Svace History Server

Version 5.0.0

Содержание

1 Введение

Сервер историй **Svacer (Svace Server)** предназначен для хранения и обработки результатов работы статического анализатора Svace. Он разработан с целью заменить старый сервер историй, встроенный в анализатор Svace.
Допускается одновременное использование старого и нового серверов.

1.1 Термины

Project	Проект, подлежащий анализу. По умолчанию именем проекта считается имя директории, в которой был запущен svace build и svace analyze. На сервере проект содержит в себе несколько branch с результатами работы анализатора
Branch (бранч)	Ветка в проекте с результатами работы анализатора. По умолчанию проект имеет одну ветку – master
Snapshot	Результат работы svace analyze, импортированный в промежуточное хранилище или на сервер. Содержит: ▪ имя проекта; ▪ имя бранча; ▪ результаты анализа (предупреждения); ▪ исходники, которые были включены в build, на котором производился анализ; ▪ вспомогательная информация для навигации по исходникам; ▪ дополнительные данные. Допускается импортирование snapshot'ов без исходного кода и дополнительной информации
Marker	Предупреждение от Svace с информацией о позиции в исходном файле
Svacer	Svace History Server, а также исполняемый файл, содержащий код backend сервера

1.2 Состав дистрибутива

bin/svacer **bin/svacer.exe**	Бинарный файл сервера и утилита для импортирования результатов. Режим работы зависит от аргументов
support/svace-migration/svace-migration.jar	Утилита подготовки данных для импорта из старого сервера историй. Требует JDK для запуска. Может использоваться JDK из состава дистрибутива Svace
extra/postgres	Portable-версия базы данных PostgreSQL
extra/docker-compose.yml	Docker-compose файл для запуска БД PostgreSQL и Svacer в docker-контейнерах

1.3 Зависимости

PostgreSQL > 10	Параметры подключения определяются аргументом при запуске сервера или переменной окружения SVACER_PG_URL. По умолчанию, используется следующий connect URL: **postgres://svace:svace@0.0.0.0:5432/svace** При отсутствии PostgreSQL можно использовать включенный в дистрибутив PostgreSQL из директории extra/postgres. Также можно использовать PostgreSQL в docker-контейнере. **Настоятельно рекомендуем** использовать выделенный сервер для хранения БД. Особенно в production
libc.so libpthread.so libdl.so linux-vdso.so	Серверная часть написана на Go с CGO_ENABLED=1 ввиду необходимости читать sqlite-файлы из результатов работы анализатора Svace
Выделенная директория для хранения object storage	Сервер использует встроенную key-value database для хранения информации об исходном коде: контент-файлов и вспомогательной информации для навигации
svace	Для импорта данных на сервер историй требуется наличие Svace на хосте, откуда происходит импорт. Svace используется для получения информации из .svace-dir

1.4 Используемые порты

Приведены значения по умолчанию. Все значения могут быть переопределены.

3002	Используется для предоставления gRPC API. gRPC API является внутренним и предназначен для загрузки данных на сервер
8080	Используется для предоставления REST API и web-сервера для предоставления web-UI

2 Установка

Для работы сервера необходим PostgreSQL. Рекомендуется использовать настроенную production-версию, но для быстрого запуска в тестовых целях в дистрибутив включена минимальная portable-версия PostgreSQL для Windows и Linux. Она находится в **./extra/postgres/<os>**.

Svacer сервер состоит из единственного бинарного файла – **svacer**. Он предоставляет как backend-часть, так и веб-сервер, который передаёт статику. По умолчанию запускается на порту 8080.

После запуска по умолчанию создается учётная запись пользователя **admin** с паролем **admin.** Изменить пароль можно в веб-интерфейсе.

Сервер имеет широкие возможности настройки параметров запуска. Узнать о них подробнее можно, прочитав **svacer --help**.

2.1 Установка и запуск из deb пакета

Скачайте .deb пакет релиза и выполните следующую команду

```
sudo apt install ./svacer_<version>_amd64.deb
```

При этом, если в репозиториях есть PostgreSQL нужной версии, он будет установлен автоматически. Если нет, то установка завершится с ошибкой о зависимости от PostgreSQL. В этом случае вам надо будет установить PostgreSQL версии не ниже 10-й (см. документацию), после чего повторить установку Svacer.

В процессе установки создаются следующие файлы и директории

- /etc/default/svacer — конфигурационный файл
- /var/log/svacer — директория для логов
- /var/lib/svacer — директория для object store

После установки Svacer требуется создать пользователя и БД PostgreSQL. Для этого перейдите в пользователя postgres, запустите psql и выполните соответствующие запросы, после чего выйдите из консоли PostgreSQL и из пользователя postgres.

```
sudo su -l postgres
psql
postgres=# create database svace;
postgres=# create user svace with encrypted password 'svace';
postgres=# grant all privileges on database svace to svace;
postgres=# alter user svace superuser;
```

В данном примере создается БД svace и права на нее выдаются пользователю svace с паролем svace, а также этому пользователю выдаются права суперюзера (это необходимо для создания расширений в PostgreSQL). При использовании этих значений дальнейшая конфигурация не требуется и можно переходить к запуску.

При использовании других имен пользователя, БД или пароля потребуется дополнительная конфигурация перед запуском Svacer: в файле **/etc/default/svacer** нужно поменять параметры подключения к БД в строке

```
SVACER_ARGS="--pg postgres://<user>:<password>@127.0.0.1:5432/<database>"
```

В этой же строке можно указывать прочие аргументы для запуска сервера Svacer.

После создания БД и настройки конфигурации Svacer запустить его можно следующими командами

```
sudo systemctl enable svacer
sudo systemctl start svacer
```

После чего проверить успешность запуска командой

```
systemctl status svacer
```

В случае успешного запуска сервер будет доступен по адресу http://localhost:8080

При установке из .deb пакета Svacer ставится в директорию, прописанную в $PATH, поэтому при запуске Svacer для импорта и загрузки результатов полный путь к исполняемому файлу **svacer** указывать не требуется, он будет доступен просто по имени.

2.2 Запуск в docker

Установите docker и docker-compose. Поскольку образы PostgreSQL и Svacer размещены на докер-хабе, для их скачивания при запуске необходимо наличие интернета.

Используйте для запуска docker-compose файл, находящийся в дистрибутиве Svacer: ./**extra/docker-compose.yml**. Перейдите в директорию, где находится этот файл и выполните команду

```
docker-compose up -d
```

При этом будет запущено два контейнера: PostgreSQL и Svacer. После запуска контейнеров веб-интерфейс Svacer будет доступен по адресу http://localhost:8080

В процессе запуска в текущей директории будут созданы две директории

- postgres_data – для хранения БД PostgreSQL
- svacer_data – для object store сервера Svacer

Эти директории будут примонтированы в соответствующие контейнеры как volumes, это необходимо для сохранения данных БД и сервера после остановки или перезапуска контейнеров.

Важное уточнение: в докере запускается только сервер Svacer. Для импорта и загрузки результатов на сервер будет нужен исполняемый файл svacer.

2.3 Установка и запуск вручную

Для установки Svacer скачайте и распакуйте архив дистрибутива (файл с названием вида **svacer_release-<version>.zip**).

2.3.1 Запуск с portable PostgreSQL

Запустить входящий в дистрибутив PostgreSQL можно, выполнив скрипт **start.cmd** или **start.sh** из директории для соответствующей ОС (**./extra/postgres/<os>**).

```
./extra/postgres/linux/start.sh
```

PostgreSQL будет запущен на порту 5432, будет создана БД svace, доступная от пользователя svace.

После этого запустите **svacer** (или **svacer.exe** для Windows). Параметр **--pg** для подключения к БД в данном случае можно не указывать, поскольку при запуске portable PostgreSQL был создан пользователь и база, прописанные в svacer как параметр по умолчанию.

```
./bin/svacer server
```

После запуска сервера веб-интерфейс Svacer будет доступен по адресу http://localhost:8080

Остановить PostgreSQL можно, запустив скрипт **stop.cmd** или **stop.sh**, соответственно.

2.3.2 Запуск с заранее настроенным PostgreSQL

При использовании не тестового сервера PostgreSQL, включенного в поставку, а предварительно настроенного на каком-либо хосте, необходимо использовать

соответствующие параметры запуска. Например, при использовании локально запущенного PostgreSQL на порту 5432 с учётной записью пользователя **svacer**, паролем **svacer123** и БД **svacer_database**, сервер необходимо запускать следующим образом:

```
./bin/svacer server --pg
postgres://svacer:svacer123@127.0.0.1:5432/svacer_database
```

После запуска сервера его веб-интерфейс будет доступен по адресу http://localhost:8080

2.4 Hooks

Для получения данных Hooks при запуске сервера нужно указать опцию **`--hooks <path to JSON file>`**. Эта опция задаёт файл, содержащий информацию о расширении сервера посредством хуков, которые пользователь может вызывать из web-интерфейса.
Каждый хук соответствует некоторому процессу или скрипту, который сервер запускает в ответ на вызов соответствующей команды из web-интерфейса.
Формат файла Hooks следующий:

```
{
  "hooks": [
    {
      "id": "<id>",
      "label": "<label >",
      "target": "<target>",
      "input": ["<param1>", "<param2>",…],
      "cmd": "<path to executable>",
      "args": ["<arg1>", "<arg2>", …]
    },
….
  ]
}
```

Табл. 1 – Параметры Hooks

Параметр	Описание
id	Идентификатор хука. Должен быть уникальным в файле
label	Имя команды, которую пользователь видит в web-интерфейсе
target	Место в UI, в которое будет добавлена команда. Сейчас поддерживается только одно значение – default. Оно соответствует вкладке Details на правой панели пользовательского интерфейса
input	Параметры, которые можно передать в запускаемый процесс, исходя из контекста вызова команды в web-интерфейсе. Поддерживаемые значения:

	▪ markerID – UUID идентификатор маркера; ▪ branchID – UUID идентификатор бранча; ▪ snapshotID – UUID идентификатор snapshot (снимка файловой системы); ▪ projectID – UUID идентификатор проекта; ▪ url – URL маркера в web-интерфейсе, на котором была вызвана команда; ▪ marker – будет заменен на полное имя временного файла, содержащего сериализованное в JSON представление маркера, который включает в себя его трассу и информацию о разметке
cmd	Полный путь к процессу, который будет запущен. Не должен включать аргументы запуска
args	Аргументы, передаваемые в запускаемый процесс. Полный список аргументов запускаемого процесса состоит из списка <args> и списка значений, соответствующих полю <input>

3 Использование

3.1 Получение справки

Запустив Svacer без параметров, можно получить информацию о доступных командах. Дополнительная информация о каждой команде доступна при использовании опции **`--help`**:

```
svacer <command name> --help
```

Общий формат командной строки:

```
svacer [global options] command [command options] [arguments...]
```

Глобальные опции относятся ко всем командам.
Опции команд имеют префикс «--». Их следует указывать до списка аргументов.

3.2 Процесс импорта

Процесс импорта результатов работы статического анализатора Svace состоит из двух шагов:

1. Дедупликация данных, предобработка, конвертирование во внутренний формат и сохранение результатов в промежуточном хранилище.
2. Загрузка данных из промежуточного хранилища на сервер.

3.2.1 Промежуточное хранилище

Промежуточное хранилище используется для хранения результатов, подготовленных для загрузки на сервер. При импорте данных в промежуточное хранилище snapshot-ам назначается уникальный ID, задается имя проекта и бранча.
Промежуточное хранилище можно использовать для аккумулирования результатов нескольких запусков анализатора, в том числе для различных проектов.
По умолчанию, промежуточное хранилище создается в директории **`.svacer-dir`** рядом с директорией **`.svace-dir`**.
Параллельный доступ к хранилищу из разных экземпляров Svacer **не допускается**. Если необходимо использовать разделяемое хранилище и конкурентный доступ, Svacer следует запускать в режиме удаленного промежуточного хранилища.
Процедура импорта с промежуточным хранилищем по умолчанию (размещается в **`<path to project>/.svacer-dir`**):

```
# <path to svace> - путь к исполняемому файлу svace из дистрибутива svace
# <path to project> - путь к проекту (директории, содержащей директорию .svace-dir
svacer import --svace <path to svace> <path to project>
```

Процедура импорта с явным указанием размещения промежуточного хранилища:

```
svacer import --store <path to store> --svace <path to svace> <path to project>
```

3.2.2 Загрузка данных на сервер

Для импорта данных из промежуточного хранилища на сервер используется команда **upload**:

```
svacer upload --user <user> --password <pwd> --host <host> --port <rest  port> --grpc <grpc port> <path to store>
```

Если отсутствуют параметры:

- **--user** и **--password**, то используется специальный системный пользователь **importer** – он предназначен для импорта данных на сервер;
- **--host**, то используется **locahost**;
- **--port** и **--grpc**, то используются их значения по умолчанию.

При загрузке данных на сервер уже загруженные snapshot-ы повторно не загружаются. Разделяемая информация (такая как исходные файлы: где один и тот же файл может использоваться в различных анализах и т. п.) также повторно не загружается.
При сбоях в ходе загрузки проводится повторная загрузка, но при этом загружаются только недостающие данные.

3.2.3 Удаленное промежуточное хранилище

Svacer можно запустить в режиме удаленного промежуточного хранилища. Взаимодействие с удаленным хранилищем выполняется с помощью gRPC-протокола:

```
svacer server --remote-store --store <path to dir> --grpc <port number>
```

Для импорта данных в удаленное промежуточное хранилище используется специальный формат аргумента:

```
svacer import --store remote:<host>:<grpc port> <path to project>
```

Для загрузки данных из удаленного промежуточного хранилища на сервер используется такой же формат спецификации параметра **--store**:

```
svacer upload --user <user> --password <pwd> --host <host> --port <rest port> --grpc <grpc port> remote:<host>:<grpc port>
```

4 Миграция

4.1 Миграция со старого сервера

В данном подразделе описана миграция со старого сервера, поставляющегося в комплекте со svace. Это не относится к миграции между версиями Svacer. Миграция данных со старого сервера на Svacer состоит из двух этапов:

1. Экспорт информации со старого сервера.
2. Импорт данных в новый.

Для экспорта информации со старого Svace сервера используется утилита **`svace-migration.jar`**:

```
java -jar svace-migration.jar <user> <password> <path to svace> <path to server dir> <path to output dir>
```

Табл. 2 – Параметры, используемые при экспорте данных

Параметр	Описание
<user>	Имя пользователя на старом сервере
<password>	Пароль пользователя на старом сервере
<path to svace>	Путь к директории, где находится Svace
<path server dir>	Директория, из которой запущен старый сервер
<path to output dir>	Директория, в которую будут записаны экспортируемые данные

Экспорт данных возможен при работающем старом сервере. Экспорт инкрементальный, поэтому ранее экспортированные результаты повторно не переносятся.

При экспорте утилита **`svace-migration.jar`** извлекает:

- все snapshot-ы в виде **`*.svres`**;
- информацию о размещении исходного кода;
- сопутствующую информацию в директории старого сервера.

Для импорта экспортированных данных используется следующая команда:

```
svacer migrate [--skip-dxr-errors] --distr <path to svace distr> --temp <path to temp folder>  --store <intermediate store> <path to exported data>
```

Табл. 3 – Параметры, используемые при импорте данных

Параметр	Описание
--skip-dxr-errors	Опциональный параметр для игнорирования ошибок в DXR-данных. Рекомендуется к использованию
--distr	Путь к дистрибутиву Svace
--temp	Директория для хранения временных файлов в ходе конверсии
--store	Путь к промежуточному хранилищу
<path to exported data>	Путь к директории, в которую были экспортированы результаты

После конвертации данных в промежуточное хранилище результат можно загрузить на новый сервер с помощью команды upload:

```
svacer upload --user <user> --password <pwd> --host <host> --port <rest  port> --grpc <grpc port> <path to store>
```

4.2 Миграция между версиями Svacer

Миграция между версиями Svacer происходит автоматически при обновлении на новую версию. Рекомендуется делать резервные копии перед обновлением (см. раздел 4.3).

Ниже описаны существенные изменения при переходе на определенную версию с более старой

- 4.0.1 и выше — миграция object store (badger) на новую версию. Может занимать существенное время при большом количестве проектов в Svacer. После перехода на версию 4.0.1 и выше object store становится несовместим с более ранними версиями
- 5.0.0 и выше — перенос DXR разметки из PostgreSQL в object store. Версии выше 5.0.0 не совместимы с более ранними версиями.

4.3 Создание и восстановление резервной копии

4.3.1 Если PostgreSQL запущен в виде сервиса

Если PostgreSQL запущен в виде сервиса (т.е. не в docker-контейнере), то для создания резервной копии БД перейдите в пользователя postgres и запустите создание дампа:

```
sudo su -l postgres
pg_dump --clean -T tmp_diff* svacer_db > svacer.sql
```

где `svacer_db` – имя БД.

В результате дамп БД будет создан в файле `svacer.sql`

Для восстановления БД из дампа также перейдите в пользователя postgres и запустите восстановление

```
sudo su -l postgres
psql -f svacer.sql svacer_db
```

4.3.2 Если PostgreSQL запущен в docker-контейнере

Если PostgreSQL запущен в docker-контейнере, то для создания резервной копии БД выполните следующую команду

```
docker exec -t distr_postgres_db_1 pg_dump --clean -T tmp_diff* -U postgres svacer_db > svacer.sql
```

где:

- `distr_postgres_db_1` – имя docker-контейнера
- `postgres` – имя пользователя, под учётной записью которого создана БД

- **svacer_db** – имя БД

Для восстановления БД используйте команду

```
cat svacer.sql | docker exec -i distr_postgres_db_1 psql -U postgres svacer_db
```

4.3.3 Если используется PostgreSQL, входящий в поставку

При использовании PostgreSQL, входящего в поставку Svacer для создания резервной копии выполните следующую команду

```
<svacer_dir>/extra/postgres/linux/bin/pg_dump –clean -U svace svace > svacer.sql
```

Для восстановления БД используйте команду

```
<svacer_dir>/extra/postgres/linux/bin/psql -f svacer.sql -U svace svace
```

4.3.4 Создание резервной копии svacer object store

Для создания резервной копии object store достаточно скопировать директорию, в которой он находится. Перед тем как это делать желательно остановить сервер Svacer.

Если директория, где хранить **object store** не была явно указана при запуске сервера, по умолчанию он будет создан в следующей директории:

- **~/.cache/svacer** – в ОС Linux
- **%LocalAppData%\svacer** – в ОС Windows
  (например, C:**\Users\myusername\AppData\Local\svacer**)

При восстановлении из резервной копии следуют поместить файлы object store в директорию по умолчанию (см. выше), либо же в другую директорию и при запуске Svacer указать путь к ней одним из способов:

1) Перед запуском Svacer установить переменную окружения SVACER_OBJECT_STORE, например:

   ```
   export SVACER_OBJECT_STORE=/home/user/data/object_store
   ```

2) При запуске Svacer передать путь к object store в параметре **--store**, например:

   ```
   ./svacer --store /home/user/data/object_store
   ```

При запуске Svacer в docker-контейнере требуется смонтировать директорию с object store в контейнер как volume и указать путь к ней в переменной **STORE** в файле **docker-compose.yml**. Допустим, директория с object store была скопирована в /home/user/data/object_store. В таком случае параметры нужно указать следующим образом

```
volumes:
  - /home/user/data:/data
environment:
  - STORE=/data/object_store
```

4.4 Перенос svacer с одного сервера на другой

1. Создайте резервную копию БД и object store в соответствии с тем, как описано в разделе 4.3
2. Перенесите созданные файлы резервных копий на нужный сервер
3. Восстановите данные из резервной копии на новом сервере

5 Работа с сервером

5.1 Вход в сервер

Для входа в сервер введите логин и пароль на странице ввода учетных данных (Рис. 1). Учётные данные по умолчанию – admin / admin.

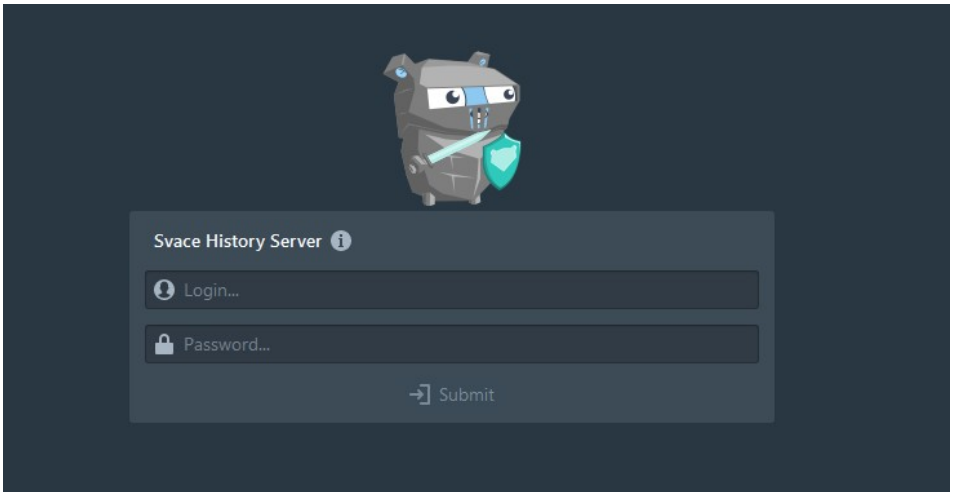

Рис. 1 – Страница ввода учетных данных

5.2 Выбор проекта и snapshot’а

После успешной авторизации выберите проект, бранч и snapshot для просмотра результатов работы анализатора Svace (Рис. 2).

По умолчанию после выбора проекта автоматически выбирается бранч master и самый свежий snapshot. Пример выбранных проекта, бранча и snapshot’а представлен на Рис. 3.

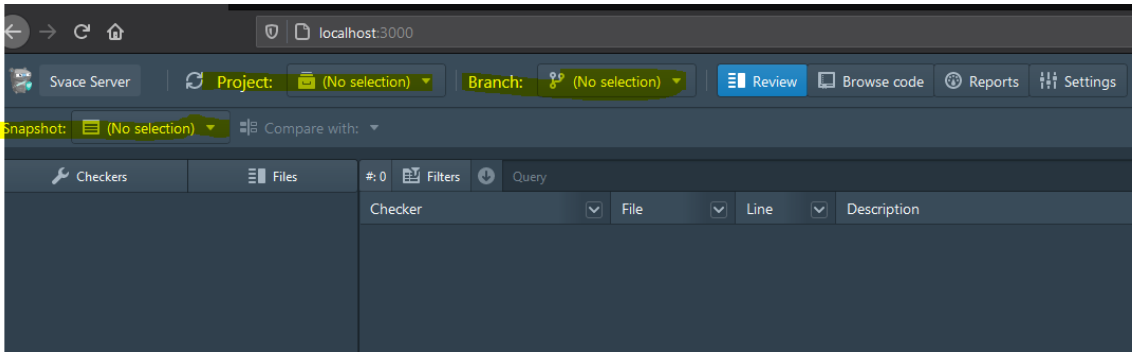

Рис. 2 – Проект, бранч и snapshot в интерфейсе

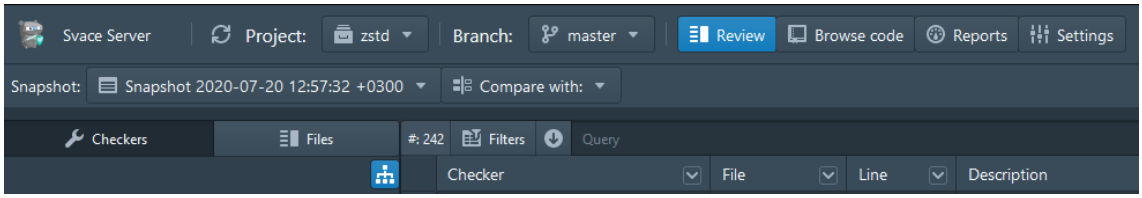

Рис. 3 – Пример выбранных проекта, бранча и snapshot’а

5.3 Виды интерфейса

Сервер историй предоставляет несколько режимов работы интерфейса, каждый из которых предназначен для выполнения определённых действий:

- **Review** – просмотр и разметка найденных предупреждений, а также сравнение snapshot'ов или отдельных предупреждений;
- **Browse Code** – просмотр snapshot'ов исходного кода, связанного с результатами анализатора Svace;
- **Reports** – формирование отчетов на основе хранимой информации;
- **Settings** – управление сервером и просмотр его состояния.

В каждом режиме интерфейс предоставляет набор панелей, которые позволяют выполнять различные действия.

5.4 Элементы управления в режиме разметки

Режим разметки предоставляет следующий набор элементов управления:

1. Левая группа панелей:

   1) Панель Checkers – показывает список чекеров Svace, сработавших в выбранном snapshot'е (Рис. 4).

      Пользователь может выбрать показ списка чекеров в виде дерева, группирующего чекеры по **checker severity** (Рис. 5).

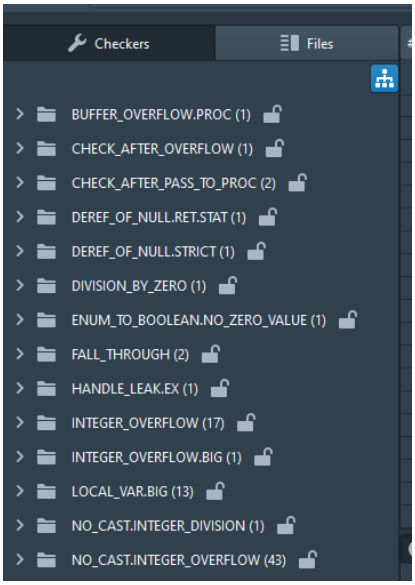

Рис. 4 – Панель Checkers. Данные в виде списка

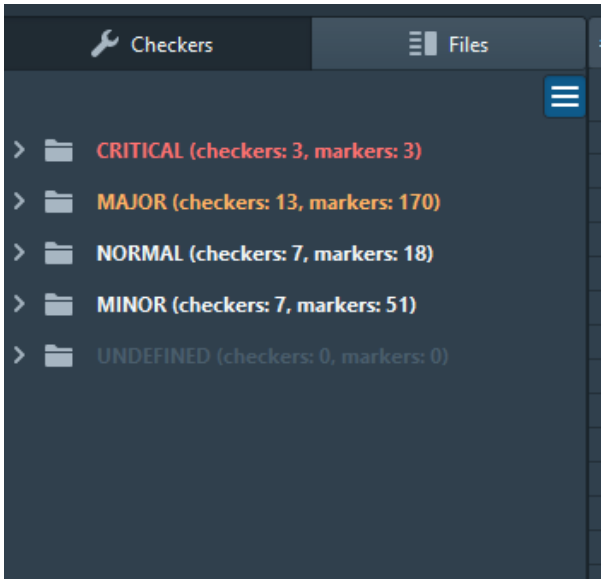

Рис. 5 – Панель Checkers. Данные в виде дерева

   2) Панель Files – показывает список файлов, для которых есть какие-либо предупреждения в snapshot (Рис. 6). Столбец ISSUES показывает число предупреждений в файле.

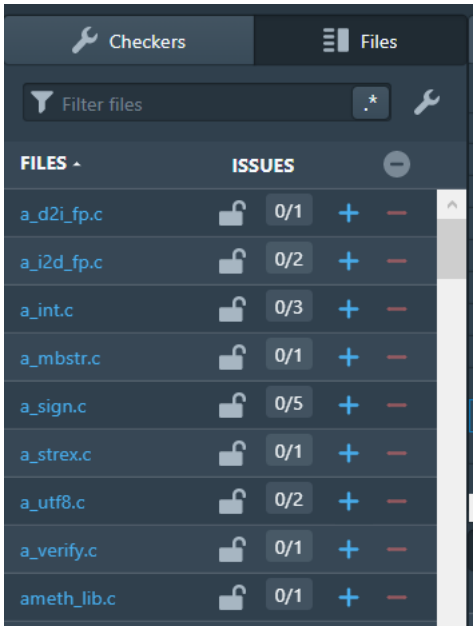

Рис. 6 – Панель Files

Действия, доступные на панели **Files**:

- клик на **FILES** или **ISSUES** – позволяет отсортировать столбцы по значениям;
- нажатие кнопок + и − – позволяют добавить или удалить соответствующий файл в фильтре для просмотра предупреждений в таблице предупреждений;
- ввод в поле Filter files имени файла или пути к нему – позволяет задать фильтрацию по имени/пути файла, а также включить фильтрацию по регулярным выражениям, нажав на кнопку .*, которая находится в правой части поля ввода.

2. Центральная группа элементов:

   1) Таблица предупреждений (Рис. 7, 1):

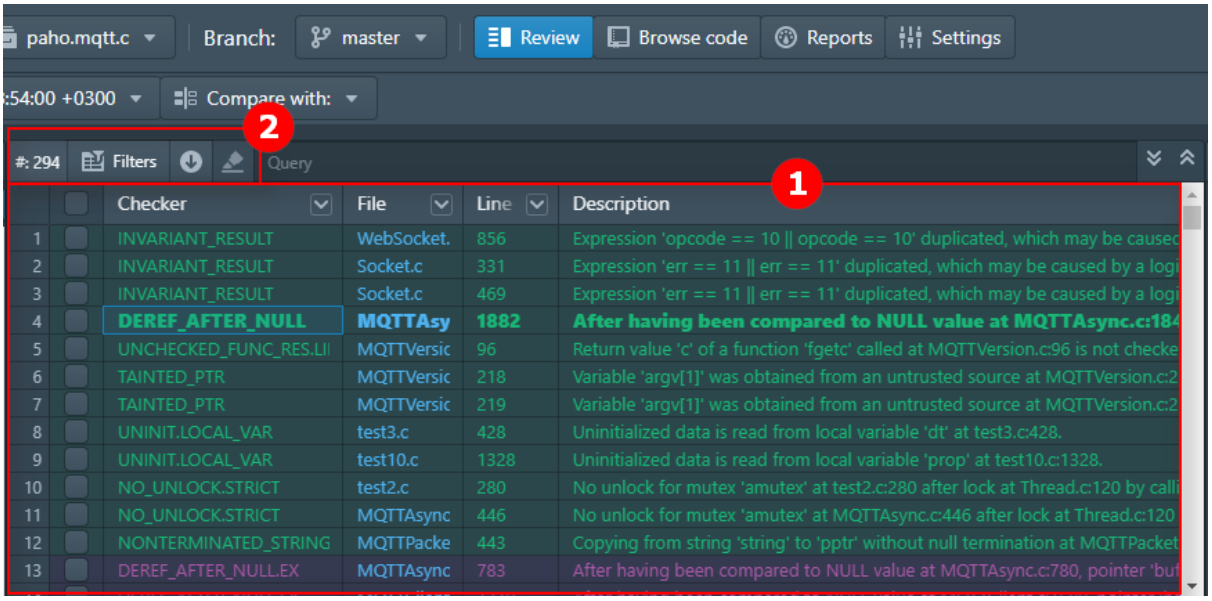

Рис. 7 – Панель и таблица предупреждений

Эта таблица является основным элементом для работы с предупреждениями. Действия, доступные в таблице:

- выбор предупреждений для групповой разметки;
- выбор активного предупреждения – двойной клик на строке в любом столбце или одинарный клик на имени файла в столбце **File**.

Также можно использовать клавиши управления курсором и Enter для выбора активного предупреждения.

2) Панель предупреждений (Рис. 7, 2)

Информация и действия, доступные на панели:

- число отображаемых в таблице предупреждений;
- фильтрация предупреждений (Рис. 8):
  - комплексная фильтрация предупреждений – выбор **Filters** – **Basic Filter**;
  - показ только предупреждений с разметкой пользователя – выбор **Filters** – **Triaged**.
- выгрузка отображаемых предупреждений в формате .csv и выгрузка отчета отображаемых предупреждений в формате .pdf (Рис. 9);
  - групповая разметка выбранных предупреждений (подробнее в разделе 5.5);
  - поиск предупреждений – ввод значения в поле query (поддерживается использование метасимволов * и ?).

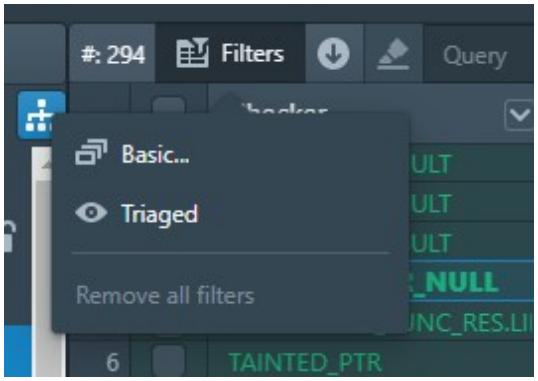

Рис. 8 – Фильтры предупреждений

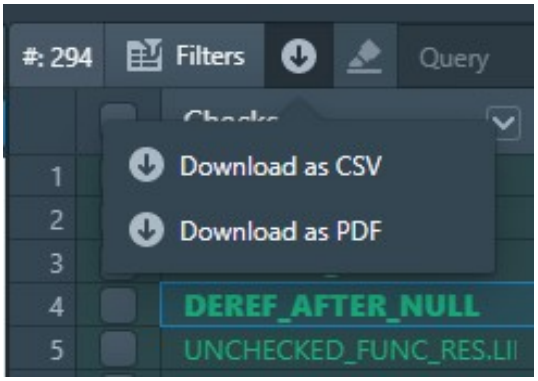

Рис. 9 – Выгрузка предупреждений

3) Панель с кодом и информацией о snapshot (Рис. 10):

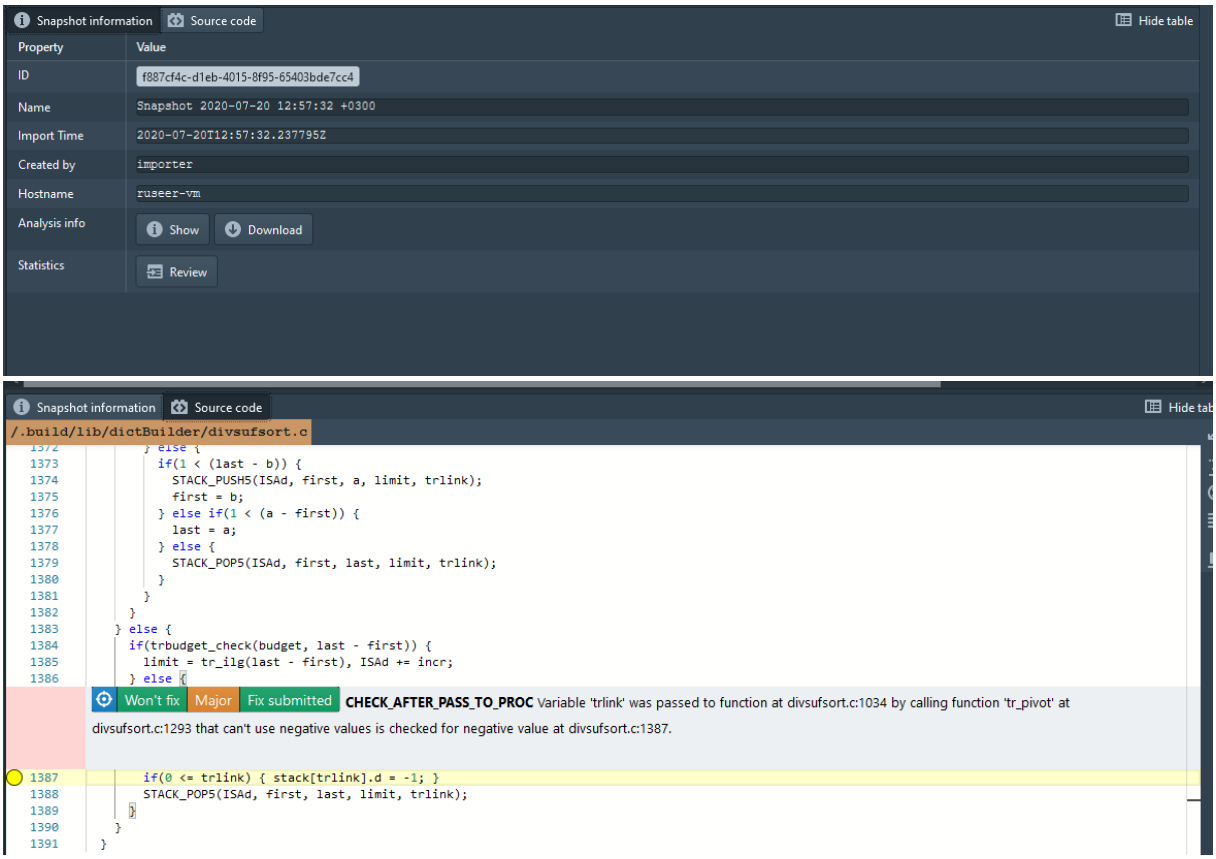

Рис. 10 – Панель с кодом и информацией о snapshot

Эта панель предоставляет информацию о snapshot и показывает положение предупреждения в исходном коде.

Кнопка **Hide Table** позволяет развернуть панель с кодом на всю среднюю панель.

В части панели с кодом секция с информацией о предупреждении содержит кнопки, которые позволяют разметить открытое предупреждение.

3. Правая группа панелей (Рис. 11).

Эта группа предоставляет дополнительную информацию о предупреждении, а также позволяет размечать предупреждения, добавлять комментарии и просматривать трассу предупреждения.

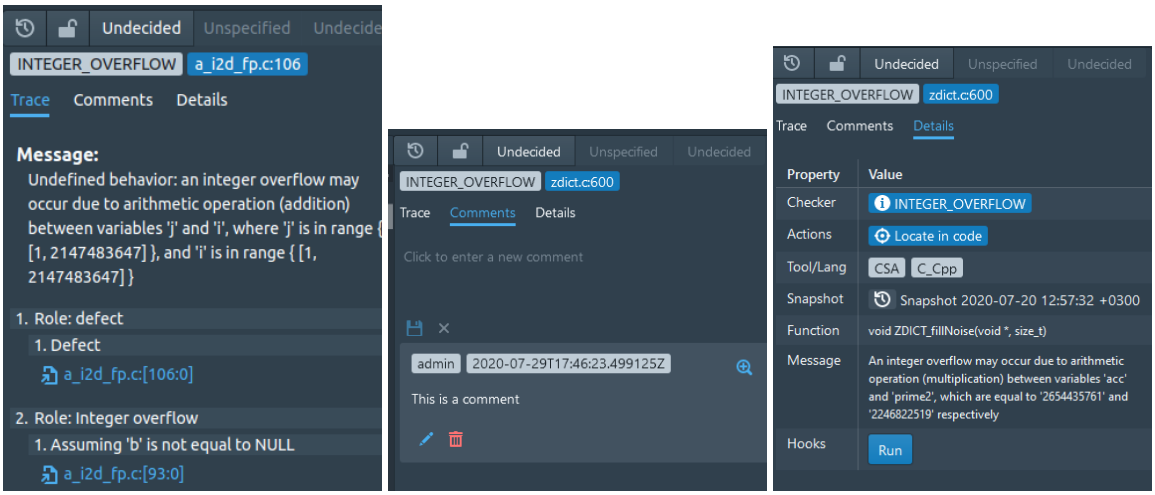

Рис. 11 – Правая группа панелей

Доступные действия:

- отображение исходного кода в панели Source code – доступно по клику на ссылку с именем файла и номером строки;

  Это позволяет всегда вернуться к нужной точке при навигации по коду.
- отображение информации (если доступна) о выбранном чекере (Рис. 12) – доступно по клику на имени чекера в закладке **Details**;

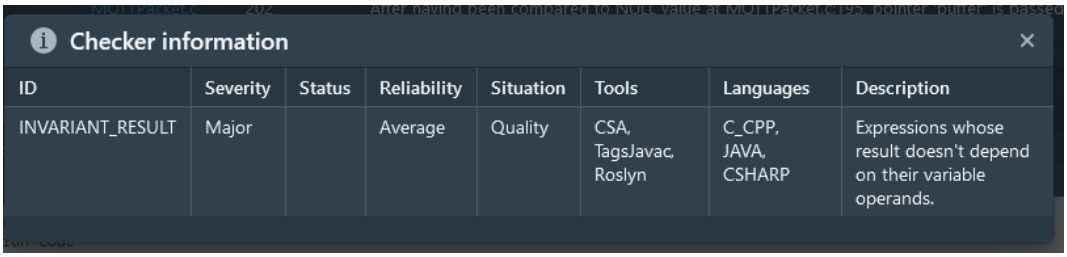

Рис. 12 – Информация о чекере

- предпросмотр кода, который соответствует строке трассы предупреждения, в отдельном окне (Рис. 13) – доступно по клику на иконку  ;
- разметка предупреждения с помощью кнопок разметки (Рис. 14);

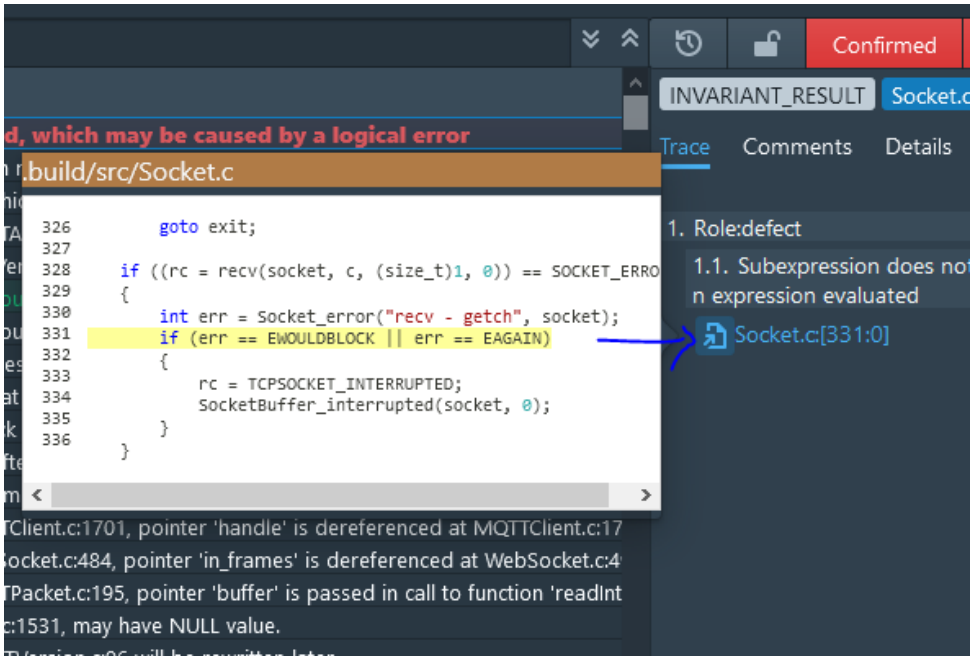

Рис. 13 – Окно предпросмотра кода трассы

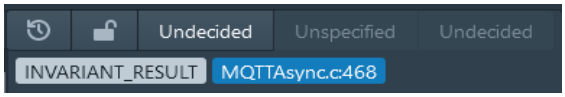

Рис. 14 – Кнопки разметки предупреждения

Разметка предупреждения заключается в установке статуса (**Confirmed**, **Won't Fix**, **False Positive**, **Unclear**) и опциональной установки **severity** и **action** (доступно только после установки **Status**).

- просмотр истории изменения разметки предупреждения (Рис. 15) – доступно по нажатии кнопки ;

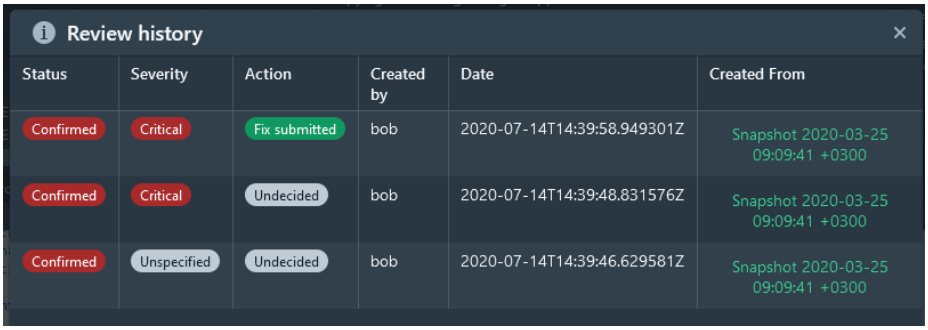

Рис. 15 – История изменения разметки предупреждения

5.5 Групповая разметка предупреждений

Для групповой разметки:

1. Выделите одно или несколько предупреждений (Рис. 16).
2. Нажмите кнопку **Review markers group**.

   Кнопка становится активной, только если есть выбранные предупреждения.

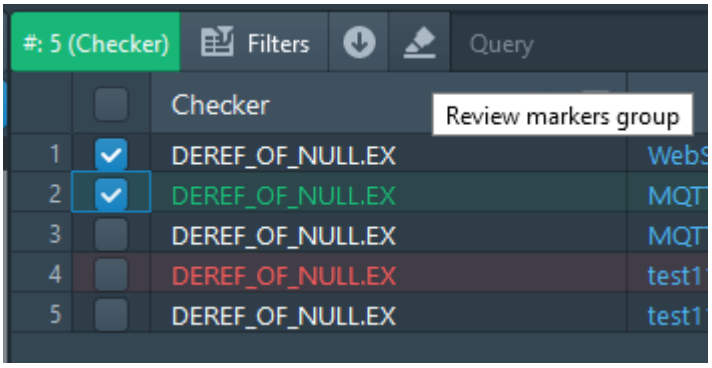

Рис. 16 – Кнопка «Review markers group»

Отобразится окно для групповой разметки (Рис. 17):

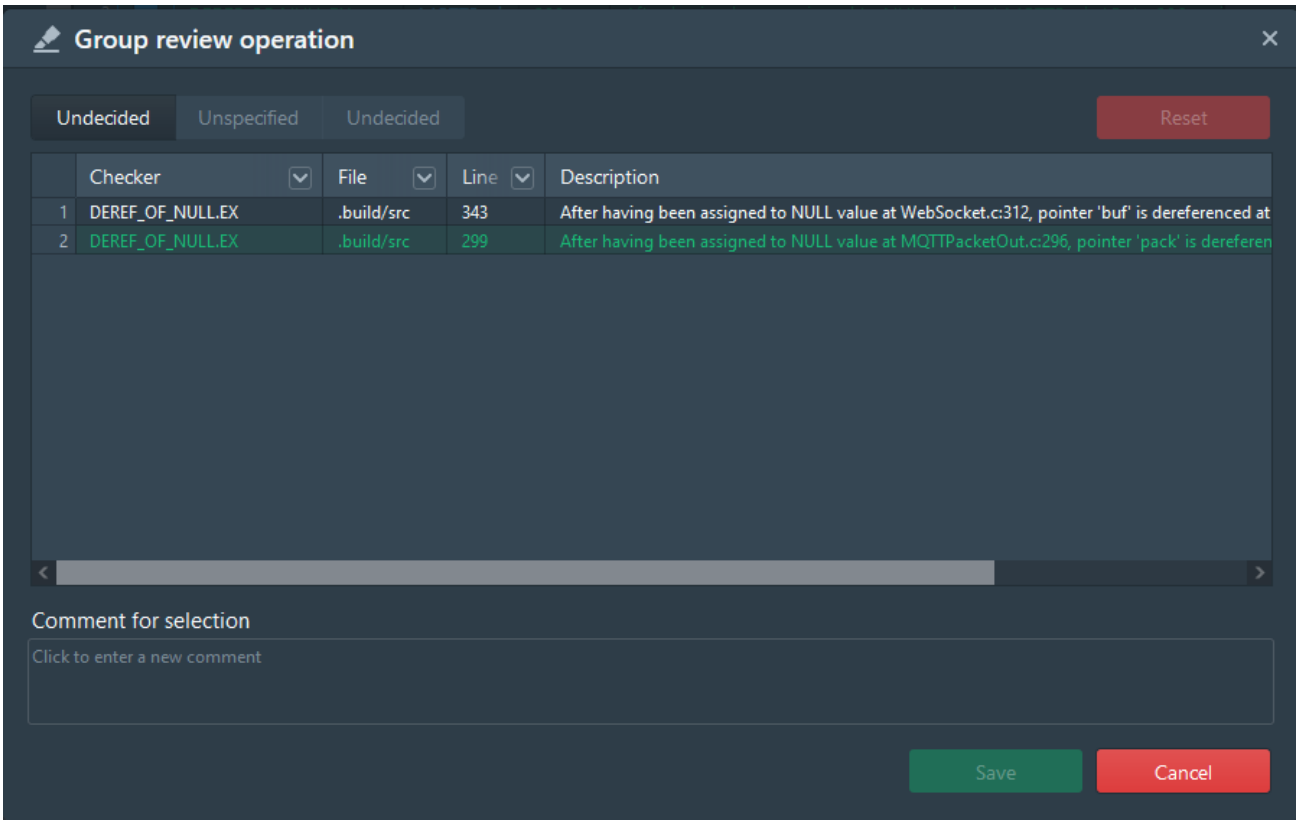

Рис. 17 – Окно для групповой разметки. Начальное состояние

3. Разметьте выбранные предупреждения.

Отображаемые в окне предупреждения окрашиваются в соответствующий цвет. Например, если пользователь поставил статус проверки **Confirmed**, то все предупреждения окрасятся в красный цвет (Рис. 18).

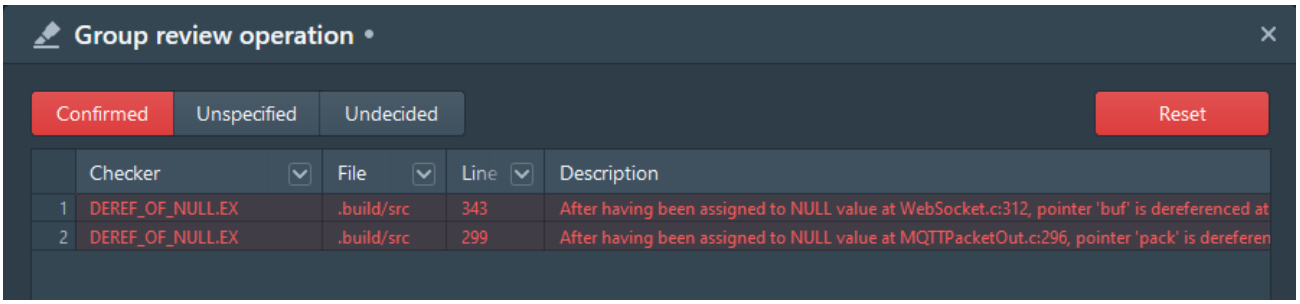

Рис. 18 – Пример размеченных предупреждений.
Установлен статус проверки «Confirmed»

4. Добавьте общий комментарий для выбранных предупреждений.

   После того, как пользователь разметил и (или) написал комментарий, станут доступными кнопки **Reset (Сбросить)** и **Save (Сохранить)**, а справа от заголовка окна появится индикатор – кружок (Рис. 19).

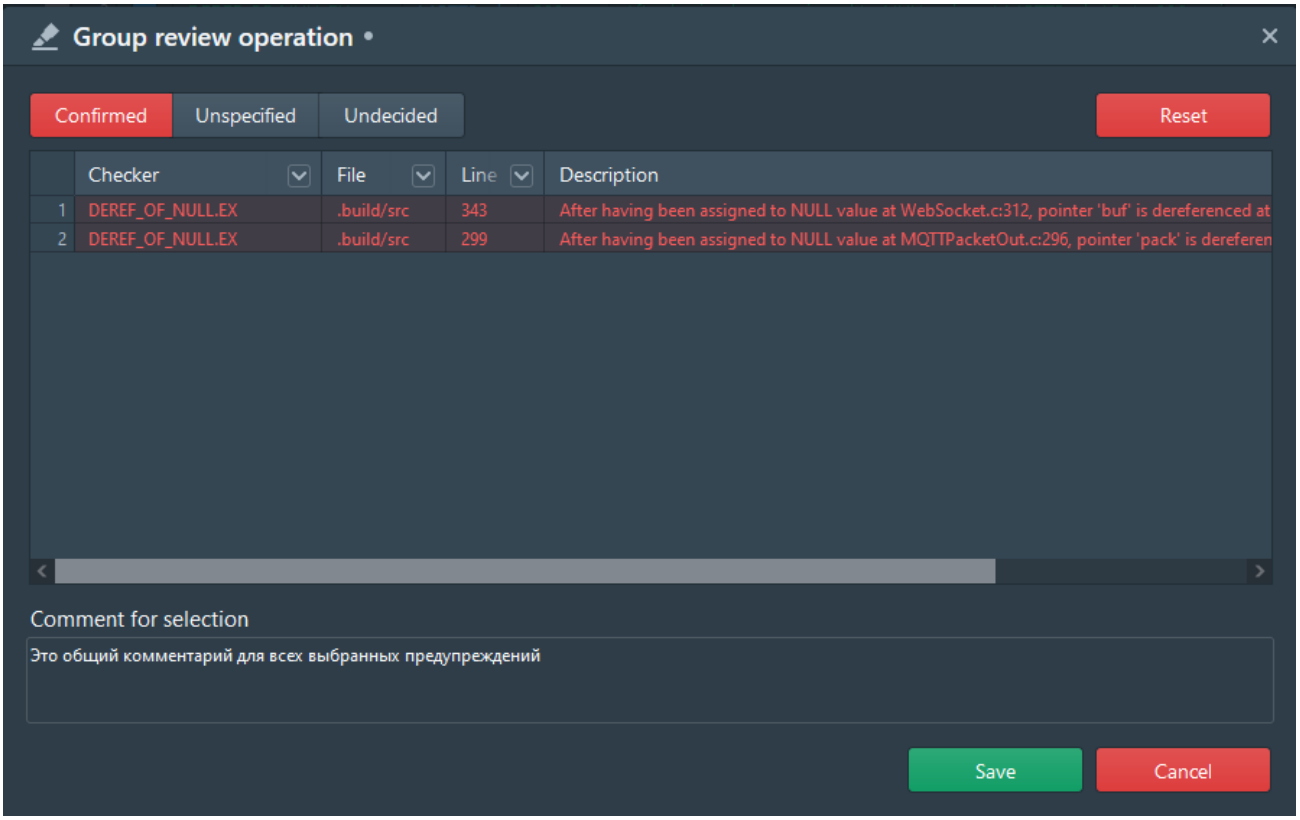

Рис. 19 – Окно для групповой разметки с кнопками

5. Если требуется отменить изменения, нажмите ✕ в правом верхнем углу окна или кнопку **Cancel (Отмена)**.

Диалоговое окно закроется и изменения не будут применены.

6. Если требуется сбросить все изменения, нажмите кнопку **Reset**.

   Данные в окне вернутся в первоначальное состояние, кнопки **Reset** и **Save** станут неактивными, а также пропадёт индикатор справа от заголовка окна (Рис. 17).

7. Нажмите кнопку **Save**, чтобы сохранить изменения разметки и сбросить выбор в таблице предупреждения (Рис. 20).

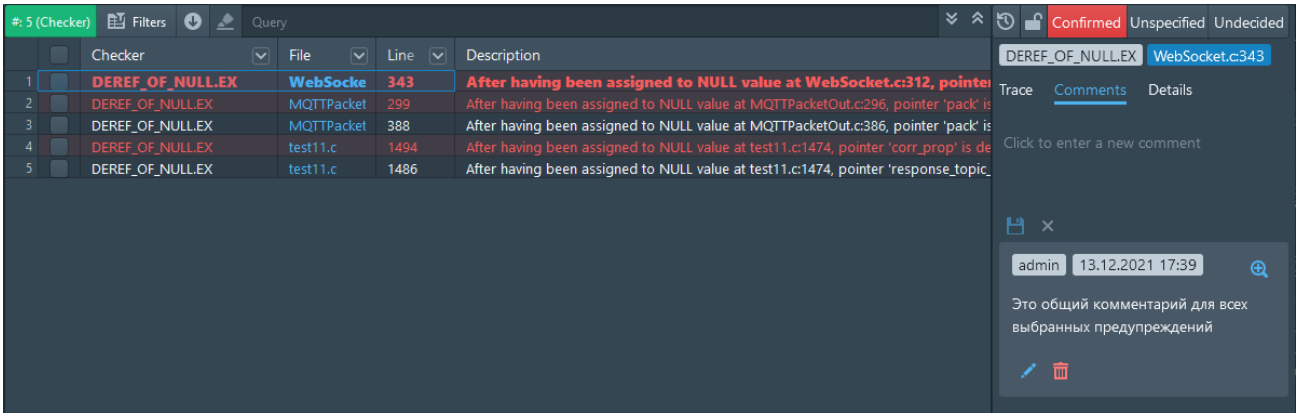

Рис. 20 – Таблица предупреждений после сохранения групповой разметки

5.6 Использование регулярных выражений

Регулярные выражения доступны в левой панели на вкладках **Files** и в **Filters>Basic filters>Files**.

Чтобы включить поиск/фильтрацию по регулярному выражению, нужно нажмите кнопку **«.*»** рядом с соответствующим полем ввода. При использовании регулярного выражения оно применяется к полному пути файла.

В **Basic filters>Files** существует возможность применить фильтр как для отображения только предупреждений из файлов, подходящих под паттерн регулярного выражения, так и для скрытия таких предупреждений. Для переключения между этими двумя режимами нажмите кнопку **«+»/«-»**, которая расположена рядом с полем ввода регулярного выражения.

Примеры применения регулярных выражений[1]:

1. Чтобы отобразить только предупреждения из файлов с расширением .c, используйте выражение:

   `\.c$`

2. Чтобы скрыть предупреждения из файлов, имя которых начинается с символа «q» и которые имеют расширения .h или .hpp:

   1) Используйте `/q.*\.h|/q.*\.hpp`
   2) Нажмите кнопку «+» рядом с полем ввода, чтобы она отобразилась как «-».

[1] Примеры приведены для **Basic filters>Files**, но на панели **Files** можно использовать такие же

3. Чтобы показать только предупреждения из файлов, которые имеют «string» в конце имени файла и с расширение из одного символа, используйте выражение:

```
/.*string\..$
```

4. Чтобы скрыть предупреждения из файлов в директориях ans1 и pem:

   1) Используйте выражение `/asn1/|/pem/`
   2) Нажмите кнопку **«+»** рядом с полем ввода, чтобы она отобразилась как **«-»**.

5. Чтобы показать только предупреждения из файлов, которые имеют в имени три цифры подряд, используйте выражение:

   ```
   /.*[0-9]{3}[^/]*
   ```

   Где конструкция `[^/]*` означает, что после трех цифр может встречаться любой символ кроме /. Это позволяет исключить директории имеющие три цифры в названии.

5.7 Блокировка разметки

Пользователь может заблокировать другому пользователю возможность размечать предупреждение или группу предупреждений. Это обеспечивает возможность разделения работы нескольких пользователей по разметке результатов без возникновения коллизий.

Для блокировки разметки используйте кнопку  :

- на панели чекеров (Рис. 21);

  В этом случае блокировка распространяется на все предупреждения чекера в выбранном проекте и бранче (для всех snapshot’ов на бранче).

- на панели файлов (Рис. 22);

  В этом случае блокировка распространяется на все предупреждения в файле в выбранном проекте и бранче (для всех snapshot’ов на бранче).

- на панели разметки (Рис. 23);

  В этом случае блокировка распространяется на все эквивалентные предупреждения в выбранном проекте и бранче (для всех snapshot’ов на бранче).

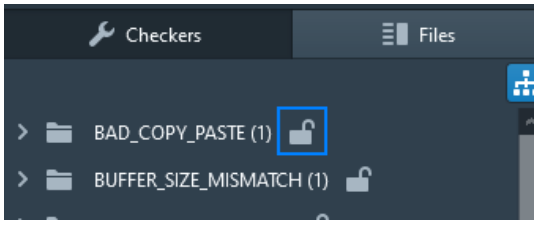

Рис. 21 – Блокировка на панели чекеров

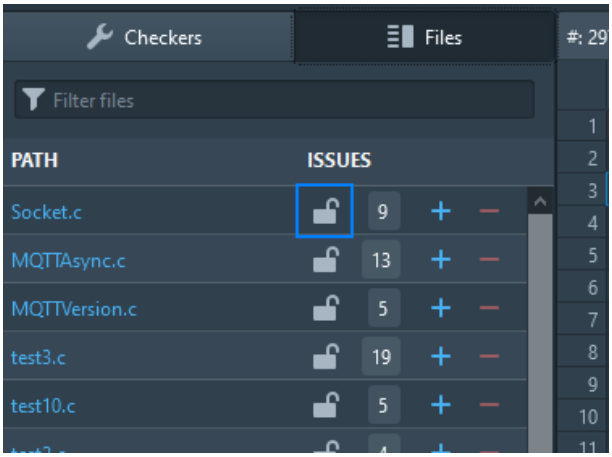

Рис. 22 – Блокировка на панели файлов

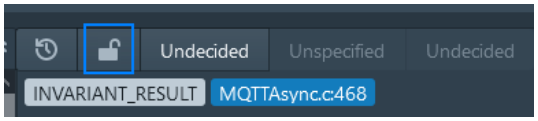

Рис. 23 – Блокировка на панели разметки

Для просмотра всех блокировок используйте панель **Settings -> Review Locks** (Рис. 24):

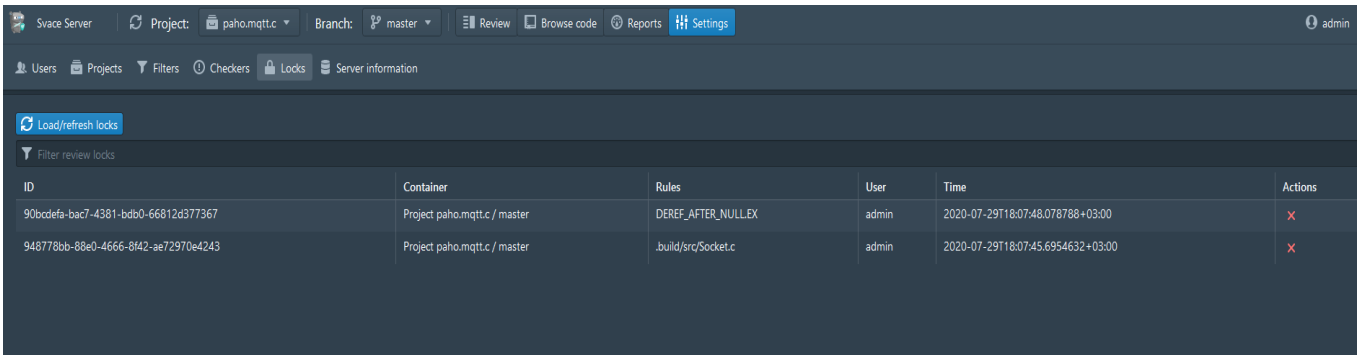

Рис. 24 –Панель Review Locks

Также на этой панели можно удалить свои блокировки.
Пользователь admin может удалять любые блокировки.

5.7.1 Импорт разметки из исходного кода

Для удобства пользователей предусмотрены следующие функции разметки предупреждения непосредственно в коде с помощью специального вида комментариев (доступно только при добавлении флага `--parse-comments` при импорте):

- добавление ревью к маркеру, созданному Svace;
- подавление маркера с конкретным CHECKER_ID;
- подавление любого маркера;

Синтаксис:

```
//svacer_review:[-]<CHECKER_ID>|r:|s:|a:|d:|c:
```

Где:

- r – status;
- s – severity;

- a – action;
- d – time (формат 2021-04-21T10:00:00Z);
- c – comment.

Комментарий вида **`// svacer_review`** (с пробелом после //) будет проигнорирован. Во всех остальных полях лишние пробелы допустимы.
Каждое поле необязательно, но если отсутствует поле **r**, то последующие поля **a** и **s** по разметке маркера будут проигнорированы.
Комментарии специального вида:

- **`//svacer_review:-`**
- игнорирование всех маркеров;
- **`//svacer_review:-INTEGER_OVERFLOW`**
- игнорирование только указанного маркера. Если в такой строке Svace определит маркер другого типа, этот комментарий не будет нести никакой смысловой нагрузки;
- **`//svacer_review:CHECK_AFTER_PASS_TO_PROC|r:False Positive|s:Major`**
- добавление к маркеру CHECK_AFTER_PASS_TO_PROC, который нашел Svace, ревью с перечисленными полями;
- **`//svacer_review:CHECK_AFTER_PASS_TO_PROC|r:False Positive|s:Major| INTEGER_OVERFLOW|r:Unclear|a:Ignore`**
- разметка сразу нескольких маркеров в одной строке: в результате будет создано ревью по обоим маркерам.

Все комментарии имеют вес, только если одновременно выполнены условия:

- комментарий указан в строке, которая соответствует строке, указанной Svace;
- указан **CHEKER_ID**, который совпадает с **CHEKER_ID**, который указан в Svace.

Если до комментария (**`some code //svacer_review: ...`**) есть что-либо помимо пробелов и табуляций, то считается, что комментарий относится к текущей строке. Иначе – к следующей строке.
Если комментарий содержит CHECKER_ID, который одинаков для строки над строкой с маркером и в конце нее, учитывается тот комментарий, который находится над строкой с маркером.

5.7.2 Экспорт разметки в исходный код

Есть возможность экспортировать разметку с сервера в исходный код с помощью команды **`export_markup`**.
Синтаксис:

```
Svacer export_markup [command options] [arguments…]
```

Флаги:

- **`--snapshot_id <snpshotID>`** – указывает id snapshot'a, разметка которого будет экспортироваться.
- **`--show-lines`** – добавляет к выводу команды номера изменившихся строк в файлах.

- Также флаги общие для всех команд (**--ssl, --user, --password, --host, --port, --grpc**)

В аргументах необходимо передать директории, в которые необходимо экспортировать разметку.

Пример команды:

```
svacer export_markup --host some_host.com --port 8080 --user admin --password password –snapshot_id 00000000-0000-0000-0000-000000000000 –show-lines /home/user/project/name /home/user/other_project/
```

Разметка экспортируется в виде комментария над строкой с маркером в таком же виде, как для импорта, указанного в предыдущем пункте (комментарии к маркеру во время экспорта не выгружаются). Таким образом экспортированная разметка может быть в дальнейшем снова импортирована из исходного кода, как в пункте выше.

5.8 Публичные запросы

Для работы с сервером через различные приложения реализованы публичные запросы, которые не будут меняться при обновлении самого сервера. Если будут происходить значимые изменения, то запрос будет помечаться как deprecated и создаваться новый такой же запрос с пометкой /api/some/request/v2.

Для каждого публичного запроса нужен регистрационный токен. Чтобы его получить нужно пройти аутентификацию.

Аутентификация – это POST запрос с basic auth на /api/login, который вернёт JWT-токен. Его нужно добавлять в заголовок (header) всех запросов (Authorization: Bearer <token>).

Публичные запросы:

- **/api/public/projects (GET)** – возвращает список всех проектов с бранчами;
- **/api/public/projects/{project_id}/branch/{branch_id}/snapshots (GET)** – возвращает список всех snapshot'ов конкретного бранча.

  Необходимые ID можно получить из предыдущего запроса.
- **/api/public/projects/{project_id}/branch/{branch_id}/snapshots/{snapshot_id}/fullmarkers (GET)** – возвращает полную информацию обо всех маркерах конкретного snapshot'а.

  У запроса есть опция **"?traces=true"** (по умолчанию false). Она добавит к каждому маркеру его трассу.
- **/api/public/diff?snapshot_v1=<id1>&snapshot_v2=<id2>&level=<number> (GET)** – выполняет сравнение snapshot'ов с указанными id.

  Если snapshot_v2 отсутствует или пустой, то сравнение будет проведено с предыдущим snapshot'ом. Количество информации определяется переменной **level**:
  - 0 – только общая информация со статистикой;
  - 1 – статистика и id маркеров;
  - >1 (default) – полная информация.

Всю эту информацию можно получить по запросу **`/api/public/help`** (без авторизации).

5.9 Функции, доступные только пользователям с ролью admin

5.9.1 Учетные записи и роли пользователей

В окне **Settings>Users** отображаются все существующие пользователи. В этом окне:

1. Чтобы добавить учётную запись (УЗ) пользователя:

   1) Нажмите кнопку **Add user**.
   2) Заполните поля появившейся формы (Рис. 25).
   3) Установите переключатель, если нужно сменить пароль при первой авторизации.

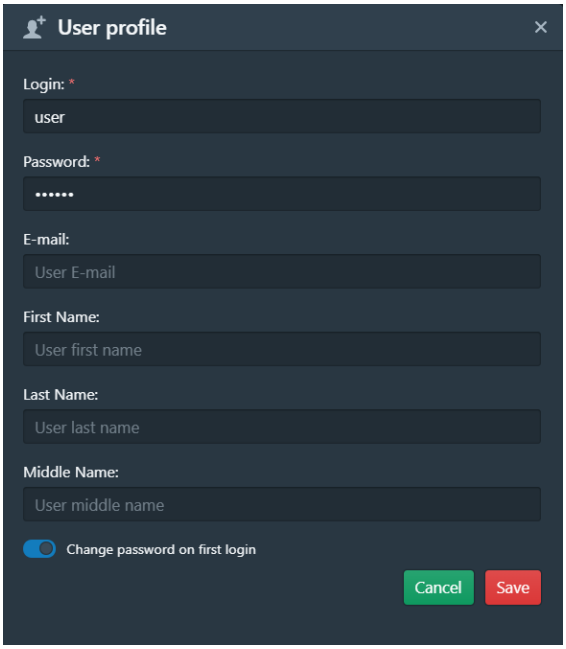

Рис. 25 – Окно создания учётной записи пользователя

2. Чтобы для существующей УЗ пользователя поменять пароль:

   1) Нажмите кнопку  в столбце **Action**.
   2) В появившейся форме (Рис. 26) введите пароль в оба поля и установите переключатель, если нужно, чтобы пользователь сменил пароль после авторизации.

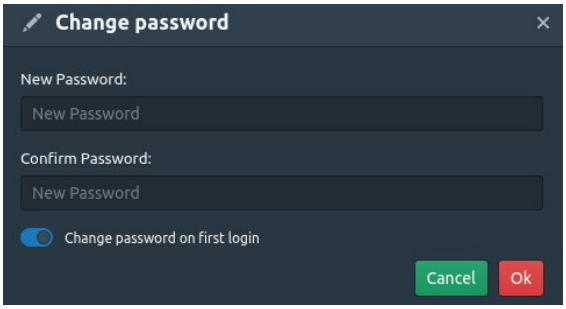

Рис. 26 – Окно смены пароля пользователя

3. Чтобы добавить роль:

   1) Нажмите кнопку **Add role**.
   2) Заполните поля появившейся формы (Рис. 27).
   3) Укажите проект, к которому относится роль (ALL, если ко всем проектам) и права, которые будут доступны пользователю.

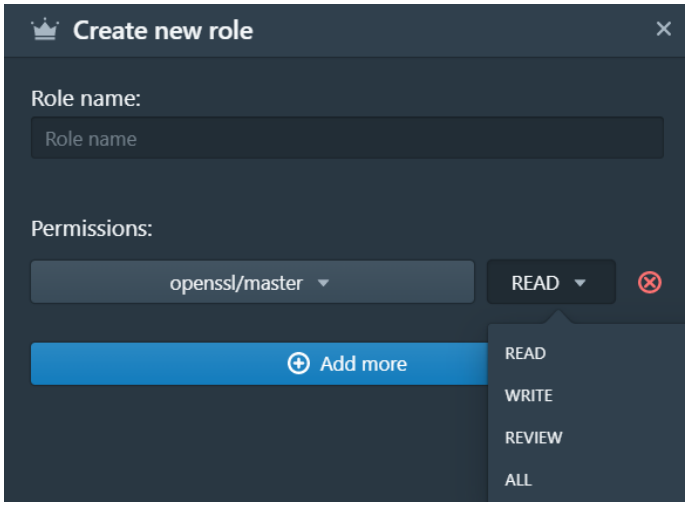

Рис. 27 – Настройка роли и прав доступа

5.9.2 Ведение списка проектов

В окне **Settings>Projects** отображаются все проекты с бранчами и snapshot'ами (Рис. 28). В этом окне доступны:

1. Добавление и удаление проекта (при импорте проект создается автоматически).
2. Клонирование бранча (если при импорте указать флаг `--branch <branch name>` ветка создается автоматически).

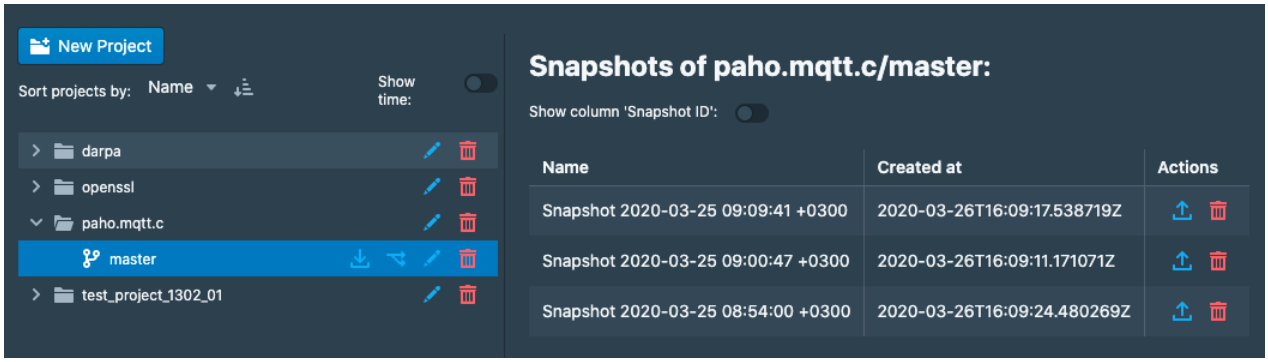

Рис. 28 – Список проектов

3. Экспорт snapshot с сервера в файл для дальнейшей загрузки его на другой сервер или в другой проект на этом же сервере (Рис. 29).

После нажатия на кнопку **Export snapshot** будет сгенерирован файл со snapshot'ом, который можно скачать, нажав соответствующую кнопку в следующем окне (Рис. 30).

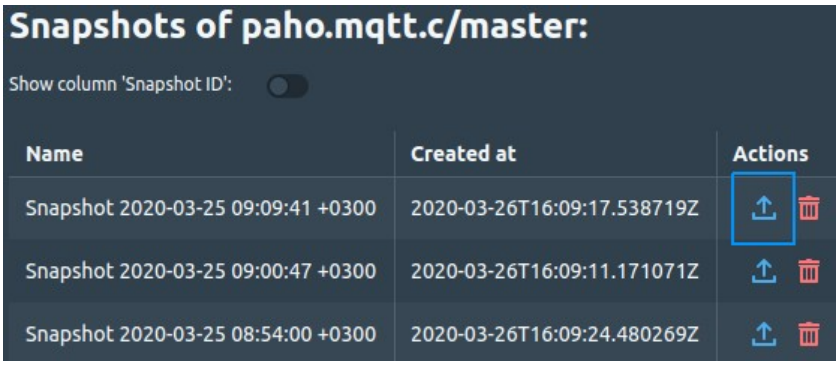

Рис. 29 – Экспорт snapshot'а

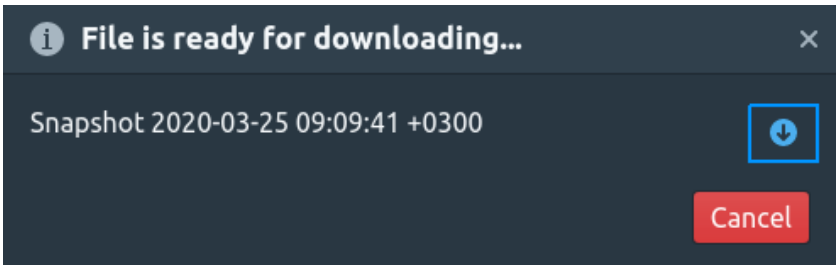

Рис. 30 – Окно скачивания файла со snapshot'ом

4. Импорт snapshot из файла. Для этого нажмите кнопку **Import snapshot** (Рис. 31) и в появившемся диалоге укажите файл для загрузки и имя нового snapshot'а.

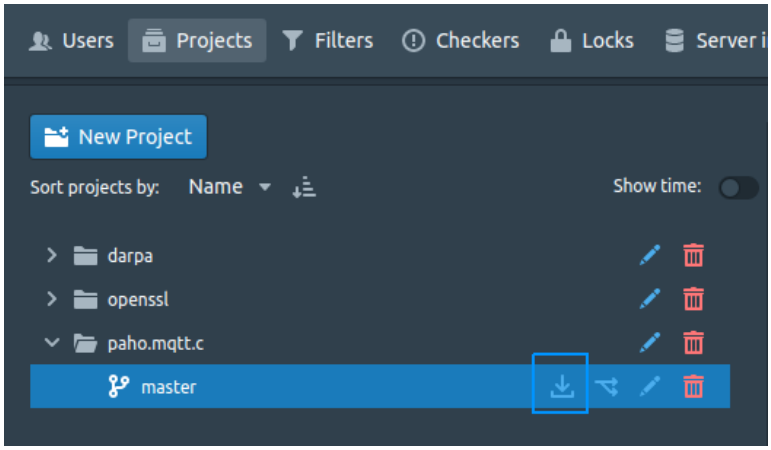

Рис. 31 – Импорт snapshot'а

5.9.3 Настройка фильтров бранчей и проектов

В окне **Settings>Filters** (Рис. 32) можно создать общие фильтры на бранч или весь проект. После применения фильтров в окне **Review** отображаются только маркеры, удовлетворяющие условиям.

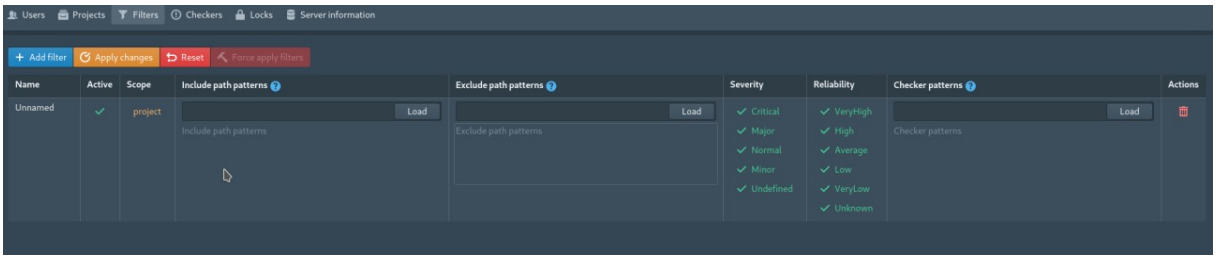

Рис. 32 – Окно фильтров бранчей и проектов

5.9.4 Просмотр информации о сервере

В окне **Settings>Server information** (Рис. 33) отображаются:

- параметры, с которыми запущен сервер;
- логи (если записываются в файл, а не выводятся в консоль);
- журнал базы данных.

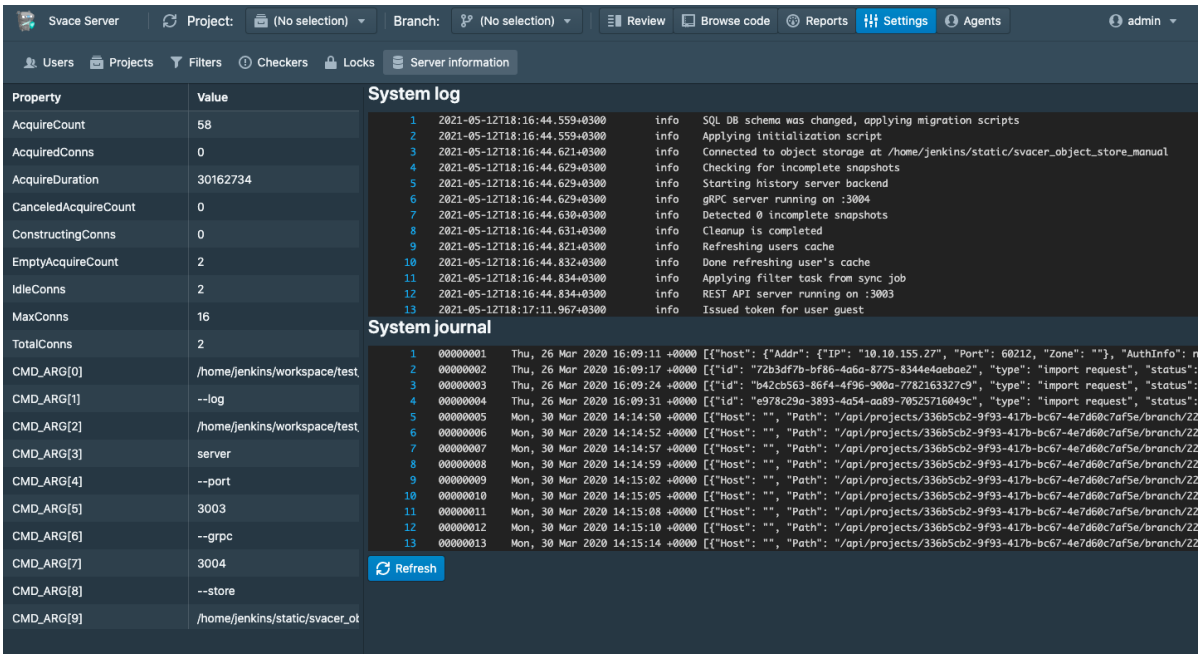

Рис. 33 – Информация о сервере