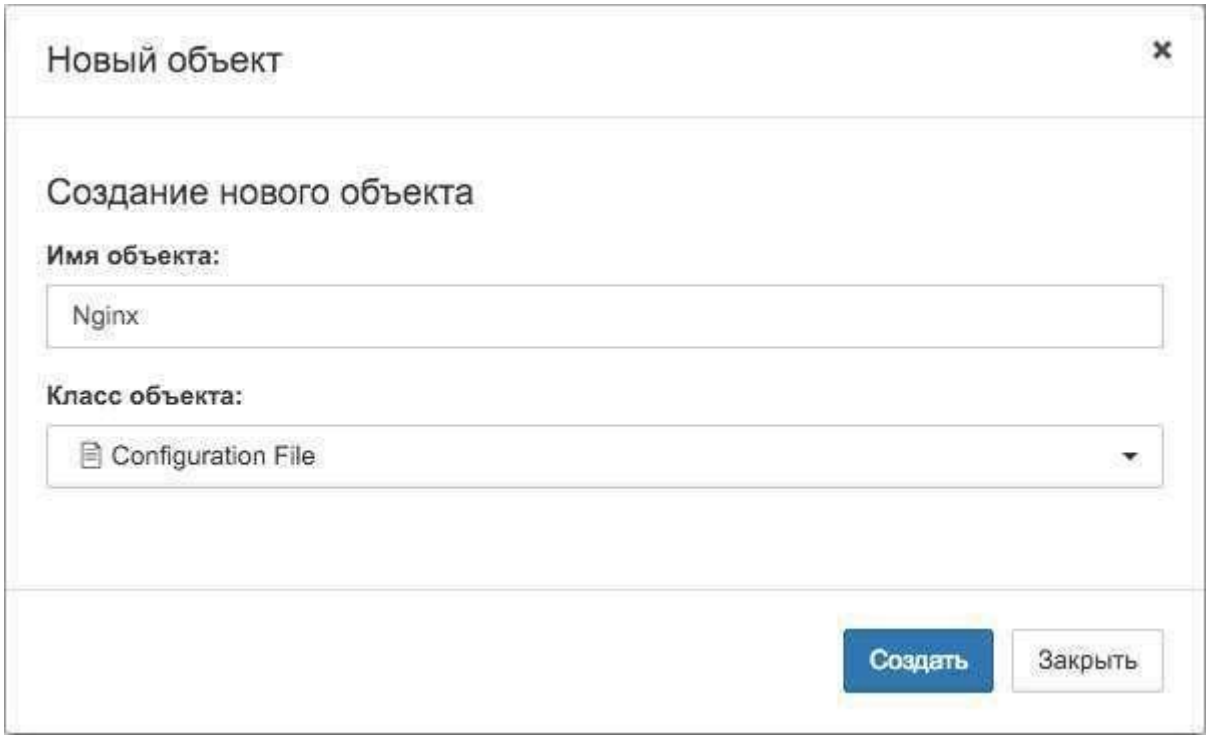


Рис. 4.8.1.1. *Окно создания нового объекта*

   1.3. Нажать кнопку 

.

2. Через REST API:

POST /node/api/objects

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/objects

curl -X POST $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "name": "New Object",
   "parent_id": "1",
   "class_id": "3"
}
EOF
```

⚠ *"parent_id" - ID родительского объекта для создаваемого.*

4.8.2 Клонирование объекта

Объект клонируется со всеми своими документами, параметрами, свойствами дочерними объектами и связями.

Клонировать объект можно двумя способами:

1. Через Web UI:
   1.1. Вызвать контекстное меню клонируемого объекта щелчком правой кнопкой мыши.
   1.2. Выбрать пункт меню "Клонировать".
2. Через REST API:

```
POST /node/api/objects/:id/clone
```

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
object_id=<...>
url=https://$saymon_hostname/node/api/objects/$object_id/clone

curl -X POST $url -u $login:$password
```

4.8.3 Удаление объекта

Удалить объект можно тремя способами:

1. Через Web UI (контекстное меню объекта):
   1.1. Вызвать контекстное меню удаляемого объекта щелчком правой кнопкой мыши.
   1.2. Выбрать пункт меню "Удалить".
   1.3. Подтвердить удаление объекта во всплывающем окне.
2. Через Web UI (режим удаления элементов):
   2.1. Нажать кнопку  на панели хлебных крошек.
   2.2. Нажать на такую же иконку на удаляемом объекте.
   2.3. Подтвердить удаление объекта во всплывающем окне.
3. Через REST API:

   DELETE /node/api/objects/:id

   Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
object_id=<...>
url=https://$saymon_hostname/node/api/objects/$object_id

curl -X DELETE $url -u $login:$password
```

4.8.4 Создание ссылки на объект

Ссылка представляет собой особый тип объекта и служит для отображения уже настроенных в инфраструктуре объектов в других её частях, например, в дашбордах.

С помощью ссылок можно создавать связи и потоки между объектами, которые находятся в разных частях инфраструктуры.

Создать ссылку можно двумя способами:

1. Через Web UI:

1.1. Нажать кнопку 

на панели инструментов.

1.2. В появившемся всплывающем окне **Новая ссылка** (Рис. 4.8.4.1) выбрать из выпадающего списка объект, на который создаётся ссылка:

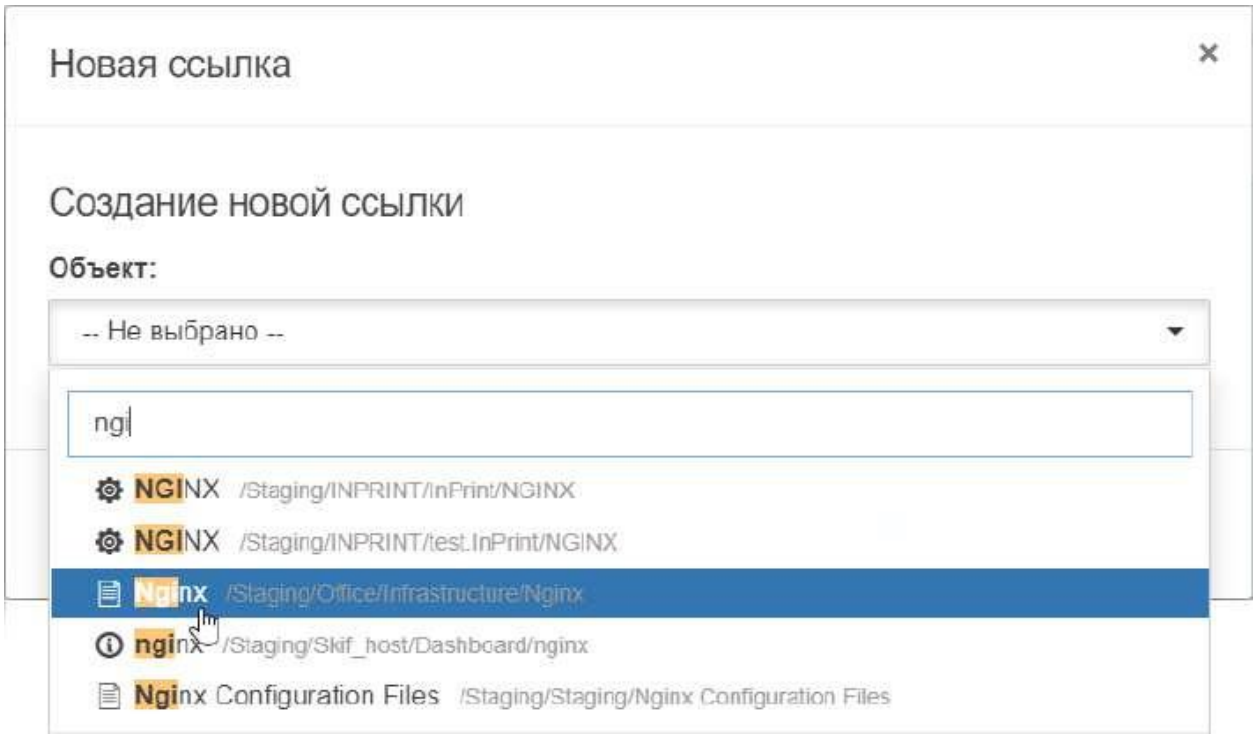


Рис. 4.8.4.1. *Окно создания нового объекта*

1.3. Нажать кнопку 

.

2. Через REST API:

POST /node/api/refs

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
url=https://$saymon_hostname/node/api/refs

curl -X POST $url -u $login:$password -H "Content-Type: application/json" \
--data '{"target": "5e60d9db630502472925fe9f", "owner": "1"}
```

4.8.5 Создание связи

Связь может отражать как физическое соединение между объектами (например, соединение сервера с маршрутизатором с помощью Ethernet-кабеля), так и логическое отношение (например, поток данных от одного программного компонента к другому).

Связи между объектами также могут являться объектами мониторинга.

Создать связь можно двумя способами:

1. Через Web UI:

1.1. Перейти в режим создания связей, нажав кнопку 

на панели инструментов.

1.2. После того, как на всех объектах появится соответствующий символ  , нажать на него на исходном объекте и, удерживая, переместить курсор на целевой объект (Рис. 4.8.5.1):

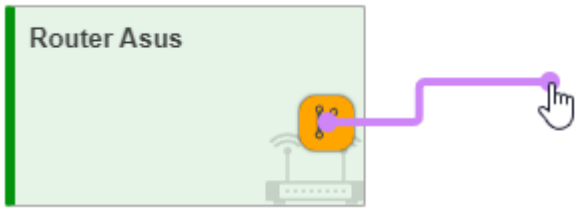


Рис. 4.8.5.1. *Создание связи*

Созданная связь отобразится между выбранными объектами (Рис. 4.8.5.2):

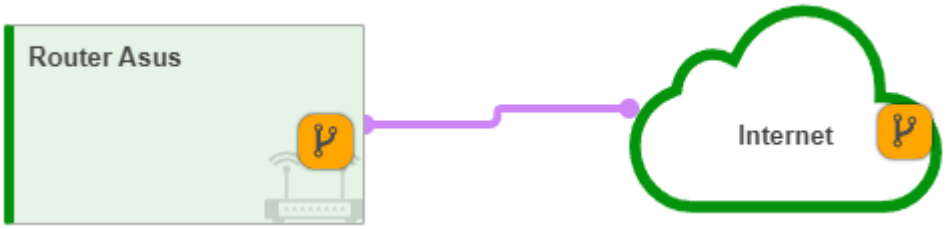


Рис. 4.8.5.2. *Связь между объектами*

1.3. Нажать кнопку 

в верхней части главного экрана для выхода из режима создания связей.

2. Через REST API:

POST /node/api/links

Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
source_obj_id=<...>
target_obj_id=<...>
url=https://$saymon_hostname/node/api/links

curl -X POST $url -u $login:$password -H "Content-Type: application/json" \
-d @- <<EOF
{
   "source": "'$source_obj_id'",
   "target": "'$target_obj_id'"
}
EOF
```

4.8.6 Удаление связи

Удалить связь можно двумя способами:

1. Через Web UI:

   1.1. Нажать кнопку  на панели хлебных крошек.

   1.2. Нажать на такую же иконку на удаляемой связи.

   1.3. Подтвердить удаление связи во всплывающем окне.

2. Через REST API:

   DELETE /node/api/links/:id

   Пример (bash):

```
login=<...>
password=<...>
saymon_hostname=<...>
link_id=<...>
url=https://$saymon_hostname/node/api/links/$link_id

curl -X DELETE $url -u $login:$password
```

4.9 Настройка интерфейса

В разделе описаны механизмы настройки элементов интерфейса для лучшей визуализации данных.

4.9.1 Выравнивание/расстановка объектов в стандартном виде

Выравнивание объектов в стандартном виде происходит благодаря сетке.

Для отображения/скрытия сетки необходимо нажать кнопку  на панели "хлебных крошек".
Настройки сетки выполняется в файле `/usr/local/saymon/client/client-config.js`

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

В секции **Grid** этого файла можно задать параметры:

- **border** - максимальный отступ границы объекта от границы сетки (в пикселях).
- **color** - цвет сетки в формате RGBA.
- **dim** - размер сетки (в пикселях).

```
return {
...
    grid: {
        dim: 20,
        color: "rgba(128, 128, 128, 0.3)",
        border: 4
    },
...
}
```

После внесения изменений в файл необходимо обновить страницу в браузере.

4.9.2 Настройка заголовка web-интерфейса

Для изменения заголовка в web-интерфейсе:

1. Открыть файл **/usr/local/saymon/client/client-config.js**.

   Подробную информацию о файле **/usr/local/saymon/client/client-config.js** *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

2. В строке **title** ввести желаемое имя:

```
return {
...
    title: '<your-new-name>',
...
}
```

3. Обновить страницу браузера.

4.9.3 Перемещение/отключение фоновой иконки объекта

Все предустановленные классы объектов имеют индивидуальную фоновую иконку. При необходимости переместить иконку, в секции **Параметры** во вкладке **Стили** объекта необходимо добавить:

```
.object-background {
background-position: 9px 8px !important;
background-size: 15px;
opacity: 1;
}

.object-caption-panel {
padding-left: 15px;
}
```

Для отключения иконки требуется добавить в ту же секцию:

```
.background-component {
display: none
}
```

4.9.4 Вертикальное отображение имени объекта

По умолчанию имя объекта отображается в его левом верхнем углу горизонтально. При необходимости отображать имя вертикально, в секции "Параметры" во вкладке "Стили" объекта необходимо добавить:

```
.js-caption {
    writing-mode: vertical-lr;
    text-orientation: upright;
    text-transform: uppercase;
    letter-spacing: 1px;
}
```

4.9.5 Редактирование стилей состояний

В процессе мониторинга в зависимости от данных, получаемых от агента, объект может менять состояние.

Каждое состояние имеет цвет. Цвет и стиль состояний можно изменить двумя способами:

1. Через Web UI:
   1.1. В панели инструментов нажать на имя пользователя и выбрать в меню пункт "Конфигурация".
   1.2. Перейти в раздел "Состояния".
   1.3. В списке состояний выбрать то, которое требуется изменить.
   1.4. При необходимости изменить имя в соответствующем поле.
   1.5. Настроить основной цвет, цвет тени, цвет строки таблицы и фон при помощи цветовой палитры или методом ввода номера цвета.
2. Через файл конфигурации:
   2.1. Создать файл: `/usr/local/saymon/saymon.local/css/saymon.local.css`.
   2.2. Открыть его в текстовом редакторе и вставить код для изменения цвета состояния или его фона. Например:

```
.state-4
   {
      background-color: rgba(255, 122, 0, 0.46);
      box-shadow: 2px 5px 10px rgba(253, 118, 7, 0.5);
      border-left: 5px solid #FD7607;
   }
.state-bg-4
   {
      background-color: #FD7607;
   }
.view-screen-element ul li.state-4
   {
      border-left: 5px solid #FD7607;
   }
.view-screen-element ul li .badge.state-4
   {
      background-color: #FD7607;
   }
```

   Номер состояния соответствует его ID.

4.10 Настройка мониторинга

Платформа "Центральный Пульт" позволяет осуществлять мониторинг с использованием различных типов проверок. Проверки настраиваются в web-интерфейсе системы.

> *Подробную информацию о каждом типе проверок см. в "Руководстве пользователя" в подразделе. 4.2.1 "Настройка процесса мониторинга и типов проверок" на стр. 43.*

Далее рассмотрены несколько примеров процесса мониторинга.

4.10.1 Мониторинг основных параметров ПК

Для мониторинга основных параметров работы сервера или ПК: CPU, File System, Memory и Network IO, достаточно выполнить несколько действий:

1. Установить агента на наблюдаемый ПК или сервер.
2. Создать объект, например, класса **Host**, в web-интерфейсе.
3. Перейти в созданный объект и добавить внутри него объекты классов:
   - **Saymon Agent**,
   - **CPU**,
   - **File System**,
   - **Memory**,
   - **Network IO**.
4. Сконфигурировать и запустить агента.

Через некоторое время информация об основных параметрах работы компьютера начнёт поступать на сервер и отображаться в web-интерфейсе системы.

4.10.2 Мониторинг процесса памяти

Для настройки мониторинга процесса памяти необходимо:

1. Установить, сконфигурировать и запустить агента на наблюдаемом ПК или сервере.
2. Создать объект, например, класса **Process**, в web-интерфейсе.
3. Перейти в созданный объект и в его секции **Мониторинг**:
    - выбрать агента, установленного ранее на данный компьютер;
    - выбрать тип проверки **Процесс по имени**;
    - заполнить необходимые поля.

Через некоторое время информация о проверяемом процессе начнёт поступать на сервер и отображаться в web-интерфейсе системы.

4.10.3 Мониторинг изменения файлов и папок

Для настройки мониторинга изменения файлов и папок сервера или ПК необходимо:

1. Установить, сконфигурировать и запустить агента на наблюдаемом ПК или сервере.
2. Создать объект, например, класса **Configuration File**, в web-интерфейсе.
3. Перейти в созданный объект и в его секции **Мониторинг**:
    - выбрать агента, установленного ранее на данный компьютер;
    - выбрать тип проверки **Конфигурационный файл / директория**;
    - указать путь к проверяемому файлу/директории.

Через некоторое время информация о проверяемом файле/директории начнёт поступать на сервер и отображаться в web-интерфейсе системы.

4.10.4 Проверка доступности web-ресурса

Данный тип мониторинга позволяет убедиться не только в работоспособности web-сайта (статус 200 OK), но и в ограничении доступа к таким ресурсам, как панель администрирования баз данных. В этом случае статус "403 Forbidden" или "404 Not Found" будет говорить о правильности настройки системы, а иной статус - о возможной угрозе безопасности системы.

Для проверки доступности и скорости отклика web-ресурса необходимо:

1. Установить, сконфигурировать и запустить агента на наблюдаемом ПК или сервере.
2. Создать объект, например, класса **Address**, в web-интерфейсе.
3. Перейти в созданный объект и в его секции **Мониторинг**:
    - выбрать агента, который будет выполнять проверку;
    - выбрать тип проверки **HTTP-запрос**;
    - выбрать тип запроса **GET**;
    - в поле **URL** указать адрес web-сайта.

Через некоторое время информация о доступности и скорости отклика наблюдаемого ресурса начнёт поступать на сервер и отображаться в web-интерфейсе системы.

4.10.5 Безагентный мониторинг web-сервера

Существует ряд случаев, при которых установка агента на сервере невозможна. В таких случаях рекомендуется написать скрипт, который с заданной периодичностью будет выполняться на сервере, собирать необходимые данные и генерировать текстовый файл с результатами в формате JSON по ссылке, доступной извне.

Для мониторинга параметров web-сервера, на который невозможно поставить агента, необходимо:

1. Написать локальный скрипт, выполняющий подготовку данных (например, в папке загрузок: `.../downloads/scripts/webserver_stat.sh`):

```
#!/bin/bash
# Сбор параметров работы web-сервера.

# использование Memory
memUsage=$(free -m | grep Mem | perl -pe 's/Mem:\s+\S+\s+(\S+).*/$1/')

# использование Swap
swapUsage=$(free -m | grep Swap | perl -pe 's/Swap:\s+\S+\s+(\S+).*/$1/')

# загрузка CPU
cpuUsage=$(uptime | awk '{print $10}' | perl -pe 's/,//')

# проверка выполнения какого-либо скрипта, например, webserver_stat.sh
scriptExec=$(ps -ef | grep webserver_stat.sh | grep -v grep | wc -l)

# Write JSON response
echo "{\"memUsageMB\":\"$memUsage\", \"swapUsageMB\":\"$swapUsage\",
\"cpuUsage\":\"$cpuUsage\", \"scriptExec\":\"$scriptExec\"}" >
webserver_stat.json
```

2. Добавить выполнение скрипта в планировщик заданий cron.
3. Установить, сконфигурировать и запустить хотя бы одного агента в инфраструктуре.
4. Создать объект, например, класса **Info**, в web-интерфейсе.
5. Перейти в созданный объект и в его секции **Мониторинг**:
   - выбрать агента, который будет выполнять проверку;
   - выбрать тип проверки **HTTP-запрос**;
   - выбрать тип запроса **GET**;
   - в поле **URL** указать адрес JSON-файла.

Через некоторое время информация о параметрах работы web-сервера начнёт поступать на сервер и отображаться в web-интерфейсе системы.

4.11 Страницы для внедрения в IFrame

Виджет IFrame, предназначенный для внедрения в интерфейс содержимого сторонних ресурсов, можно также использовать и со специально настроенными информационными страницами внутри Центрального Пульта.

4.11.1 Страница с авариями

В адресной строке страницы аварий возможно указать дополнительные URL-параметры, влияющие на представление панели элементов управления и отображение данных в таблице.

Пример для отключения всех элементов управления, кроме кнопки экспорта данных в Excel, и применения ранее сохранённого фильтра **Major** (вместо **127.0.0.1** указать адрес нужного сервера Центрального Пульта):

http://127.0.0.1/incidents.html?header=off&exportSection=on&filterName=Major

Параметры элементов управления (часть 1 из 2):

Параметр	Описание
header	Включает/выключает отображение панели, содержащей элементы управления фильтрацией данных в таблицах; • on - отображение панели включено, • off - отображение панели выключено. По умолчанию - on - отображение панели включено. *При выключенном отображении панели элементов управления выключаются все описанные в таблице элементы. В этом случае, при необходимости включить отдельный элемент, нужно отдельно вписать его в адресную строку со значением **on**.*
cache	Включает/выключает загрузку сделанных пользователем настроек из кэша браузера: будут ли настройки сохраняться при обновлении страницы браузера; • on - загрузка настроек включена, • off - загрузка настроек выключена.
entityHref	Включает/выключает отображение ссылок на аварийные элементы в столбце Объект/связь; • on - отображение ссылок включено, • off - отображение ссылок выключено.
exportSection	Включает/выключает отображение кнопки экспорта данных в Excel; • on - отображение кнопки экспорта данных включено, • off - отображение кнопки экспорта данных выключено.
filterSection	Включает/выключает отображение секции расширенного фильтра; • on - отображение секции фильтра включено, • off - отображение секции фильтра выключено.

Параметры элементов управления (часть 2 из 2):

Параметр	Описание
logo	Включает/выключает отображение логотипа; • on - отображение логотипа включено, • off - отображение логотипа выключено.
presetSection	Включает/выключает отображение секции представлений; • on - отображение секции представлений включено, • off - отображение секции представлений выключено.
propertyFieldsSection	Включает/выключает отображение секции с выбором полей свойств; • on - отображение секции с выбором полей свойств включено, • off - отображение секции с выбором полей свойств выключено.
typeSelect	Включает/выключает отображение селектора выбора типа данных - выбор между активными авариями и историей аварий; • on - отображение селектора включено, • off - отображение селектора выключено.

Параметры фильтра:

Параметр	Описание
filterName	Имя сохранённого фильтра, который сразу будет применён к отображаемым данным.
timeFilter	Включает/выключает отображение дополнительного фильтра по временному диапазону; • on - временной фильтр включен, • off - временной фильтр выключен. По умолчанию - off - временной фильтр выключен.
type	Выбор типа отображаемых данных – список активных или исторических аварий; • active - отображать список активных аварий, • history - отображать список исторических аварий. По умолчанию отображается последний выбранный тип.

4.11.2 Страница с графиками

В системе возможно настроить специальную страницу с набором графиков из одного или нескольких объектов (Рис. 4.11.2.1):

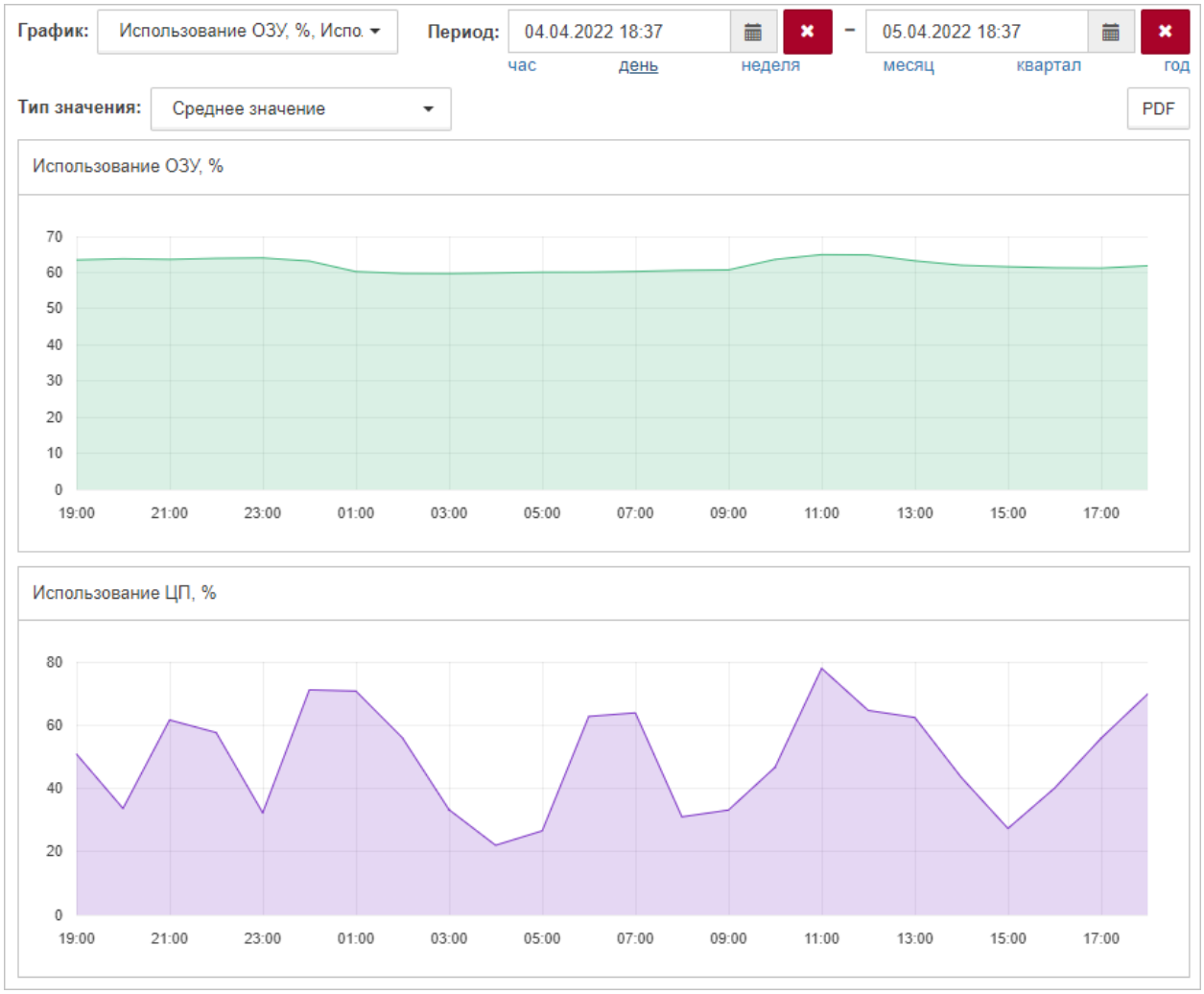


Рис. 4.11.2.1. *Пример настроенной страницы графиков*

В верхней части страницы представлены элементы управления (Рис. 4.11.2.2):

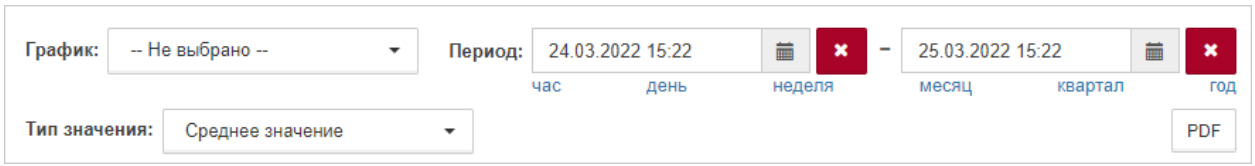


Рис. 4.11.2.2. *Элементы управления страницы графиков*

Элементы управления страницы графиков:

Параметр	Описание
График	Выпадающий список, где можно отметить отображаемые графики подключенных метрик.
Период	Временной период для отображения графиков. При необходимости можно быстро ввести период за последние • час, • день, • неделю, • месяц, • квартал, • год, нажав соответствующую кнопку под полями выбора периода.
Тип значения	Способ вычисления значений точек графиков при достаточно коротком периоде обновления данных: • **среднее значение** - значения точек графика рассчитываются как среднее арифметическое значений метрики; • **минимальное значение** - значения точек графиков соответствуют минимальным значениям метрики; • **максимальное значение** - значения точек графиков соответствуют максимальным значениям метрики.
PDF	Кнопка сохранения страницы в PDF-формате.

Страница с графиками доступна по ссылке вида

*http://**<адрес сервера Центрального Пульта>**/iframe-widgets.html?#widget/selectable-chart/<ID основного объекта>/**<дополнительные параметры>***

где

- **<адрес сервера Центрального Пульта>** - IP-адрес или FQDN инсталляции Центрального Пульта;
- **<ID основного объекта>** - идентификатор объекта, в свойствах которого хранится список метрик и их отображаемые названия;
- **<дополнительные параметры>** - настройки, которые определяют первоначальный вид страницы при её загрузке.

Пример:

*http://**127.0.0.1**/iframe-widgets.html?#widget/selectable-chart/**5fcc9a4de0a5393f45fb99f3**/*
chart=61a662bdde9e7970b38b340c:percentageUsage.combined&
chart=TOTAL.percentUsed&period=last-24hours

В данном примере:

- сервер работает на локальной машине и доступен по адресу **127.0.0.1**;
- основным объектом указан объект с идентификатором **5fcc9a4de0a5393f45fb99f3**;
- при загрузке страницы отображаются график метрики **percentageUsage.combined** объекта с идентификатором **61a662bdde9e7970b38b340c** и график метрики **TOTAL.percentUsed** основного объекта за прошедшие сутки.

Возможные метрики для построения графиков необходимо указать в свойствах основного объекта.

В имени свойства указывается имя метрики в формате:

chart:<имя метрики>

При необходимости добавить метрику из другого объекта нужно добавить идентификатор объекта:

chart:<ID внешнего объекта>:<имя метрики>

В значении свойства задаётся отображается имя графика:

<желаемое название отображаемого графика>

Примеры:

Имя свойства: chart:TOTAL.percentUsed

Значение свойства: Использование ОЗУ, %

Описание: График метрики ***TOTAL.percentUsed*** основного объекта

Имя свойства: chart:61a662bdde9e7970b38b340c:percentageUsage.combined

Значение свойства: Использование ЦП, %

Описание: График метрики ***percentageUsage.combined*** объекта с ID ***61a662bdde9e7970b38b340c***

Дополнительные параметры страницы графиков:

Параметр	Возможные значения	Описание
aggregate	**aggregate=avg** - выбираются средние значения, **aggregate=max** - выбираются максимальные значения, **aggregate=min** - выбираются минимальные значения.	Тип значения, применяемый при загрузке страницы. По умолчанию применяется функция усреднения.
chart	chart:<имя метрики основного объекта> **chart:packetsTransmitted** chart:<ID нужного объекта>:<имя метрики нужного объекта> **chart:61a662bdde9e7970b38b340c:freeMemory** *Используемые в URL метрики должны быть заранее настроены в свойствах основного объекта.*	Графики, которые будут отображаться при загрузке страницы. Если не указать данный параметр, то по умолчанию откроется страница без графиков.
period	**period=last-hour** - за последний час, **period=last-24hours** - за последний день, **period=last-7days** - за последнюю неделю, **period=last-30days** - за последний месяц, **period=last-90days** - за последний квартал, **period=last-365days** - за последний год.	Период времени для построения графиков, выбираемый при загрузке страницы. По умолчанию отображаются графики за последний час.
widget-instance-id	Значение параметра - имя представления, уникальное в рамках основного объекта. В качестве значения может быть использован любой набор символов. Примеры: • **widget-instance-id=1** - сохранение настроек в представлении с именем **1**; • **widget-instance-id=example** - сохранение настроек в представлении с именем **example**.	Сохранение настроек страницы в отдельном представлении с уникальным именем. По умолчанию настройки не записываются в представление.

Параметр **widget-instance-id** позволяет сохранять все производимые пользователем настройки в представлении с соответствующим именем.

Имена представлений задаются пользователями произвольно.

> *Функционал будет полезен при необходимости одновременного просмотра графиков одних и тех же метрик с различными настройками отображения.*

Если встроить одну и ту же страницу с графиками в несколько IFrame-виджетов, то все изменения, вносимые через элементы управления на одном из виджетов, будут применяться и к остальным. Изменения с настройками отображения страницы можно зафиксировать в отдельном представлении с помощью параметра **widget-instance-id**. Имя представления соответствует значению параметра.

При указании **widget-instance-id** в дополнительных параметрах, изменения, вносимые через другие IFrame-виджеты с той же страницей с графиками, не будут влиять на настройки, сохранённые в представлении.

В настройках IFrame-виджета можно менять значение параметра **widget-instance-id**, чтобы создать еще одно представление или восстановить настройки из ранее сохранённого.

Пример ссылки на страницу с графиками с использованием **widget-instance-id**:

*http://**127.0.0.1**/iframe-widgets.html?#widget/selectable-chart/5fcc9a4de0a5393f45fb99f3/**widget-instance-id=default***

В данном примере:

- сервер работает на локальной машине и доступен по адресу 127.0.0.1;
- основным объектом указан объект с идентификатором **5fcc9a4de0a5393f45fb99f3**;
- настройки отображения страницы сохраняются в представлении с именем **default**.

4.11.3 Страница с журналом сессий

Журнал пользовательских сессий можно открывать по прямой ссылке, которая также может быть использована для встраивания в виджет IFrame:

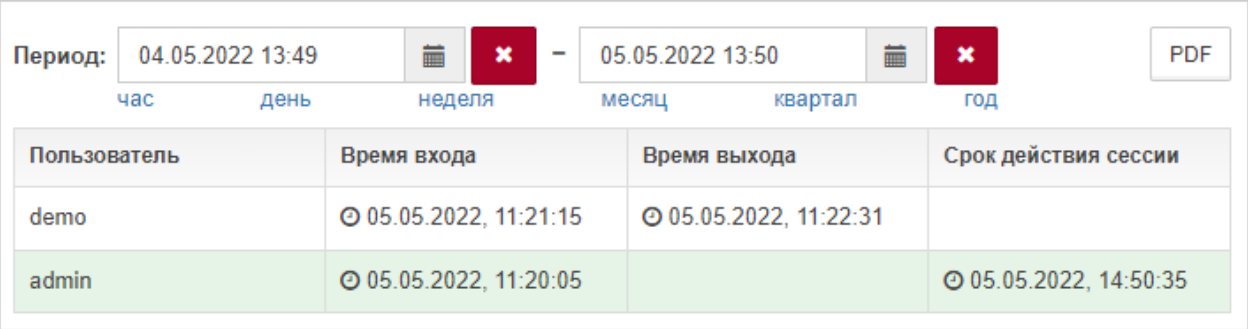


Рис. 4.11.3.1. *Пример настроенной страницы с журналом сессий*

Ссылка имеет следующую структуру:

*https://**<адрес сервера Центрального Пульта>**/iframe-widgets.html?#widget/session-log/1/**<дополнительные параметры>***

где:

- **<адрес сервера Центрального Пульта>** - IP-адрес или FQDN инсталляции Центрального Пульта;
- **<дополнительные параметры>** - настройки, которые определяют вид страницы.

Пример:

*https://**127.0.0.1**/iframe-widgets.html?#widget/session-log/1/**period=last-30days&limit=10***

В данном примере:

- сервер работает на локальной машине и доступен по адресу 127.0.0.1;
- при загрузке страницы устанавливается временной период за последний месяц;
- выдача ограничена 10 записями.

Дополнительные параметры страницы журнала сессий:

Параметр	Описание
limit	Максимальное количество возвращаемых записей. По умолчанию выдача ограничена 30 записями.
period	Период времени для отображения журнала сессий при загрузке страницы. По умолчанию отображаются сессии за последний час. Параметр может принимать следующие значения: • **period=last-hour** - за последний час, • **period=last-24hours** - за последний день, • **period=last-7days** - за последнюю неделю, • **period=last-30days** - за последний месяц, • **period=last-90days** - за последний квартал, • **period=last-365days** - за последний год.

4.12 Открытие терминала удалённого доступа

Для объектов класса **Host**, а также для объектов с хотя бы одним из заданных свойств

- address (приоритетнее);
- IP,

в контекстном меню доступны дополнительные пункты:

- **Открыть SSH терминал**,
- **Открыть telnet терминал**.

Если IP-адрес указан в свойствах объекта, при выборе соответствующего пункта меню происходит подключение к терминалу.

По умолчанию используются следующие порты:

- **22** для SSH;
- **23** для telnet.

При необходимости использовать другой порт номер порта указывается в свойствах объекта для соответствующего протокола:

- **ssh_port**
- **telnet_port**

Логин пользователя также можно заранее указать в свойствах для соответствующего протокола:

- **ssh_login**
- **telnet_login**

Для объектов класса **Host**, если в свойствах объекта IP-адрес не задан, при выборе соответствующего пункта меню открывается окно (Рис. 4.12.1), в котором необходимо:

- ввести IP-адрес объекта в поле "Адрес";
- указать номер порта в поле "Порт";
- ввести логин пользователя в поле "Пользователь".

> ⚠ *При установленном слайдере "Сохранить" - "Да" введённые данные сохранятся в свойствах объекта.*

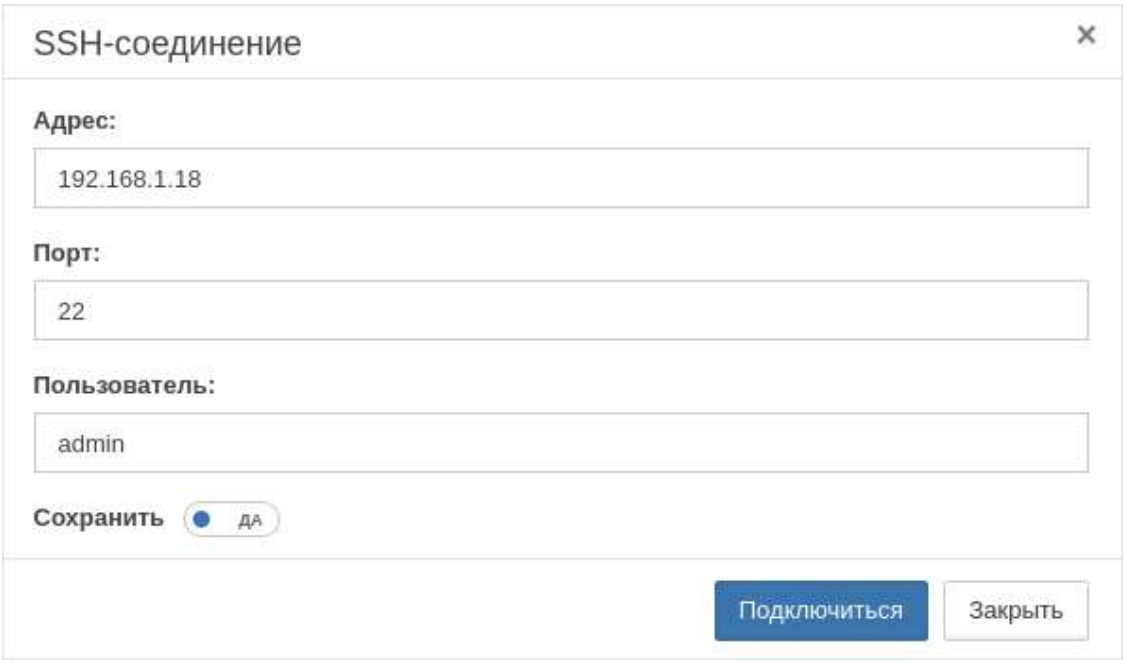


Рис. 4.12.1. *Пример окна подключения к SSH-терминалу*

После ввода данных необходимо нажать кнопку Подключиться.

4.13 Переход к web-интерфейсу устройства

Для объектов с хотя бы одним из следующих свойств (в порядке уменьшения приоритета)

- web_link,
- address,
- IP,

в контекстном меню доступен пункт "Переход по ссылке". При выборе данного пункта происходит переход по указанному в значении свойства IP-адресу или URL.

4.14 Импорт и экспорт данных

Система предусматривает возможность создания резервных копий, восстановления и переноса основных данных между различными инсталляциями.

Экспорт данных

Для экспорта данных из MongoDB в архив на существующей инсталляции системы необходимо выполнить следующую команду:

```
mongodump -d saymon -o mongodb_backup
```

Для экспорта настроек конфигурации необходимо сделать копии:

- директории с конфигурационными файлами сервера `/etc/saymon`;
- конфигурационного файла клиента `/usr/local/saymon/client/client-config.js`.

> *Подробную информацию о файле* `/usr/local/saymon/client/client-config.js` *см. в подразделе 4.1.4 "Настройка элементов web-интерфейса" на стр. 51.*

Импорт данных

Для восстановления или импорта данных в MongoDB на новой инсталляции системы необходимо выполнить следующую команду:

```
mongorestore --drop --noIndexRestore mongodb_backup
echo flushall | redis-cli
```

Для импорта настроек конфигурации необходимо вставить:

- файлы из ранее сохранённой директории с конфигурационными файлами сервера в директорию `/etc/saymon`;
- конфигурационный файл клиента в директорию `/usr/local/saymon/client`.

5 Проблемы в работе системы и способы их решения

Раздел содержит информацию об известных проблемах, которые могут возникать при работе с платформой, и способы их решения.

5.1 Недостаточно места на виртуальной машине с сервером

Условия проблемы:

- есть виртуальная машина с сервером Центрального Пульта;
- нет места на виртуальной машине с сервером Центрального Пульта.

Решение проблемы:

1. Понять содержимое и объём занимаемого места:

```
sudo du -h / | sort -h
```

2. Просмотреть список папок в *stdout:*
   - если много места занимает папка */var/log/saymon*, то можно уменьшить количество хранимых лог-файлов правкой */etc/logrotate.d/saymon* для *saymon-server.log:* ***rotate X*** и */opt/saymon-agent/conf/logback-upstart.xml* для *saymon-agent.*.log*:***<maxHistory>10</maxHistory>;***
   - если много места занимают данные из MongoDB, то зайти в базу данных и оценить размеры коллекций:

```
mongo saymon

function getReadableFileSizeString(fileSizeInBytes) {
    var i = -1;
    var byteUnits = [' kB', ' MB', ' GB', ' TB', 'PB', 'EB', 'ZB', 'YB'];
    do {
        fileSizeInBytes = fileSizeInBytes / 1024;
        i++;
    } while (fileSizeInBytes > 1024);
    return Math.max(fileSizeInBytes, 0.1).toFixed(1) + byteUnits[i];
};

var collectionNames = db.getCollectionNames(), stats = [];
collectionNames.forEach(function (n) { stats.push(db.getCollection(n).stats()); });
stats = stats.sort(function(a, b) { return b['size'] - a['size']; });
for (var c in stats) { print(stats[c]['ns'] + ": " + getReadableFileSizeString(stats[c]
['size']) + " (" + getReadableFileSizeString(stats[c]['storageSize']) + ")"); }
```

В наиболее объёмных коллекциях используется timestamp, следующей командой можно удалить из коллекции stateHistory массив данных за рамками глубины хранения:

```
db.stateHistory.remove({timestamp:{$gt:1477994233000}})
```

После описанных выше действий место в системе не освободится, так как MongoDB аллоцирует дисковое пространство. Требуется сделать бекап и восстановить базу:

```
mongodump
sudo rm -rf /var/lib/mongodb/*
sudo mongorestore dump/ --dbpath /var/lib/mongodb/
sudo chown -R mongodb:mongodb /var/lib/mongodb
sudo service mongod restart
```

- если много места занимают данные Open TSDB, не вынесенные из Docker-контейнера, их можно вынести:

```
sudo docker exec -it opentsdb bash
cd /data/hbase/hbase-root
tar zcvf hbase-root.tar.gz hbase-root
scp hbase-root.tar.gz saymon@*host_ip*:/opt/.
exit
cd /opt/ && tar xvf hbase-root.tar.gz
sudo docker stop opentsdb
sudo docker rm opentsdb
sudo docker run -d -p 127.0.0.1:4242:4242 --restart=always –
volume /opt/hbase-root/:/data/hbase/hbase-root/ --name=opentsdb
rossinno/opentsdb
```

5.2 Отсутствие подключения агента к серверу

Условия проблемы:

- агент не подключается к серверу;
- запись в логе:
  12.10.2016 07:45:59.431 [pool-1-thread-1] WARN n.r.s.agent.connection.RedisBackend - Redis connection failed (will retry in 5 seconds): JedisDataException: ERR max number of clients reached

Решение проблемы:

1. Проверить на сервере проблему локально:

```
# redis-cli -a 'пароль_от_redis_в_кавычках' info clients | grep connected_clients| sed -e 's/connected_clients://g'
Error: Connection reset by peer
```

2. Проверить проблему локально через redis-cli:

```
# redis-cli
127.0.0.1:6379> auth пароль_от_redis
(error) ERR max number of clients reached
127.0.0.1:6379> q
```

3. Рестарт Redis-сервера:

```
# service redis-server restart
Stopping redis-server: redis-server
Starting redis-server: redis-server
```

5.3 Проверка работы MongoDB

Проверка наличия процесса в памяти:

```
ps -ef | grep mongod
mongodb 1147 1 0 Nov02 ? 04:23:16 /usr/bin/mongod –
config /etc/mongod.conf
```

Остановка, запуск и рестарт процесса:

```
sudo service mongod stop
sudo service mongod start
sudo service mongod restart
```

5.4 Проверка работы MySQL

Проверка пароля MySQL (действие на хосте с сервером):

```
cat /etc/saymon/saymon-server.conf
```

Просмотр секции db{}:

```
"db" : {
"host" : "localhost",
"user" : "user",
"password" : "password",
"database" : "saymondb"
},
```

5.5 Проверка работы Redis

Проверка наличия процесса в памяти:

```
ps -ef | grep redis
redis 1763 1 0 Aug10 ? 00:37:11 /usr/bin/redis-server 0.0.0.0:6379
root 1786 1 0 Aug10 ? 00:00:00 /usr/bin/stunnel4 /etc/stunnel/redis-client.conf
root 1787 1 0 Aug10 ? 00:00:00 /usr/bin/stunnel4 /etc/stunnel/redis-client.conf
...
```

Остановка, запуск и рестарт процесса:

```
sudo service redis-server stop
sudo service redis-server start
sudo service redis-server restart
```

Номер порта, на котором осуществляется процесс:

```
sudo netstat -lnp | grep redis
tcp 0 0 0.0.0.0:6379 0.0.0.0:* LISTEN 1763/redis-server 0
```

или в конфигурационном файле:

```
cat /etc/saymon/saymon-server.conf | grep cache -A 4
"cache": {
    "auth_pass": "12!@easy",
    "host": "127.0.0.1",
    "port": 6379
},
...
```

Проверка доступности (открытости) порта:

```
sudo iptables -L INPUT -n -v --line-numbers
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num pkts bytes target prot opt in out source destination
1 15M 3082M ACCEPT tcp – * * 0.0.0.0/0 0.0.0.0/0 tcp dpt:6379
```

Добавление порта в список открытых и запись нового правила:

```
sudo iptables -I INPUT 1 -m state --state NEW -p tcp --dport 6379 -j ACCEPT
sudo bash -c "/sbin/iptables-save > /etc/iptables.rules"
```

5.6 500 Internal Server Error и отсутствие графиков

Условия проблемы:

- вместо графиков возникает ошибка 500.

Решение проблемы:

Необходимо перезапустить OpenTSDB:

less /var/log/opentsdb/opentsdb.log - (здесь можно увидеть какие-то ошибки)
sudo service opentsdb stop
sudo service hbase restart
sudo service opentsdb start

5.7 Ошибка работы HTTP-проверки

Условия проблемы:

- HTTP-проверка адреса https://xxx.xxx не работает и возникает ошибка.

Решение проблемы:

Данная проблема возникает при использовании агента в связке с Java 1.6.

Есть два варианта решения:

1. Обновить Java, установленную в операционной системе, до версии 1.7 или 1.8.
2. Скачать и установить последнюю версию агента со встроенной Java.

Приложение А (обязательное)

Права на операции

Основные права (часть 1 из 4):

Название	Описание	Запись в конфигурационном файле сервера
Выполнение операций	Включает секцию **Операции** и возможность их запуска.	"execute-operations"
Поиск и групповые операции	Включает возможность выполнения поиска и групповых операций. *Необходимы права на "Управление объектами, свойствами и документами"*	"run-bulks
Управление объектами, свойствами и документами	Включает • создание, • клонирование, • переименование, • изменение размеров, • перемещение, • удаление • объектов, • виджетов объектов, • ссылок, • подложек. Включает просмотр и редактирование всех секций объектов, кроме секции **Мониторинг**.	"manage-objects"
Создание объектов	Включает • создание, • клонирование объектов.	"create-objects"
Модификация объектов	Включает • переименование, • изменение размеров, • перемещение • объектов, • виджетов объектов, • ссылок, • подложек.	"modify-objects"
Удаление объектов	Включает удаление объектов.	"delete-objects"

Основные права (часть 2 из 4):

Название	Описание	Запись в конфигурационном файле сервера
Управление связями, свойствами и документами	Включает • создание, • изменение, • удаление • связей, • виджетов связей. Включает просмотр и редактирование всех секций связей, кроме секции **Мониторинг**.	"manage-links"
Создание связей	Включает создание связей.	"create-links"
Модификация связей	Включает • изменение связей, • создание, изменение и удаление виджетов связей.	"modify-links"
Удаление связей	Включает удаление связей.	"delete-links"
Управление потоками	Включает создание, изменение и удаление потоков. *Для изменения потоков требуются права на просмотр секции "Параметры".*	"manage-flows"
Создание потоков	Включает создание потоков.	"create-flows"
Модификация потоков	Включает изменение потоков. *Требуются права на просмотр секции "Параметры".*	"modify-flows"
Удаление потоков	Включает удаление потоков.	"delete-flows"

Основные права (часть 3 из 4):

Название	Описание	Запись в конфигурационном файле сервера
Управление документами объектов и связей	Включает редактирование секции "Документы". *Требуются права на просмотр секции "Документы".*	"manage-documents"
Создание документов	Включает прикрепление ссылок на web-страницы.	"create-documents"
Модификация документов	Включает редактирование документов.	"modify-documents"
Удаление документов	Включает удаление документов.	"delete-documents"
Загрузка документов	Включает прикрепление PDF-файлов.	"upload-documents"
Управление свойствами объектов и связей	Включает редактирование секции **Свойства**. *Требуются права на просмотр секции "Свойства".*	"manage-properties"
Создание свойств	Включает создание свойств.	"create-properties"
Модификация свойств	Включает модификацию свойств.	"modify-properties"
Удаление свойств	Включает удаление свойств.	"delete-properties"
Управление классами	Включает кнопку  в секциях • Условия перехода состояний, • Условия генерации аварий. Включает в окне конфигурации системы вкладку **Управление классами**.	"manage-classes"
Управление операциями	Включает редактирование секции **Операции**.	"manage-operations"
Управление настройками мониторинга	Включает просмотр и редактирование секции **Мониторинг**.	"manage-service-properties"

Основные права (часть 4 из 4):

Название	Описание	Запись в конфигурационном файле сервера
Управление скриптами	Включает возможность использовать скрипт с указанным текстом при проверке **Выполнение программы / скрипта**. Включает в окне конфигурации системы вкладку **Скрипты**.	"manage-scripts"

Просмотр секций:

Секция	Запись в конфигурационном файле сервера
Данные	"view-section-stat"
Свойства	"view-section-properties"
Документы	"view-section-documents"
История состояний	"view-section-state-history"
Графики	"view-section-history-graph"
Мониторинг	"view-section-monitoring"
Параметры	"view-section-entity-settings"
Условия перехода состояний	"view-section-state-conditions"
Действия при смене состояний	"view-section-state-triggers"
Виджеты	"view-section-widgets"
История изменений	"view-section-audit-log"
Изменения конфигурации	"view-section-config-log"
Операции	"view-section-operations"
История операций	"view-section-operations-history
Условия генерации аварий	"view-section-incident-conditions"
Правила формирования данных	"view-section-stat-rules"

Права на администрирование системы:

Название	Описание	Запись в конфигурационном файле сервера
Управление конфигурацией системы	Включает в окне конфигурации системы вкладки: • Журнал событий, • Шаблоны уведомлений, • Аварии, • Состояния, • Парольные политики.	"manage-configuration"
Обновление агентов	Позволяет пользователю обновлять отдельных агентов кнопкой и всех агентов на вкладке **Агенты** в окне конфигурации системы.	"upload-agent-updates"
Управление пользователями	Позволяет управлять всеми пользователями системы.	"manage-users"
Управление журналом событий	Позволяет • назначать ответственного, • изменять текст событий, • удалять события в журнале событий SNMP.	"manage-event-log"
Управление историческими данными	Позволяет • добавлять, • редактировать, • удалять комментарии на графиках.	"manage-history"
Запуск автоопределения объектов	Позволяет использовать сервисный REST-метод **node/api/discovery**.	"run-discovery"
Чтение журнала сессий	Включает в окне конфигурации системы вкладку **Журнал сессий**.	"read-session-log"

Приложение Б (обязательное)

Параметры исполняемых файлов пользовательских сенсоров и скриптов, выполняемых при создании/удалении объектов

Возможные поля файла настроек параметров (часть 1 из 3):

Поле	Описание
title	Отображаемое в web-интерфейсе имя исполняемого файла. Опциональное поле. По умолчанию - идентификатор сенсора или скрипта. *Если идентификатор содержит путь (исполняемый файл вложен в подпапки), имя будет отображаться с путём.* Пример: *путь к файлу: (путь до папки с исполняемыми файлами)/cisco/task_1.sh* *отображаемое имя скрипта: cisco - task_1*
excludeForClasses	Список ID классов объектов, к которым неприменим данный сенсор. Опциональное поле. По умолчанию сенсор доступен для всех классов объектов. ***Поле применяется только для пользовательских мониторинговых сенсоров.***
icon	Иконка из набора Font Awesome, отображаемая рядом с именем проверки. Опциональное поле. По умолчанию -  - 'fa fa-cubes fa-fw' ***Поле применяется только для пользовательских мониторинговых сенсоров.***
args	Настройки аргументов, которые задаются пользователем через web-интерфейс и передаются исполняемому файлу.
args.id	Имя аргумента, передаваемое исполняемому файлу. Опциональное поле.
args.default	Значение аргумента по умолчанию, подставляемое в соответствующее поле. Опциональное поле.
args.description	Описание аргумента, отображаемое в web-интерфейсе. Опциональное поле.
args.name	Название аргумента, отображаемое в web-интерфейсе.

Возможные поля файла настроек параметров (часть 2 из 3):

Поле	Описание
args.required	Признак обязательности заполнения значения аргумента. • false - значение опционально, • true - значение обязательно. Опциональное поле. По умолчанию - **false**.
args.options	Применимо только для типа **select** (*args.type.select*). Определяет набор значений, отображаемых в выпадающем списке. По умолчанию при создании проверки выбирается первая опция. Чтобы установить иную опцию значением по умолчанию, необходимо указать значение поля *args.options.value* в поле *args.default*. Исполняемому файлу передаются: • имя аргумента (*args.id*); • значение аргумента (*args.options.value*). Если *args.id* не задано, то исполняемому файлу передаётся только значение выбранного аргумента (*args.options.value*). Если значение аргумента не задано (*args.options.value*), то исполняемому файлу также не передаётся имя аргумента (*args.id*). *Данную логику можно использовать для реализации пункта "Не выбрано" и разделителей в выпадающем списке.*
args.options.description	Описание значения аргумента, отображаемое в выпадающем списке. Опциональное поле. По умолчанию отображается значение аргумента (*args.options.value*).
args.options.value	Значение аргумента. Опциональное поле. При отсутствии значения исполняемому файлу также не передаётся имя аргумента. *Такие опции можно использовать для реализации пункта "Не выбрано" и разделителей в выпадающем списке.*
args.type	Тип значения аргумента. Опциональное поле. По умолчанию - text.

Возможные поля файла настроек параметров (часть 3 из 3):

Поле	Описание
args.type.checkbox	Переключатель. По умолчанию выключен. При включении исполняемому файлу передаётся значение, указанное в поле *args.id* (имя аргумента). Если это поле отсутствует, то исполняемому файлу передаётся порядковый номер аргумента (отсчёт начинается с 0).
args.type.password	Отображает текстовое поле для ввода значения аргумента с маскировкой введённых символов.
args.type.select	Отображает выпадающий список с предопределёнными значениями аргумента. Список значений задается в поле *args.options*. Для данного типа не отображается описание аргумента (поле *args.description*). Вместо этого необходимо использовать соответствующее поле для каждого значения аргумента (*args.options.description*).
args.type.text	Отображает текстовое поле для ввода значения аргумента.
args.type.textarea	Отображает текстовое поле для ввода значения аргумента. Позволяет вводить многострочный текст.

Пример содержимого файла настроек со всеми типами аргументов (часть 1 из 2):

```
{
    "title": "Example",
    "icon": "fa fa-folder fa-fw",
    "excludeForClasses": [1,2],
    "args": [
        {
            "name": "Unnamed (positional) arg"
        },
        {
            "name": "Named arg",
            "id": "--name"
        },
        {
            "name": "Behold the description on the right",
            "description": "Here I am!",
            "id": "--description"
        },
        {
            "name": "Arg with defaul value",
            "description": "and description",
            "id": "--defval",
            "default": "default_value"
        },
        {
```

Пример содержимого файла настроек со всеми типами аргументов (часть 2 из 2):

```
            "name": "Required arg",
            "description": "This field is required to fill in",
            "id": "--required",
            "required": true
        },
        {
            "name": "Type - Text",
            "description": "This and all above fields have got text type",
            "id": "--text",
            "type": "text"
        },
        {
            "name": "Type - Text Area",
            "description": "This field is text area. \nIt is resizable in most browsers. \nAnd allows to enter
            multiline texts.",
            "id": "--textarea",
            "type": "textarea"
        },
        {
            "name": "Type - Checkbox",
            "description": "This is the checkbox type",
            "id": "--checkbox",
            "type": "checkbox"
        },
        {
            "name": "Type - Password",
            "description": "This is the password field",
            "id": "--pass",
            "default": "qwerty",
            "type": "password"
        },
        {
            "name": "Type - Select",
            "description": "This is the select type. This description is not displayed in the web interface. Use
            description for each option instead.",
            "id": "--select",
            "default": "option2",
            "type": "select",
            "options": [
                {
                    "description": "== Not selected =="
                },
                {
                    "value": "option1", "description": "OK"
                },
                {
                    "description": "== Divider =="
                },
                {
                    "value": "option2", "description": "Warning"
                },
                {
                    "value": "option3", "description": "Error"
                }
            ]
        }
    ]
}
```

www.cpult.ru

2022

РОССИННО