
FindFace Security

Выпуск 4.2.1

NtechLab

июн. 26, 2023

1	Руководство администратора	3
1.1	Архитектура	3
1.2	Системные требования	7
1.3	Принцип лицензирования	8
1.4	Развертывание FindFace Security	9
1.5	Первые шаги после установки	35
1.6	Работа с FindFace Security	38
1.7	Расширенный функционал	73
1.8	Обслуживание и устранение неисправностей	104
1.9	Приложения	128
2	Руководство оператора	175
2.1	Веб-интерфейс	175
2.2	Идентификация по базам данных	175
2.3	Идентификация лиц в режиме реального времени	178
2.4	Эпизоды событий	184
2.5	Работа с досье	187
2.6	Видеостена	190
2.7	Мобильный веб-интерфейс	191
3	Руководство по интеграции	197
3.1	HTTP API	197
3.2	Вебхуки	198
3.3	Интеграции с партнерами	209
3.4	Плагины	218
	Содержание модулей Python	233
	Алфавитный указатель	235

FindFace Security — это система биометрической видеоидентификации, предназначенная для автоматизации основной деятельности сотрудников служб безопасности и гостеприимства. В основе FindFace Security лежит [FindFace Enterprise Server](#), передовая технология распознавания лиц на базе искусственного интеллекта. FindFace Security представляет собой готовый к использованию продукт, который может использоваться в таких областях, как транспорт, розничная торговля, банковское обслуживание, индустрия развлечений, спортивные мероприятия, организация мероприятий, сервисы знакомств, видеонаблюдение, общественная и корпоративная безопасность и др.

FindFace Security обнаруживает и идентифицирует человеческие лица и уведомляет ответственных лиц об их появлении. FindFace Security также распознает такие атрибуты лица, как пол, возраст, эмоции, очки, наличие медицинской маски, борода и др., и отображает данную информацию для каждого обнаруженного на видеоизображении лица.

Интегрированная в FindFace Security антиспуфинговая система гарантирует, что перед камерой находится живой человек, и исключает возможность мошенничества с использованием фотографии лица на бумаге или экране мобильного устройства.

Раннее распознавание прихода нежелательных персон и VIP-гостей позволяет решать следующие задачи:

- снижение операционных потерь от мошеннических действий;
- снижение репутационных потерь и предотвращение конфликтных ситуаций;
- повышение качества обслуживания клиентов, в частности VIP-гостей;
- предотвращение потенциально опасных ситуаций, угрожающих жизни и здоровью людей.

FindFace Security поддерживает интеграцию сторонних решений через [HTTP API](#), [вебхуки](#) и [плагины](#), так что вы с легкостью сможете усовершенствовать свою текущую систему или приложение, добавив в них функционал распознавания лиц.

Полный список возможностей:

- Архитектура на основе AI.
- Быстрая и надежная биометрическая идентификация в реальном времени по базам данных досье.
- Повышенная производительность и отказоустойчивость в высоконагруженных системах с большим количеством подключенных камер и клиентов.
- Поддержка как видеопотоков с камер, так и архивных файлов.
- Быстрое создание базы данных досье.
- Распределение базы данных досье между несколькими серверами с синхронизацией и репликацией.
- Настройка содержимого досье.
- Поддержка дедупликации событий и досье.
- Верификация лиц.
- AI-распознавание пола, возраста, эмоций, очков, бороды, медицинской маски и других атрибутов лица.
- AI-детектор живых лиц (Liveness).
- AI-распознавание персон.
- Статистика в реальном времени по лицам и силуэтам на видеоизображении.
- Видеонаблюдение.

- Поиск по базам данных.
- Расширенный набор поисковых фильтров.
- Расширенное управление пользователями.
- Аутентификация по паролю или сертификату.
- CPU- и GPU-ускорение на ваш выбор.
- Удобный консольный инсталлятор и дружелюбный интерфейс.
- Развертывание на одном или нескольких серверах.
- Возможность лицензирования в открытых и закрытых системах.
- Интеграция через HTTP API, вебхуки и плагины на python.
- Партнерские интеграции с популярными системами.
- Мобильное приложение.

Данное руководство будет для вас наиболее полезным, если вы:

- администратор FindFace Security,
- сотрудник службы безопасности,
- сотрудник службы гостеприимства,
- специалист, обслуживающий систему и ее комплекс технических средств,
- специалист по системной интеграции, который собирается интегрировать технологию распознавания лиц в свою систему.

1.1 Архитектура

Хотя взаимодействие с FindFace Security происходит в основном через веб-интерфейс, не забудьте уделить немного времени изучению архитектуры программного комплекса. Эти знания необходимы для развертывания, интеграции, обслуживания и устранения проблем при работе FindFace Security.

В этой главе:

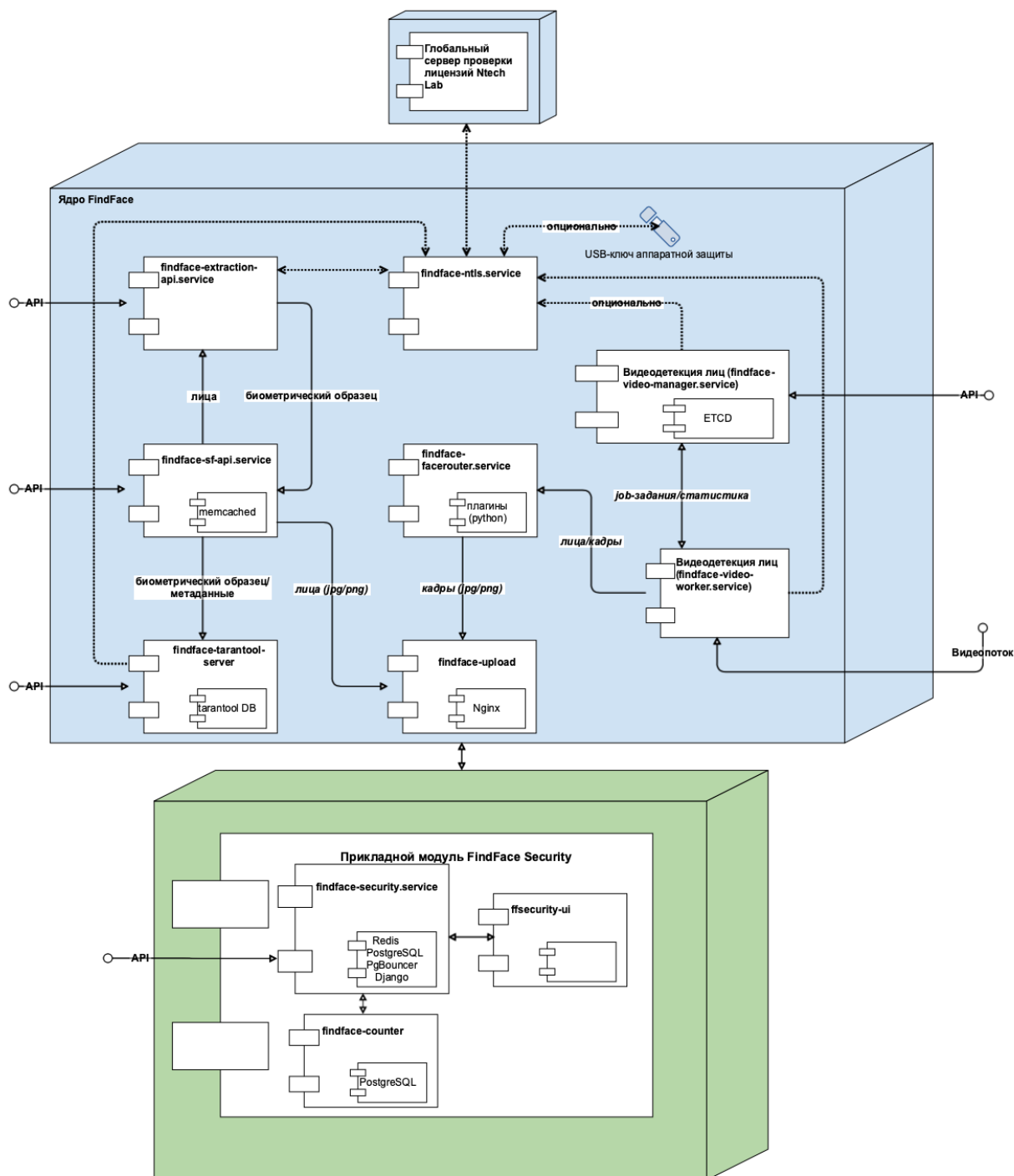
- *Архитектурные элементы*
 - *Схема архитектуры*
 - *Ядро FindFace*
 - *Прикладной модуль FindFace Security*
- *Развертывание на одиночном сервере или в кластере*
- *Аппаратное ускорение на CPU и GPU*

1.1.1 Архитектурные элементы

FindFace Security состоит из следующих основных архитектурных элементов:

- Ядро FindFace, передовая технология распознавания лиц на базе искусственного интеллекта, которая может использоваться в качестве отдельного продукта [FindFace Enterprise Server](#).
- FindFace Security, готовый прикладной модуль к FindFace Enterprise Server.

Схема архитектуры



Ядро FindFace

Ядро FindFace включает в себя следующие компоненты:

Ком- по- нент	Описание	По- став- щик
findface-extraction-api	Сервис, использующийся для обнаружения лиц на статических изображениях и извлечения из лиц биометрических образцов (векторов признаков). Сервис также выполняет распознавание таких атрибутов лица, как пол, возраст, эмоции, борода, очки, медицинская маска и др., а также распознавание человеческих силуэтов (если были заданы соответствующие настройки). Работа на базе нейронных сетей. Поддерживается CPU- или GPU-ускорение.	Собственная разработка NtechLab
findface-sf-api	Сервис, реализующий HTTP API обнаружения и распознавания лиц.	
findface-tarantool-server	Сервис, обеспечивающий взаимодействие между сервисом findface-sf-api и биометрической базой данных (базой, в которой хранятся биометрические образцы) на основе Tarantool.	
findface-upload	Веб-сервер на базе NginX, используемый как хранилище исходных изображений, миниатюр лиц и нормализованных изображений лиц.	
findface-facerouter	Сервис, который используется для задания правил обработки обнаруженных видео лиц. В FindFace Security функции findface-facerouter выполняются findface-security (см. <i>Прикладной модуль FindFace Security</i>). Однако в целях интеграции вы можете установить и настроить данный компонент (см. <i>Плагины</i>).	
findface-video-manager	Сервис, являющийся частью модуля видеодетекции лиц, через который осуществляется управление детекцией лиц на видео, а именно задаются настройки и список видеопотоков для обработки.	
findface-video-worker	Сервис, часть модуля видеодетекции лиц, который распознает лица на видео и отправляет их нормализованные изображения, соответствующие видеокadres и метаданные (такие как ID камеры и время обнаружения лица) в сервис findface-facerouter для дальнейшей обработки в соответствии с заданными правилами. Поддерживается CPU- и GPU-ускорение.	
findface-ntls	Локальный сервер лицензий, который проверяет подлинность <i>лицензии</i> FindFace Security, взаимодействуя с глобальным сервером лицензий NtechLab. Для закрытых систем поддерживается работа с аппаратными лицензионными ключами. Поддерживается лицензирование через прокси-сервер.	
Tarantool	Стороннее программное обеспечение, на основе которого реализована биометрическая база данных, хранящая извлеченные биометрические образцы (векторы признаков) и события обнаружения лиц. Системные данные, досье, пользовательские аккаунты и настройки камер хранятся в PostgreSQL (часть прикладного модуля FindFace Security).	Tarantool
etcd	Стороннее программное обеспечение, реализующее распределенное хранилище ключей для компонента findface-video-manager . Используется в качестве координационной службы в распределенной системе, обеспечивая отказоустойчивость видеодетектора лиц.	etcd
NginX	Стороннее программное обеспечение, которое реализует веб-интерфейсы системы.	nginx
memcached	Стороннее программное обеспечение, реализующее сервис кэширования данных в оперативной памяти на основе хеш-таблицы. Используется для временного хранения извлеченных биометрических образцов перед их записью в базу данных Tarantool.	memcached

Прикладной модуль FindFace Security

Прикладной модуль FindFace Security включает в себя следующие компоненты:

Ком- по- нент	Описание	По- став- щик
findface- security	Компонент, обеспечивающий доступ конечного пользователя к функциям ядра FindFace. Обеспечивает взаимодействие между ядром FindFace и веб-интерфейсом, функционирование системы в целом, HTTP и веб-сокет, обновление биометрической базы данных, отправку уведомлений о событиях, объединение событий в эпизоды, работу вебхуков и счетчики. Включает в себя следующие внутренние сервисы: проверка лицензии, менеджер вебхуков, группировщик персон, менеджер эпизодов, менеджер счетчиков.	Собственная раз- ра- ботка NtechLab
ffsecurity- ui	Основной веб-интерфейс, использующийся для взаимодействия с FindFace Security. Позволяет работать с событиями идентификации лиц, искать лица в базах данных, управлять камерами, пользователями, досье и списками наблюдения, собирать статистику в реальном времени и многое другое.	
findface- counter	Сервис, используемый для дедупликации лиц.	
PostgreSQL	Базовое программное обеспечение, реализующее основную базу данных системы. В данной базе хранятся детализированные досье персон с разбиением по категориям (спискам наблюдения), а также данные внутреннего характера, такие как профили пользователей FindFace Security, настройки видеокамер и пр. Биометрические данные лиц и события идентификации лиц хранятся в Tarantool (часть ядра FindFace).	PostgreSQL
Pgbouncer	Базовое программное обеспечение, обеспечивающее облегченный пул соединений для PostgreSQL. Необязательный компонент, используется для увеличения производительности базы данных при высокой нагрузке.	PgBouncer
Redis	Стороннее программное обеспечение, которое реализует брокер сообщений внутри findface-security.	Redis
Django	Стороннее программное обеспечение, реализующее веб-фреймворк для веб-интерфейса FindFace Security.	Django

См. также:

Подробнее о компонентах

1.1.2 Развертывание на одиночном сервере или в кластере

Вы можете развернуть FindFace Security как на одиночном сервере, так и в кластерной среде. При выборе последнего варианта доступны следующие схемы развертывания:

- Центральный сервер FindFace Security, взаимодействующий с несколькими дополнительными серверами для обработки видео (с одним установленным компонентом **findface-video-worker**).
- Полностью распределенная архитектура FindFace Security. Может потребоваться балансировка нагрузки.

Подробнее см. раздел *Типичная установка в кластере*.

1.1.3 Аппаратное ускорение на CPU и GPU

Сервисы **findface-extraction-api** и **findface-video-worker** могут использовать как CPU-, так и GPU-ускорение. Нужный тип ускорения выбирается во время установки из консольного *инсталлятора*.

Если установка FindFace Security выполняется из *apt-пенозитория*, на CPU-сервере нужно развернуть пакеты **findface-extraction-api** и/или **findface-video-worker-cpu**, а на GPU-сервере пакеты **findface-extraction-api-gpu** и/или **findface-video-worker-gpu**.

Важно: Для выбора конфигурации оборудования см. *Системные требования*.

Важно: Если разрешение используемой камеры превышает 1280x720 пикселей, настоятельно рекомендуется использовать пакет с ускорением на GPU `findface-video-worker-gpu`.

Примечание: *Liveness-детектор* на CPU работает гораздо медленнее, чем на GPU.

1.2 Системные требования

Для расчета характеристик серверов FindFace Security используйте приведенные ниже системные требования.

Совет: Сначала обязательно ознакомьтесь с *архитектурой* FindFace Security.

В этой главе:

- *Базовая конфигурация*

1.2.1 Базовая конфигурация

Важно: Если разрешение используемой камеры превышает 1280x720 пикселей, настоятельно рекомендуется использовать пакет с ускорением на GPU `findface-video-worker-gpu`.

	Минимальная	Рекомендуемая
CPU	Intel Core i5 CPU с 4-мя физическими ядрами 2.8 ГГц	Intel Xeon E5v3 с 6-ю физическими ядрами, лучший или похожий процессор
	На собственные нужды FindFace Security требуется 2 ядра HT > 2.5 ГГц. Характеристики также зависят от количества камер. Для одной камеры 720p@25FPS требуется 2 ядра >2.5 ГГц. Поддержка AVX2	
GPU (опционально)	Nvidia Geforce® GTX 980 4 Гб	Nvidia Geforce® GTX 1080+ с 8+ Гб RAM
	Поддерживаемые серии: GeForce (Maxwell, Pascal, Turing и выше), Tesla (Maxwell, Pascal, Volta v100, Turing и выше)	
RAM	10 Гб	16+ Гб
	На собственные нужды FindFace Security требуется 8 Гб. Потребление памяти также зависит от количества камер. Для одной камеры 720p@25FPS требуется 2 Гб RAM	
HDD	16 Гб	16+ Гб
	На собственные нужды операционной системы и FindFace Security требуется 15 Гб. Суммарный объем определяется в зависимости от требуемой глубины архива событий в базе данных и в логе из расчета 1.5 Мб на 1 событие	
Операционная система	Ubuntu 16.04, 18.04, только x64	

Совет: Для более точного подбора конфигурации свяжитесь с нашими техническими экспертами по адресу support@ntechlab.com.

1.3 Принцип лицензирования

FindFace Security лицензируется по следующим критериям:

1. В ходе работы FindFace Security биометрические образцы извлекаются из лиц, обнаруженных на видео, и из фотографий, загружаемых в досье, а также при построении т. н. центроидов в процессе *кластеризации персон*.

В целом схема лицензирования выглядит следующим образом:

- События: 1 событие распознавания лица на видео = 1 лицо в лицензии.
 - Досье: 1 фотография в досье = 1 лицо в лицензии.
 - Персоны: 1 персона = 1 лицо в лицензии.
2. Количество камер в системе.
 3. Количество экземпляров моделей, используемых `findface-extraction-api`.
 4. Распознавание атрибутов лица: пол/возраст/эмоции/очки/борода/медицинская маска.
 5. Распознавание живых лиц в реальном времени (Liveness).
 6. Интеграции с партнерами.

Вы можете выбрать между онлайн-лицензированием и лицензированием в закрытой сети:

- Онлайн-лицензирование выполняется через Глобальный менеджер лицензий NtechLab `license. ntechlab.com` и требует наличия стабильного интернет-соединения и DNS. После отключения от

интернета система продолжит работать в автономном режиме в течение 4-х часов. Уведомите своего менеджера, если требуется продлить данный период (до 2-х дней).

- Для лицензирования в закрытой сети необходимо наличие USB-порта на физическом сервере с компонентом `findface-ntls` (сервер лицензирования в составе *ядра FindFace*) для подключения предоставляемого USB-ключа аппаратной защиты.

Для обеспечения функционирования системы достаточно одного экземпляра `findface-ntls`. Если по какой-либо причине ваша система нуждается в большем количестве серверов лицензирования, заблаговременно сообщите об этом своему менеджеру NtechLab, чтобы предотвратить блокировку системы.

См.также:

Лицензирование

1.4 Развертывание FindFace Security

Для вашего удобства мы предлагаем несколько вариантов развертывания FindFace Security:

- Развертывание из консольного инсталлятора
- Пошаговое развертывание из apt-репозитория

Важно: Первый после развертывания запуск сервисов `findface-extract-api-gpu` и `findface-video-worker-gpu` может занять продолжительное время, до 45 минут, из-за процесса кэширования.

1.4.1 Развертывание из консольного инсталлятора

Для развертывания FindFace Security используется консольный инсталлятор.

Совет: Перед тем как приступить к развертыванию, обязательно ознакомьтесь с *системными требованиями*.

Важно: Для успешного функционирования системы после установки из инсталлятора IP-адрес сервера должен быть статическим. Для того чтобы сделать IP-адрес статическим, откройте файл `etc/network/interfaces` и измените текущую запись для основного сетевого интерфейса так, как показано в примере ниже. Не забудьте заменить адреса в примере на актуальные с учетом настроек сети.

```
sudo vi /etc/network/interfaces

iface eth0 inet static
address 192.168.112.144
netmask 255.255.255.0
gateway 192.168.112.254
dns-nameservers 192.168.112.254
```

Перезапустите сетевые интерфейсы.

```
sudo service networking restart
```

С осторожностью редактируйте файл `etc/network/interfaces`. Перед тем как приступить к редактированию, ознакомьтесь с [инструкцией по настройке сетей Ubuntu](#).

Для разворачивания FindFace Security из инсталлятора выполните следующие действия:

1. Загрузите файл инсталлятора `findface-security-and-server-4.2.1.run`.
2. Поместите файл `.run` в любой каталог на сервере установки (например, `/home/username`).
3. Из данного каталога сделайте файл `.run` исполняемым.

```
chmod +x findface-security-and-server-4.2.1.run
```

4. Запустите файл `.run`.

```
sudo ./findface-security-and-server-4.2.1.run
```

Инсталлятор задаст вам несколько вопросов, после чего проверит, соответствует ли сервер системным требованиям. Вопросы следующие:

1. Устанавливаемый продукт: FindFace Security.
2. Тип установки:
 - 1: установить FindFace Security на одиночном физическом сервере.
 - 2: установить FindFace Security в качестве центрального сервера и настроить его на взаимодействие с дополнительными удаленными серверами `findface-video-worker`.

Совет: Для отдельной установки `findface-video-worker` см. [Дополнительное разворачивание findface-video-worker на удаленных серверах](#).

- 3: установить только apt-репозиторий для [пошагового разворачивания](#) в будущем.

Важно: При данном типе установке модели нейронных сетей, необходимые для функционирования `findface-extraction-api`, не устанавливаются. Обязательно установите их [вручную](#) на серверах с `findface-extraction-api`.

- 4: полностью настраиваемая установка (установка нужных пакетов).

Важно: Обязательно [установите](#) модели нейронных сетей на серверах с `findface-extraction-api`.

3. Тип пакета `findface-video-worker`: CPU или GPU.
4. Тип пакета `findface-extraction-api`: CPU или GPU.

Ответы на вопросы будут сохранены в файл `/tmp/<findface-installer-*>.json`. Вы можете отредактировать его и использовать для установки FindFace Security на других серверах, не отвечая повторно на вопросы инсталлятора.

При выборе установки одиночного сервера FindFace Security, его компоненты будут автоматически установлены, настроены и запущены в соответствии со следующей конфигурацией:

Важно: В случае чистой установки инсталлятор автоматически настроит `findface-extraction-api` на использование нейронной сети `ifruit_320`. В противном случае

вам будет предложено сделать выбор между ifruit_320 и предыдущей моделью. Категорически не рекомендуется использовать инсталлятор для обновления системы. См. инструкции в разделе *Обновление FindFace Security до 4.2.*

Сервис	Конфигурация
postgresql-9.5 (Ubuntu 16.04) или postgresql-10 (Ubuntu 18.04)	Устанавливается и запускается.
redis-server	Устанавливается и запускается.
etcd	Устанавливается и запускается.
pgbouncer	Устанавливается и запускается.
memcached	Устанавливается и запускается.
nginx	Устанавливается и запускается.
django	Устанавливается и запускается как веб-фреймворк для веб-интерфейса FindFace Security.
findface-ntls	Устанавливается и запускается.
findface-tarantool-server	Устанавливается и запускается. Количество экземпляров (шардов) рассчитывается по формуле: $N = \max(\min(\text{mem_mb} // 2000, \text{cpu_cores}), 1)$, т.е. оно равно размеру оперативной памяти в Мб, разделенному на 2000, или количеству физических ядер процессора (но не менее 1 шарда).
findface-extraction-api	Устанавливается и запускается.
findface-sf-api	Устанавливается и запускается.
findface-upload	Устанавливается.
findface-video-manager	Устанавливается и запускается (CPU/GPU-ускорение).
findface-video-worker-*	Устанавливается и запускается.
findface-data-*	Модели нейронных сетей для распознавания лиц и их атрибутов (пол, возраст, эмоции, очки, борода, медицинская маска и др.). Устанавливаются.
findface-gpudetector-data/	Данные гри-детектора NTechLab. Устанавливается.
python3-ntech.ffsecurity-client	Библиотека клиента Python, используемая в API FindFace Security. Устанавливается.
findface-security	Устанавливается и запускается.
findface-counter	Устанавливается и запускается.
jq	Устанавливается. Используется для структурирования API-ответов от FindFace Security в формате JSON.

По завершении установки в консоль будет выведена информация, необходимая для использования FindFace Security:

Совет: Обязательно сохраните эти данные: они вам понадобятся.

```
#####
#                               Installation is complete                               #
                                                                                       (continues on next page)
```

(продолжение с предыдущей страницы)

```
#####
- upload your license to http://172.20.77.17/#/license/
- user interface: http://172.20.77.17/
  superuser:      admin
  password:       admin
  documentation:  http://172.20.77.17/doc/
```

5. Загрузите файл лицензии через основной веб-интерфейс `http://<IP_адрес_сервера>/#/license`. Для доступа в веб-интерфейс используйте логин и пароль администратора, выведенные в консоли.

Примечание: IP-адрес сервера в ссылках на веб-интерфейсы FindFace имеет вид 127.0.0.1 или <IP_адрес_в_сети>, в зависимости от того, принадлежит ли сервер к сети.

Важно: Не передавайте данные **superuser** (Супер Администратора) третьим лицам. Для администрирования системы создайте назначаемого администратора. Отличие назначаемого администратора от Супер Администратора в том, что последний не может лишиться прав администратора даже при смене роли.

6. Для того чтобы автоматически установить FindFace Security на других серверах, не отвечая на вопросы инсталлятора, используйте файл `/tmp/<findface-installer-*>.json`. Запустите инсталлятор следующей командой:

```
sudo ./findface-security-and-server-4.2.1.run -f /tmp/<findface-installer-*>.json
```

Совет: Пример данного файла можно посмотреть в разделе *Файл с параметрами установки*.

1.4.2 Пошаговое развертывание из apt-репозитория

Данный раздел содержит подробную информацию о пошаговом развертывании компонентов FindFace Security. Выполните приведенные ниже инструкции, придерживаясь заданного порядка.

В этом разделе:

- Установка apt-репозитория
- Установка необходимого стороннего ПО
- Обеспечение лицензирования
- Развертывание основной базы данных
- Развертывание ядра FindFace
- Развертывание прикладного модуля FindFace Security и биометрической базы данных

Установка apt-репозитория

Прежде всего установите apt-репозиторий FindFace следующим образом:

1. Загрузите файл инсталлятора `findface-security-and-server-4.2.1.run`.
2. Поместите файл `.run` в любой каталог на сервере установки (например, `/home/username`).
3. Из данного каталога сделайте файл `.run` исполняемым.

```
chmod +x findface-security-and-server-4.2.1.run
```

4. Запустите файл `.run`.

```
sudo ./findface-security-and-server-4.2.1.run
```

Инсталлятор задаст вам несколько вопросов, после чего проверит, соответствует ли сервер системным требованиям. Вопросы следующие:

1. Устанавливаемый продукт: **FindFace Security**.
2. Тип установки: `repo: Don't install anything, just set up the APT repository`.
3. Устанавливаемые модели нейронных сетей (при необходимости). Для того чтобы выбрать модели, сначала снимите выделение, введя в командной строке `-*`, затем введите через пробел порядковые номера нужных моделей, например: `1 3`. Введите `done` для сохранения выбора и перехода к следующему шагу.

Важно: Должна быть установлена как минимум одна модель для биометрии лица.

После этого apt-репозиторий FindFace будет автоматически установлен.

Установка необходимого стороннего ПО

Для работы FindFace Security необходимо стороннее программное обеспечение PostgreSQL, PgBouncer, Redis, etcd и memcached. Выполните следующие действия:

1. Установите пакеты с указанным сторонним ПО следующим образом:

На Ubuntu 16.04:

```
sudo apt update
sudo apt install -y postgresql-9.5 redis-server etcd memcached pgbouncer
```

На Ubuntu 18.04:

```
sudo apt update
sudo apt install -y postgresql-10 redis-server etcd memcached pgbouncer
```

2. Откройте файл конфигурации `memcached`. Установите максимальный размер памяти в мегабайтах, используемый для хранения элементов: `-m 512`. Установите максимальный размер элемента: `-I 16m`. Если один или оба этих параметра отсутствуют, добавьте их в файл.

```
sudo vi /etc/memcached.conf
```

```
-m 512  
-I 16m
```

3. Предоставьте надежный пароль пользователю ntech (9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3 в примере ниже). Внесите учетные данные в список пользователей pgbouncer.

```
echo "ntech" "9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3" | sudo tee -a /etc/pgbouncer/userlist.txt
```

4. Настройте pgbouncer. Откройте файл /etc/pgbouncer/pgbouncer.ini и добавьте базу данных ffsecurity в раздел databases. Настройте другие параметры, как показано в примере ниже.

```
sudo vi /etc/pgbouncer/pgbouncer.ini  
  
[databases]  
ffsecurity = dbname=ffsecurity host=localhost port=5432 user=ntech  
  
...  
auth_type = plain  
pool_mode = transaction  
max_client_conn = 16384  
syslog = 1  
listen_port = 5439
```

5. Добавьте сервисы стороннего ПО в автозагрузку Ubuntu и запустите их:

На Ubuntu 16.04:

```
sudo systemctl enable postgresql@9.5-main.service redis-server etcd.service memcached.service  
↳pgbouncer.service  
sudo systemctl start postgresql@9.5-main.service redis-server etcd.service memcached.service  
↳pgbouncer.service
```

На Ubuntu 18.04:

```
sudo systemctl enable postgresql@10-main.service redis-server etcd.service memcached.service  
↳pgbouncer.service  
sudo systemctl start postgresql@10-main.service redis-server etcd.service memcached.service  
↳pgbouncer.service
```

Обеспечение лицензирования

См.также:

Принцип лицензирования

Вы получаете файл лицензии вместе с установочными пакетами FindFace Security. Для лицензирования в закрытой сети вам также будет предоставлен ключ аппаратной защиты.

Лицензирование FindFace Security обеспечивается следующим образом:

1. Разверните findface-ntls, сервер лицензий в составе ядра FindFace.

Важно: Система на базе FindFace Security может включать в себя только один экземпляр `findface-ntls`.

Совет: В файле конфигурации `findface-ntls` вы можете изменить папку для хранения файла лицензии и настроить удаленный доступ к веб-интерфейсу `findface-ntls`, используемому для управления лицензией. Подробнее см. [findface-ntls](#).

```
sudo apt update
sudo apt install -y findface-ntls
sudo systemctl enable findface-ntls.service && sudo systemctl start findface-ntls.service
```

2. Загрузите файл лицензии через веб-интерфейс `findface-ntls` одним из следующих способов:

- Откройте веб-интерфейс `findface-ntls`: `http://<NTLS_IP_address>:3185/#/`. Загрузите файл лицензии.

Совет: Впоследствии используйте основной веб-интерфейс FindFace Security, чтобы посмотреть информацию о лицензиях, обновить или продлить лицензию (*Настройки -> Лицензия*).

- Непосредственно положите файл лицензии в предназначенную для этого папку (по умолчанию, `/opt/ntech/license`, может быть изменена в файле конфигурации `/etc/findface-ntls.cfg`).

3. При лицензировании в закрытой системе вставьте USB-ключ аппаратной защиты в USB-порт.

4. Если лицензируемые компоненты установлены на удаленных серверах, впоследствии укажите IP-адрес сервера `findface-ntls` в их файлах конфигурации. Подробнее см. [findface-extraction-api](#), [findface-tarantool-server](#), [Видеодетекция лиц: findface-video-manager и findface-video-worker](#).

См.также:

[Просмотр и обновление лицензии](#)

Развертывание основной базы данных

Основная база данных FindFace Security построена на PostgreSQL. Для того чтобы развернуть основную базу данных, выполните следующие действия:

1. Откройте список пользователей `pgbouncer` `/etc/pgbouncer/userlist.txt`. Скопируйте пароль пользователя `ntech` (`9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3` в примере ниже).

```
sudo cat /etc/pgbouncer/userlist.txt

"ntech" "9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3"
```

2. В консоли PostgreSQL создайте пользователя `ntech` под скопированным паролем, а также базы данных `ffsecurity` и `ffcounter` PostgreSQL.

```
sudo -u postgres psql

postgres=# CREATE ROLE ntech WITH LOGIN PASSWORD '9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3';
```

(continues on next page)

(продолжение с предыдущей страницы)

```
postgres=# CREATE DATABASE ffsecurity WITH OWNER ntech ENCODING 'UTF-8' LC_COLLATE='en_US.UTF-8' LC_CTYPE='en_US.UTF-8' TEMPLATE template0;

postgres=# CREATE DATABASE ffcounter WITH OWNER ntech ENCODING 'UTF-8' LC_COLLATE='C.UTF-8' LC_CTYPE='C.UTF-8' TEMPLATE template0;
```

Совет: Для выхода из консоли PostgreSQL введите \q и нажмите **Enter**.

3. Разрешите авторизацию в PostgreSQL по UID клиента сокета. Перезапустите PostgreSQL.

На Ubuntu 16.04:

```
echo 'local all ntech peer' | sudo tee -a /etc/postgresql/9.5/main/pg_hba.conf

sudo systemctl restart postgresql@9.5-main.service
```

На Ubuntu 18.04:

```
echo 'local all ntech peer' | sudo tee -a /etc/postgresql/10/main/pg_hba.conf

sudo systemctl restart postgresql@10-main.service
```

Развертывание ядра FindFace

Для развертывания ядра FindFace выполните следующие действия:

Совет: Вы можете найти описание компонентов ядра FindFace и их параметров конфигурации в разделах *Архитектура* и *Подробно о компонентах*.

1. Установите компоненты ядра FindFace:

```
sudo apt update
sudo apt install -y findface-tarantool-server findface-extraction-api findface-sf-api
findface-upload findface-video-manager findface-video-worker-cpu
```

Примечание: Для того чтобы установить компонент `findface-extraction-api` с GPU-ускорением, вместо `findface-extraction-api` в команде введите `findface-extraction-api-gpu`.

Примечание: Для того чтобы установить компонент `findface-video-worker` с GPU-ускорением, вместо `findface-video-worker-cpu` в команде введите `findface-video-worker-gpu`. Если на физическом сервере установлено несколько видеокарт, см. *Использование нескольких видеокарт*.

Важно: Обязательно *установите* модели нейронных сетей на серверах с findface-extraction-api.

2. В файле конфигурации findface-sf-api включите параметр allow-return-facen.

```
sudo vi /etc/findface-sf-api.ini

...
limits:
    ...
    allow-return-facen: true
...
```

3. В файле конфигурации findface-extraction-api включите модели распознавания атрибутов лица, таких как пол, возраст, эмоции, очки, борода и медицинская маска, в зависимости от ваших нужд. Удостоверьтесь, что для каждой модели вы указали правильный тип ускорения CPU или GPU: он должен совпадать с типом ускорения findface-extraction-api. Обратите внимание, что findface-extraction-api на CPU может работать только с CPU-моделями, в то время как findface-extraction-api на GPU поддерживает как CPU-, так и GPU-модели. Подробнее см. *Распознавание атрибутов лица*.

```
sudo vi /etc/findface-extraction-api.ini

models:
    age: faceattr/age.v1.cpu.fnk
    emotions: faceattr/emotions.v1.cpu.fnk
    face: face/ifruit_320.cpu.fnk
    gender: faceattr/gender.v2.cpu.fnk
    beard: faceattr/beard.v0.cpu.fnk
    glasses3: faceattr/glasses3.v0.cpu.fnk
    medmask3: faceattr/medmask3.v1.cpu.fnk
```

Доступны следующие модели:

Атрибут лица	Ускоре- ние	Параметр в файле конфигурации
биометрия лица	CPU	face: face/ifruit_320.cpu.fnk face: face/ifruit_160.cpu.fnk
	GPU	face: face/ifruit_320.gpu.fnk face: face/ifruit_160.gpu.fnk
возраст	CPU	age: faceattr/age.v1.cpu.fnk
	GPU	age: faceattr/age.v1.gpu.fnk
пол	CPU	gender: faceattr/gender.v2.cpu.fnk
	GPU	gender: faceattr/gender.v2.gpu.fnk
эмоции	CPU	emotions: faceattr/emotions.v1.cpu.fnk
	GPU	emotions: faceattr/emotions.v1.gpu.fnk
очки	CPU	glasses3: faceattr/glasses3.v0.cpu.fnk
	GPU	glasses3: faceattr/glasses3.v0.gpu.fnk
борода	CPU	beard: faceattr/beard.v0.cpu.fnk
	GPU	beard: faceattr/beard.v0.gpu.fnk
медицинская мас- ка	CPU	medmask3: faceattr/medmask3.v1.cpu.fnk
	GPU	medmask3: faceattr/medmask3.v1.gpu.fnk

Совет: Для того чтобы отключить модель распознавания, передайте в соответствующий параметр пустое значение. Не удаляйте сам параметр, поскольку в этом случае будет выполняться поиск модели по умолчанию.

```
models:
  gender: ""
  age: ""
  emotions: ""
```

- Откройте файл конфигурации `/etc/findface-video-worker-cpu.ini` (`/etc/findface-video-worker-gpu.ini`). В параметре `mgr-static` укажите IP-адрес сервера `findface-video-manager`, который будет обеспечивать `findface-video-worker` настройками и списком видеопотоков для обработки. В параметре `capacity` укажите максимальное количество видеопотоков, которое может быть обработано `findface-video-worker`. В разделе `streamer` укажите IP-адрес и порт для доступа к *видеостене*. Значение `port` должно быть 18999.

```
sudo vi /etc/findface-video-worker-cpu.ini
sudo vi /etc/findface-video-worker-gpu.ini

mgr-static=127.0.0.1:18811

capacity=10

[streamer]
#-----
## streamer/shots webserver port, 0=disabled
## type:number env:CFG_STREAMER_PORT longopt:--streamer-port
port = 18999
## streamer url - how to access this worker on streamer_port
## type:string env:CFG_STREAMER_URL longopt:--streamer-url
url = 127.0.0.1:18999
```

- Добавьте сервисы ядра FindFace в автозагрузку Ubuntu и запустите их.

```
sudo systemctl enable findface-extraction-api findface-sf-api findface-video-manager findface-
↪video-worker-cpu
sudo systemctl start findface-extraction-api findface-sf-api findface-video-manager findface-
↪video-worker-cpu
```

Развертывание прикладного модуля FindFace Security и биометрической базы данных

Для развертывания прикладного модуля FindFace Security, выполните следующие действия:

- Установите компоненты `findface-security`, `ffsecurity-ui` и `findface-counter`. Добавьте сервис `findface-counter` в автозагрузку Ubuntu и запустите его.

```
sudo apt update
sudo apt install -y ffsecurity ffsecurity-ui findface-counter
sudo systemctl enable findface-counter && sudo systemctl start findface-counter
```

- Перенесите схему базы данных из FindFace Security в PostgreSQL, создайте группы пользователей с *предустановленными* правами и первого пользователя с правами администратора (т. н. Супер Администратора).

Важно: Супер Администратор не может лишиться прав администратора даже при смене роли.

```
sudo findface-security migrate
sudo findface-security create_groups
sudo findface-security create_default_user
```

3. Создайте структуру биометрической базы данных на основе Tarantool.

```
sudo findface-security make_tnt_schema | sudo tee /etc/ffsecurity/tnt_schema.lua
```

4. Откройте файл конфигурации `/etc/tarantool/instances.enabled/FindFace.lua`. Проверьте, содержит ли файл команду `dofile`, а также определения `meta_indexes` и `meta_scheme`, как показано в примере ниже.

```
sudo vi /etc/tarantool/instances.enabled/FindFace.lua

dofile("/etc/ffsecurity/tnt_schema.lua")
-- host:port to bind, HTTP API
FindFace = require("FindFace")
FindFace.start("127.0.0.1", 8001, {
    license_ntls_server="127.0.0.1:3133",
    meta_indexes=meta_indexes,
    meta_scheme = meta_scheme
})
```

5. Добавьте сервис `findface-tarantool-server` в автозагрузку Ubuntu и запустите его.

```
sudo systemctl enable tarantool@FindFace.service && sudo systemctl start tarantool@FindFace.
↪service
```

6. Откройте файл конфигурации `/etc/ffsecurity/config.py`. Задайте следующие параметры:

- **SERVICE_EXTERNAL_ADDRESS:** IP-адрес или URL FindFace Security, являющимся приоритетным для Genetec и вебхуков. Если параметр не задан, система использует для работы с данным функционалом **EXTERNAL_ADDRESS**. Для использования Genetec и вебхуков обязательно укажите по крайней мере один из параметров **SERVICE_EXTERNAL_ADDRESS/EXTERNAL_ADDRESS**.
- **EXTERNAL_ADDRESS:** (Опционально) IP-адрес или URL, который используется для доступа к веб-интерфейсу FindFace Security. Если параметр не задан, система автоматически определяет его как внешний IP-адрес. Для доступа в FindFace Security вы можете использовать оба IP-адреса: как автоопределенный, так и указанный в **EXTERNAL_ADDRESS**.
- **VIDEO_DETECTOR_TOKEN:** придумайте токен и укажите его в данном параметре, чтобы авторизовать модуль видеодетекции лиц.
- **VIDEO_MANAGER_ADDRESS:** IP-адрес сервера `findface-video-manager`.
- **NTLS_HTTP_URL:** IP-адрес сервера `findface-ntls`.
- **ROUTER_URL:** IP-адрес сервера `findface-security`, который будет получать обнаруженные на видео лица от экземпляров `findface-video-worker`. Адрес указывается внутренний или внешний, в зависимости от сети, в которой `findface-video-worker` взаимодействует с `findface-security`. Измените порт по умолчанию с учетом *настроек перенадресации* с HTTP на HTTPS или вообще не указывайте его, оставив только IP-адрес.
- **SF_API_ADDRESS:** IP-адрес сервера `findface-sf-api`.

- DATABASES (раздел): заполните его следующим образом: 'PORT': 5439, 'USER': 'ntech', 'PASSWORD': '<пароль из /etc/pgbouncer/userlist.txt>' (см. [Установка необходимого стороннего ПО](#)).

Совет: Если необходимо обеспечить безопасность данных, включите *SSL-шифрование*.

Совет: При необходимости установите 'IGNORE_UNMATCHED': True, чтобы отключить запись события в базу данных, если обнаруженное лицо отсутствует в списках наблюдения (верификация дала отрицательный результат). Данную настройку рекомендуется использовать при большом количестве посетителей. Пороговая степень схожести при верификации лиц определяется параметром CONFIDENCE_THRESHOLD.

Совет: Рекомендуется отредактировать значение параметра MINIMUM_DOSSIER_QUALITY. Данный параметр определяет минимальное качество лица на фотографии в досье. Если качество лица хуже минимального, пользователь не сможет загрузить такую фотографию в досье. Прямые изображения лиц анфас считаются наиболее качественными. Им соответствуют значения вблизи 0, как правило, отрицательные (такие как -0.00067401276, например). Перевернутые лица и лица, повернутые под большими углами, характеризуются отрицательными значениями от -5 и меньше. По умолчанию 'MINIMUM_DOSSIER_QUALITY': -2, что соответствует среднему качеству.

Важно: Если вы включили модели распознавания в файле конфигурации findface-extraction-api, добавьте следующую строку в секцию FFSECURITY: 'EVENTS_FEATURES': ['gender', 'age', 'emotions', 'beard', 'glasses', 'medmask'], в зависимости от того, какие модели были включены. Данная строка должна быть расположена между строками SF_API_ADDRESS и LIVENESS_THRESHOLD, как показано в примере ниже. Подробнее см. [Распознавание атрибутов лица](#).

```
sudo vi /etc/ffsecurity/config.py

# =====
# FindFace Security configuration file
# =====
#
# This config file is written in Python's syntax and interpreted at FindFace Security
# service startup. You have to restart the service in order to apply changes.
#
# If you have any questions or suggestions, please contact us at support@ntechlab.com
# =====
# GENERAL SETTINGS
# =====
# enables additional logs
DEBUG = False
# media files directory
MEDIA_ROOT = "/var/lib/ffsecurity/uploads"
# static files directory
STATIC_ROOT = "/var/lib/ffsecurity/static"
# language code
LANGUAGE_CODE = 'en-us'
# time zone
```

(continues on next page)

(продолжение с предыдущей страницы)

```

TIME_ZONE = 'UTC'
# Database is used by FindFace Security to store cameras,
# camera groups, watchlists and so on. Only PostgreSQL is supported.
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'DISABLE_SERVER_SIDE_CURSORS': True,
        'NAME': 'ffsecurity',
        'PORT': 5439, 'USER': 'ntech', 'PASSWORD': '9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3'
    }
}
# Signature key for session encryption
# Use pwgen -sncy 50 1/tr "" ". " to generate your own unique key
SECRET_KEY = 'd1c22e3b368f2f8be4caa95b3540ce0c'
# =====
# FINDFACE SECURITY SETTINGS
# =====
# SERVICE_EXTERNAL_ADDRESS is prioritized for FFSecurity webhooks and Genetec plugin.
# EXTERNAL_ADDRESS is used instead if SERVICE_EXTERNAL_ADDRESS is not provided.
# You must provide either SERVICE_EXTERNAL_ADDRESS or EXTERNAL_ADDRESS in order
# to be able to work with FFSecurity webhooks and Genetec plugin.
SERVICE_EXTERNAL_ADDRESS = 'http://127.0.0.1'
# EXTERNAL_ADDRESS is used to access objects created inside FFSecurity via external links.
EXTERNAL_ADDRESS = ''
# - Base FFSecurity settings -
FFSECURITY = {
    # findface-video-worker authorization token
    'VIDEO_DETECTOR_TOKEN': 'fdc0fea20cf0a734c5e2de1886fc07b4',
    # base face matching confidence threshold
    'CONFIDENCE_THRESHOLD': 0.739,
    # episodes specific matching threshold that is used to join faces in an episode
    'EPISODES_THRESHOLD': 0.689,
    # minimum face quality sufficient to add it to a dossier
    'MINIMUM_DOSSIER_QUALITY': -2,
    # skip all unmatched faces
    'IGNORE_UNMATCHED': False,
    # matched events older than EVENTS_MAX_MATCHED_AGE will be automatically
    # deleted (every night at 1:17 am by default)
    'EVENTS_MAX_MATCHED_AGE': 30,
    # same as above but for unmatched events
    'EVENTS_MAX_UNMATCHED_AGE': 30,
    # same as EVENTS_MAX_MATCHED_AGE but for matched full frame images only (thumbnails won't
    ↳ be deleted)
    'EVENTS_MAX_FULLFRAME_UNMATCHED_AGE': 30,
    # same as above but for unmatched full frame images only (thumbnails won't be deleted)
    'EVENTS_MAX_FULLFRAME_MATCHED_AGE': 30,
    # NTLS licence server url
    'NTLS_HTTP_URL': 'http://127.0.0.1:3185',
    # findface-video-worker face posting address,
    # it must be set to either FFSecurity EXTERNAL_ADDRESS (by default)
    # or findface-facerouter url (in some specific cases)
    'ROUTER_URL': 'http://127.0.0.1',
    # send serialized dossiers, dossier-lists, camera and camera groups in webhooks
    'VERBOSE_WEBHOOKS': False,
    'THUMBNAIL_JPEG_QUALITY': 75,
    # FFServer services urls

```

(continues on next page)

(продолжение с предыдущей страницы)

```

'VIDEO_MANAGER_ADDRESS': 'http://127.0.0.1:18810',
'SF_API_ADDRESS': 'http://127.0.0.1:18411',
'FFCOUNTER_ADDRESS': 'http://127.0.0.1:17300',
# additional events features.
# make sure that corresponding extractors
# are licensed and enabled at findface-extraction-api config file.
# available features are: gender, age, emotions, beard, glasses, medmask.
'EVENTS_FEATURES': [],
# feature specific confidence thresholds
'LIVENESS_THRESHOLD': 0.75,
'EMOTIONS_THRESHOLD': 0.25,
'BEARD_THRESHOLD': 0.7,
# -- Persons configuration --
# rrule (recurrence rule) for scheduling persons clusterization
'PERSONS_CLUSTERIZATION_SCHEDULE': 'RRULE:FREQ=DAILY;INTERVAL=1;WKST=MO;BYHOUR=0;
↳BYMINUTE=0',
# face to person matching confidence threshold
'PERSONS_CONFIDENCE_THRESHOLD': 0.739,
# minimum face quality for handle face in person service
'PERSON_EVENT_MIN_QUALITY': -2,
# counters full frame saving options:
# `always` - save always
# `detect` - save only if faces or silhouettes have been detected
# `never` - never save full frames
'COUNTERS_SAVE_FULLFRAME': 'always',
'COUNTERS_FULLFRAME_JPEG_QUALITY': 75,
'COUNTERS_THUMBNAIL_JPEG_QUALITY': 75,
# -- Optional parameters --
# Edit CUSTOM_FIELDS section to customize dossier content.
# Below is an example for integration FindFace Security with Sigur.
# 'CUSTOM_FIELDS': {
#     'dossier_meta': {
#         'items': [
#             {
#                 'name': 'personid',
#                 'default': '',
#                 'label': 'PersonID',
#                 'display': ['list', 'form'],
#                 'description': 'Sigur person ID'
#             },
#             {
#                 'name': 'firstname',
#                 'default': '',
#                 'label': 'First Name',
#                 'display': ['list', 'form'],
#                 'description': 'Sigur first name'
#             },
#             {
#                 'name': 'lastname',
#                 'default': '',
#                 'label': 'Last Name',
#                 'display': ['list', 'form'],
#                 'description': 'Sigur last name'
#             },
#             {
#                 'name': 'version',

```

(continues on next page)

(продолжение с предыдущей страницы)

```

#         'default': '',
#         'label': 'Version',
#         'display': ['list', 'form'],
#         'description': 'Sigur photo version'
#     }
# ],
#     'filters': [
#         {
#             'name': 'personid',
#             'label': 'Sigur person ID filter',
#             'field': 'personid'
#         }
#     ]
# }
# },
# maximum event age in seconds than could be added to an episode.
# 'EPISODE_SEARCH_INTERVAL': 60,
# If none of these events matched, new episode is created.
# maximum episode duration (episode is closed after)
# 'EPISODE_MAX_DURATION': 300,
# if no new event added to an episode during this timeout, episode will be closed.
# 'EPISODE_EVENT_TIMEOUT': 30,
# maximum created thumbnail width
# 'THUMBNAIL_MAX_WIDTH': 320,
# social backend url. Contact support for additional information.
# 'SOCIAL_BACKEND': None,
# 'SOCIAL_HEADERS': {},
# unacknowledged events notification interval
# 'UNACKNOWLEDGED_NOTIFY_INTERVAL': 1,
}
# - FindFace Security user interface configuration dictionary -
FFSECURITY_UI_CONFIG = {
    "event": {
        "features": {
            "f_gender_class": ["male", "female"],
            "age": {
                "f_age_gte": "",
                "f_age_lte": ""
            },
        },
        "f_emotions_class": ["angry", "disgust", "fear", "happy", "sad", "surprise",
↪ "neutral"],
        "f_glasses_class": ["none", "eye", "sun"],
        "f_beard_class": ["none", "beard"],
        "f_liveness_class": ["real", "fake"],
        "f_medmask_class": ["none", "correct"],
    }
}
}
# -- ASGI-server configuration --
# consult support before changing these settings.
# per worker thread pool size.
ASGI_THREADS = 32
UVICORN_SETTINGS = {
    # worker processes count, 'auto' sets it to logical cpu count
    'workers': 'auto',
    'host': 'localhost',

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    'port': 8002,
    # websocket worker processes count,
    # 'auto' sets it to logical cpu count, but not more than 8.
    'ws-workers': 'auto',
    'ws-host': 'localhost',
    'ws-port': 8003,
}
# disable unused services to increase
# overall system performance in some cases.
SERVICES = {
    "ffsecurity": {
        "episodes": True,
        "webhooks": True,
        "persons": False,
    }
}
# =====
# FINDFACE SECURITY PLUGINS
# =====
# Uncomment lines below to enable plugins. Please consult documentation for
# a plugin specific settings.
# ===== Axxon =====
# INSTALLED_APPS.append('ffsecurity_axxon')
# FFSECURITY['AXXON'] = [
#     {
#         'name': 'server_name',
#         'api': 'http://example.com/',
#         'rtsp': 'rtsp://example.com:554/',
#         'user': 'user',
#         'password': 'password',
#     }
# ]
# FFSECURITY_UI_CONFIG['dossier'] = {
#     'video': True,
# }
# ===== Genetec =====
# INSTALLED_APPS.append('ffsecurity_genetec')
# ===== Sova =====
# INSTALLED_APPS.append('ffsecurity_sova')
# ===== CryptoPRO authentication =====
# INSTALLED_APPS.append('ffsecurity_cproauth')
# REST_FRAMEWORK['DEFAULT_AUTHENTICATION_CLASSES'] = [
#     'ffsecurity.auth.TokenAuthentication',
#     'ffsecurity_cproauth.auth.CryptoProOrTokenAuthentication'
# ]
# ===== DossierLists sync =====
# INSTALLED_APPS.append('ffsecurity_sync')
# token must be identical on master and slave
# use pwgen -s 64 1
# SYNC_TOKEN = 'change_me'
# SYNC_TIME = {
#     # 24 hour format
#     'hour': 3,
#     'minute': 0,
# }

```

7. Используя команду `pwgen -sncу 50 1|tr “” “.”`, сгенерируйте ключ подписи для шифрования

сессии (используется в Django) и задайте его в параметре SECRET_KEY.

8. Запустите сервисы.

```
sudo systemctl enable findface-security
sudo systemctl start findface-security
```

9. Отключите сервер nginx, активный по умолчанию, и добавьте сервер `findface-security` в список включенных серверов. Перезапустите nginx.

```
sudo rm /etc/nginx/sites-enabled/default

sudo ln -s /etc/nginx/sites-available/ffsecurity-nginx.conf /etc/nginx/sites-enabled/

sudo nginx -s reload
```

1.4.3 Дополнительное развертывание findface-video-worker на удаленных серверах

Для отдельной установки сервиса `findface-video-worker` выполните следующие действия:

Совет: Перед тем как приступить к развертыванию, обязательно ознакомьтесь с *системными требованиями*.

Совет: Если на сервере несколько видеокарт, перед развертыванием `findface-video-worker-gpu` изучите раздел *Использование нескольких видеокарт*.

1. Загрузите файл инсталлятора `findface-security-and-server-4.2.1.run`.
2. Поместите файл `.run` в любой каталог на сервере установки (например, `/home/username`).
3. Из данного каталога сделайте файл `.run` исполняемым.

```
chmod +x findface-security-and-server-4.2.1.run
```

4. Запустите файл `.run`.

```
sudo ./findface-security-and-server-4.2.1.run
```

Инсталлятор задаст вам несколько вопросов, после чего проверит, соответствует ли сервер системным требованиям. Вопросы следующие:

1. Устанавливаемый продукт: FindFace Video Worker.
2. Тип пакета `findface-video-worker`: CPU или GPU.
3. IP-адрес центрального сервера FindFace Security.

После этого процесс установки будет автоматически запущен.

Примечание: Ответы на вопросы будут сохранены в файл `/tmp/<findface-installer-*>.json`. Отредактируйте его и используйте для установки `findface-video-worker` на других серверах, не отвечая повторно на вопросы инсталлятора.

Примечание: Если `findface-ntls` и/или `findface-video-manager` будут установлены на серверах, отличных от сервера `findface-security`, укажите их IP-адреса в файле конфигурации `findface-video-worker` после установки компонента.

```
sudo vi /etc/findface-video-worker-cpu.ini
sudo vi /etc/findface-video-worker-gpu.ini
```

В параметре `ntls-addr` укажите IP-адрес сервера `findface-ntls`.

```
ntls-addr=127.0.0.1:3133
```

В параметре `mgr-static` укажите IP-адрес сервера `findface-video-manager`, который будет обеспечивать `findface-video-worker` настройками и списком видеопотоков для обработки.

```
mgr-static=127.0.0.1:18811
```

Совет: Для того чтобы автоматически установить `findface-video-worker` на других серверах, не отвечая на вопросы инсталлятора, используйте файл `/tmp/<findface-installer-*>.json`. Запустите инсталлятор следующей командой:

```
sudo ./findface-security-and-server-4.2.1.run -f /tmp/<findface-installer-*>.json
```

Пример файла `/tmp/<findface-installer-*>.json` можно посмотреть в разделе *Файл с параметрами установки*.

1.4.4 Установка моделей нейронных сетей

Для обнаружения и идентификации лиц и их атрибутов (пол, возраст, эмоции, борода, очки, медицинская маска и т. д.) `findface-extraction-api` использует нейронные сети.

Если необходим ручной запуск установки моделей, выполните следующие действия:

1. Запустите подготовленный файл `findface-security-and-server-4.2.1.run`.

```
sudo ./findface-security-and-server-4.2.1.run
```

2. Тип установки: `Fully customized installation`.
3. Выберите устанавливаемый компонент FindFace Security: `findface-data`. Для этого сначала снимите выделение со всех компонентов, введя в командной строке `-*`, затем введите порядковый номер компонента: `1`. Введите `done` для сохранения выбора и перехода к следующему шагу.
4. Выберите модели для установки. После этого процесс установки будет автоматически запущен.

Примечание: Вы можете найти установленные модели для распознавания лиц и атрибутов лиц в каталогах `/usr/share/findface-data/models/face/` и `/usr/share/findface-data/models/faceattr/` соответственно.

```
ls /usr/share/findface-data/models/face/
ifruit_160.cpu.fnk ifruit_320.cpu.fnk ifruit_160.gpu.fnk ifruit_320.gpu.fnk
```

(continues on next page)

(продолжение с предыдущей страницы)

```
ls /usr/share/findface-data/models/faceattr/
age.v1.cpu.fnk age.v1.gpu.fnk beard.v0.cpu.fnk beard.v0.gpu.fnk emotions.v1.cpu.fnk emotions.
v1.gpu.fnk gender.v2.cpu.fnk gender.v2.gpu.fnk glasses3.v0.cpu.fnk glasses3.v0.gpu.fnk,
medmask3.v1.cpu.fnk, medmask3.v1.gpu.fnk, liveness.v3.gpu.fnk
```

1.4.5 Полностью настраиваемая установка

Консольный инсталлятор FindFace Security предоставляет несколько вариантов установки, в том числе полностью настраиваемый вариант (установку отдельно выбранных пакетов). Данный вариант в основном используется при развертывании FindFace Security в сильно распределенной среде.

Для запуска полностью настраиваемой установки нужно ответить на вопросы инсталлятора следующим образом:

- Устанавливаемый продукт: FindFace Security.
- Тип установки: Fully customized installation.
- Устанавливаемые компоненты FindFace Security: для того чтобы выбрать нужные, сначала снимите выделение со всех компонентов, введя в командной строке `-*`, затем введите порядковые номера нужных компонентов через пробел, например: `1 7` (для выбора `findface-data` и `findface-extraction-api`), `13` (`findface-tarantool-server`) или `9` (`findface-upload`). Введите `done` для сохранения выбора и перехода к следующему шагу.
- Связанные вопросы, такие как как тип ускорения: CPU или GPU.

1.4.6 Типичная установка в кластере

Данный раздел посвящен развертыванию FindFace Security в кластерной среде.

Совет: Если после прочтения данного раздела у вас остались вопросы, не стесняйтесь задать их нашим экспертам по адресу support@ntechlab.com.

Развертывание FindFace Security в кластере может быть необходимо по следующим причинам:

- Нужно распределить высокую нагрузку при обработке видео.
- Требуется обработка видеопотоков от группы камер в месте их физического расположения.

Примечание: Актуально для сетей гостиниц, магазинов, при наличии нескольких проходных в одном здании и др.

См.также:

Привязка группы камер к экземпляру `findface-video-worker`

- Нужно распределить высокую нагрузку при извлечении биометрических образцов.
- В поиске задействовано большое количество лиц, что требует реализации распределенной базы данных.

Перед тем как приступить к разворачиванию, постройте архитектурную схему с учетом будущей нагрузки системы и выделенных под нее аппаратных ресурсов (см. *Системные требования*). Наиболее распространенной схемой является следующая:

- Центральный сервер с установленными компонентами `findface-ntls`, `findface-security`, `findface-sf-api`, `findface-video-manager`, `findface-upload`, `findface-video-worker`, `findface-extraction-api`, `findface-tarantool-server`, а также сторонним программным обеспечением.
- Несколько дополнительных серверов для обработки видео с установленным компонентом `findface-video-worker`.
- (При необходимости) Несколько дополнительных серверов для извлечения биометрических образцов с установленным компонентом `findface-extraction-api`.
- (При необходимости) Дополнительные серверы базы данных с несколькими шардами Tarantool на каждом.

Инструкции в настоящем разделе приведены для описанной выше наиболее часто встречающейся схемы разворачивания в кластере. В высоконагруженных системах также может потребоваться распределить обработку API-запросов, т. е. организовать несколько серверов `findface-sf-api` и `findface-video-manager`. В этом случае руководствуйтесь инструкциями в разделе *Полностью настраиваемая установка*.

Разворачивание FindFace в кластерной среде состоит из следующих этапов:

- *Разворачивание центрального сервера*
- *Разворачивание серверов обработки видео*
- *Разворачивание биометрических серверов*
- *Распределение нагрузки между биометрическими серверами*
- *Организация распределенной базы данных*
- *Настройка сетевого взаимодействия*

Разворачивание центрального сервера

Для разворачивания центрального сервера FindFace Security выполните следующие действия:

1. На выделенном физическом сервере *установите* FindFace Security из инсталлятора следующим образом:
 - Устанавливаемый продукт: **FindFace Security**.
 - Тип установки: **Single server, multiple video workers**. В этом случае FindFace Security будет установлен в качестве центрального сервера и настроен на взаимодействие с дополнительными удаленными экземплярами `findface-video-worker`.
 - Тип ускорения `findface-video-worker` (на центральном сервере): CPU или GPU, в зависимости от конфигурации оборудования.
 - Тип ускорения `findface-extraction-api` (на центральном сервере): CPU или GPU, в зависимости от конфигурации оборудования.

По завершении установки в консоль будет выведена информация, необходимая для использования FindFace Security:


```
#####
#                               Installation is complete                               #
#####
- upload your license to http://172.20.77.17/#/license/
- user interface: http://172.20.77.17/
  superuser:      admin
  password:       admin
  documentation:  http://172.20.77.17/doc/
```

- Загрузите файл лицензии через веб-интерфейс центрального сервера `http://<IP_адрес_сервера>/#/license`. Для доступа в веб-интерфейс используйте логин и пароль администратора, выведенные в консоли.

Примечание: IP-адрес сервера в ссылках на веб-интерфейсы FindFace имеет вид 127.0.0.1 или <IP_адрес_в_сети>, в зависимости от того, принадлежит ли сервер к сети.

Важно: Не передавайте данные `superuser` (Супер Администратора) третьим лицам. Для администрирования системы создайте назначаемого администратора. Отличие назначаемого администратора от Супер Администратора в том, что последний не может лишиться прав администратора даже при смене роли.

- Разрешите лицензируемым сервисам обращаться к серверу лицензирования `findface-ntls` с любого IP-адреса. Для этого, откройте файл конфигурации `/etc/findface-ntls.cfg` и установите `listen = 0.0.0.0:3133`.

```
sudo vi /etc/findface-ntls.cfg

# Listen address of NTLS server where services will connect to.
# The format is IP:PORT
# Use 0.0.0.0:PORT to listen on all interfaces
# This parameter is mandatory and may occur multiple times
# if you need to listen on several specific interfaces or ports.
listen = 0.0.0.1:3133
```

Развертывание серверов обработки видео

На дополнительном сервере для обработки видео установите экземпляр `findface-video-worker`, руководствуясь *пошаговыми инструкциями*. Ответьте на вопросы инсталлятора следующим образом:

- Устанавливаемый продукт: FindFace Video Worker.
- Тип ускорения `findface-video-worker`: CPU или GPU, в зависимости от конфигурации оборудования.
- FindFace Security IP address: IP-адрес центрального сервера.

После этого процесс установки будет автоматически запущен. Ответы на вопросы инсталлятора будут сохранены в файл `/tmp/<findface-installer-*>.json`. Используйте данный файл, чтобы установить FindFace Video Worker на других серверах, не отвечая на вопросы инсталлятора повторно. Для этого запустите инсталлятор командой:

```
sudo ./findface-security-and-server-4.2.1.run -f /tmp/<findface-installer-*>.json
```

Примечание: Если `findface-ntls` и/или `findface-video-manager` установлены на других серверах, чем `findface-security`, укажите их IP-адреса после установки в файле конфигурации `findface-video-worker`.

```
sudo vi /etc/findface-video-worker-cpu.ini
sudo vi /etc/findface-video-worker-gpu.ini
```

В параметре `ntls-addr` укажите IP-адрес сервера `findface-ntls`.

```
ntls-addr=127.0.0.1:3133
```

В параметре `mgr-static` укажите IP-адрес сервера `findface-video-manager`, который будет обеспечивать `findface-video-worker` настройками и списком видеопотоков для обработки.

```
mgr-static=127.0.0.1:18811
```

Развертывание биометрических серверов

На дополнительном сервере для извлечения биометрических образцов установите экземпляр `findface-extraction-api` из консольного инсталлятора. Ответьте на вопросы инсталлятора следующим образом:

- Устанавливаемый продукт: FindFace Security.
- Тип установки: Fully customized installation.
- Устанавливаемые компоненты FindFace Security: `findface-extraction-api` и `findface-data`. Для того чтобы их выбрать, сначала снимите выделение со всех компонентов, введя в командной строке `*`, затем введите порядковые номера `findface-extraction-api` и `findface-data` через пробел: `1 7`. Введите `done` для сохранения выбора и перехода к следующему шагу.
- Тип ускорения `findface-extraction-api`: CPU или GPU.
- Необходимость в изменении файла конфигурации `findface-extraction-api`: укажите IP-адрес сервера `findface-ntls`.
- Устанавливаемые модели нейронных сетей: CPU/GPU-модель для извлечения биометрических данных лица (обязательна для установки) и (опционально) CPU/GPU модели для распознавания пола, возраста, эмоций, очков, бороды и медицинской маски. Для того чтобы выбрать нужные модели, сначала снимите установленное по умолчанию выделение всех моделей, введя в командной строке `*`, затем введите порядковые номера нужных моделей через пробел, например: `8 2` для выбора соответственно GPU-модели для извлечения биометрических данных и GPU-модели для распознавания возраста. Введите `done` для сохранения выбора и перехода к следующему шагу. Удостоверьтесь, что для каждой модели выбран правильный тип ускорения CPU или GPU: он должен совпадать с типом ускорения `findface-extraction-api`. Обратите внимание, что `findface-extraction-api` на CPU может работать только с CPU-моделями, в то время как `findface-extraction-api` на GPU поддерживает как CPU-, так и GPU-модели. Подробнее см. *Распознавание атрибутов лица*.

Доступны следующие модели:

Атрибут лица	Ускоре- ние	Пакет
биометрия лица	CPU	findface-data-ifruit-320-cpu_3.0.0_all.deb findface-data-ifruit-160-cpu_3.0.0_all.deb
	GPU	findface-data-ifruit-320-gpu_3.0.0_all.deb findface-data-ifruit-160-gpu_3.0.0_all.deb
возраст	CPU	findface-data-age.v1-cpu_3.0.0_all.deb
	GPU	findface-data-age.v1-gpu_3.0.0_all.deb
пол	CPU	findface-data-gender.v2-cpu_3.0.0_all.deb
	GPU	findface-data-gender.v2-gpu_3.0.0_all.deb
эмоции	CPU	findface-data-emotions.v1-cpu_3.0.0_all.deb
	GPU	findface-data-emotions.v1-gpu_3.0.0_all.deb
очки	CPU	findface-data-glasses3.v0-cpu_3.0.0_all.deb
	GPU	findface-data-glasses3.v0-gpu_3.0.0_all.deb
борода	CPU	findface-data-beard.v0-cpu_3.0.0_all.deb
	GPU	findface-data-beard.v0-gpu_3.0.0_all.deb
медицинская маска	CPU	findface-data-medmask3.v1-cpu_3.0.0_all.deb
	GPU	findface-data-medmask3.v1-gpu_3.0.0_all.deb

После этого процесс установки будет автоматически запущен. Ответы на вопросы инсталлятора будут сохранены в файл `/tmp/<findface-installer-*>.json`. Используйте данный файл, чтобы установить `findface-extraction-api` на других серверах, не отвечая на вопросы инсталлятора повторно.

```
sudo ./findface-security-and-server-4.2.1.run -f /tmp/<findface-installer-*>.json
```

После развертывания биометрических серверов *распределите* между ними нагрузку.

Распределение нагрузки между биометрическими серверами

Распределение нагрузки между несколькими биометрическими серверами выполняется через балансировщик нагрузки. Приведенная ниже пошаговая инструкция демонстрирует балансировку нагрузки с помощью `nginx` в режиме `round-robin` для 3-х экземпляров `findface-extraction-api`, расположенных на различных физических серверах. Один экземпляр установлен на центральном сервере FindFace Security (172.168.1.9), 2 других на дополнительных удаленных серверах (172.168.1.10, 172.168.1.11). Если в системе присутствует большее количество биометрических серверов, балансировка нагрузки выполняется по аналогии.

Совет: Вы можете использовать любой удобный вам балансировщик нагрузки. Руководство по его использованию ищите в соответствующей справочной документации.

Для балансировки нагрузки между экземплярами `findface-extraction-api` выполните следующие действия:

1. Назначьте т. н. сервер шлюза для балансируемой группы биометрических серверов. Им может стать центральный сервер FindFace Security (рекомендуется) или любой другой сервер с установленным `nginx`.

Важно: Вам нужно будет указать IP-адрес шлюза при настройке *распределенной сети* FindFace Security.

Совет: Вы можете установить nginx следующим образом:

```
sudo apt update
sudo apt install nginx
```

2. На сервере шлюза создайте новый файл конфигурации Nginx.

```
sudo vi /etc/nginx/sites-available/extapi
```

3. Вставьте следующий текст в созданный файл конфигурации. В директиве `upstream` (`upstream extapibackends`) замените примерные IP-адреса на актуальные IP-адреса биометрических серверов. В директиве `server` в параметре `listen` укажите номер слушающего порта сервера шлюза. Вам потребуется указать данный порт при настройке *распределенной сети* FindFace Security.

```
upstream extapibackends {
    server 172.168.1.9:18666; ## ``findface-extraction-api`` on principal server
    server 172.168.1.10:18666; ## 1st additional extraction server
    server 127.168.1.11:18666; ## 2nd additional extraction server
}
server {
    listen 18667;
    server_name extapi;
    client_max_body_size 64m;
    location / {
        proxy_pass http://extapibackends;
        proxy_next_upstream error;
    }
    access_log /var/log/nginx/extapi.access_log;
    error_log /var/log/nginx/extapi.error_log;
}
```

4. Включите балансировщик нагрузки в nginx.

```
sudo ln -s /etc/nginx/sites-available/extapi /etc/nginx/sites-enabled/
```

5. Перезапустите nginx.

```
sudo service nginx restart
```

6. На центральном сервере и каждом из дополнительных биометрических серверов откройте файл конфигурации `/etc/findface-extraction-api.ini`. Замените адрес `localhost` в параметре `listen` на адрес, который вы указали до этого в директиве `upstream extapibackends` файла конфигурации Nginx `/etc/nginx/sites-available/extapi`. В нашем примере адрес 1-го дополнительного биометрического сервера должен быть заменен на следующий:

```
sudo vi /etc/findface-extraction-api.ini

listen: 172.168.1.10:18666
```

7. Перезапустите `findface-extraction-api` на центральном сервере и каждом дополнительном биометрическом сервере.

```
sudo systemctl restart findface-extraction-api.service
```

Балансировка нагрузки успешно настроена. Не забудьте указать актуальный IP-адрес и слушающий порт сервера шлюза при настройке *распределенной сети* FindFace Security.

Организация распределенной базы данных

Компонент `findface-tarantool-server` соединяет базу данных Tarantool и компонент `findface-sf-api`, передавая результаты поиска от базы данных в `findface-sf-api` для дальнейшей обработки. Для увеличения скорости поиска на каждом сервере с базой данных Tarantool могут быть созданы многочисленные сегменты (шарды) `findface-tarantool-server`. Их параллельное функционирование приводит к значительному увеличению производительности. Каждый шард может обрабатывать приблизительно до 10 000 000 лиц. При развертывании `findface-tarantool-server` из инсталлятора шарды создаются автоматически с учетом аппаратной конфигурации физического сервера.

Для того чтобы распределить биометрическую базу данных, установите `findface-tarantool-server` на каждом сервере базы данных. Ответьте на вопросы инсталлятора следующим образом:

- Устанавливаемый продукт: FindFace Security.
- Тип установки: Fully customized installation.
- Устанавливаемые компоненты FindFace Security: `findface-tarantool-server`. Для того чтобы его выбрать, сначала снимите выделение со всех компонентов, введя в командной строке `-*`, затем введите порядковый номер `findface-tarantool-server: 13`. Введите `done` для сохранения выбора и перехода к следующему шагу.

После этого процесс установки будет автоматически запущен. Ответы на вопросы инсталлятора будут сохранены в файл `/tmp/<findface-installer-*>.json`. Используйте данный файл, чтобы установить `findface-tarantool-server` на других серверах, не отвечая на вопросы инсталлятора повторно.

```
sudo ./findface-security-and-server-4.2.1.run -f /tmp/<findface-installer-*>.json
```

В результате установки шарды `findface-tarantool-server` будут автоматически установлены в количестве $N = \max(\min(\text{mem_mb} // 2000, \text{cpu_cores}), 1)$, т. е. равном размеру оперативной памяти в Мб, разделенному на 2000, или количеству физических ядер процессора (но не менее 1 шарда).

Обязательно укажите IP-адреса и порты шардов при настройке *распределенной сети* FindFace Security. Для того чтобы узнать номера портов, на каждом сервере базы данных выполните следующую команду:

```
sudo cat /etc/tarantool/instances.enabled/*shard* | grep -E ".start|(listen =)"
```

Будет возвращен следующий результат:

```
listen = '127.0.0.1:33001',
FindFace.start("127.0.0.1", 8101, {
  listen = '127.0.0.1:33002',
FindFace.start("127.0.0.1", 8102, {
```

Номера портов шардов указаны в секции `FindFace.start`: 8101, 8102 и т. д.

Настройка сетевого взаимодействия

После развертывания компонентов FindFace Security настройте их взаимодействие по сети. Выполните следующие действия:

1. Откройте файл конфигурации `/etc/findface-sf-api.ini`:

```
sudo vi /etc/findface-sf-api.ini
```

Задайте следующие параметры:

Параметр	Описание
extraction-api -> extraction-api	IP-адрес и слушающий порт сервера, являющегося <i>шлюзом</i> для биометрических серверов с настроенной балансировкой нагрузки.
storage-api -> shards -> master	IP-адрес и порт мастера шарда findface-tarantool-server . Остальные шарды прописываются по аналогии.
upload_url	Путь в WebDAV nginx, по которому в компонент findface-upload будут отправляться исходные изображения, миниатюры и нормализованные изображения лиц.

```
...
extraction-api:
  extraction-api: http://172.168.1.9:18667

...
webdav:
  upload-url: http://127.0.0.1:3333/uploads/

...
storage-api:
  ...
  shards:
    - master: http://172.168.1.9:8101/v2/
      slave: ''
    - master: http://172.168.1.9:8102/v2/
      slave: ''
    - master: http://172.168.1.12:8101/v2/
      slave: ''
    - master: http://172.168.1.12:8102/v2/
      slave: ''
    - master: http://172.168.1.13:8102/v2/
      slave: ''
    - master: http://172.168.1.13:8102/v2/
      slave: ''
```

- Откройте файл конфигурации `/etc/ffsecurity/config.py`.

```
sudo vi /etc/ffsecurity/config.py
```

Задайте следующие параметры:

Параметр	Описание
SERVICE_EXTERNAL_ADDRESS	Адрес FindFace Security, являющимся приоритетным для Genetec и вебхуков. Если параметр не задан, система использует для работы с данным функционалом EXTERNAL_ADDRESS. Для использования Genetec и вебхуков обязательно укажите по крайней мере один из параметров SERVICE_EXTERNAL_ADDRESS/ EXTERNAL_ADDRESS.
EXTERNAL_ADDRESS	(Опционально) IP-адрес или URL, который используется для доступа к веб-интерфейсу FindFace Security. Если параметр не задан, система автоматически определяет его как внешний IP-адрес. Для доступа в FindFace Security вы можете использовать оба IP-адреса: как автоопределенный, так и указанный в EXTERNAL_ADDRESS.
VIDEO_DETECTOR_TOKEN	Токен и укажите его в данном параметре, чтобы авторизовать модуль видеодетекции лиц.
VIDEO_MANAGER_ADDRESS	Адрес сервера findface-video-manager.
NTLS_HTTP_URL	Адрес сервера findface-ntls.
ROUTER_URL	Внешний IP-адрес сервера findface-security, который будет получать обнаруженные лица от экземпляра(ов) findface-video-worker.
SF_API_ADDRESS	Адрес сервера findface-sf-api.
EXTRACTION_API	Адрес и слушающий порт сервера, являющегося <i>шлюзом</i> для биометрических серверов с настроенной балансировкой нагрузки.

```

sudo vi /etc/ffsecurity/config.py

...
# SERVICE_EXTERNAL_ADDRESS prioritized for webhooks and genetec
SERVICE_EXTERNAL_ADDRESS = 'http://localhost'
EXTERNAL_ADDRESS = 'http://127.0.0.1'

...
FFSECURITY = {
    'VIDEO_DETECTOR_TOKEN': '7ce2679adfc4d74edcf508bea4d67208',
    ...
    'EXTRACTION_API': 'http://172.168.1.9:18667/',
    'VIDEO_MANAGER_ADDRESS': 'http://127.0.0.1:18810',
    ...
    'NTLS_HTTP_URL': 'http://127.0.0.1:3185',
    'ROUTER_URL': 'http://172.168.1.9',
    ...
    'SF_API_ADDRESS': 'http://127.0.0.1:18411',
    ...
}

```

На этом установка FindFace Security в кластерной среде будет завершена.

1.5 Первые шаги после установки

После успешного развертывания FindFace Security пришло время открыть *веб-интерфейс* и начать работу. В этой главе вы найдете рекомендуемую последовательность шагов, следуя которой вы сможете задействовать весь спектр функциональных возможностей FindFace Security.

В этой главе:

- *Добавление камер*
- *Составление списков наблюдения и загрузка досье*
- *Создание пользователей и предоставление им прав*
- *Мониторинг лиц*
- *Организация видеонаблюдения*
- *Сбор статистики*
- *Работа FindFace Security*
- *Базовое обслуживание системы*
- *Больше возможностей*

1.5.1 Добавление камер

1. *Создайте новую группу камер* или используйте созданную по умолчанию. Группа камер — это системный объект, который позволяет группировать камеры с учетом их физического расположения. К примеру, камеры, расположенные на входе в здание, можно объединить в одну группу и работать с ними как с единым целым.
2. *Добавьте камеры* в созданную группу и *проверьте их работу*.

Дополнительно:

1. Настройте систему на обработку видеопотоков с группы камер в месте их физического расположения. Это может быть актуально в распределенной архитектуре. *Подробнее.*
2. Рассмотрите возможность включения дедупликации событий, если сцены наблюдения камер в группе перекрываются. Данная функция позволяет исключить регистрацию одинаковых событий в пределах одной группы камер. *Подробнее.*

1.5.2 Составление списков наблюдения и загрузка досье

1. *Создайте новый список наблюдения* или используйте созданный по умолчанию. Список наблюдения — это системный объект, с помощью которого выполняется классификация людей по произвольным критериям: черный список, розыск, VIP, персонал и т. д.
2. Загрузите досье и добавьте их в созданный список наблюдения *вручную, пакетно через веб-интерфейс* или используя функцию *пакетной загрузки через консоль*.

Дополнительно:

1. *Распределите базу данных досье* между несколькими серверами. База данных досье будет доступна для редактирования на головном сервере, а для чтения и мониторинга — на ведомых.
2. *Настройте содержимое досье*. Создайте дополнительные поля, вкладки и поисковые фильтры.

1.5.3 Создание пользователей и предоставление им прав

1. Ознакомьтесь со списком *предустановленных пользовательских ролей* и при необходимости *создайте новые*.
2. *Добавьте пользователей* в систему и предоставьте им полномочия.

1.5.4 Мониторинг лиц

По умолчанию FindFace Security отслеживает только незарегистрированные в системе лица, т. е. лица, для которых *отсутствуют совпадения с досье*. Для того чтобы включить отслеживание лиц из пользовательского списка наблюдения, сделайте это список *активным*. Вы также можете включить звуковое оповещение и сделать обязательным принятие вручную связанных со списком событий.

Дополнительно:

1. Добавьте информации в события, включив распознавание пола, возраста, эмоций, бороды, медицинской маски и очков. *Подробнее*.
2. Защитите систему от спуфинга, включив функцию распознавания живых лиц (Liveness). *Подробнее*.

1.5.5 Организация видеонаблюдения

Настройте *раскладку камер* для базового видеонаблюдения.

1.5.6 Сбор статистики

Настройте *счетчики* для сбора статистики лиц и силуэтов в реальном времени.

1.5.7 Работа FindFace Security

1. *Идентифицируйте обнаруженные на видеоизображении лица*, проверяя их в реальном времени на совпадение с лицами из списков наблюдения. Работайте с историей событий, используя различные фильтры.
2. Работайте с *эпизодами*. Эпизод — это набор событий идентификации, в которых фигурируют лица одного и того же человека, обнаруженные в течение определенного периода времени. Поскольку информация о событиях отображается на вкладке *События* в произвольном порядке, обработка большого количества разнородных событий может быть делом затруднительным и неэффективным. С функцией Эпизоды, система использует искусственный интеллект для группировки входящих событий на основе времени обнаружения и схожести лиц. Это позволяет с легкостью обрабатывать разнородные события даже в больших количествах.
3. Выполняйте идентификацию (поиск) лиц по следующим базам данных:
 - База данных обнаруженных лиц. *Подробнее*.
 - База данных досье. *Подробнее*.
4. *Ищите в архивных видео* лица из списков наблюдения.
5. *Сравнивайте лица* вручную, чтобы проверить их на принадлежность одному человеку.

6. Используйте *мобильное приложение*.

Дополнительно:

Включите *распознавание персон*. В системных базах данных появится новая сущность «person event» (событие появления персоны), связанная со всеми *эпизодами*, в которых фигурировало лицо того или иного человека. Вы можете работать с галереей персон по аналогии со списками событий и эпизодов.

1.5.8 Базовое обслуживание системы

1. *Настройте* автоматическое удаление старых событий, эпизодов и полных кадров.
2. При необходимости *удалите* старые события, эпизоды и полные кадры вручную.
3. Регулярно создавайте *резервную копию* базы данных.

1.5.9 Больше возможностей

1. Настройте *вебхуки* для автоматической отправки уведомлений об определенных событиях, эпизодах и записях счетчика на заданный URL-адрес. При наступлении нужного события FindFace Security отправит HTTP-запрос на URL-адрес, указанный в настройках вебхука. Вебхуки можно использовать для решения разнообразных задач, например, для уведомления пользователя об определенном событии, вызова определенных действий на целевом веб-сайте, при решении задач безопасности, таких как удаленное автоматическое управление доступом и др. *Подробнее*.
2. Задействуйте функции FindFace Security через *HTTP API*.
3. Ознакомьтесь со списком наших *партнерских интеграций*.
4. Задействуйте *плагины*, что задать собственные правила обработки обнаруженных на видео лиц.

См.также:

- *Управление видеокамерами*
- *Мониторинг лиц. Управление базой данных досъе*
- *Управление пользователями*
- *Идентификация лиц в офлайн видео*
- *Счетчики лиц и силуэтов*
- *Расширенный функционал*
- *Обслуживание и устранение неисправностей*

1.6 Работа с FindFace Security

Работа с FindFace Security выполняется через веб-интерфейс. Для того чтобы отобразить веб-интерфейс, в адресной строке браузера введите базовый адрес веб-интерфейса и пройдите авторизацию.

Примечание: Базовый адрес задается при *установке* FindFace Security.

Важно: Для первого входа в систему после развертывания FindFace Security используйте учетную запись администратора, созданную при *установке*. Для создания других пользователей см. раздел *Управление пользователями*.

Веб-интерфейс имеет удобный и интуитивный дизайн и обеспечивает доступ к следующим функциям:

- Объединение камер в группы в зависимости от их физического расположения. Добавление и настройка камеры. См. *Управление камерами*.
- Управление списками наблюдения. Создание досье вручную и пакетно. См. *Мониторинг лиц*, *Управление базой данных досье*.
- *Управление пользователями и ролями*.
- *Идентификация лиц в видеофайле*.
- *Статистика лиц и силуэтов*. Сбор статистики в реальном времени по лицам и силуэтам на видео.
- *Основные настройки*. Изменение порога срабатывания системы (порога положительной верификации лиц) и настройка автоматической очистки базы данных от старых событий.
- *Сравнение 2-х лиц*. Проверка 2-х лиц на идентичность.
- Идентификация лиц по базе событий в *режиме реального времени* как на живом (видеопоток), так и на архивном (видеофайл) видео. *Эпизоды событий*. Идентификация лиц *по базе событий и досье*. *Видеонаблюдение*.

1.6.1 Управление видеокамерами

Для настройки видео-идентификации лиц добавьте камеры в FindFace Security, сгруппировав их с учетом расположения.

Примечание: Права на создание групп камер и камер настраиваются в разрешениях пользователя (см. *Управление пользователями*).

В этой главе:

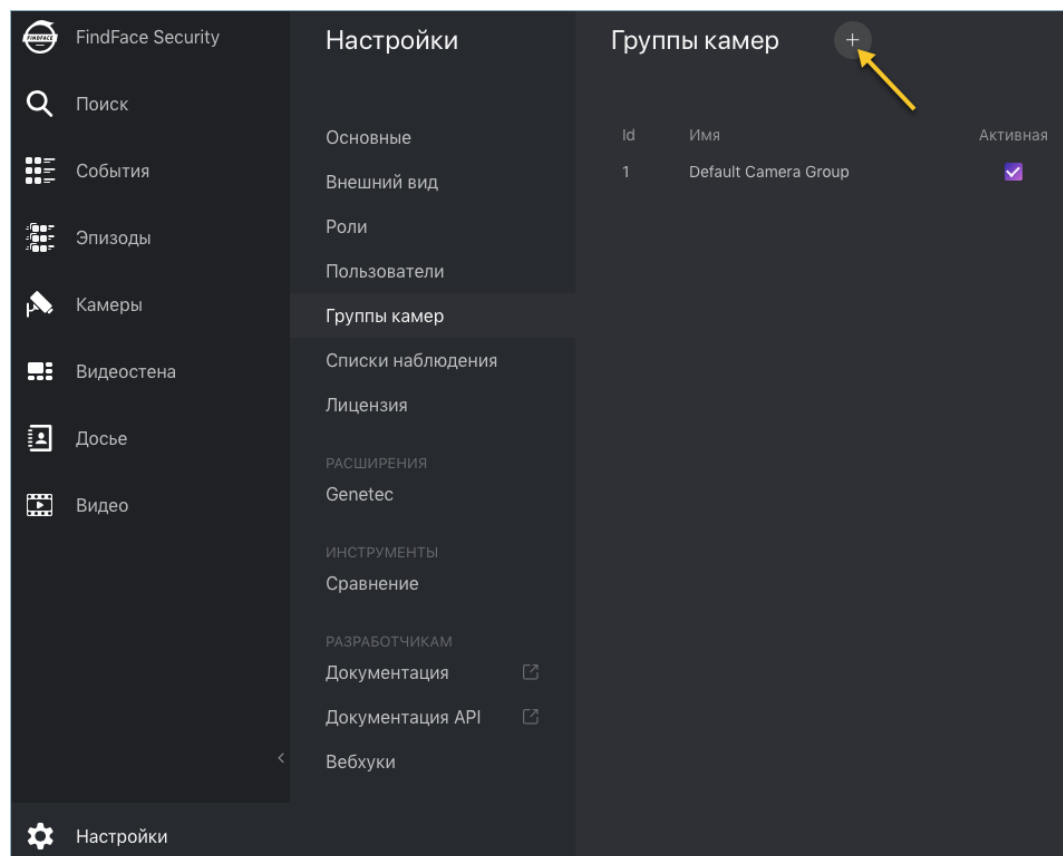
- *Создание группы камер*
- *Добавление камеры*
- *Мониторинг работы камер*

Создание группы камер

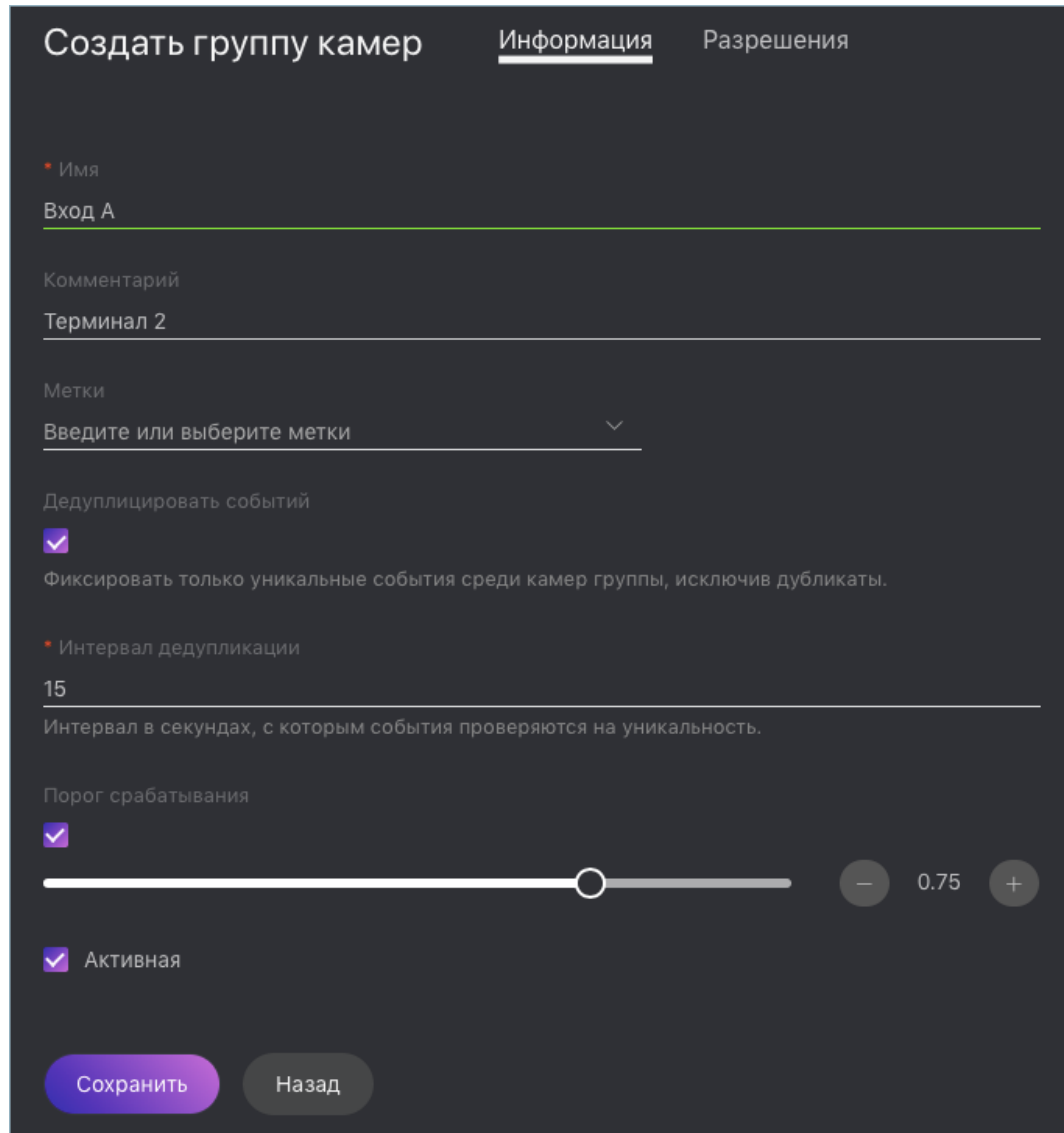
Совет: В системе доступна группа камер по умолчанию.

Для создания группы камер выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Группы камер*.
2. Нажмите **+**.



3. На вкладке *Информация*, введите имя группы и при необходимости комментарий к ней.



4. Если вам нужно выделить определенный экземпляр `findface-video-worker` для обработки видеопотоков с данной группы камер, создайте или выберите из уже созданных одну или несколько меток.

Примечание: Для того чтобы завершить выделение, перечислите метки в файле конфигурации `findface-video-worker`. Подробнее см. [Привязка группы камер к экземпляру `findface-video-worker`](#).

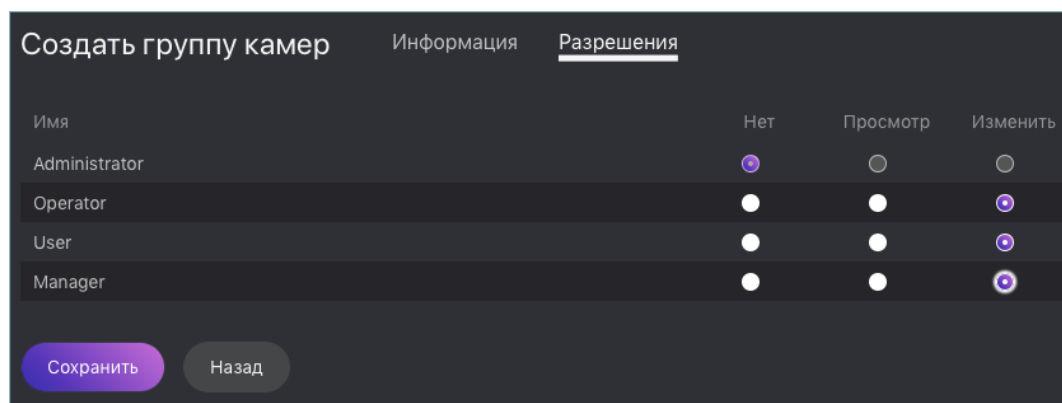
5. Если события от камер, принадлежащих одной группе, требуется дедуплицировать, т. е. исключить одинаковые события, поставьте флажок *Дедуплицировать события* и задайте в секундах интервал дедупликации (интервал, с которым события проверяются на уникальность).

Предупреждение: Используйте дедупликацию очень осторожно. Если камеры из одной группы наблюдают разные сцены, некоторые лица могут быть пропущены. Подробнее см. [Дедупликация событий](#).

- По умолчанию на всех группах камер в системе используется *универсальный порог* срабатывания. Для того чтобы установить индивидуальный порог для группы камер, поставьте флажок *Порог срабатывания* и укажите нужное пороговое значение.

Важно: Для большинства случаев распознавания лиц значение, установленное по умолчанию для универсального порога срабатывания, является оптимальным. Не рекомендуется самостоятельно менять его. Обязательно проконсультируйтесь с нашими техническими экспертами по адресу support@ntechlab.com.

- Поставьте флажок *Активная*.
- Нажмите *Сохранить*.
- На вкладке *Разрешения* назначьте права на работу с группой камер, указав, пользователям с какими ролями разрешено изменять/просматривать ее настройки.

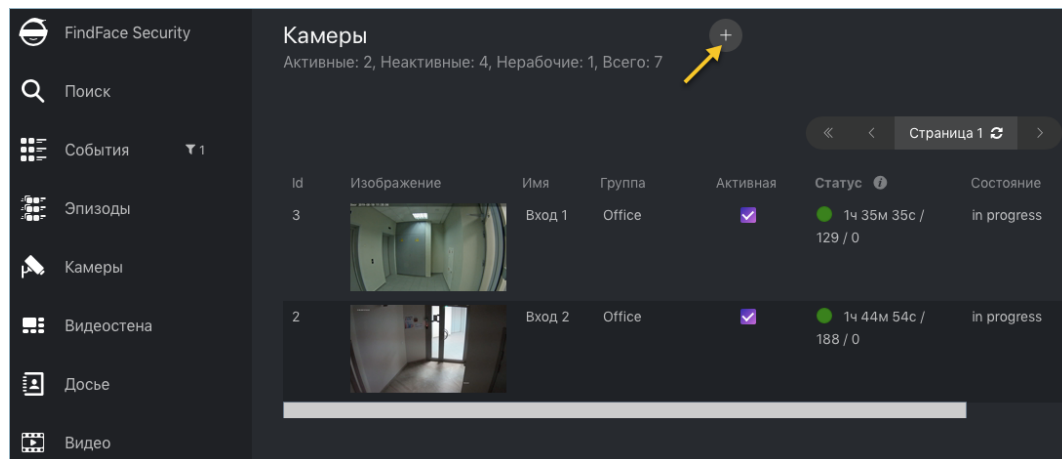


- Нажмите *Сохранить*.

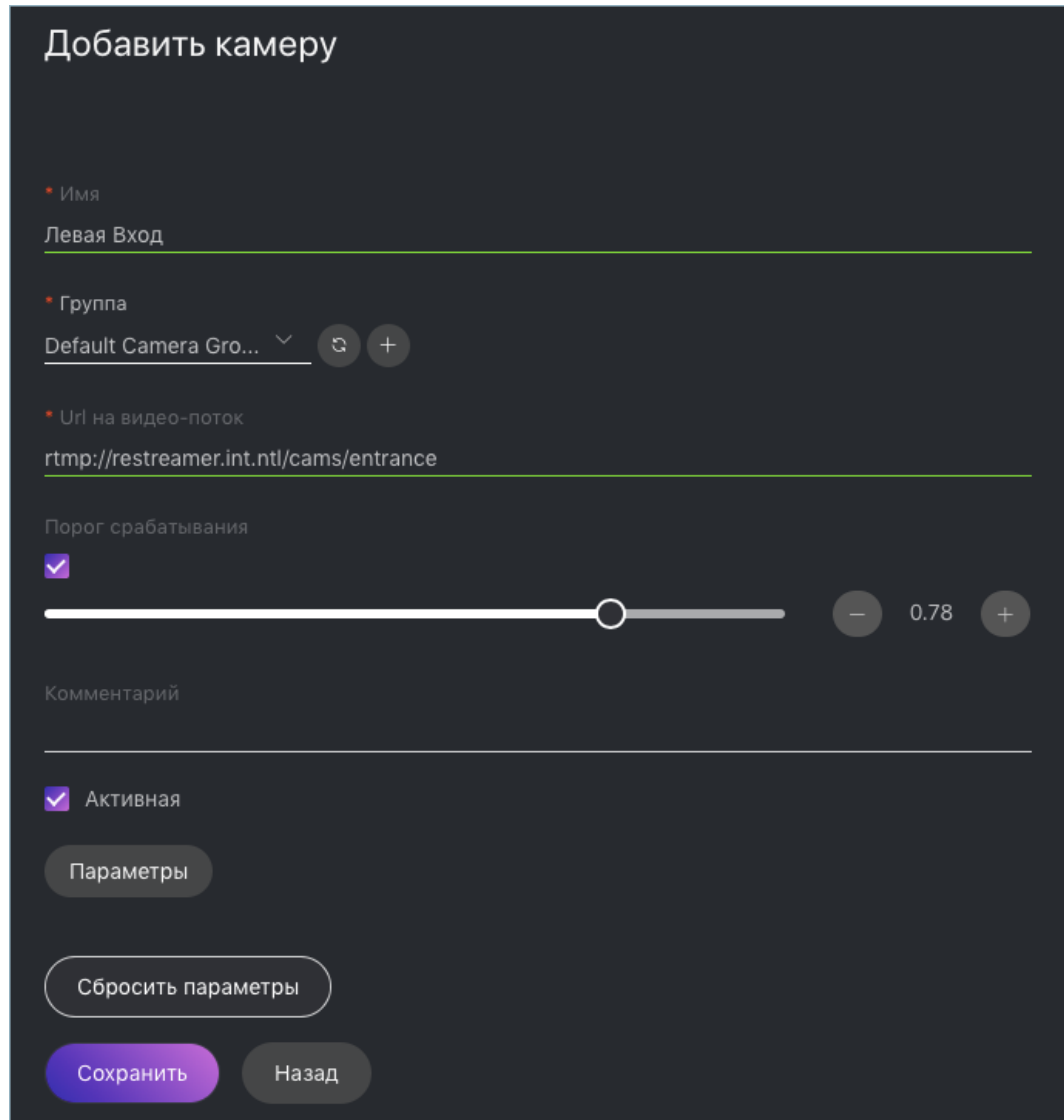
Добавление камеры

Для добавления камеры в группу выполните следующие действия:

- Перейдите на вкладку *Камеры*.
- Нажмите *+*.



- Введите название камеры и добавьте ее в одну из групп. При необходимости введите комментарий к камере.



- Задайте URL камеры или адрес видеофайла для обработки, например, `file:///data/some.mp4`.
- По умолчанию на всех камерах в системе используется *универсальный порог* срабатывания. Для того чтобы установить индивидуальный порог для камеры, поставьте флажок *Порог срабатывания* и укажите нужное пороговое значение.

Важно: Для большинства случаев распознавания лиц значение, установленное по умолчанию для универсального порога срабатывания, является оптимальным. Не рекомендуется самостоятельно менять его. Обязательно проконсультируйтесь с нашими техническими экспертами по адресу support@ntechlab.com.

- Поставьте флажок *Активная*.
- Для того чтобы настроить обработку видео, нажмите на кнопку *Параметры* и внесите изменения:
 - Минимальное качество изображения лица (filter_min_quality):* Минимальное качество

изображения лица для отправки на сервер. Определяется эмпирически: отрицательные значения вблизи 0 = наиболее качественные прямые изображения лиц анфас, -1 = хорошее качество, -2 = удовлетворительное качество, отрицательные значения -5 и меньше = перевернутые лица и лица, повернутые под большими углами, распознавание может быть неэффективным.

- *Минимальный размер лица (filter_min_face_size)*: Минимальный размер лица в пикселях для отправки на сервер. Если 0, фильтр выключен.
- *Максимальный размер лица (filter_max_face_size)*: Максимальный размер лица в пикселях для отправки на сервер.
- *Качество сжатия (jpeg_quality)*: Качество сжатия полного кадра для отправки.
- *Опции FFMPEG (ffmpeg_params)*: Опции ffmpeg для видеопотока. Задаются массивом строк ключ=значение, например, "rtsp_transport=tcp ss=00:20:00".
- *Буферный режим (overall)*: Буферный режим. Отправлять для лица один кадр наилучшего качества.
- *Временной интервал (realtime_post_interval)*: Временной интервал в миллисекундах, в течение которого в режиме реального времени выбирается лучший кадр с лицом.
- *Отправлять лучший кадр (realtime_post_every_interval)*: Если true, отправлять лучший кадр в каждом интервале времени realtime_post_interval в режиме реального времени. Если false, отправлять лучший кадр, только если его качество улучшилось по сравнению с предыдущим отправленным кадром.
- *Время ожидания ответа на запрос (router_timeout_ms)*: Время ожидания в миллисекундах ответа на отправленный запрос с лицом.
- *Получать временные метки из потока (use_stream_timestamp)*: Если true, отправлять на сервер временные метки полученные из потока. Если false, отправлять текущие дату и время.
- *Прибавлять к временным меткам (start_stream_timestamp)*: Прибавлять указанное количество секунд к временным меткам из потока.
- *Ограничение скорости проигрывания (play_speed)*: Если меньше нуля, то скорость не ограничивается. В остальных случаях поток читается со скоростью play_speed. Не применимо для потоков с камер видеонаблюдения.
- *Регион слежения (ROT)*: Включает детектирование и отслеживание лиц только внутри заданной прямоугольной области. Используйте данную опцию, чтобы уменьшить нагрузку на сервер.
- *Регион захвата лица (ROI)*: Включает отправку на сервер лиц, обнаруженных только внутри интересующей области.

Совет: Для задания ROT/ROI удобно использовать визуальный мастер. Сначала создайте камеру без ROT/ROI, затем откройте ее для редактирования и нажмите на кнопку *Параметры*. Вы увидите визуальный мастер.

При необходимости задайте опциональные параметры обработки видео. Для это нажмите на кнопку *Дополнительные параметры*.

- *Формат FFMPEG (ffmpeg_format)*: Передать формат FFMPEG (mxg, flv, и т. д.), если он не может быть автоматически определен.

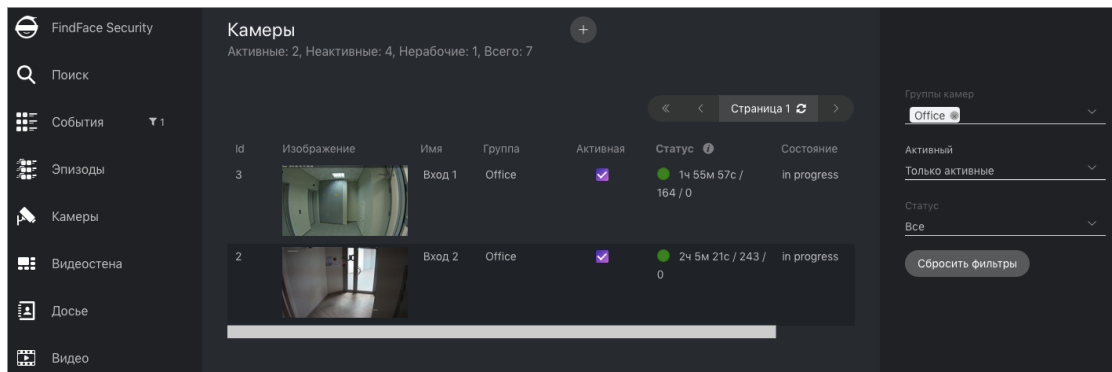
- *Проверять SSL-сертификат (router_verify_ssl)*: Если true, проверять SSL-сертификат сервера при отправке на него лиц через https. Если false, может быть принят самоподписанный сертификат.
- *Минимальная интенсивность движения (imotion_threshold)*: Минимальная интенсивность движения, которая будет регистрироваться детектором движения.

8. Нажмите *Сохранить*.

Примечание: Каждая камера связана с так называемым job-заданием, задачей на обработку видео, содержащей параметры конфигурации и данные видеопотока, которая назначается определенному экземпляру `findface-video-worker`. Данная задача может быть перезапущена (см. [Мониторинг работы камер](#)).

Мониторинг работы камер

Мониторинг работы камер выполняется на вкладке *Камеры*.



Статусы камер:

- Зеленый: идет обработка видеопотока с камеры, проблем не обнаружено.
- Желтый: камера работает менее 30 секунд или имеют место одна или несколько ошибок при отправке лиц.
- Красный: камера не работает.
- Серый: камера отключена.

Для каждой камеры приводятся следующие статистические данные по обработке видеопотока: длительность обработки/количество успешно отправленных лиц/количество лиц, обработанных с ошибками после последнего перезапуска job-задания.

Примечание: Каждая камера связана с так называемым job-заданием, задачей на обработку видео, содержащей параметры конфигурации и данные видеопотока, которая назначается определенному экземпляру `findface-video-worker`. Данная задача может быть перезапущена.

Для перезапуска job-задания откройте настройки камеры и нажмите на кнопку *Перезапустить*. При этом количество ошибок будет обнулено.

При большом количестве камер в системе используйте следующие фильтры:

- *Группы камер*,

- *Активный,*
- *Статус.*

См.также:

- *Привязка группы камер к экземпляру findface-video-worker*
- *Дедупликация событий*

1.6.2 Мониторинг лиц. Управление базой данных досье

Данная глава посвящена мониторингу обнаруженных лиц и работе с базой данных досье. Каждое досье содержит одну или несколько фотографий персоны и классифицируется по принадлежности к тому или иному списку наблюдения, например, к черному или белому в самом простом случае. Вы можете создать несколько списков наблюдения, например, в зависимости от уровня опасности или, наоборот, статуса персоны.

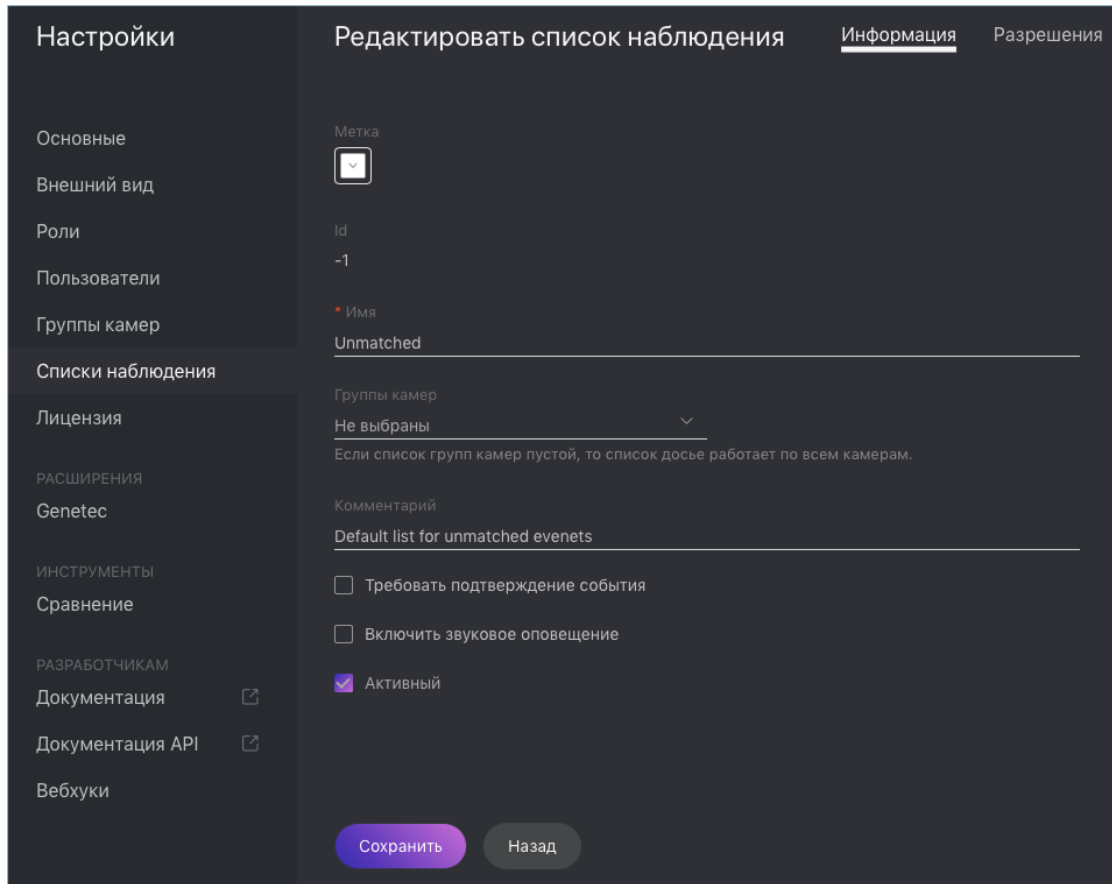
Совет: Для автоматического создания большого количества досье используйте *функционал пакетной загрузки*.

В этом разделе:

- *Мониторинг незарегистрированных лиц*
- *Создание списка наблюдения*
- *Создание досье вручную*
- *Пакетная загрузка фотографий*
- *Фильтрация досье по спискам наблюдения*
- *Очистка базы данных досье*

Мониторинг незарегистрированных лиц

Базовая конфигурация FindFace Security уже содержит предустановленный список наблюдения, предназначенный для мониторинга незарегистрированных в системе лиц, т. е. лиц, для которых отсутствуют совпадения с досье. Данный список наблюдения не может быть удален из системы. Для редактирования настроек или деактивации списка, перейдите на вкладку *Настройки*. Выберите *Списки наблюдения* и откройте настройки списка, щелкнув *Unmatched* в таблице.



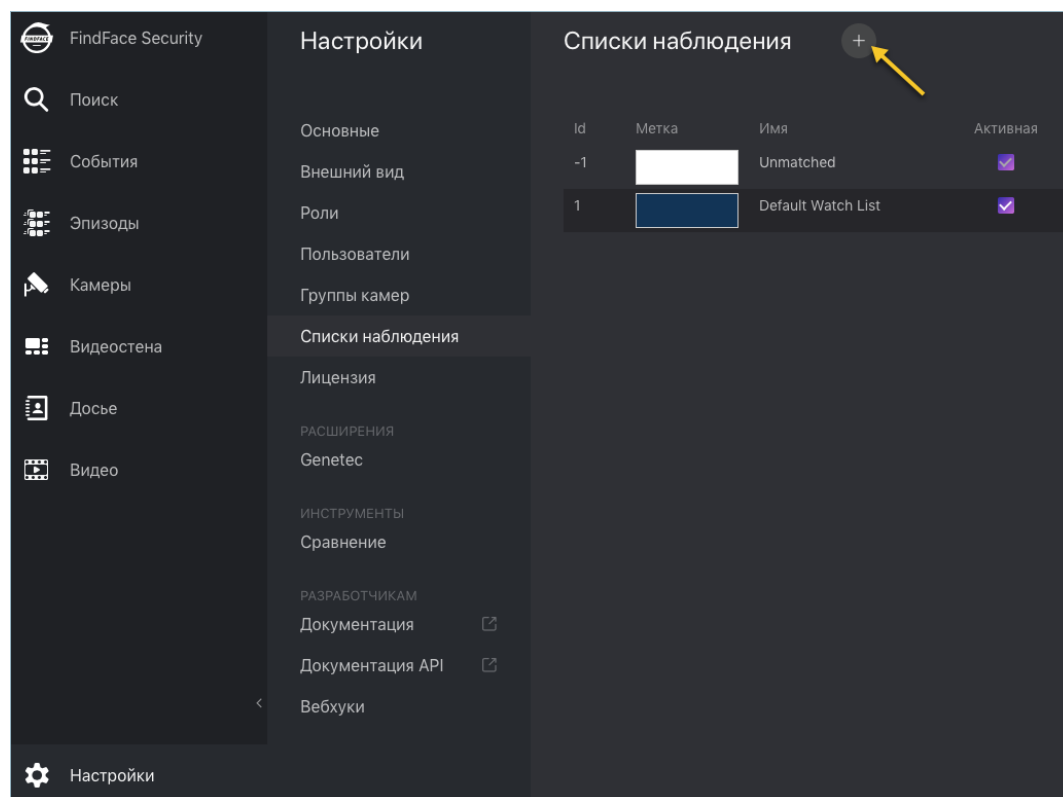
Примечание: Для того чтобы отобразить в списке событий только незарегистрированные лица, выберите значение *Unmatched* в фильтре *Списки наблюдения* на вкладке *События* (подробнее см. *Идентификация лиц в режиме реального времени*).

Создание списка наблюдения

Для создания пользовательского списка наблюдения выполните следующие действия:

Совет: Помимо списка наблюдения *Unmatched* для мониторинга незарегистрированных лиц, в системе по умолчанию уже создан список для мониторинга лиц по базе досье. Данный список наблюдения не может быть удален из системы.

1. Перейдите на вкладку *Настройки*. Выберите *Списки наблюдения*.
2. Нажмите +.



3. В палитре *Метка* выберите цвет, который будет использоваться в событиях распознавания персон из данного списка. Правильно выбранный цвет повышает быстроту реагирования оператора на событие.

Создать список наблюдения Информация Разрешения

Метка

Имя
Уголовный Розыск

Группы камер
Не выбраны

Если список групп камер пустой, то список досье работает по всем камерам.

Комментарий

☒ Порог срабатывания

☒ Требовать подтверждение события

☒ Включить звуковое оповещение

☒ Активный

Сохранить Назад

4. Введите название списка наблюдения. При необходимости добавьте комментарий.
5. Выберите группу камер, которая будет использоваться для мониторинга списка наблюдения. Если группа камер не выбрана, список наблюдения будет отслеживаться всеми активными камерами в системе.
6. Поставьте флажок *Требовать подтверждение*, если для данного списка оператор должен в обязательном порядке подтвердить принятие события.
7. При необходимости включите звук при появлении события для данного списка.
8. По умолчанию ко всем спискам наблюдения в системе применяется *универсальный порог срабатывания*. Для того чтобы задать индивидуальный порог для списка наблюдения, поставьте флажок *Порог срабатывания* и укажите нужное значение.

Важно: Для большинства случаев распознавания лиц значение, установленное по умолчанию для универсального порога срабатывания, является оптимальным. Не рекомендуется самостоятельно менять его. Обязательно проконсультируйтесь с нашими техническими экспертами по

адресу support@ntechlab.com.

9. Поставьте флажок *Активный*.
10. Нажмите *Сохранить*.
11. На вкладке *Разрешения* назначьте права на список наблюдения, указав роли пользователей, которые смогут изменять/просматривать его настройки.

Имя	Нет	Просмотр	Изменить
Administrator	☒	☐	☐
Operator	☐	☒	☐
User	☐	☐	☒
Manager	☐	☐	☒
Оператор Кат. 1	☐	☐	☒

Сохранить Назад

12. Нажмите *Сохранить*.

Создание досье вручную

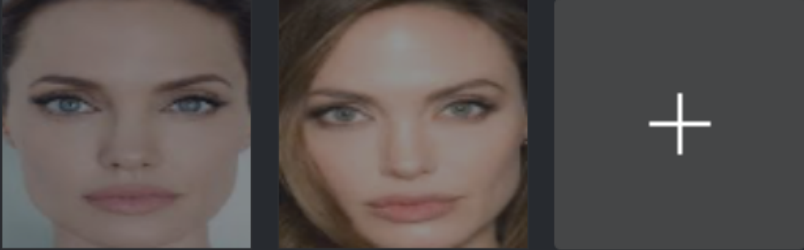
Для создания досье вручную выполните следующие действия:

1. В веб-интерфейсе перейдите на вкладку *Досье*.
2. Нажмите *+*.

Создать досье

У вас много досье для загрузки? Попробуйте [Пакетную загрузку досье](#)

Фотографии



Файлы

 Проверки_2.docx, 9
8.97KB



Имя

Лара Крофт

Комментарий

Списки наблюдения

Default Watch List 

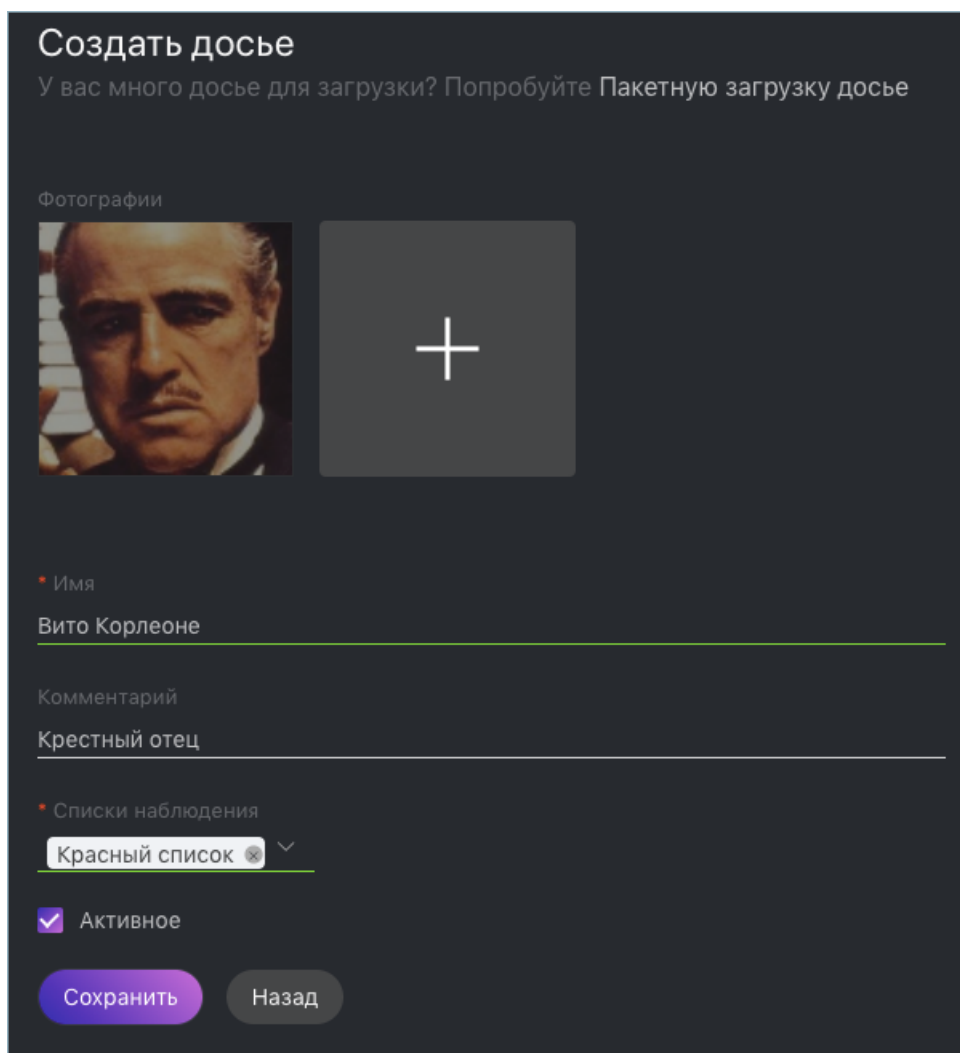
☒ Активное

Сохранить

Назад

- Добавьте одну или несколько фотографий, имеющие отношение к делу файлы и введите имя человека. При необходимости добавьте комментарий.

Важно: Лицо на фотографии должно быть надлежащего качества, т. е. в близком к анфас положении. Расстояние между зрачками: 60 пх. Поддерживаемые форматы: WEBP, JPG, BMP, PNG. При несоответствии фотографии требованиям будет выведено сообщение с описанием ошибки.



4. Из раскрывающегося списка *Списки наблюдения* выберите список (или несколько списков, по очереди), в который следует добавить досье.
5. Поставьте флажок *Активное*. Если досье неактивно, оно не будет использоваться для идентификации лица в режиме реального времени.
6. Нажмите *Сохранить*. Если похожее досье уже существует в базе данных, вы сможете выбрать, объединить ли его с новым досье, отменить создание нового досье, в любом случае создать новое.

Пакетная загрузка фотографий

Для создания большого количества досье используйте функционал пакетной загрузки. Выполните следующие действия:

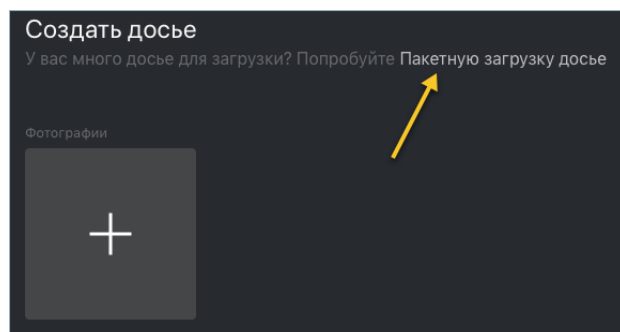
Совет: Если количество досье превышает 10000, используйте *Пакетная загрузка фотографий через консоль*.

Важно: Лицо на фотографиях должно быть надлежащего качества, т. е. в близком к анфас положении. Расстояние между зрачками: 60 px. Поддерживаемые форматы: WEBP, JPG, BMP, PNG. При несоответствии фотографии требованиям будет выведено сообщение с описанием ошибки.

1. В веб-интерфейсе перейдите на вкладку *Досье*.
2. Нажмите *+*.

The screenshot shows a web form titled "Создать досье" (Create Dossier). Below the title is a subtitle: "У вас много досье для загрузки? Попробуйте Пакетную загрузку досье" (You have many dossiers to upload? Try Batch upload dossier). The form is divided into two main sections: "Фотографии" (Photos) and "Файлы" (Files). The "Фотографии" section contains two photo thumbnails of a woman and a large grey square with a white plus sign for adding more photos. The "Файлы" section shows a file named "Проверки_2.docx, 9 8.97KB" and a similar grey square with a plus sign for adding more files. Below these sections are input fields for "Имя" (Name) with the value "Лара Крофт" (Lara Croft), "Комментарий" (Comment), and "Списки наблюдения" (Watch Lists) with a dropdown menu showing "Default Watch List". There is a checked checkbox for "Активное" (Active). At the bottom are two buttons: "Сохранить" (Save) in a purple pill shape and "Назад" (Back) in a grey pill shape.

3. Нажмите *Пакетная загрузка досье*.



4. Выберите фотографии для загрузки пофайлово или укажите папку с фотографиями.

5. Имена файлов с фотографиями можно использовать как основу для имен и/или комментариев в создаваемых досье. Выберите нужный вариант(ы). Затем настройте правило формирования имени и/или комментария, добавив пользовательский префикс и/или постфикс к имени файла.

Совет: Во избежание слияние 3-х слов в одно, используйте символ подчеркивания или пробел в префиксе и постфиксе.

6. Из раскрывающегося списка *Списки наблюдения* выберите список (или несколько списков, по очереди), в который следует добавить создаваемые досье.
7. В параметре *Параллельная загрузка* задайте количество потоков загрузки фотографий. Чем больше потоков, тем быстрее будет завершена загрузка, однако также потребуется и большее количество ресурсов.
8. Из раскрывающегося списка *Групповая фотография* выберите, как должна поступить система при наличии нескольких лиц на фотографии: отклонить фотографию или загрузить самое большое лицо.
9. Для запуска пакетного создания досье нажмите на кнопку *Старт*.

Важно: Для просмотра лога пакетной загрузки нажмите на кнопку *Журналы*. Затем при необходимости можно скачать лог в формате `.csv`.

Журналы пакетной загрузки

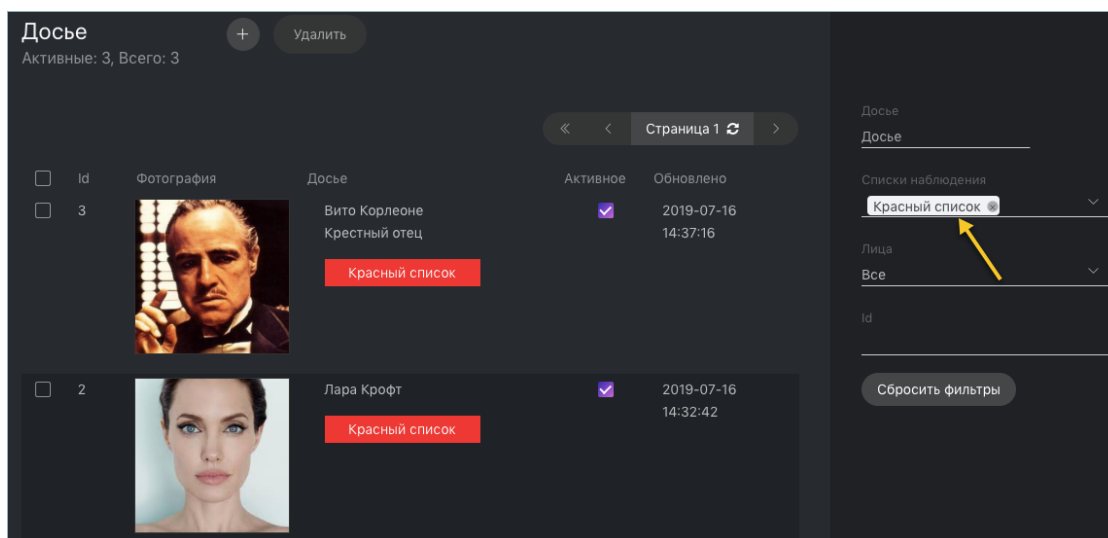
Назад Удалить << < Страница 1 ↻ > >>

☐	Id	Имя	Создано	Количество успешных	Количество ошибок	Скачать csv
☐	3	admin-1562050597638000331	2019-07-02 11:56:36	3	0	Скачать
☐	2	admin-1562037158774000980	2019-07-02 08:12:38	120	2	Скачать
☐	1	admin-1558353930674000104	2019-05-20 17:05:30	97	3	Скачать

<< < Страница 1 ↻ > >>

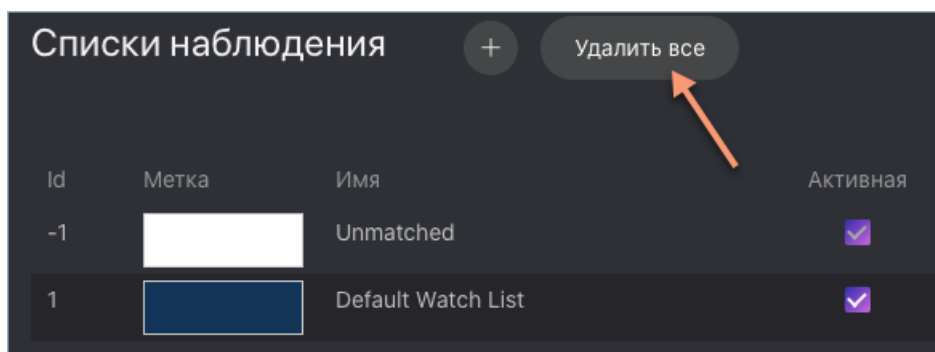
Фильтрация досье по спискам наблюдения

Все созданные в FindFace Security досье отображаются на вкладке *Досье*. Используйте фильтр *Списки наблюдения*, чтобы отфильтровать досье по спискам.



Очистка базы данных досье

Вы можете очистить базу данных досье в один клик целиком. Для этого перейдите на вкладку *Настройки*. Выберите *Списки наблюдения*. Нажмите *Удалить все*.



1.6.3 Управление пользователями

В этой главе:

- *Предустановленные роли*
- *Создание новой роли*
- *Главная и дополнительная роль пользователя*
- *Создание пользователя*
- *Деактивация или удаление пользователя*

Предустановленные роли

Для работы с FindFace Security предусмотрены следующие предустановленные роли:

- Администратор. Обладает полными правами на *управление видеокамерами*, *базой данных досье*, событий, пользователями FindFace Security, а также полным доступом ко всем остальным функциям.

Важно: Супер Администратор не может лишиться прав администратора даже при смене роли.

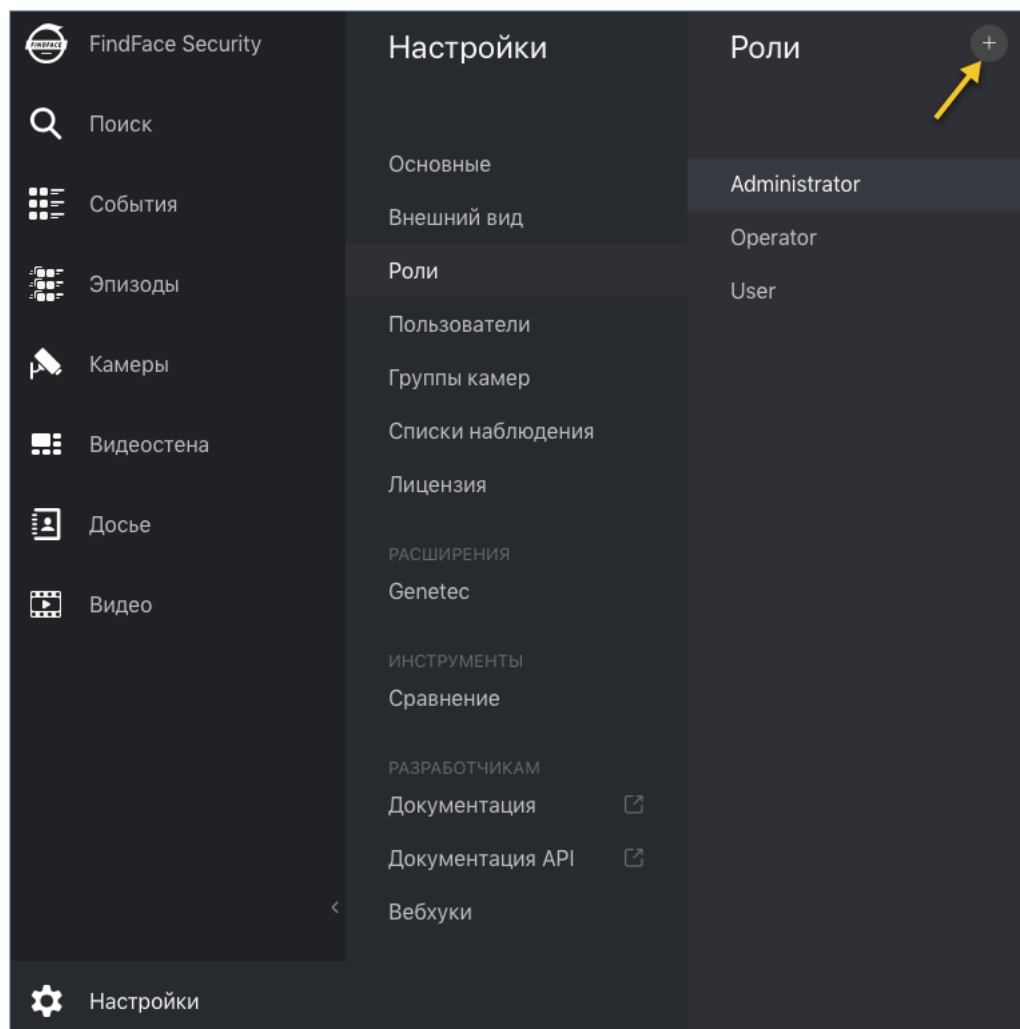
- Оператор. Обладает правами на *создание досье вручную*, подтверждение событий и поиск лиц в базах событий и досье. Остальная информация доступна в режиме чтения. *Пакетное* создание досье невозможно.
- Пользователь. Обладает правами только на подтверждение событий и поиск лиц в базе событий и досье. Остальная информация доступна в режиме чтения.

Вы можете изменить привилегии предустановленных ролей, а также создать новые роли.

Создание новой роли

Для создания новой роли выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Роли*.
2. Нажмите *+*.



3. На вкладке *Информация* задайте имя роли.

4. Нажмите *Сохранить*. Рядом с вкладкой *Информация* появятся дополнительные вкладки. На данных вкладках можно задать права на определенные списки наблюдения (вкладка *Списки наблюдения*) и группы камер (*Группы камер*), а также привилегии на работу с системными функциями и объектами (*Разрешения*).

Примечание: Например, если вы устанавливаете **Нет** для определенной группы камер на вкладке *Группы камер*, пользователи с данной ролью не смогут работать с **этой** группой камер. Установка **Нет** для `cameragroup` на вкладке *Разрешения* не позволит пользователям просматривать и работать со **всеми** группами камер.

Примечание: Право на событие складывается из прав на соответствующую камеру и список наблюдения. Для просмотра событий, для которых не найдены совпадения с досье, требуются только права на камеру.

Полный список объектов FindFace Security:

- `dossierlist`: *списки наблюдения*;
- `dossier`: *досье*;
- `dossierface`: *фото в досье*;
- `cameragroup`: *группа камер*;
- `camera`: *камера*;
- `listevent`: *список событий*;
- `eventepisode`: *эпизоды*
- `uploadlist`: список фотографий в *пакетной загрузке*;
- `upload`: элемент (фото) в пакетной загрузке;
- `user`: *пользователь*;
- `webhook`: *вебхук*;
- `videosource`: *идентификация лиц в видеофайлах*.
- `counter`: *счетчики лиц и силуэтов*
- `person`: *галерея персон*

Вы также можете включать и отключать права на следующий функционал:

- `configure_sova`: конфигурация интеграции с системой Сова
- `configure_genetec`: конфигурация *интеграции с Genetec*;
- `configure_ntls`: конфигурация *сервера лицензий findface-ntls*.
- `batchupload_dossier`: *пакетное создание досье*;
- `view_runtimeetting`: просмотр *основных настроек* FindFace Security;
- `change_runtimeetting`: изменение основных настроек FindFace Security

Информация	Списки наблюдения	Группы камер	Разрешения	
Имя	Просмотр	Изменить	Добавить	Удалить
dossierlist	✓	✓	✓	✓
dossier	✓	✓	✓	✓
dossierface	✓	✓	✓	✓
cameragroup	✓	✓	✓	✓
camera	✓	✓	✓	✓
listevent	✓	✓	✓	✓
eventepisode	✓	✓	✓	✓
uploadlist	✓	✓	✓	✓
upload	✓	✓	✓	✓
user	✓	✓	✓	✓
webhook	✓	✓	✓	✓
videosource	✓	✓	✓	✓
counter	✓	✓	✓	✓
person	✓	✓	✓	✓
				Активное
configure_sova				✓
configure_genetec				✓
configure_ntls				✓
batchupload_dossier				✓
view_runtime-setting				✓
change_runtime-setting				✓

Главная и дополнительная роль пользователя

Вы можете назначить пользователю привилегии, используя следующие роли:

- *Главная роль*: основная роль пользователя, обязательная для назначения. Пользователю можно назначить только одну главную роль.
- *Роль*: дополнительная роль пользователя, необязательная для назначения. Одному пользователю можно назначить несколько ролей. Связанные с ними права будут добавлены к правам, предоставляемым главной ролью.

Все пользователи, принадлежащие к определенной основной роли, автоматически получают доступ к группам камер (и камерам внутри группы) и спискам наблюдения (и досье в этих списках), созданным

пользователем с той же основной ролью, с учетом привилегий, определенных их дополнительными ролями.

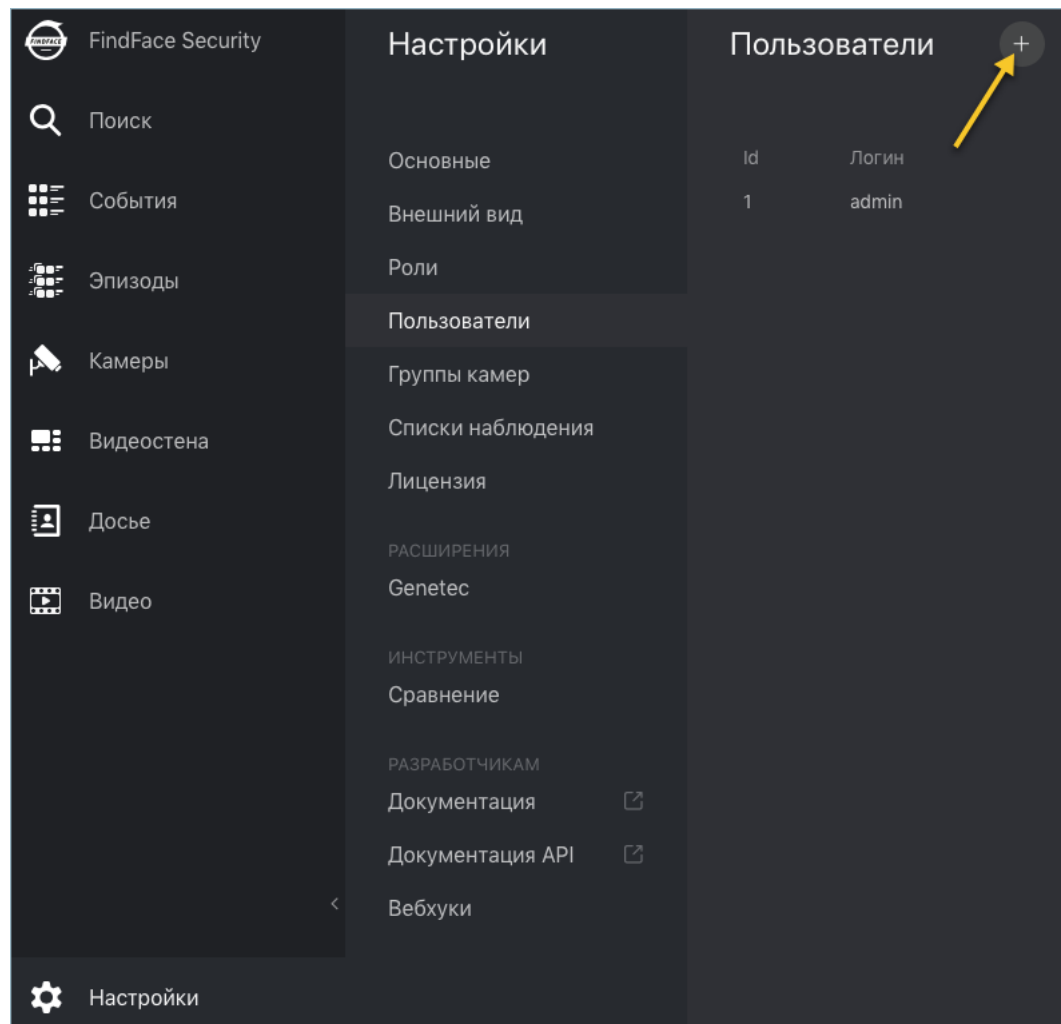
См.также:

Создание пользователя

Создание пользователя

Для создания нового пользователя выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Пользователи*.
2. Нажмите **+**.



3. Задайте имя пользователя, логин и пароль. При необходимости добавьте комментарий.
4. Из раскрывающегося меню *Roles* выберите одну или несколько пользовательских ролей. Назначьте они из них основной.

Создать пользователя

Имя
Иванов Николай Петрович

Логин
n.ivanov@ntechlab.com

Пароль
narkyw-huccab-bumBe7 Strong Password

Подтверждение пароля
narkyw-huccab-bumBe7 Strong Password

Роли
Оператор Кат. 1 Главная роль

Добавить роль

Комментарий

Активный
☒

Создать Назад

5. Поставьте флажок *Активный*.

6. Нажмите *Создать*.

Деактивация или удаление пользователя

Для того чтобы деактивировать пользователя, снимите флажок *Активный* в списке пользователей (*Настройки -> Пользователи*).

Для удаления пользователя из FindFace Security щелкните по его логину в списке. Нажмите *Удалить*.

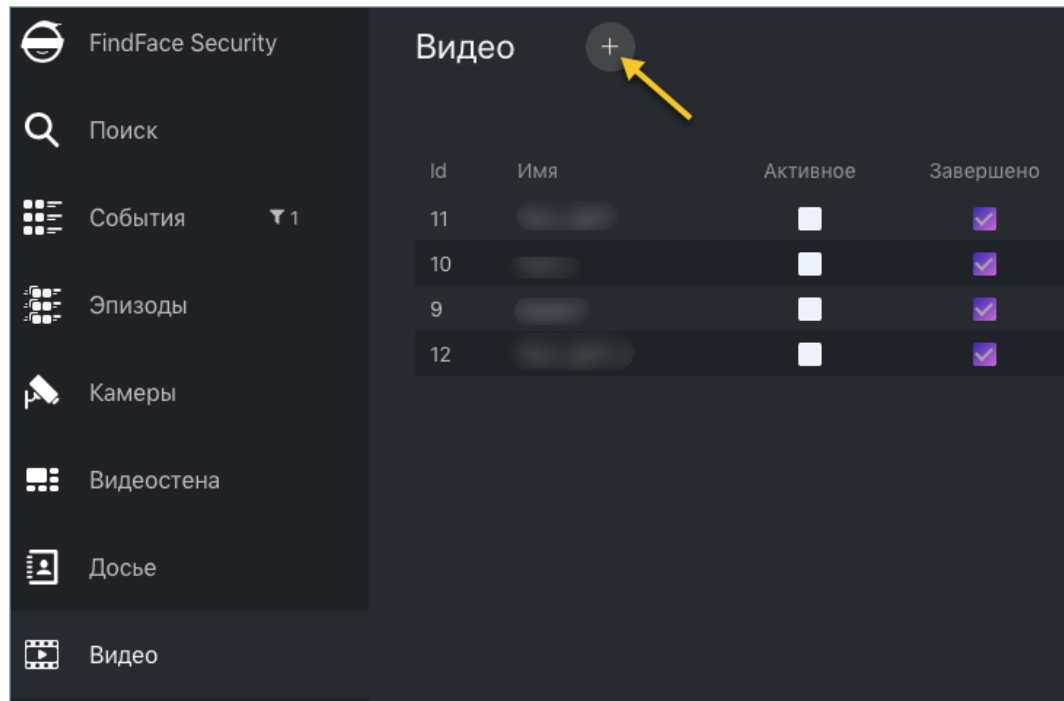
1.6.4 Идентификация лиц в офлайн видео

Помимо идентификации лица в режиме реального времени, FindFace Security также позволяет выполнять идентификацию лиц в офлайн видео. Данная функциональность имеет широкий спектр

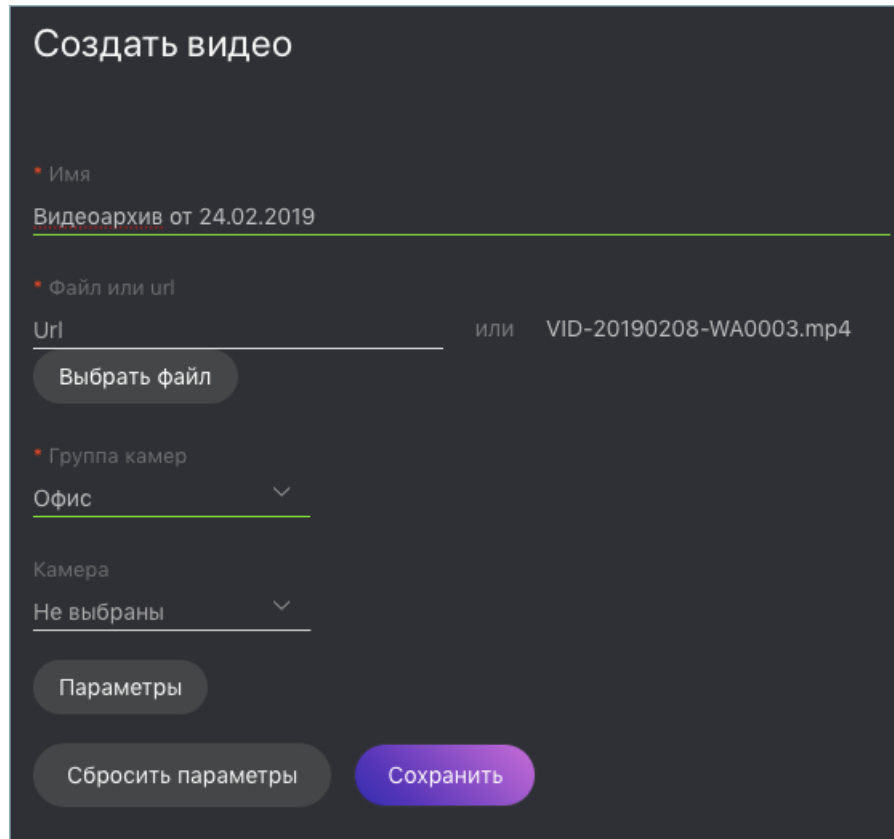
возможных применений, среди которых наиболее распространенным случаем является обнаружение и распознавание лиц в архивных видео.

Для идентификации лица в оффлайн видео выполните следующие действия:

1. Создайте *группу камер* с базовыми настройками.
2. Укажите данную группу камер в настройках тех *списков наблюдения*, лица в которых нужно найти на видео.
3. Создайте видео в FindFace Security, загрузив его из файла или онлайн-хранилища/облака. Для этого, перейдите на вкладку *Видео*.
4. Нажмите *+*.



5. Укажите название видео.



Создать видео

• Имя
Видеоархив от 24.02.2019

• Файл или url
Url или VID-20190208-WA0003.mp4
Выбрать файл

• Группа камер
Офис

Камера
Не выбраны

Параметры

Сбросить параметры Сохранить

6. Укажите URL видеоизображения в онлайн-хранилище или выберите видеофайл.
7. Выберите только что созданную группу камер.
8. (Опционально) Выберите камеру, с которой будут ассоциироваться лица, найденные на видео.
9. (Опционально) Укажите параметры обработки видео по аналогии с параметрами обработки видеопотока с *камеры*.
10. Нажмите *Сохранить* для загрузки видео на сервер.



Id
1

* Имя
Видеоархив от 24.02.2019

* Файл или url
Url _____ или

* Группа камер
Офис

Камера
Не выбраны

- После того как видео загрузится, нажмите *Обработать* для запуска процедуры идентификации лиц. Для просмотра найденных лиц перейдите на вкладку *События* и отфильтруйте список событий по связанной группе камер.

1.6.5 Счетчики лиц и силуэтов

Важно: Для подсчета силуэтов нужно предварительно включить их *детектирование*.

FindFace Security позволяет подсчитывать лица и силуэты на подключенных камерах. Данную функциональность можно использовать в различных целях, например, для подсчета людей в очередях и зонах ожидания, мониторинга общественных собраний, предотвращения скопления людей и т. д.

Метод подсчета основан на временных срезах. Это означает, что счетчик считает лица и силуэты на статических скриншотах, которые делаются с заданным интервалом. Счетчик показывает, как количество лиц и силуэтов меняется со временем.

В этом разделе:

- *Настройка счетчиков*
- *Создание счетчика*
- *График счетчика*
- *Работа с записями счетчика*
- *Настройка вебхука для счетчика*

Настройка счетчиков

Для настройки счетчиков откройте файл конфигурации `findface-security` и измените следующие параметры:

- `COUNTERS_SAVE_FULLFRAME` определяет параметры сохранения полных кадров при работе счетчиков: `always` (всегда), `detect` - сохранять, только если были обнаружены лица или силуэты, `never` (никогда).
- `COUNTERS_FULLFRAME_JPEG_QUALITY`: качество JPEG полных кадров,
- `COUNTERS_THUMBNAIL_JPEG_QUALITY`: качество JPEG миниатюр.

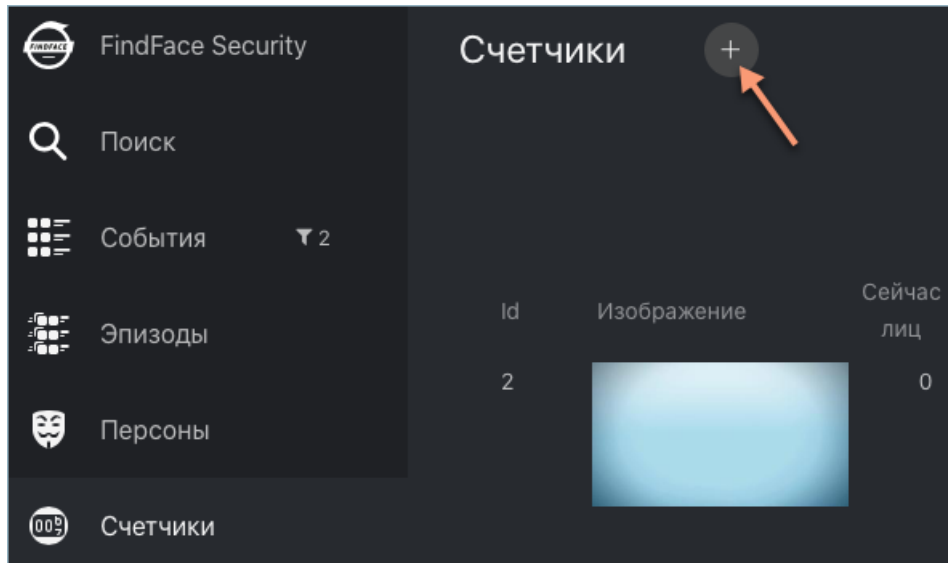
```
sudo vi /etc/ffsecurity/config.py

# counters full frame saving options:
# `always` - save always
# `detect` - save only if faces or silhouettes have been detected
# `never` - never save full frames
'COUNTERS_SAVE_FULLFRAME': 'always',
'COUNTERS_FULLFRAME_JPEG_QUALITY': 75,
'COUNTERS_THUMBNAIL_JPEG_QUALITY': 75,
...
```

Создание счетчика

Для создания счетчика выполните следующие действия:

1. Перейдите на вкладку *Счетчики*.
2. Нажмите `+`.



3. Укажите имя счетчика.

4. Выберите камеру для подсчета.
5. Задайте интервал между двумя последовательными скриншотами, используемыми для подсчета лиц и силуэтов.
6. Установите флажок *Определять лица* для подсчета лиц.
7. Установите флажок *Определять силуэты* для подсчета силуэтов. Детектирование силуэтов должно быть *включено*.

8. Убедитесь, что для счетчика установлен флажок *Активный*.
9. Нажмите *Сохранить*. В результате появятся две новые вкладки.
10. (Опционально) Перейдите на вкладку *Регион захвата*, чтобы задать регион отслеживания лиц и силуэтов в поле зрения камеры.

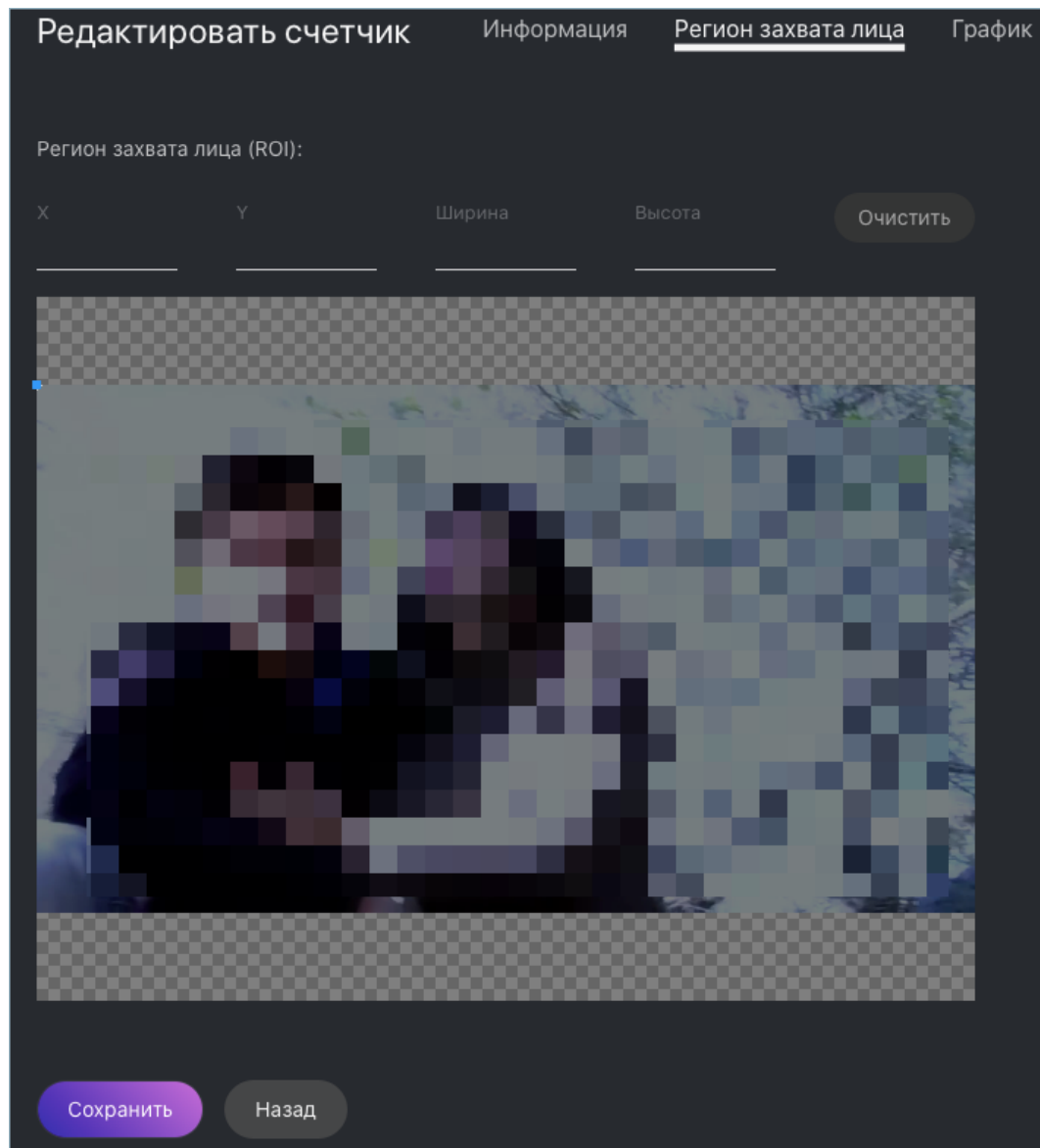
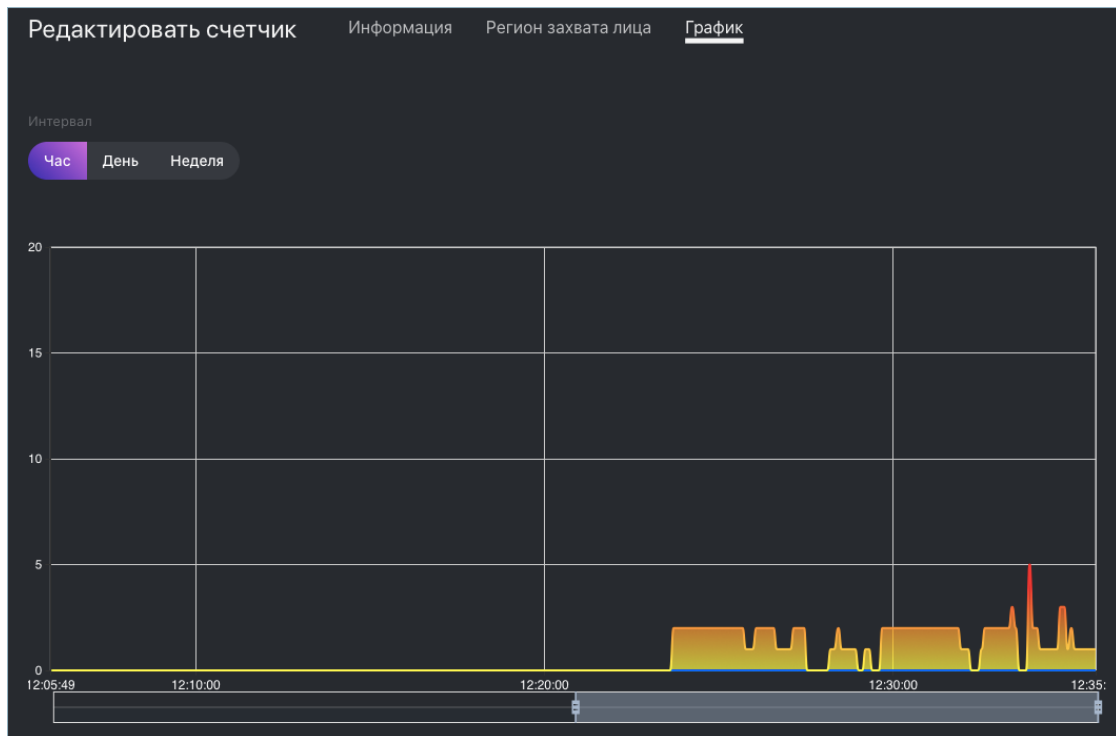


График счетчика

Для того чтобы увидеть график счетчика за последние час, сутки или неделю, перейдите на вкладку *График* в настройках счетчика.



Работа с записями счетчика

Статические скриншоты, сделанные счетчиком, с указанием количества лиц и силуэтов на них сохраняются как т. н. записи счетчика.

Для просмотра записей счетчика перейдите на вкладку *Счетчики* и нажмите на нужный счетчик.

FindFace Security					
Записи счетчика					
Поиск События 2 Эпизоды Персоны Счетчики Камеры Видеостена Досье Видео					
Id	Изображение	Лица	Силуэты	Дата	
51890		1	0	2020-06-07 12:28:31	
51888		2	0	2020-06-07 12:28:25	
51886		1	0	2020-06-07 12:28:18	
51884		1	0	2020-06-07 12:28:12	

Счетчик

Офис

Камеры

Не выбраны

Группы камер

Не выбраны

Старт

Конiec

Количество лиц

От

Количество силуэтов

От

Для работы с записями счетчика используйте следующие фильтры:

- Счетчик
- Камеры
- Группы камер
- Период времени

- Количество лиц в записи
- Количество силуэтов в записи
- Id записи

Настройка вебхука для счетчика

Для того чтобы автоматизировать процесс сбора статистики, *настройте вебхук* на срабатывание, если запись счетчика содержит определенное количество лиц и силуэтов.

См.также:

- *Детектирование силуэтов*
- *Вебхуки*

1.6.6 Основные настройки

Для изменения порога срабатывания при верификации лиц, порога срабатывания для эпизодов, качества JPEG миниатюр и настройки автоматического удаления старых лиц (событий и эпизодов) из базы данных, перейдите на вкладку *Настройки*. Выберите *Основные*. После внесения изменений нажмите *Обновить*.

Основные настройки

Порог срабатывания

Порог срабатывания в эпизодах

Качество jpeg миниатюр

Очистка событий

☒ Включено

Удалять события с совпадениями старше

Удалять события без совпадений старше

Удалять полные кадры событий с совпадениями старше

Удалять полные кадры событий без совпадений старше

В этом разделе:

- *Универсальный порог срабатывания*

- Порог срабатывания для эпизодов
- Качество JPEG миниатюр
- Автоматическое удаление старых событий и эпизодов

Универсальный порог срабатывания

FindFace Security принимает решение о совпадении обнаруженного лица с лицом из досье (т. е. о положительной верификации) на основании предустановленной пороговой степени схожести. По умолчанию установлено оптимальное пороговое значение, равное 0.739. При необходимости вы можете изменить данное значение.

Примечание: Чем выше пороговая степень схожести, тем меньше шансов на положительную ложную верификацию человека, однако некоторые подходящие фотографии могут также не пройти верификацию.

Совет: Вы можете настроить порог срабатывания индивидуально для каждой *камеры, группы камер и списка наблюдения*.

Важно: Для большинства случаев распознавания лиц значение, установленное по умолчанию для универсального порога срабатывания, является оптимальным. Не рекомендуется самостоятельно менять его. Обязательно проконсультируйтесь с нашими техническими экспертами по адресу support@ntechlab.com.

Порог срабатывания для эпизодов

При формировании эпизода система ищет в биометрической базе данных *недавние* события с похожими лицами выше определенной пороговой степени схожести. Порог по умолчанию установлен на 0.689. При необходимости вы можете изменить это значение.

Качество JPEG миниатюр

В зависимости от качества JPEG, файлы миниатюр могут в конечном счете занять значительную часть дискового пространства. Используйте вкладку *Основные* для настройки параметра.

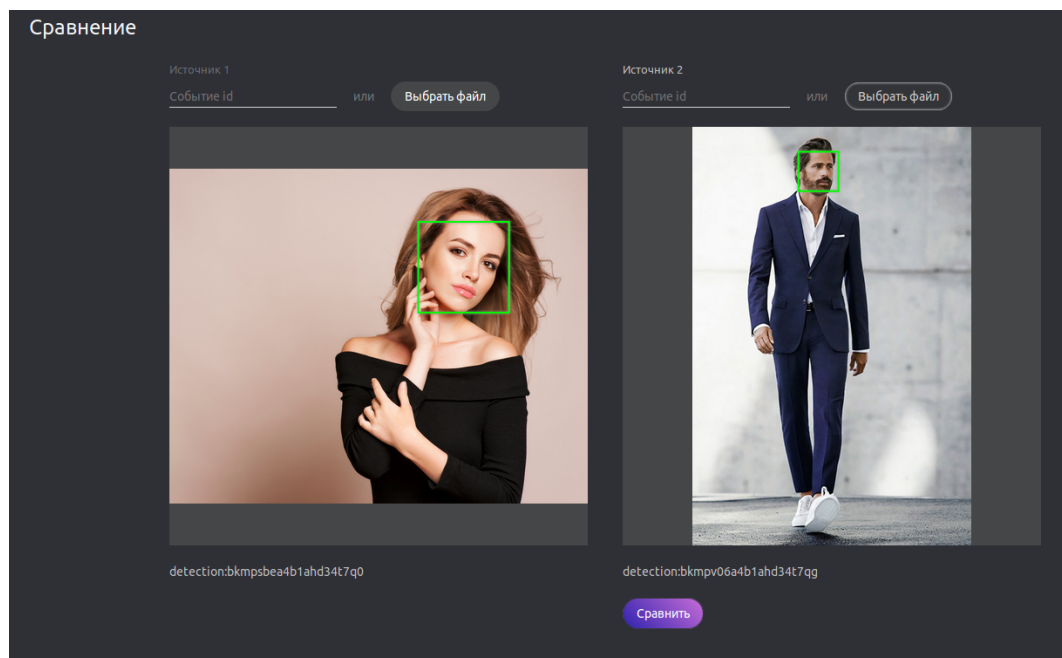
Автоматическое удаление старых событий и эпизодов

Используйте эту же вкладку, чтобы задать расписание автоматического удаления старых событий и эпизодов из базы данных. Можно настроить удаление событий и эпизодов по разным расписаниям в зависимости от наличия совпадений с базой данных досье, а также выбрать удаление только полных кадров.

1.6.7 Сравнение лиц

FindFace Security позволяет выполнять сравнение 2-х лиц. Выполните следующие действия:

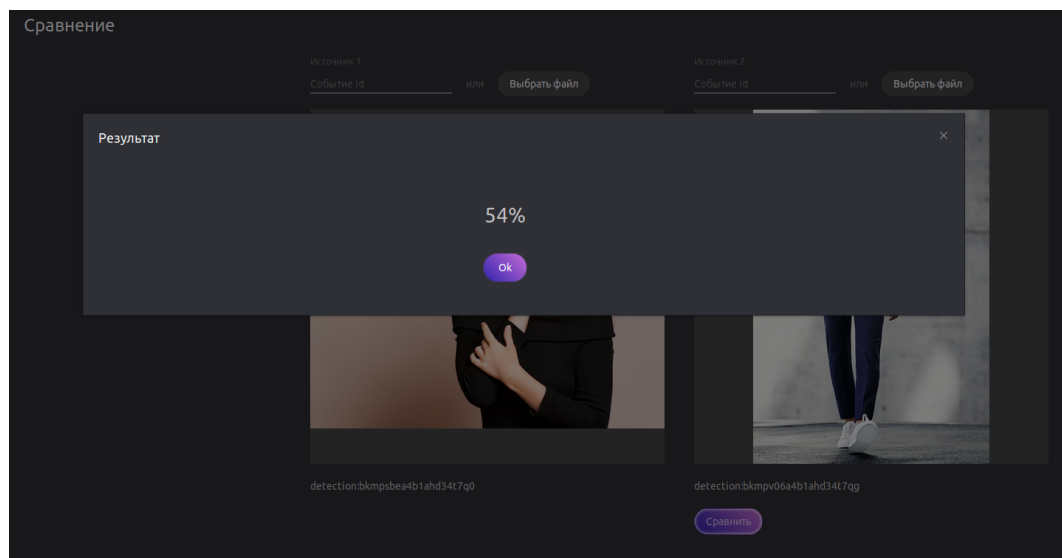
1. Перейдите на вкладку *Настройки*. Выберите *Сравнение*.



2. Укажите id событий, лица из которых нужно сравнить, и/или загрузите фотографии с лицами.

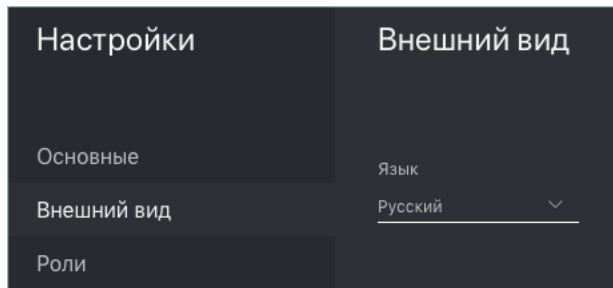
Совет: Узнать ID события можно на вкладке *События*.

3. Нажмите *Сравнить*. В результате будет отображена вероятность принадлежности лиц одному человеку.



1.6.8 Переключение языка

Для того чтобы переключить язык интерфейса, перейдите в меню *Настройки* -> *Внешний вид*.



1.7 Расширенный функционал

1.7.1 Настройка эпизодов

В этом разделе:

- *Об эпизодах*
- *Параметры эпизода*
- *Назначение прав на эпизоды*

Об эпизодах

Эпизод — это набор событий идентификации, в которых фигурируют лица одного и того же человека, обнаруженные в течение определенного периода времени.

Эпизоды бывают двух типов:

- **LIVE**: открытый на данный момент эпизод, в который могут добавлены новые события.
- **Закрытый**: закрытый эпизод, добавление событий невозможно.

Параметры эпизода

Для настройки эпизодов вам понадобится файл конфигурации **findface-security**. Добавьте следующие параметры в секцию **FFSECURITY**:

- **EPISODE_SEARCH_INTERVAL**: период времени, предшествующий событию, в течение которого система ищет в биометрической базе данных события с похожими лицами. Если такого события не найдено, система создает новый эпизод. В противном случае она выбирает наиболее подходящее событие из открытого (LIVE) эпизода, отсортировав 100 последних похожих лиц.

Примечание: Порог срабатывания в эпизодах отличается от порога при верификации лиц в событиях. См. *Основные настройки*.

- **EPISODE_MAX_DURATION**: максимальная продолжительность эпизода в секундах. По истечении этого времени эпизод автоматически закрывается.

- EPISODE_EVENT_TIMEOUT: максимальное время в секундах с момента добавления последнего события в эпизод. По истечении этого времени эпизод автоматически закрывается.

```
sudo vi /etc/ffsecurity/config.py

...

FFSECURITY = {
    ...
    'EPISODE_SEARCH_INTERVAL': 60,
    'EPISODE_MAX_DURATION': 300,
    'EPISODE_EVENT_TIMEOUT': 30,
    ...
}

...
```

См.также:

Для того чтобы посмотреть, как работают эпизоды, перейдите на вкладку *Эпизоды*. Подробнее см. *Эпизоды событий*.

Назначение прав на эпизоды

Пользователь получает уведомление о новом эпизоде, если у него есть права на открывающее этот эпизод событие. Просмотр новых событий в эпизоде также требует соответствующих прав.

Право на событие состоит из прав на соответствующие камеру и список наблюдения.

Примечание: Чтобы увидеть несопоставленные с досье события, вам понадобятся только права на камеру.

Для управления правами на объект *Эпизод* перейдите в разрешения для соответствующей роли и настройте разрешение `eventepisode`.

Совет: См. *Управление пользователями*.

Информация	Списки наблюдения	Группы камер	Разрешения	
Имя	Просмотр	Изменить	Добавить	Удалить
dossierlist	✓	✓	✓	✓
dossier	✓	✓	✓	✓
dossierface	✓	✓	✓	✓
cameragroup	✓	✓	✓	✓
camera	✓	✓	✓	✓
listevent	✓	✓	✓	✓
eventepisode	✓	✓	✓	✓
uploadlist	✓	✓	✓	✓
upload	✓	✓	✓	✓
user	✓	✓	✓	✓
webhook	✓	✓	✓	✓
videosource	✓	✓	✓	✓
counter	✓	✓	✓	✓
person	✓	✓	✓	✓
				Активное
configure_sova				✓
configure_genetec				✓
configure_ntls				✓
batchupload_dossier				✓
view_runtime_setting				✓
change_runtime_setting				✓

1.7.2 Привязка группы камер к экземпляру findface-video-worker

Часто в распределенной архитектуре обработку видеоизображения с группы камер требуется выполнять локально, не обращаясь к центральному серверу и не перераспределяя видеопотоки между удаленными экземплярами `findface-video-worker`.

Примечание: Например, это может быть актуальным для сетей гостиниц, магазинов, при наличии нескольких проходных в одном здании и т. д.

В этом случае группу камер привязывают к локально установленному экземпляру `findface-video-worker`.

Выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Группы камер*.
2. Откройте настройки группы камер.
3. В поле *Метки* создайте или выберите из уже созданных одну или несколько меток для привязки группы камер к экземпляру `findface-video-worker`. Сохраните изменения.
4. Откройте файл конфигурации экземпляра `findface-video-worker` и укажите в нем заданные метки в формате `имя_метки=true` (`terminal_1` в примере ниже).

```
sudo vi /etc/findface-video-worker-cpu.ini
sudo vi /etc/findface-video-worker-gpu.ini

labels = terminal_1=true
```

5. Перезапустите `findface-video-worker`.

```
sudo systemctl restart findface-video-worker-cpu.service
sudo systemctl restart findface-video-worker-gpu.service
```

Примечание: Если камере присвоена метка, то видеопоток с нее может обрабатываться как экземпляром `findface-video-worker(-gpu)` с аналогичной меткой, так и экземпляром `findface-video-worker(-gpu)` без меток.

Предупреждение: Если камера с меткой обрабатывается экземпляром `findface-video-worker(-gpu)` без меток и появляется свободный экземпляр с меткой, камера автоматически на него не переключится. Чтобы переключить камеру, перезапустите экземпляр `findface-video-worker(-gpu)` с меткой.

1.7.3 Распределенная база данных досье

В распределенной архитектуре часто необходимо, чтобы база данных досье распределялась по нескольким серверам.

В текущей реализации база данных досье доступна для редактирования только на главном сервере, т. н. master-сервере. Master-сервер синхронизирован с несколькими дополнительными slave-экземплярами FindFace Security. На slave-серверах база данных досье доступна только для чтения и мониторинга.

Важно: Досье можно удалить на slave-сервере, если master-сервер недоступен.

Важно: Если в списке наблюдения будущего slave-сервера уже есть досье, они будут заменены на досье с master-сервера во время первой синхронизации. Вся информация в предыдущих досье будет потеряна.

В этом разделе:

- *Настройка синхронизации Master/Slave*
- *Репликация списка наблюдения master -> slave*
- *Задание времени синхронизации*
- *Отмена репликации и синхронизации списка наблюдения*
- *Включение функционала в веб-интерейсе*

Настройка синхронизации Master/Slave

Для настройки синхронизации master/slave выполните следующие действия:

1. На master-сервере откройте файл конфигурации **findface-security**. Придумайте токен синхронизации и укажите его в параметре **SYNC_TOKEN**.

```
sudo vi /etc/ffsecurity/config.py

...

SYNC_TOKEN = 'ABC_123456789'
```

2. Скопируйте токен синхронизации в файл конфигурации **findface-security** на slave-серверах.
3. Убедитесь, что параметр **EXTERNAL_ADDRESS** в файлах конфигурации **findface-security** на slave-серверах содержит тот же адрес, что и на master-сервере.

Синхронизация master/slave теперь настроена и будет активирована, как только вы настроите репликацию списка наблюдения с master-сервера на slave.

Репликация списка наблюдения master -> slave

Для того чтобы реплицировать список наблюдения от master-сервера на slave-экземпляры, отправьте на slave POST-запрос со следующими параметрами в теле:

- **remote_dossier_list**: id исходного списка наблюдения на master-сервере
- **remote_url**: URL master-сервера
- **slave_dossier_list**: id списка наблюдения на slave-сервере, который будет репликой исходного списка наблюдения.

```
POST /sync/dossier-lists/
{remote_dossier_list: 1,
remote_url: "http://172.17.46.14",
slave_dossier_list: 3}
```

Задание времени синхронизации

По умолчанию реплицированные списки наблюдения на slave-серверах синхронизируются с master-сервером каждую ночь в 3:00. Время синхронизации можно изменить следующим образом:

1. Откройте файл конфигурации **findface-security** на master-сервере.

```
sudo vi /etc/ffsecurity/config.py
```

2. Вставьте следующие строки и задайте собственный график репликации.

```
...  
  
SYNC_TIME = {  
    'hour': 3, # 24 hour format  
    'minute': 0,  
}
```

Отмена репликации и синхронизации списка наблюдения

Для того чтобы отменить репликацию и синхронизацию списка наблюдения, отправьте на slave-сервер API-запрос с {id} соответствующего списка на slave-сервере:

```
DELETE /sync/dossier-lists/{id}/
```

Включение функционала в веб-интерейсе

По умолчанию вы можете включать и отключать репликацию списка наблюдения только через API. Для того чтобы данный функционал был также доступен через веб-интерфейс, выполните следующие действия:

1. Откройте файл конфигурации **findface-security** на master-сервере.

```
sudo vi /etc/ffsecurity/config.py
```

2. Включите плагин **ffsecurity_sync**, добавив строку **INSTALLED_APPS.append('ffsecurity_sync')** в раздел плагинов:

```
...  
  
# integration plugins  
...  
INSTALLED_APPS.append('ffsecurity_sync')
```

3. Выполните аналогичные действия на каждом slave-сервере.
4. На каждом сервере выполните перенос основной архитектуры базы данных из FindFace Security в PostgreSQL и перезапустите службу **findface-security**.

```
sudo findface-security migrate  
sudo systemctl restart findface-security.service
```

1.7.4 Пользовательские вкладки, поля и фильтры в досье

Часто необходимо, чтобы досье содержало дополнительные вкладки и поля, по которым можно было осуществлять поисковые запросы.

Для добавления пользовательских вкладок и полей в досье выполните следующие действия:

1. Подготовьте список пользовательских вкладок и полей для добавления в досье.

2. Откройте файл конфигурации `findface-security`.

```
sudo vi /etc/ffsecurity/config.py
```

3. В разделе `FFSECURITY` раскомментируйте секцию `CUSTOM_FIELDS` и измените ее примерное содержимое с учетом следующих правил:

- `'items'`: список полей в досье. Опишите каждое поле следующими параметрами:
 - `'name'`: внутреннее имя поля, `string`.
 - `'label'`: название поля в веб-интерфейсе, `string`.
 - `'display'`: формат отображения (`form` или `list`), `string` или `array`.
 - `'tab'`: вкладка, на которой располагается поле. Если не задана, поле появится на главной странице досье (той, что с фотографией).
 - `'editable'`: редактируемость поля, `boolean`.
 - `'type'`: тип данных поля, `string`. Возможные значения:
 - * `list`: требует задания `items`, дополнительного параметра для списков (см. ниже), ожидает объекты `{id, name}` в словарях;
 - * `valuelist`: ожидает элементы примитивных типов.
 - * `objectlist`: позволяет создавать массивы объектов нужного типа.
 - * `datetime`: примитивный тип данных, отображаемый как список `datetime`.
 - * `date`: примитивный тип данных, отображаемый как выбор даты.
 - * `boolean`: примитивный тип данных, отображаемый как флажок.
 - * `string`: примитивный тип данных `string`.
 - дополнительные параметры для списков (`type=list`, `type=valuelist`):
 - * `multiple`: возможность выбора нескольких элементов в списке, `boolean`.
 - * `items`: словарь, используемый как источник данных для списка.
 - * `allow_create`: возможность добавления новых элементов в список, `boolean`.
 - * `custom_id`: пользовательское поле для `id` (`type=list`).
 - дополнительные параметры для списков объектов (`type=objectlist`).
 - * `object`: объекты, используемые как источник данных для списка объектов.
 - * `simple`: указывает, что поле ожидает данные примитивного типа вместо объектов, например, ожидает строки с телефонными номерами.
- `'filters'`: список фильтров для поиска по пользовательским полям. Параметры:
 - `'name'`: внутреннее имя фильтра,
 - `'label'`: название фильтра в веб-интерфейсе,
 - `'field'`: связанное поле в формате `[имя поля]`.
- `'tabs'`: список вкладок в досье. Первая вкладка в списке соответствует главной странице досье.

```

FFSECURITY = {

...

  # Edit CUSTOM_FIELDS section to customize dossier content.
  # Below is an example for integration FindFace Security with Sigur.
  'CUSTOM_FIELDS': {
    'dossier_meta': {
      'items': [
        {
          'name': 'personid',
          'default': '',
          'label': 'PersonID',
          'display': ['list', 'form'],
          'description': 'Sigur person ID'
        },
        {
          'name': 'firstname',
          'default': '',
          'label': 'First Name',
          'display': ['list', 'form'],
          'description': 'Sigur first name'
        },
        {
          'name': 'lastname',
          'default': '',
          'label': 'Last Name',
          'display': ['list', 'form'],
          'description': 'Sigur last name'
        },
        {
          'name': 'version',
          'default': '',
          'label': 'Version',
          'display': ['list', 'form'],
          'description': 'Sigur photo version'
        }
      ],
      'filters': [
        {
          'name': 'personid',
          'label': 'Sigur person ID filter',
          'field': 'personid'
        }
      ]
    }
  },
},

```

4. Вы увидите, что новая форма досье появилась в веб-интерфейсе.

1.7.5 Пакетная загрузка фотографий через консоль

Помимо веб-интерфейса, для пакетной загрузки фотографий в базу данных досье можно использовать поставляемую вместе с FindFace Security утилиту `findface-security-uploader`. Используйте утилиту, когда требуется загрузить большое количество фотографий (более 10000).

Совет: Для вызова справки `findface-security-uploader` выполните команду:

```
findface-security-uploader --help
```

Выполните следующие действия:

1. Подготовьте CSV- или TSV-файл со списком фотографий и метаданными.

Важно: В качестве источника метаданных файл должен иметь следующий формат: `путь к фотографии | метаданные`.

Для подготовки TSV-файла можно использовать скрипт, аналогичный данному или команду `find`.

Примечание: Как скрипт, так и команда в примерах создают файл `images.tsv` с данными в формате `полный путь к файлу с фотографией | метаданные`. В качестве метаданных будет создана строка с именем файла.

Для запуска скрипта на создание TSV-файла со списком фотографий из указанного каталога (`/home/user/25_celeb/` в примере) выполните следующую команду:

```
python3 tsv_builder.py /home/user/25_celeb/
```

Пример использования команды `find`:

```
find photos/ -type f -iname '*.g' | while read x; do y="${x%.*}"; printf "%s\t%s\n" "$x" "${y#
↪#*/}"; done
```

2. Создайте файл задания (job-файл) из CSV- или TSV-файла, используя метод `add` утилиты. В результате в текущем каталоге будет создан файл `enroll-job.db`.

```
findface-security-uploader add images.tsv
```

Опции `add`:

- `-format`: формат файла, по умолчанию `tsv`,
- `-delimiter`: используемый разделитель, по умолчанию `"\t"` для TSV-файла, `","` для формата CSV.

Примечание: Файл `job` представляет собой sqlite-базу, которая может быть открыта в консоли `sqlite3`.

3. Выполните задание `job`, используя метод `run` утилиты.

```
findface-security-uploader run --dossier-lists 2 --api http://127.0.0.1:80 --user admin --
↪password password
```

Опции `run`:

- `-parallel`: количество потоков загрузки фотографий, по умолчанию 10. Чем больше потоков, тем быстрее будет завершена загрузка, однако также потребуется и большее количество ресурсов.

- **-api:** URL API компонента findface-security, по умолчанию `http://127.0.0.1:80/`.
- **-user:** имя пользователя.
- **-password:** пароль.
- **-dossier-lists:** перечень разделенных запятой id списков наблюдения, в которые нужно добавить фотографии.
- **-failed:** в случае неудачи при обработке job-файла исправьте ошибку и повторите попытку с данной опцией.

1.7.6 Дедупликация событий

В этом разделе:

- *Включение дедупликации*
- *Алгоритм работы дедупликации*

Рассмотрите возможность включения дедупликации, чтобы исключить дублирование событий распознавания лиц в пределах одной группы камер.

Включение дедупликации

Для того чтобы активировать функцию дедупликации, выполните следующие действия:

1. Для каждой камеры в группе включите буферный режим детектирования лиц. См. подробнее *Добавление камер*.
2. Перейдите на вкладку *Настройки*. Выберите *Группы камер*.
3. Откройте настройки группы камер.
4. Поставьте флажок *Дедуплицировать события* и задайте в секундах интервал дедупликации.

Алгоритм работы дедупликации

Алгоритм дедупликации работает следующим образом:

1. В буферном режиме за один сеанс отслеживания на камере сервер получает одно, наилучшее, изображение лица (сеанс отслеживания продолжается до тех пор, пока лицо не исчезнет из поля зрения камеры).
2. Если в пределах одной группы камер произошло несколько сеансов отслеживания на одной или нескольких камерах в течение указанного интервала дедупликации, FindFace Security обработает полученные изображения лиц следующим образом:
 - Если в течение предшествующего периода, равного интервалу дедупликации, есть совпадение с досье, FindFace Security удаляет вновь полученное изображение. В противном случае изображение сохраняется в базе данных.
 - При выполнении дедупликации не совпавших с досье лиц FindFace Security учитывает как сходство между лицами на различных изображениях, так и качество лиц. FindFace Security удаляет все изображения похожих лиц в пределах интервала дедупликации, если они более

низкого качества, чем первое в данном интервале. Если новое лицо более высокого качества, оно сохраняется. Это гарантирует, что система дедуплицирует события, не пропуская высококачественные лица, необходимые для видеоаналитики.

1.7.7 Распознавание живых лиц в реальном времени (Liveness)

Примечание: *Liveness-детектор* на CPU работает гораздо медленнее, чем на GPU.

Для обнаружения фальшивых лиц и предотвращения фото-атак используйте интегрированную антиспуфинговую систему, отличающую живые лица от их изображений. Алгоритм анализирует несколько последовательных кадров, регистрируя изменения в мимике и текстуре кожи, и благодаря этому определяет, является ли лицо перед камерой живым или фальшивым. Это исключает возможность мошенничества с использованием изображения лица на бумаге или экране мобильного устройства.

Liveness-детектор оценивает живость лица с определенным уровнем достоверности и возвращает оценку достоверности вместе с бинарным результатом **Живой человек/изображение**, в зависимости от установленного порога достоверности.

В этом разделе:

- Включение Liveness-детектора
- Настройка порога Liveness
- Информация о живом лице в веб-интерфейсе

Включение Liveness-детектора

Для включения Liveness-детектора выполните следующие действия:

1. Откройте файл конфигурации `/etc/findface-video-worker-gpu.ini` (`/etc/findface-video-worker-cpu.ini`). В разделе `liveness` укажите путь к модели нейронной сети (`fnk`) и нормализатору (`norm`), которые будут использоваться в liveness-детекторе.

```
sudo vi /etc/findface-video-worker-gpu.ini

#-----
[liveness]
#-----
## path to liveness fnk
## type:string env:CFG_LIVENESS_FNK longopt:--liveness-fnk
fnk = /usr/share/findface-data/models/faceattr/liveness.v3.gpu.fnk

## path to normalization for liveness
## type:string env:CFG_LIVENESS_NORM longopt:--liveness-norm
norm = /usr/share/findface-data/models/facenorm/crop2x.v2_maxsize400.gpu.fnk
```

```
sudo vi /etc/findface-video-worker-cpu.ini
```

```
#-----
```

(continues on next page)

(продолжение с предыдущей страницы)

```
[liveness]
#-----
## path to liveness fnk
## type:string env:CFG_LIVENESS_FNK longopt:--liveness-fnk
fnk = /usr/share/findface-data/models/faceattr/liveness.v3.cpu.fnk

## path to normalization for liveness
## type:string env:CFG_LIVENESS_NORM longopt:--liveness-norm
norm = /usr/share/findface-data/models/facenorm/crop2x.v2_maxsize400.cpu.fnk
```

2. Перезапустите findface-video-worker.

```
sudo systemctl restart findface-video-worker-gpu
sudo systemctl restart findface-video-worker-cpu
```

Настройка порога Liveness

При необходимости вы можете настроить пороговое значение Liveness в файле конфигурации `/etc/ffsecurity/config.py`. Liveness-детектор оценивает “живость” лица с определенной достоверностью. В зависимости от порогового значения достоверности, он возвращает бинарный результат **Живой человек** или **Изображение**.

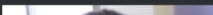
Примечание: Значение по умолчанию является оптимальным. Перед изменением порога проконсультируйтесь у наших специалистов по адресу support@ntechlab.com.

```
sudo vi /etc/ffsecurity/config.py

'LIVENESS_THRESHOLD': 0.75,
```

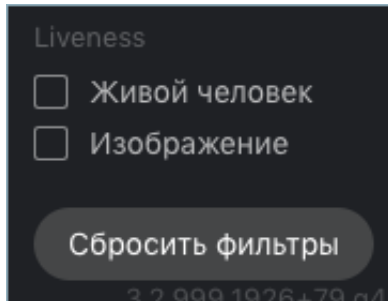
Информация о живом лице в веб-интерфейсе

После настройки Liveness-детектора для каждого события будет отображаться оценка liveness обнаруженного лица.

4178559 4273888 33633		No matches Unmatched	fake (liveness): 0.00	2019-04-30 17:29:59 (Camera not...)
4178559 4263150 90320		No matches Unmatched	real (liveness): 0.99	2019-04-30 17:30:00 (Camera not...)
4178559		No matches		2019-04-30

Примечание: Liveness может принимать значение `null`, если детектор живых лиц отключен или на предоставленном изображении невозможно достоверно оценить Liveness.

Используйте фильтр *Liveness* для просмотра событий только с живыми людьми или только с изображениями, если имели место спуфинговые атаки.



1.7.8 Распознавание атрибутов лица

FindFace Security позволяет распознавать в реальном времени такие атрибуты лица, как пол, возраст, эмоции, очки, борода и медицинская маска. Данный функционал доступен как на GPU-, так и на CPU-видеодетекторе лиц.

В этом разделе:

- *Включение распознавания атрибутов лица*
- *Отображение результатов распознавания атрибутов в событиях*
- *Атрибуты лиц в событиях*

Включение распознавания атрибутов лица

Важно: На данном шаге включается распознавание атрибутов лица на уровне HTTP API.

Для того чтобы включить автоматическое распознавание атрибутов лица, откройте файл конфигурации `/etc/findface-extraction-api.ini` и включите соответствующие модели: пол, возраст, эмоции, очки, борода и медицинская маска. Удостоверьтесь, что для каждой модели вы указали правильный тип ускорения CPU или GPU: он должен совпадать с типом ускорения `findface-extraction-api`. Обратите внимание, что `findface-extraction-api` на CPU может работать только с CPU-моделями, в то время как `findface-extraction-api` на GPU поддерживает как GPU-, так и CPU-модели.

```
sudo vi /etc/findface-extraction-api.ini

models:
  age: faceattr/age.v1.cpu.fnk
  emotions: faceattr/emotions.v1.cpu.fnk
  face: face/ifruit_320.cpu.fnk
```

(continues on next page)

(продолжение с предыдущей страницы)

```
gender: faceattr/gender.v2.cpu.fnk
beard: faceattr/beard.v0.cpu.fnk
glasses3: faceattr/glasses3.v0.cpu.fnk
medmask3: faceattr/medmask3.v1.cpu.fnk
```

Доступны следующие модели:

Примечание: Вы можете найти модели для распознавания атрибутов лица в каталоге `/usr/share/findface-data/models/faceattr/`.

```
ls /usr/share/findface-data/models/faceattr/
age.v1.cpu.fnk age.v1.gpu.fnk beard.v0.cpu.fnk beard.v0.gpu.fnk emotions.v1.cpu.fnk emotions.
v1.gpu.fnk gender.v2.cpu.fnk gender.v2.gpu.fnk glasses3.v0.cpu.fnk glasses3.v0.gpu.fnk
medmask3.v1.cpu.fnk medmask3.v1.gpu.fnk liveness.v3.gpu.fnk
```

Атрибут лица	Ускоре- ние	Параметр в файле конфигурации
биометрия лица	CPU	face: face/ifruit_320.cpu.fnk face: face/ifruit_160.cpu.fnk
	GPU	face: face/ifruit_320.gpu.fnk face: face/ifruit_160.gpu.fnk
возраст	CPU	age: faceattr/age.v1.cpu.fnk
	GPU	age: faceattr/age.v1.gpu.fnk
пол	CPU	gender: faceattr/gender.v2.cpu.fnk
	GPU	gender: faceattr/gender.v2.gpu.fnk
эмоции	CPU	emotions: faceattr/emotions.v1.cpu.fnk
	GPU	emotions: faceattr/emotions.v1.gpu.fnk
очки	CPU	glasses3: faceattr/glasses3.v0.cpu.fnk
	GPU	glasses3: faceattr/glasses3.v0.gpu.fnk
борода	CPU	beard: faceattr/beard.v0.cpu.fnk
	GPU	beard: faceattr/beard.v0.gpu.fnk
медицинская мас- ка	CPU	medmask3: faceattr/medmask3.v1.cpu.fnk
	GPU	medmask3: faceattr/medmask3.v1.gpu.fnk

Совет: Для того чтобы отключить модель распознавания, передайте в соответствующий параметр пустое значение. Не удаляйте сам параметр, поскольку в этом случае будет выполняться поиск модели по умолчанию.

```
models:
  gender: ""
  age: ""
  emotions: ""
```

Перезапустите `findface-extraction-api`.

```
sudo systemctl restart findface-extraction-api
```

После включения моделей обязательно *настройте* отображение атрибутов в веб-интерфейсе.

Отображение результатов распознавания атрибутов в событиях

Для отображения результатов распознавания атрибутов лица в списке событий, добавьте следующую строку в секцию FFSECURITY: “ ‘EVENTS_FEATURES’: [‘gender’, ‘age’, ‘emotions’, ‘beard’, ‘glasses’, ‘medmask’]”, в зависимости от того, какие модели были включены.

Предупреждение: Данная строка должна быть расположена между строками SF_API_ADDRESS и LIVENESS_THRESHOLD, как показано в примере ниже.

```
sudo vi /etc/ffsecurity/config.py

...
FFSECURITY = {
...
'SF_API_ADDRESS': 'http://127.0.0.1:18411',
'EVENTS_FEATURES': ['gender', 'age', 'emotions', 'beard', 'glasses', 'medmask'],
'LIVENESS_THRESHOLD': 0.75,
'BEARD_THRESHOLD': 0.7,
}
```

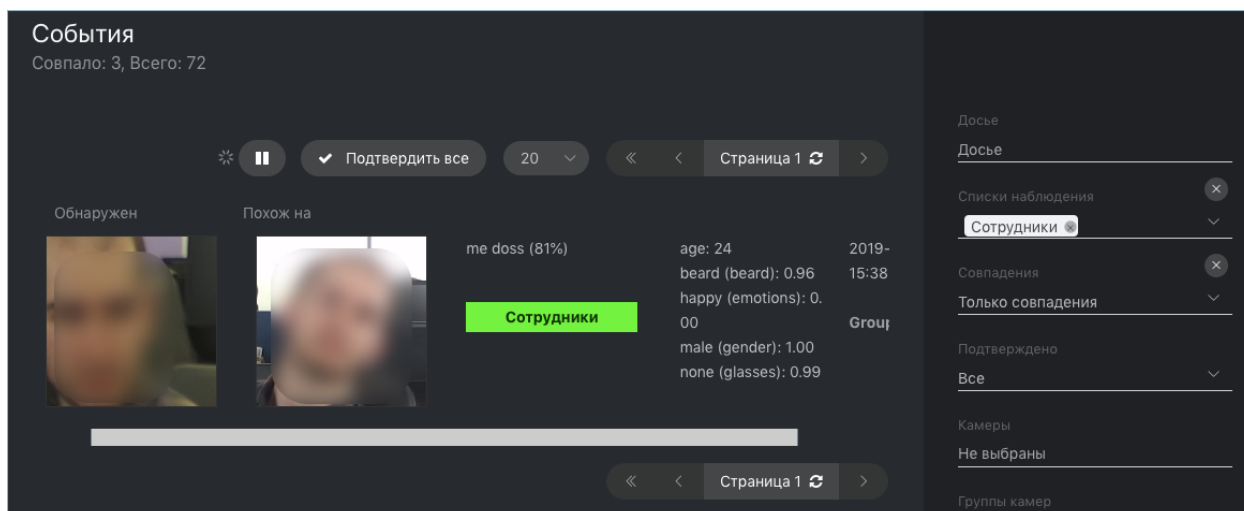
Перезапустите findface-security.

```
sudo systemctl restart findface-security
```

Атрибуты лиц в событиях

После настройки распознавания атрибутов лиц результат распознавания будет выводиться для каждого обнаруженного лица в следующем формате:

Атрибут лица	Формат результата	Пример
Возраст	Атрибут: возраст : количество лет	возраст: 33
Пол	Результат: мужской/женский (атрибут: пол): уверенность алгоритма в результате	женский (пол): 0.95
Эмоции	Результат: злость/отвращение/страх/счастье/грусть/удивление (атрибут: эмоции): уверенность алгоритма в результате	счастье (эмоции): 0.99
Очки	Результат: медицинские/солнечные/нет (атрибут: очки): уверенность алгоритма в результате	нет (очки): 0.87
Борода	Результат: борода/нет (атрибут: борода): уверенность алгоритма в результате	нет (борода): 0.91
Медицинская маска	Результат: нет/есть (атрибут: медицинская маска): уверенность алгоритма в результате	нет (медицинская маска): 0.93



При необходимости выполните фильтрацию событий по атрибутам лица.

1.7.9 Детектирование силуэтов

Для сбора *статистики* человеческих силуэтов в реальном времени необходимо предварительно включить их детектирование.

В этом разделе:

- Включение детектирования силуэтов

Включение детектирования силуэтов

Для включения детектирования человеческих силуэтов выполните следующие действия:

1. Откройте файл конфигурации `/etc/findface-extraction-api` и добавьте новый детектор `edie` (`edie_rc2.cpu.fnk/edie_rc2.gpu.fnk`) с соответствующими настройками, как показано в примере ниже.

```
sudo vi /etc/findface-extraction-api.ini

detectors:
  max_batch_size: 1
  instances: 1
  models:
    cheetah:
      model: facedet/cheetah.cpu.fnk
      options:
        min_object_size: 32
        resolutions:
          - 256x256
          - 384x384
          - 512x512
          - 768x768
```

(continues on next page)

(продолжение с предыдущей страницы)

```

- 1024x1024
- 1536x1536
- 2048x2048
edie:
  model: pedet/edie_rc2.cpu.fnk
  options:
    min_object_size: 16
    resolutions:
      - 1280x720
      - 1920x1080

```

Примечание: Значения опций детектора `edie`, приведенные в примере, считаются оптимальными. Вы можете настроить их в зависимости от сцен наблюдения используемых камер. Свяжитесь с нашими специалистами по адресу support@ntechlab.com, если у вас есть вопросы.

2. Перезапустите `findface-extraction-api`.

```
sudo systemctl restart findface-extraction-api
```

1.7.10 Использование нескольких видеокарт

Если на физическом сервере установлено несколько видеокарт, вы можете создать дополнительные экземпляры `findface-extract-api-gpu` или `findface-video-worker-gpu` и распределить их по одному экземпляру на карту.

В этом разделе:

- *Распределение экземпляров `findface-extraction-api-gpu` по видеокартам*
- *Привязка `findface-video-worker-gpu` к дополнительной видеокарте*

Распределение экземпляров `findface-extraction-api-gpu` по видеокартам

Для распределения экземпляров `findface-extraction-api-gpu` по нескольким видеокартам выполните следующие действия:

1. Остановите исходный сервис `findface-extraction-api-gpu`.

```
sudo service findface-extraction-api stop
```

2. Создайте несколько копий файла конфигурации `findface-extract-api-gpu`, в зависимости от того, какое количество видеокарт будет использоваться для извлечения биометрических образцов. Добавьте соответствующие номера устройств GPU к именам новых файлов конфигурации, как показано в примере ниже (устройства GPU №0 и №6).

```

/etc/findface-extraction-api@0.ini
/etc/findface-extraction-api@6.ini

```

3. Откройте новые файлы конфигурации. Укажите номера устройств GPU, а также номера слушающих портов.

```
sudo vi /etc/findface-extraction-api@0.ini

listen: 127.0.0.1:18666
...

gpu_device: 0
...
```

```
sudo vi /etc/findface-extraction-api@6.ini

listen: 127.0.0.1:18667
...

gpu_device: 6
...
```

4. Запустите новые сервисы.

```
sudo service findface-extraction-api@0 start
sudo service findface-extraction-api@6 start
```

Привязка findface-video-worker-gpu к дополнительной видеокарте

Для создания дополнительного экземпляра `findface-video-worker-gpu` и его привязки к свободной видеокарте выполните следующие действия:

1. Отобразите статус исходного сервиса `findface-video-worker-gpu`, выполнив команду:

```
sudo systemctl status findface-video-worker-gpu.service
```

2. Найдите полный путь к сервису в строке `Loaded: loaded (/lib/systemd/system/findface-video-worker-gpu.service; enabled; vendor preset: enabled)`. В приведенном примере это `findface-video-worker-gpu.service` (имя может быть другим). Создайте копию сервиса под новым именем.

```
sudo cp /lib/systemd/system/findface-video-worker-gpu.service /lib/systemd/system/findface-
↪video-worker-gpu2.service`
```

3. Таким же образом создайте под новым именем копию исходного файла конфигурации.

```
sudo cp /etc/findface-video-worker-gpu.ini /etc/findface-video-worker-gpu2.ini
```

4. Откройте только что созданный файл конфигурации и актуализируйте номер используемого GPU-устройства.

```
sudo vim /etc/findface-video-worker-gpu2.ini

## cuda device number
device_number = 1
```

5. Откройте новый сервис и укажите только что созданный файл конфигурации.

```
sudo vim /lib/systemd/system/findface-video-worker-gpu2.service

ExecStart=/usr/bin/findface-video-worker-gpu --config /etc/findface-video-worker-gpu2.ini
```

6. Для применения изменений перезагрузите демон `systemd`.

```
sudo systemctl daemon-reload
```

7. Добавьте новый сервис в автозагрузку.

```
sudo systemctl enable findface-video-worker-gpu2.service

Created symlink from /etc/systemd/system/multi-user.target.wants/findface-video-worker-gpu2.
↪service to /lib/systemd/system/findface-video-worker-gpu2.service
```

8. Запустите новый сервис.

```
sudo systemctl start findface-video-worker-gpu2.service
```

9. Проверьте статус обоих сервисов `findface-video-worker-gpu`.

```
sudo systemctl status findface-video-worker-* | grep -i 'Active:' -B 3

findface-video-worker-gpu2.service - findface-video-worker-gpu daemon
  Loaded: loaded (/lib/systemd/system/findface-video-worker-gpu2.service; enabled; vendor
  ↪preset: enabled)
  Active: active (running) since Thu 2019-07-18 10:32:02 MSK; 1min 11s ago
...

findface-video-worker-gpu.service - findface-video-worker-gpu daemon
  Loaded: loaded (/lib/systemd/system/findface-video-worker-gpu.service; enabled; vendor
  ↪preset: enabled)
  Active: active (running) since Mon 2019-07-15 15:18:33 MSK; 2 days ago
```

1.7.11 Прямые API-запросы к базе данных Tarantool

Вы можете использовать HTTP API для извлечения напрямую данных из базы данных Tarantool.

В этом разделе:

- *Общие сведения*
- *Добавление лица*
- *Удаление лица*
- *Поиск лица*
- *Редактирование метаданных и/или вектора признаков*
- *Получение списка галерей*
- *Получение информации о галерее*

- *Создание галереи*
- *Удаление галереи*

Общие сведения

API-запросы к базе данных Tarantool нужно отправлять по адресу `http://<tarantool_host_ip:port>`.

Совет: Порт для API-запросов можно узнать в разделе `FindFace.start` файла конфигурации Tarantool:

```
cat /etc/tarantool/instances.enabled/FindFace.lua

##8001:
FindFace.start("127.0.0.1", 8001)
```

Примечание: В случае если FindFace Security развернут на одиночном физическом сервере, база данных Tarantool по умолчанию будет доступна только локально (127.0.0.1). Если необходимо открыть доступ к базе данных Tarantool с удаленного сервера, *внесите изменения* в файл конфигурации `findface-tarantool-server`.

API-запросы к Tarantool могут содержать следующие параметры в сегментах пути:

- `:ver`: версия API (v2 на данный момент).
- `:name`: имя галереи.

Совет: Для получения списка имен галерей на шарде введите следующую команду в адресном поле браузера:

```
http://<tarantool_host_ip:shard_port>/stat/list/1/99
```

Та же самая команда в консоли:

```
curl <tarantool_host_ip:shard_port>/stat/list/1/99 \q jq
```

Вы также можете получить список имен галерей, отправив в Tarantool прямой запрос:

```
echo 'box.space.galleries:select()' | tarantoolctl connect <tarantool_host_ip:shard_port>
```

Имейте в виду, что при значительном количестве шардов в системе произвольно выбранный шард может не включать в себя все существующие галереи. В этом случае отобразите список галерей на нескольких шардах.

Добавление лица

```
POST /:ver/faces/add/:name
```


Параметры в теле:

Массив лиц в представлении JSON со следующими полями:

- "id": id лица в галерее, uint64_t,
- "facen": необработанный вектор признаков, base64,
- "meta": метаданные лица, словарь.

Возвращает:

- HTTP 200 и пустое тело в случае успеха.
- HTTP 404 с описанием ошибки, если галерея с заданным именем не существует.
- HTTP с отличным от 200 статусом и описанием ошибки в теле в случае неудачи.

Пример**Запрос**

```
curl -D - -s 'http://localhost:8001/v2/faces/add/testgal' --data '[
  {
    "id": 9223372036854776000,
    "facen": "qgI3vZRv/z...Np09MdHavW1WuT0=",
    "meta": {
      "cam_id": "223900",
      "person_name": "Mary Ostin",
    }
  }
]
```

Ответ

```
HTTP/1.1 200 Ok
Content-length: 1234
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive
```

Удаление лица

```
POST /v2/faces/delete/:name
```

Параметры в теле:

Массив в представлении JSON из списка id лиц, подлежащих удалению.

Возвращает:

- HTTP 200 и пустое тело в случае успеха.
- HTTP 404, если лицо с заданным id не найдено в галерее.
- HTTP с отличным от 200 статусом и описание ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -s 'http://localhost:8001/v2/faces/delete/testgal' --data '[1, 4, 922, 3]'
```

Ответ

```
HTTP/1.1 200 Ok
Content-length: 111
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive
```

Поиск лица

```
POST /v2/faces/search/:name
```

Параметры в теле:

Поисковый запрос в представлении JSON со следующими полями:

- **limit**: максимальное количество лиц в ответе.
- **sort**: включает сортировку по следующим параметрам: **id**: по возрастанию id, **-id** по убыванию id, **-score**: по убыванию степени схожести (если поиск выполняется по схожим векторам признаков).
- **filter** (фильтры):
 - **facen**: опциональный фильтр по схожести вектора признаков. Передайте словарь со следующими полями: **data**: вектор признаков в формате base64; **score**: диапазон качества лица, поддерживаются только запросы с правой границей 1 (максимальное качество).
 - **id** и **meta/<meta_key>**: фильтры по пользовательскому id лиц и содержимому поля **meta**. Для задания фильтра используются следующие операторы:
 - * **range**: диапазон значений, только для числовых полей.
 - * **set**: набор значений, одно из которых должно присутствовать в id или метаданных, для числовых и строковых полей.
 - * **subset**: набор значений, каждое из которых должно присутствовать в id или метаданных, для числовых и строковых полей.
 - * **like**: аналогично **like** в SQL-запросах: поддерживаются только 'aa%' или 'aa%' или '%aa%'. Только для полей **string** и **set[string]**. При использовании **set[string]** фильтр вернет результат, если хотя бы одно из значений прошло проверку.

* `ilike` (только для полей `string` и `set[string]`): аналогично `like`, но без учета регистра.

Возвращает:

- В случае успеха массив JSON с лицами. Значение в заголовке `X-search-stat` показывает, был ли использован быстрый индекс для поиска: `with_index` или `without_index`.

Примечание: В API v2 быстрый индекс не используется.

- HTTP с отличным от 200 статусом и описание ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -s 'http://localhost:8001/v2/testgal/search' --data '{
  "limit": 2,
  "sort": {
    "score": -1
  },
  "filter": {
    "facen": {
      "data": "qgI3vZRv/z0BQTk9rcir0yZrNp09MdHavW1WuT0=",
      "score": [0.75, 1]
    },
    "id": {
      "range": [922337203685400000, 9223372036854999000]
    },
    "meta": {
      "person_id": {
        "range": [444, 999]
      },
      "cam_id": {
        "set": ["12767", "8632", "23989"]
      }
    }
  }
}'
```

Ответ

```
HTTP/1.1 200 Ok
Content-length: 1234
X-search-stat: without_index
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive

{
  "results": [
    {
      "facen": " qgI3vZRv/z0BQTk9rcir0yZrNp09MdHavW1WuT0=",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
"meta": {
  "timestamp": 0,
  "photo_hash": "",
  "person_id": 777,
  "cam_id": "8632"
},
"score": 0.9964,
"id": 9223372036854776000
}
]
```

Редактирование метаданных и/или вектора признаков

```
POST /v2/faces/update/:name
```

Параметры в теле:

Массив лиц в представлении JSON со следующими полями:

- "id": id лица, uint64_t,
- "facen": (опционально) новый вектор признаков, base64. Если параметр отсутствует или null, поле в базе данных не обновляется.
- "meta": словарь, в котором передаются новые метаданные. Если поле meta отсутствует или null, оно не обновляется в базе данных.

Возвращает:

- HTTP 200 и словарь со всеми параметрами лица, в том числе неизменными, в случае успеха.
- HTTP 404 с описанием ошибки, если лица с таким id не существует.
- HTTP с отличным от 200 статусом и описанием ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -s 'http://localhost:8001/v2/faces/update/sandbox' --data ' [{"id":1,"facen":null,"meta":{"m:timestamp":1848}}] '
```

Ответ

```
HTTP/1.1 200 Ok
Content-length: 151
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive
```

(continues on next page)

(продолжение с предыдущей страницы)

```
{
  "meta": {
    "m:timestamp": 1848,
    "normalized_id": "1_b9pkrf00mjt6h1vmq1kg.png",
    "m:cam_id": "a9f7a973-f07e-469d-a3bd-41ddd510b26f",
    "feat": {
      "score": 0.123
    },
    "id": 1,
    ...
  }
}
```

Получение списка галерей

```
POST /v2/galleries/list
```

Возвращает:

Массив с галереями, для каждой из которой возвращается имя (**name**) и число лиц (**faces**).

Пример

Запрос

```
curl -D - -s -X POST http://localhost:8001/v2/galleries/list
```

Ответ

```
HTTP/1.1 200 Ok
Content-length: 42
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive

{
  "results": [
    {
      "name": "testgal",
      "faces": 2
    }
  ]
}
```

Получение информации о галерее

```
POST /v2/galleries/get/:name
```

Возвращает:

- HTTP 200 и словарь с параметрами галереи в случае успеха.
- HTTP 404 с описанием ошибки, если галереи с таким именем не существует.
- HTTP с отличным от 200 статусом и описанием ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -s -X POST http://localhost:8001/v2/galleries/get/testgal
```

```
HTTP/1.1 200 Ok
Content-length: 11
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive

{"faces":2}
```

Создание галереи

```
POST /v2/galleries/add/:name
```

Возвращает:

- HTTP 200 и пустое тело в случае успеха.
- HTTP с отличным от 200 статусом и описание ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -X POST -s 'http://localhost:8001/v2/galleries/add/123'
```

Ответ

```
HTTP/1.1 409 Conflict
Content-length: 57
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive

{"error":{"message":"gallery already exists","code":409}}
```

Удаление галереи

```
POST /v2/galleries/delete/:name
```

Возвращает:

- HTTP 200 и пустое тело в случае успеха.
- HTTP с отличным от 200 статусом и описание ошибки в теле в случае неудачи.

Пример

Запрос

```
curl -D - -X POST -s 'http://localhost:8001/v2/galleries/delete/123'
```

Ответ

```
HTTP/1.1 204 No content
Content-length: 0
Server: Tarantool http (tarantool v1.7.3-673-g23cc4dc)
Connection: keep-alive
```

1.7.12 Распознавание персон

FindFace Security может автоматически распознавать лица одного и того же человека (персоны) и кластеризовать их. В этом случае в системных базах данных появляется сущность «person event» (событие появления персоны), связанная со всеми *эпизодами*, в которых фигурировало лицо той или иной персоны. Работа с галереей персон производится на вкладке *Персоны*.

Важно: По умолчанию кластеризация персон отключена. *Включите и настройте ее* через файл конфигурации `findface-security`.

В этом разделе:

- *Методы кластеризации*
- *Включение и настройка кластеризации персон*
- *Работа с галереей персон*

Методы кластеризации

FindFace Security использует следующие методы группировки лиц, принадлежащих одному и тому же человеку:

- Динамическая кластеризация. Происходит на лету после закрытия эпизода. Результат динамической кластеризации отображается в реальном времени на вкладке *Персоны*.

Примечание: Технические детали заключаются в следующем. Не каждый эпизод является подходящим: число событий в нем должно быть равно или больше значения параметра `PERSON_EVENT_MIN_EPISODE_EVENTS` (задается через файл конфигурации `findface-security`). Если эпизод соответствует данному требованию, система выбирает событие наилучшего качества и выполняет следующие операции:

- Создает новую сущность `PersonEvent` в основной базе данных PostgreSQL. Сущность содержит метаданные события, биометрический образец и миниатюру лица, а также ссылку на родительский эпизод.
- Выполняет поиск похожего центроида лица в галерее `person_events` биометрической базы данных Tarantool. Центроид лица — это виртуальный биометрический образец, усредненный по всем обнаруженным лицам персоны. Если похожий центроид найден, система обновляет его, используя биометрический образец из нового события. В противном случае создается новый центроид.

- Кластеризация по расписанию. Мы рекомендуем планировать ее на поздние ночные часы, так как процесс занимает много времени и ресурсов.

Примечание: Расписание задается в формате RRULE в параметре `PERSONS_CLUSTERIZATION_SCHEDULE` файла конфигурации `findface-security`. В остальном техническая реализация напоминает динамический метод. Однако качество центроида лица при кластеризации по расписанию лучше, поскольку в этом случае центроиды усредняются по большому массиву накопленных биометрических образцов.

Важно: Кластеризация по расписанию полностью перезаписывает содержимое галереи персон, включая `id`.

Включение и настройка кластеризации персон

По умолчанию кластеризация персон отключена. Для включения данной функции откройте файл конфигурации `findface-security` и внесите следующие изменения в раздел `SERVICES`:

```
sudo vi /etc/ffsecurity/config.py

...
SERVICES = {
    "ffsecurity": {
        ...
        "persons": True,
    }
    ...
}
```

В результате в веб-интерфейсе FindFace Security появится вкладка *Персоны*.

В том же файле конфигурации можно изменить следующие параметры:

- `PERSON_EVENT_MIN_QUALITY`: минимальное качество лиц, используемых в кластеризации персон.
- `PERSON_EVENT_MIN_EPISODE_EVENTS`: минимальное количество событий в эпизодах, участвующих в кластеризации персон.
- `PERSONS_CONFIDENCE_THRESHOLD`: пороговая степень схожести для принятия решения о принадлежности лица персоне.

Предупреждение: Перед изменением данного параметра проконсультируйтесь с нашими специалистами по адресу support@ntechlab.com.

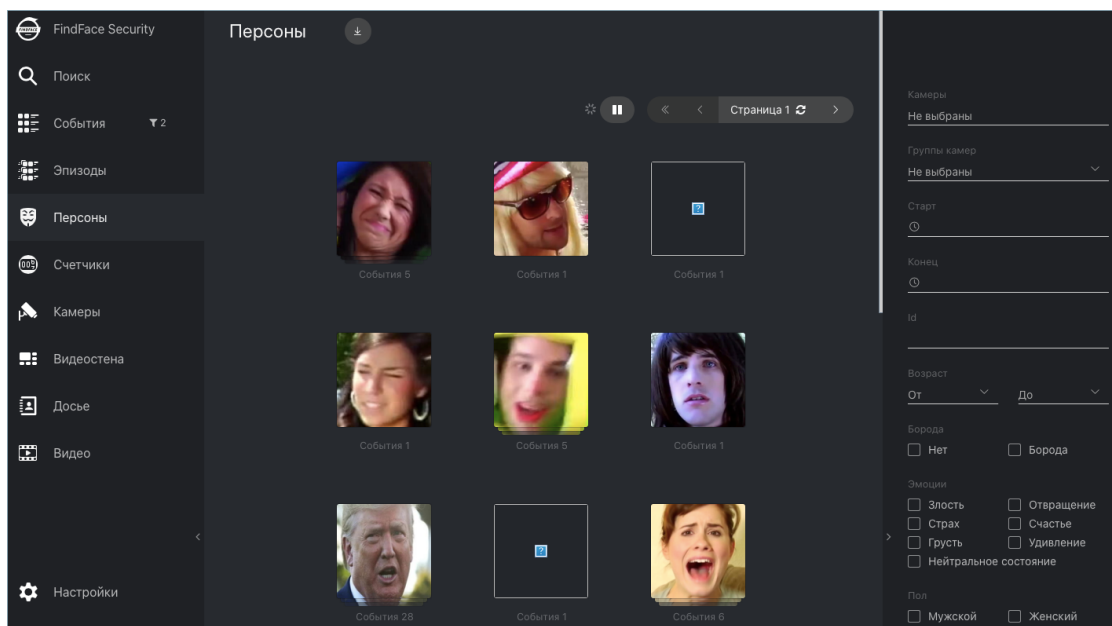
- **PERSONS_CLUSTERIZATION_SCHEDULE:** рекуррентное правило (RRULE), задающее расписание плановой кластеризации персон.

Совет: См. калькулятор RRULE.

```
# -- Persons configuration --
# rrule (recurrence rule) for scheduling persons clusterization
'PERSONS_CLUSTERIZATION_SCHEDULE': 'RRULE:FREQ=DAILY;INTERVAL=1;WKST=MO;BYHOUR=0;BYMINUTE=0',
# face to person matching confidence threshold
'PERSONS_CONFIDENCE_THRESHOLD': 0.739,
# minimum face quality for handle face in person service
'PERSON_EVENT_MIN_QUALITY': -2,
```

Работа с галереей персон

Для просмотра галереи персон перейдите на вкладку *Персоны*.

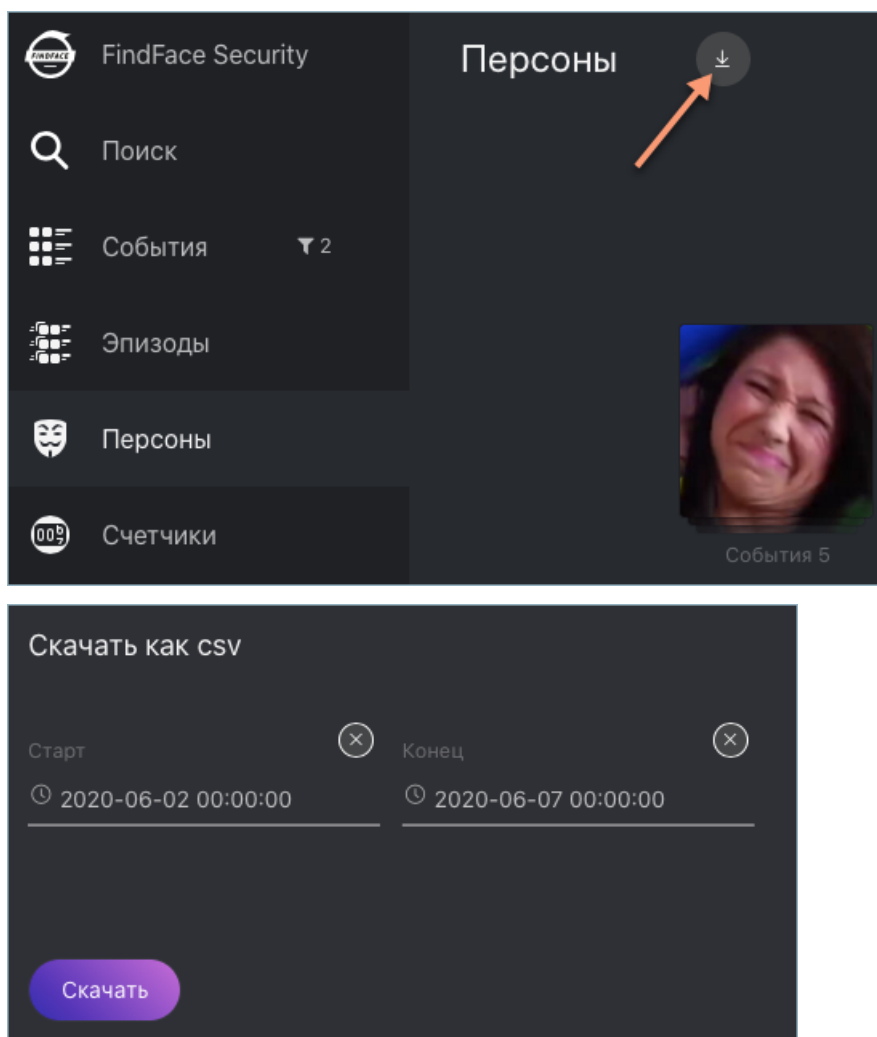


Для работы с галереей персон используйте следующие фильтры:

- Камеры
- Группы камер
- Период времени
- Id персоны
- Атрибуты лица (если используются)
- Liveness (если используется)

Вы можете скачать галерею персон в CSV-формате. Для этого нажмите





См.также:

- *Файл конфигурации findface-security*
- *Вебхуки*

1.7.13 Аутентификация по сертификатам КриптоПро

По умолчанию вход пользователей в систему осуществляется по логину и паролю. При необходимости можно включить аутентификацию по сертификатам КриптоПро.

В этом разделе:

- *Настройка аутентификации по сертификатам КриптоПро*
- *Загрузка сертификата пользователя*

Настройка аутентификации по сертификатам КриптоПро

Для настройки аутентификации по сертификатам КриптоПро выполните следующие действия:

1. Загрузите на сервер FindFace Security архивы КриптоПро CSP 4.0 для Linux (x64, deb) и КриптоПро ЭЦП SDK 2.0 Linux x64.
2. Распакуйте загруженные архивы.
3. Из директории `/opt/ffsecurity/lib/python3.5/site-packages/ffsecurity_cproauth` запустите скрипт `build.sh`.

```
chmod +x build.sh
sudo ./build.sh
```

По требованию скрипта установите дополнительные deb-пакеты из распакованных архивов.

```
Please install lsb-cprocsp-devel from CryptoPro CSP ( https://www.cryptopro.ru/
↳products/csp/downloads )

apt install lsb-cprocsp-devel
...
```

4. Откройте файл конфигурации `findface-security`.

```
sudo vi /etc/ffsecurity/config.py
```

5. Добавьте в него следующие настройки:

```
...
UVICORN_SETTINGS = {
...
'proxy_headers': True
}
...
INSTALLED_APPS.append('ffsecurity_cproauth')
#REST_FRAMEWORK['DEFAULT_AUTHENTICATION_CLASSES'] = ['ffsecurity.auth.TokenAuthentication',
↳'ffsecurity_cproauth.auth.CryptoProOrTokenAuthentication']
```

6. Перезапустите `findface-security`.

```
sudo systemctl restart findface-security.service
```

На этом настройка аутентификации по сертификатам КриптоПро будет завершена. Можно приступить к загрузке сертификатов в систему.

Загрузка сертификата пользователя

Для того чтобы загрузить сертификат пользователя в систему, выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Пользователи*.
2. Откройте карточку пользователя.
3. Нажмите *Добавить сертификат*.

1.8 Обслуживание и устранение неисправностей

1.8.1 Обновление FindFace Security до 4.2

Предупреждение: Начиная с версии 4.1, FindFace Security использует новую версию Tarantool. Биометрическая база данных из предыдущих версий FindFace Security (4.0 и более ранних) НЕКОМПАТИБИЛЬНА с FindFace Security 4.2. При обновлении продукта до 4.2 обязательно используйте функцию *резервного копирования / восстановления* (см. полный алгоритм ниже).

Для обновления FindFace Security с любой предыдущей версии до 4.2 выполните следующие действия:

1. Откройте файл конфигурации `findface-security`. Сохраните для последующего использования значения следующих параметров: `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN`, `ROUTER_URL`.

```
sudo vi /etc/ffsecurity/config.py

EXTERNAL_ADDRESS = "http://172.20.77.58"

...
# use pwgen -sncy 50 1/tr "" "." to generate your own unique key
SECRET_KEY = 'c8b533847bbf7142102de1349d33a1f6'

FFSECURITY = {
    'VIDEO_DETECTOR_TOKEN': '381b0f4a20495227d04185ab02f5085f',
    ...
    'ROUTER_URL': 'http://172.20.77.58',
    ...
}
```

2. Остановите сервис `findface-security`.

```
sudo systemctl stop findface-security*
```

3. Создайте резервную копию биометрической базы данных на основе Tarantool в любой выбранной директории, например, `/etc/findface_dump`.

Совет: Подробнее см. *Резервное копирование и восстановление хранилищ данных*.

```
mkdir -p /etc/findface_dump
cd /etc/findface_dump
sudo findface-storage-api-dump -config /etc/findface-sf-api.ini
```

4. Установите apt-репозиторий с новой версией FindFace Security, используя консольный инсталлятор согласно инструкции в *этом разделе*.
5. Установите пакет `pgbouncer` следующим образом:

```
sudo apt update
sudo apt install -y pgbouncer
```

6. Предоставьте надежный пароль пользователю `ntech` (9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3 в примере ниже). Внесите учетные данные в список пользователей `pgbouncer`.

```
echo "ntech" "9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3" | sudo tee -a /etc/pgbouncer/userlist.txt
```

7. Настройте pgbouncer. Откройте файл `/etc/pgbouncer/pgbouncer.ini` и добавьте базу данных `ffsecurity` в раздел `databases`. Настройте другие параметры, как показано в примере ниже.

```
sudo vi /etc/pgbouncer/pgbouncer.ini
[databases]
ffsecurity = dbname=ffsecurity host=localhost port=5432 user=ntech
...
; ip address or * which means all ip-s
listen_addr = 127.0.0.1
listen_port = 5439
...
; any, trust, plain, crypt, md5
auth_type = plain
...
; When server connection is released back to pool:
;   session      - after client disconnects
;   transaction  - after transaction finishes
;   statement    - after statement finishes
pool_mode = transaction
...
; total number of clients that can connect
max_client_conn = 16384
...
;; Syslog settings
syslog = 1
syslog_facility = daemon
syslog_ident = pgbouncer
```

8. Скопируйте пароль пользователя `ntech` (`9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3` в примере). В PostgreSQL установите скопированный пароль для роли `ntech`. Откройте интерактивный терминал PostgreSQL. Появится строка `postgres=#`. После знака `#` введите следующую команду: `ALTER ROLE ntech PASSWORD '<скопированный пароль>'`.

```
sudo -u postgres psql

postgres=# ALTER ROLE ntech PASSWORD '9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3';
```

9. Через интерактивный терминал PostgreSQL создайте базу данных `ffcounter` в PostgreSQL.

```
postgres=# CREATE DATABASE ffcounter WITH OWNER ntech ENCODING 'UTF-8' LC_COLLATE='C.UTF-8'
↳ LC_CTYPE='C.UTF-8' TEMPLATE template0;
```

10. Создайте и настройте `pgbouncer.service`.

```
sudo touch /etc/systemd/system/pgbouncer.service
sudo vi /etc/systemd/system/pgbouncer.service
```

Вставьте следующий код:

```
[Unit]
Description=Pgbouncer service
After=postgresql.service
Before=findface-security.service
[Service]
User=postgres
```

(continues on next page)

(продолжение с предыдущей страницы)

```
Group=postgres
ExecStart=/usr/sbin/pgbouncer "/etc/pgbouncer/pgbouncer.ini"
[Install]
WantedBy=multi-user.target
```

11. Добавьте `pgbouncer.service` в автозагрузку и перезапустите сервис:

```
sudo systemctl enable pgbouncer.service
sudo systemctl restart pgbouncer.service
```

12. Установите из репозитория новые сервисы FindFace Security в соответствии с текущей архитектурной схемой.

CPU-версия:

```
sudo apt update
sudo apt install ffsecurity ffsecurity-ui findface-extraction-api findface-ntls findface-sf-
↪api findface-tarantool-server findface-upload findface-video-manager findface-video-worker-
↪cpu findface-counter
```

GPU-версия:

```
sudo apt update
sudo apt install ffsecurity ffsecurity-ui findface-extraction-api-gpu findface-ntls findface-
↪sf-api findface-tarantool-server findface-upload findface-video-manager findface-video-
↪worker-gpu findface-gpudetector-data findface-counter
```

Важно: Для FindFace Security 4.2 на GPU необходимы драйверы `nvidia-440` и `cuda 10.2`. Убедитесь, что установлены правильные версии драйверов.

Важно: В какой-то момент вам будет предложено выбрать, какую версию конфигурационного файла `findface-security` использовать. Выберите вариант `Install the packages maintainer's version`.

13. Добавьте сервис `findface-counter` в автозагрузку.

```
sudo systemctl enable findface-counter
```

14. Откройте файл конфигурации `findface-security` и вставьте в него сохраненные значения параметров `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN` и `ROUTER_URL`. Заполните раздел `DATABASES` по аналогии: `'PORT': 5439`, `'USER': 'ntech'`, `'PASSWORD': '9T3g1nXy9yx3y8MIGm9fbef3dia8UTc'` (пароль из `/etc/pgbouncer/userlist.txt`).

```
sudo vi /etc/ffsecurity/config.py
...
# Database is used by FindFace Security to store cameras,
# camera groups, watchlists and so on. Only PostgreSQL is supported.
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'DISABLE_SERVER_SIDE_CURSORS': True,
        'NAME': 'ffsecurity',
```

(continues on next page)

(продолжение с предыдущей страницы)

```

        'PORT': 5439, 'USER': 'ntech', 'PASSWORD': '9T3g1nXy9yx3y8MIGm9fbef3dia8UTc3',
    }
}
...
# Use pwgen -sncy 50 1/tr "" "." to generate your own unique key
SECRET_KEY = '002231ccb690586f4d33e98322c591bb'
...
SERVICE_EXTERNAL_ADDRESS = 'http://172.20.77.58'
# EXTERNAL_ADDRESS is used to access objects created inside FFSecurity via external links.
EXTERNAL_ADDRESS = 'http://172.20.77.58'
...
# findface-video-worker authorization token
'VIDEO_DETECTOR_TOKEN': '8977e1b0067d43f6c908d0bf60363255',
...
# findface-video-worker face posting address,
# it must be set to either FFSecurity EXTERNAL_ADDRESS (by default)
# or findface-facerouter url (in some specific cases)
'ROUTER_URL': 'http://127.0.0.1:80',

```

15. Измените структуру базы данных Tarantool с помощью файла `tnt_schema.lua` из новой версии.

```
sudo findface-security make_tnt_schema | sudo tee /etc/ffsecurity/tnt_schema.lua
```

16. Остановите шарды `findface-tarantool-server`. Очистите директории с данными для всех активных шардов.

```

sudo systemctl stop 'tarantool@*'

sudo rm /opt/ntech/var/lib/tarantool/shard-*/{index,snapshots,xlogs}/*

```

17. Перейдите в директорию с файлами конфигурации Tarantool `/etc/tarantool/instances.enabled/`. В каждый файл конфигурации `*.lua` добавьте определение `meta_indexes`, как показано в примере ниже.

```

sudo vi /etc/tarantool/instances.enabled/*.lua

FindFace.start("127.0.0.1", 8001, {
    license_ntls_server="127.0.0.1:3133",
    meta_indexes=meta_indexes,
    meta_scheme = meta_scheme
})

```

18. Перезапустите шарды `findface-tarantool-server`.

```

TNT=$(ls /etc/tarantool/instances.enabled/ | cut -c 7,8,9)
for i in $TNT; do sudo systemctl restart tarantool@shard-$i.service ; done

```

19. Восстановите базу данных Tarantool из резервной копии.

```

cd /etc/findface_dump

for x in *.json; do sudo findface-storage-api-restore -config /etc/findface-sf-api.ini < "$x";
↪ done

```

20. Перезагрузите сервисы.

```
sudo systemctl restart findface-security.service
sudo systemctl restart findface-ntls findface-extraction-api findface-video-worker* findface-
↪video-manager findface-sf-api findface-counter
```

21. Перенесите схему базы данных из FindFace Security в PostgreSQL, заново создайте группы пользователей с *предустановленными* правами и первого пользователя с правами администратора.

```
sudo findface-security migrate
sudo findface-security create_groups
sudo findface-security create_default_user
```

22. Перезапустите PostgreSQL.

На Ubuntu 16.04:

```
sudo systemctl restart postgresql@9.5-main.service
```

На Ubuntu 18.04:

```
sudo systemctl restart postgresql@10-main.service
```

1.8.2 Резервное копирование и восстановление хранилищ данных

Этот раздел посвящен резервному копированию и восстановлению хранилищ данных FindFace Security. Система использует следующие хранилища:

- Биометрическая база данных на основе Tarantool, в которой хранятся биометрические образцы (векторы признаков) и события идентификации лиц.
- Основная база данных системы на основе PostgreSQL, в которой хранятся внутренние данные системы, досье, учетные записи пользователей и настройки камер.
- Каталог `/var/lib/ffsecurity/uploads`, в котором хранятся загруженные в досье фотографии, видеофайлы, а также такие артефакты событий, как полные кадры, миниатюры лиц и нормализованные изображения лиц.
- Каталог `/var/lib/ffupload/`, в котором хранятся только такие артефакты событий, как миниатюры лиц.

В этом разделе:

- *Резервное копирование и восстановление биометрической базы данных*
 - *Утилиты*
 - *Резервное копирование базы данных*
 - *Восстановление базы данных*
- *Резервное копирование и восстановление основной базы данных*

- Резервное копирование артефактов

Резервное копирование и восстановление биометрической базы данных

В биометрической базе данных на основе Tarantool есть 3 галереи:

- `ffsec_dossier_face`: биометрические образцы, извлеченные из фотографий в досье.
- `ffsec_events`: биометрические образцы, извлеченные из лиц, обнаруженных на видео.
- `ffsec_persons`: центроиды персон (виртуальные биометрические образцы, усредненные по всем лицам персоны) и метаданные.

Функционал резервного копирования и восстановления базы данных позволяет при необходимости восстанавливать содержимое данных галерей.

Для предотвращения потери данных создание резервной копии биометрической базы данных рекомендуется выполнять по крайней мере 1 раз в неделю.

Резервную копию базы данных также нужно создать перед *миграцией* системы на другую биометрическую модель.

Утилиты

Для резервного копирования и восстановления биометрической базы данных FindFace Security необходимы следующие утилиты:

1. резервное копирование: `findface-storage-api-dump`,
2. восстановление: `findface-storage-api-restore`.

Данные утилиты автоматически устанавливаются вместе с компонентом `findface-sf-api`.

Резервное копирование базы данных

Для резервного копирования биометрической базы данных используйте утилиту `findface-storage-api-dump` следующим образом:

Важно: Сервисы `findface-tarantool-server` и `findface-sf-api` должны быть активны.

Примечание: Резервное копирование можно также применить к распределенной базе данных. В этом случае утилита `findface-storage-api-dump` создаст резервные копии всех шардов, указанных в `/etc/findface-sf-api.ini`.

1. На сервере с установленным `findface-sf-api` создайте каталог для хранения резервных копий (`/etc/findface_dump` в примере ниже).
2. Запустите утилиту `findface-storage-api-dump` следующей командой:

```
sudo findface-storage-api-dump -output-dir=/etc/findface_dump -config /etc/findface-sf-api.ini
```

Утилита создаст резервные копии всех галерей и запишет их в указанный каталог в виде файлов с соответствующими именами `ffsec_dossier_face.json`, `ffsec_events.json` и `ffsec_monitoring.json`. Эти файлы содержат все данные, необходимые для полного восстановления галерей.

Восстановление базы данных

Для восстановления биометрической базы данных выполните следующие действия:

1. Используя HTTP API, создайте галереи в базе данных: `ffsec_dossier_face`, `ffsec_events`, `ffsec_monitoring`.

Совет: См. [HTTP API](#).

```
curl -D - -X POST -s 'http://localhost:8101/v2/galleries/add/ffsec_dossier_face'
curl -D - -X POST -s 'http://localhost:8101/v2/galleries/add/ffsec_events'
curl -D - -X POST -s 'http://localhost:8101/v2/galleries/add/ffsec_monitoring'
```

2. Запустите утилиту `findface-storage-api-restore` для всех файлов в папке с резервными копиями:

```
sudo findface-storage-api-restore -config /etc/findface-sf-api.ini /etc/findface_dump/*.json
```

Процесс восстановления можно при необходимости прервать с сохранением выполненной работы. Для того чтобы продолжить процесс после прерывания, снова запустите утилиту `findface-storage-api-restore`.

См. также:

- [Опции резервного копирования базы данных](#)
- [Опции восстановления базы данных](#)

Резервное копирование и восстановление основной базы данных

Для резервного копирования базы данных PostgreSQL выполните команду:

```
sudo -u postgres pg_dump ffsecurity > ffsecurity_postgres_backup.sql
```

Для восстановления базы данных PostgreSQL выполните команду:

```
psql -d ffsecurity -f ffsecurity_postgres_backup.sql
```

Резервное копирование артефактов

Артефакты FindFace Security, такие как загруженные фотографии, видеофайлы и артефакты событий (полные кадры, миниатюры лиц и нормализованные изображения лиц) хранятся в следующих каталогах:

- `/var/lib/ffsecurity/uploads`
- `/var/lib/ffupload/`

Для резервного копирования артефактов выполните команду:

```
tar -cvzf var_lib_ffsecurity_uploads.tar.gz /var/lib/ffsecurity/uploads
tar -cvzf var_lib_ffupload.tar.gz /var/lib/ffupload/
```

1.8.3 Миграция на другую модель биометрического образца

Совет: Не стесняйтесь обращаться к нашим специалистам по вопросам миграции по адресу support@ntechlab.com.

Важно: Перед миграцией обязательно создайте *резервную копию* базы данных.

Иногда вам может потребоваться миграция экземпляра FindFace Security на другую модель нейронной сети. Как правило, данная процедура необходима при обновлении до последней версии продукта.

Для того чтобы мигрировать систему на другую модель биометрического образца, используйте утилиту `findface-sf-api-migrate`. Для того чтобы передать настройки миграции, запустите ее с опцией `-config` и укажите файл конфигурации, показанный в примере ниже.

```
findface-sf-api-migrate -config <migration.ini>
```

Пример файла конфигурации:

```
extraction-api:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 0s
  extraction-api: http://127.0.0.1:18666
storage-api-from: # current location of the gallery
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  max-idle-conns-per-host: 20
  shards:
    - master: http://127.0.0.1:8001/v2/
      slave: ""
storage-api-to:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  max-idle-conns-per-host: 20
  shards:
    - master: http://127.0.0.1:8002/v2/
      slave: ""
workers_num: 3
faces_limit: 100
extraction_batch_size: 8
normalized_storage:
```

(continues on next page)

(продолжение с предыдущей страницы)

```

type: webdav
enabled: True
webdav:
  upload-url: http://127.0.0.1:3333/uploads/
s3:
  endpoint: 172.20.77.75:9000
  bucket-name: sf-api-normalized
  access-key: W0G6EQT6MC3BZC8136DW
  secret-access-key: XnottrdxRFp70wfEGdkvKgkzKZ3mEa2Y9bYmob4I
  secure: False
  region: ""
  operation-timeout: 10
  public-url: 123

```

Параметр	Описание
extraction-api -> extraction-api	Адрес findface-extraction-api с новой моделью биометрического образца в файле конфигурации.
storage-api-from	Прежнее хранилище биометрических образцов (до миграции)
storage-api-to	Новое хранилище биометрических образцов (после миграции)
normalized_storage -> upload-url	Хранилище нормализованных изображений лиц

1.8.4 Изменение структуры биометрической базы данных

В некоторых случаях вам может потребоваться применить новую структурную схему к биометрической базе данных Tarantool, например, при обновлении до последней версии продукта или если необходимо усовершенствовать структуру базы данных, добавив в нее дополнительные параметры, расширенные метаданные лиц и т. д.

В этом разделе:

- *О структуре биометрической базы данных*
- *Изменение структуры*

О структуре биометрической базы данных

В FindFace Security структура базы данных задается через файл `tnt_schema.lua`.

Структура представляет собой набор полей, каждое из которых описывается следующими параметрами:

- `id`: id поля;
- `name`: название поля, должно совпадать с названием соответствующего параметра лица;
- `field_type`: тип данных;
- `default`: значение по умолчанию. Если значение по умолчанию больше `'1e14 - 1'`, то его следует записывать в виде строки, т. е. `"123123..."` вместо `123123...`

Используемый по умолчанию файл `tnt_schema.lua` приведен ниже:

```
meta_scheme = {
  -- internal.normalized_id:
  {
    default = '',
    field_type = 'string',
    id = 1,
    name = 'normalized_id',
  },
  -- internal.feat:
  {
    default = '',
    field_type = 'string',
    id = 2,
    name = 'feat',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:acknowledged:
  {
    default = 0,
    field_type = 'unsigned',
    id = 3,
    name = 'm:acknowledged',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:acknowledged_by:
  {
    default = 0,
    field_type = 'unsigned',
    id = 4,
    name = 'm:acknowledged_by',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:acknowledged_date:
  {
    default = 0,
    field_type = 'unsigned',
    id = 5,
    name = 'm:acknowledged_date',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:acknowledged_reaction:
  {
    default = '',
    field_type = 'string',
    id = 6,
    name = 'm:acknowledged_reaction',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:camera:
  {
    default = 0,
    field_type = 'unsigned',
    id = 7,
    name = 'm:camera',
  },
  -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:camera_group:
  {
    default = 0,
    field_type = 'unsigned',
    id = 8,
    name = 'm:camera_group',
  },
}
```

(continues on next page)

(продолжение с предыдущей страницы)

```

},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:confidence:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 9,
    name = 'm:confidence',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:created_date:
{
    default = 0,
    field_type = 'unsigned',
    id = 10,
    name = 'm:created_date',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:episode:
{
    default = 0,
    field_type = 'unsigned',
    id = 11,
    name = 'm:episode',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:episode_open:
{
    default = 0,
    field_type = 'unsigned',
    id = 12,
    name = 'm:episode_open',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_age:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 13,
    name = 'm:f_age',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_beard_class:
{
    default = '',
    field_type = 'string',
    id = 14,
    name = 'm:f_beard_class',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_beard_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 15,
    name = 'm:f_beard_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_emotions_class:
{
    default = '',
    field_type = 'string',
    id = 16,
    name = 'm:f_emotions_class',
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_emotions_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 17,
    name = 'm:f_emotions_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_gender_class:
{
    default = '',
    field_type = 'string',
    id = 18,
    name = 'm:f_gender_class',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_gender_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 19,
    name = 'm:f_gender_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_glasses_class:
{
    default = '',
    field_type = 'string',
    id = 20,
    name = 'm:f_glasses_class',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_glasses_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 21,
    name = 'm:f_glasses_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_liveness_class:
{
    default = '',
    field_type = 'string',
    id = 22,
    name = 'm:f_liveness_class',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_liveness_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 23,
    name = 'm:f_liveness_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_medmask_class:
{
    default = '',
    field_type = 'string',
    id = 24,
    name = 'm:f_medmask_class',
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_medmask_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 25,
    name = 'm:f_medmask_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_race_class:
{
    default = '',
    field_type = 'string',
    id = 26,
    name = 'm:f_race_class',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:f_race_score:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 27,
    name = 'm:f_race_score',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:thumbnail:
{
    default = '',
    field_type = 'string',
    id = 28,
    name = 'm:thumbnail',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:frame:
{
    default = '',
    field_type = 'string',
    id = 29,
    name = 'm:frame',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:frame_coords_bottom:
{
    default = 0,
    field_type = 'unsigned',
    id = 30,
    name = 'm:frame_coords_bottom',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:frame_coords_left:
{
    default = 0,
    field_type = 'unsigned',
    id = 31,
    name = 'm:frame_coords_left',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:frame_coords_right:
{
    default = 0,
    field_type = 'unsigned',
    id = 32,
    name = 'm:frame_coords_right',
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:frame_coords_top:
{
    default = 0,
    field_type = 'unsigned',
    id = 33,
    name = 'm:frame_coords_top',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:pk:
{
    default = 0,
    field_type = 'unsigned',
    id = 34,
    name = 'm:pk',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:matched:
{
    default = 0,
    field_type = 'unsigned',
    id = 35,
    name = 'm:matched',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:matched_dossier:
{
    default = 0,
    field_type = 'unsigned',
    id = 36,
    name = 'm:matched_dossier',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:matched_face:
{
    default = 0,
    field_type = 'unsigned',
    id = 37,
    name = 'm:matched_face',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:matched_lists:
{
    default = {},
    field_type = 'set[unsigned]',
    id = 38,
    name = 'm:matched_lists',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:normalized_photo:
{
    default = '',
    field_type = 'string',
    id = 39,
    name = 'm:normalized_photo',
},
-- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:quality:
{
    default = "10000000000000000000",
    field_type = 'unsigned',
    id = 40,
    name = 'm:quality',
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    },
    -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:scores:
    {
        default = '',
        field_type = 'string',
        id = 41,
        name = 'm:scores',
    },
    -- <class 'ffsecurity.entities_tnt.event.models.ListEvent'>.m:video_source:
    {
        default = 0,
        field_type = 'unsigned',
        id = 42,
        name = 'm:video_source',
    },
    -- <class 'ffsecurity.entities_tnt.dossier_face.models.DossierFace'>.m:dossier:
    {
        default = 0,
        field_type = 'unsigned',
        id = 43,
        name = 'm:dossier',
    },
    -- <class 'ffsecurity.entities_tnt.dossier_face.models.DossierFace'>.m:modified_date:
    {
        default = 0,
        field_type = 'unsigned',
        id = 44,
        name = 'm:modified_date',
    },
    -- <class 'ffsecurity.entities_tnt.dossier_face.models.DossierFace'>.m:source_photo:
    {
        default = '',
        field_type = 'string',
        id = 45,
        name = 'm:source_photo',
    },
    -- <class 'ffsecurity.entities_tnt.dossier_face.models.DossierFace'>.m:source_photo_name:
    {
        default = '',
        field_type = 'string',
        id = 46,
        name = 'm:source_photo_name',
    },
}
-- Fields referenced by multiple models: m:frame_coords_left, m:frame_coords_right, m:frame_coords_
-- top, m:pk, m:created_date, m:frame_coords_bottom, m:thumbnail
meta_indexes = {'m:episode', 'm:episode_open'}
```

Изменение структуры

Для изменения структуры базы данных выполните следующие действия:

1. Остановите сервис `findface-security`.

```
sudo systemctl stop findface-security.service
```

- Создайте резервную копию биометрической базы данных в любой выбранной директории, например, /tmp/dump.

Совет: Подробнее см. *Резервное копирование и восстановление хранилищ данных*.

```
mkdir -p /tmp/dump
cd /tmp/dump
sudo findface-storage-api-dump -config /etc/findface-sf-api.ini
```

- Подготовьте файл `tnt_schema.lua` с новой структурой базы данных.
- Измените структуру базы данных с помощью подготовленного файла `tnt_schema.lua`.

```
sudo findface-security make_tnt_schema | sudo tee /etc/ffsecurity/tnt_schema.lua
```

- Откройте файл конфигурации Tarantool. Убедитесь, что перед секцией `FindFace.start` добавлена строка `dofile("/etc/ffsecurity/tnt_schema.lua")`, а переменная `meta_scheme=meta_scheme` определена в параметрах `FindFace.start`.

```
sudo vi /etc/tarantool/instances.enabled/<shard_00N>.lua

dofile("/etc/ffsecurity/tnt_schema.lua")

FindFace.start("127.0.0.1", 8101, {
    license_ntls_server="127.0.0.1:3133",
    meta_scheme = meta_scheme
})
```

- Удалите базу данных Tarantool (базу по умолчанию или шарды).

```
sudo rm -f /opt/ntech/var/lib/tarantool/default/{index,snapshots,xlogs}/*

sudo rm -f /opt/ntech/var/lib/tarantool/shard-001/{index,snapshots,xlogs}/*
...
sudo rm -f /opt/ntech/var/lib/tarantool/shard-00N/{index,snapshots,xlogs}/*
```

- Восстановите базу данных Tarantool из резервной копии.

Важно: Если некоторые прежние поля отсутствуют в новой структуре базы данных, сначала потребуется вручную удалить соответствующие данные из резервной копии.

```
cd /tmp/dump
for x in *.json; do curl -X POST "http://127.0.0.1:18411/v2/galleries/${x%.json}"; done
for x in *.json; do sudo findface-storage-api-restore -config /etc/findface-sf-api.ini < "$x";
↪ done
```

- Запустите сервис `findface-security`.

```
sudo systemctl start findface-security.service
```

1.8.5 Удаление экземпляра продукта

Вы можете автоматически удалить FindFace Security вместе с базой данных с помощью скрипта `ffsec_uninstall.sh`. Перед удалением будут созданы резервные копии файлов конфигурации и ба-

зы данных.

Выполните следующие действия:

1. Загрузите скрипт `ffsec_uninstall.sh` в любой каталог на сервере установки (например, в `/home/username/`).
2. Из данного каталога сделайте скрипт исполняемым.

```
chmod +x ffsec_uninstall.sh
```

3. Запустите скрипт.

```
sudo ./ffsec_uninstall.sh
```

4. Ответьте **all** на вопрос интерактивного мастера удаления, чтобы полностью удалить FindFace Security вместе с базой данных.

1.8.6 Проверка статуса компонентов

Проверьте статус компонентов, если вы столкнулись с проблемой в системе.

Компонент	Команда для просмотра статуса сервиса
<code>findface-extraction-api</code>	<code>sudo systemctl status findface-extraction-api.service</code>
<code>findface-sf-api</code>	<code>sudo systemctl status findface-sf-api.service</code>
<code>findface-tarantool-server</code>	<code>sudo systemctl status tarantool@FindFace.service</code>
<code>findface-video-manager</code>	<code>sudo systemctl status findface-video-manager.service</code>
<code>findface-video-worker</code>	<code>sudo systemctl status findface-video-worker*.service</code>
<code>findface-ntls</code>	<code>sudo systemctl status findface-ntls</code>
<code>findface-security</code>	<code>sudo systemctl status findface-security*</code>
<code>etcd</code>	<code>sudo systemctl status etcd.service</code>
<code>NginX</code>	<code>sudo systemctl status nginx.service</code>
<code>memcached</code>	<code>sudo systemctl status memcached.service</code>
<code>postgresql</code>	<code>sudo systemctl status postgresql*</code>
<code>redis</code>	<code>sudo systemctl status redis.service</code>

1.8.7 Лог-файлы

При разборе нештатных ситуаций используйте логи, содержащие подробную детализировку всех событий, произошедших в системе.

Компонент	Команда для просмотра лога
<code>findface-extraction-api</code>	<code>sudo tail -f /var/log/syslog grep extraction-api</code>
<code>findface-sf-api</code>	<code>sudo tail -f /var/log/syslog grep sf-api</code>
<code>findface-tarantool-server</code>	<code>sudo tail -f /var/log/tarantool/FindFace.log</code>
<code>findface-video-manager</code>	<code>sudo tail -f /var/log/syslog grep video-manager</code>
<code>findface-video-worker</code>	<code>sudo tail -f /var/log/syslog grep video-worker</code>
<code>findface-security</code>	<code>sudo tail -f /var/log/syslog grep findface-security</code>
<code>findface-ntls</code>	<code>sudo tail -f /var/log/syslog grep ntls</code>
<code>findface-security</code>	<code>sudo tail -f /var/log/syslog grep security</code>
<code>etcd</code>	<code>sudo tail -f /var/log/syslog grep etcd</code>

Вы также можете посмотреть аудит-логи для каждого компонента. Для этого используйте команду `journalctl -u <component>`, например:

```
journalctl -u findface-extraction-api
```

Важно: Для того чтобы включить хранение аудит-логов на жестком диске в ОС Ubuntu, в файле `/etc/systemd/journald.conf` раскомментируйте и измените параметр **Storage** следующим образом:

```
sudo vi /etc/systemd/journald.conf
...
[Journal]
Storage=persistent
```

При необходимости также раскомментируйте и измените значение параметра **SystemMaxUse**. Данный параметр определяет в процентах максимальный объем логов на жестком диске (по умолчанию 10%).

```
SystemMaxUse=15
```

Для того чтобы просмотреть аудит-логи в ОС Ubuntu, выполните следующую команду:

```
journalctl -o verbose SYSLOG_IDENTIFIER=ffsecurity
```

При расшифровке аудит-логов в первую очередь обращайте внимание на следующие параметры:

- **REQUEST_USER:** пользователь, который выполнил изменения;
- **REQUEST_PATH:** URL запроса;
- **REQUEST_DATA:** данные запроса.

Ниже приведен пример лога создания досье с `id=1879` пользователем `admin`:

```
Fr 2017-12-22 17:53:32.436258 MSK [s=0b5566699751426983e13241301205e9;i=e26015;
↪b=907c34cc1fde4398af63bb575587d9ba;m=246f620c449;t=560eefaf59bc5;x=ed60a136c8fc6362]
  PRIORITY=6
  _UID=123
  _GID=130
  _CAP_EFFECTIVE=0
  _BOOT_ID=907c34cc1fde4398af63bb575587d9ba
  _MACHINE_ID=a3eea61c03e041ef8e64d5c72f5fce40
  _HOSTNAME=ntechadmin
  SYSLOG_IDENTIFIER=ffsecurity
  THREAD_NAME=MainThread
  _TRANSPORT=journal
  _PID=6579
  _COMM=findface-securi
  _EXE=/opt/ffsecurity/bin/python3
  _CMDLINE=/opt/ffsecurity/bin/python /opt/ffsecurity/bin/findface-security runworker
  _SYSTEMD_CGROUP=/system.slice/system-findface\x2dsecurity\x2dworker.slice/findface-security-
↪worker@4.service
  _SYSTEMD_UNIT=findface-security-worker@4.service
  _SYSTEMD_SLICE=system-findface\x2dsecurity\x2dworker.slice
  CODE_FILE=/opt/ffsecurity/lib/python3.5/site-packages/ffsecurity/mixins.py
  CODE_LINE=94
  CODE_FUNC=finalize_response
  REQUEST_USER=admin
  LOGGER=ffsecurity.audit
```

(continues on next page)

(продолжение с предыдущей страницы)

```
MESSAGE=N8Be05i1 POST /dossier-faces/ 201 by admin
REQUEST_DATA={"dossier": "'1879'", "source_photo": "<InMemoryUploadedFile: 14927016033292449.
↪ jpeg (image/jpeg)>"}
REQUEST_PATH=/dossier-faces/
REQUEST_ID=N8Be05i1
_SOURCE_REALTIME_TIMESTAMP=1513954412436258
```

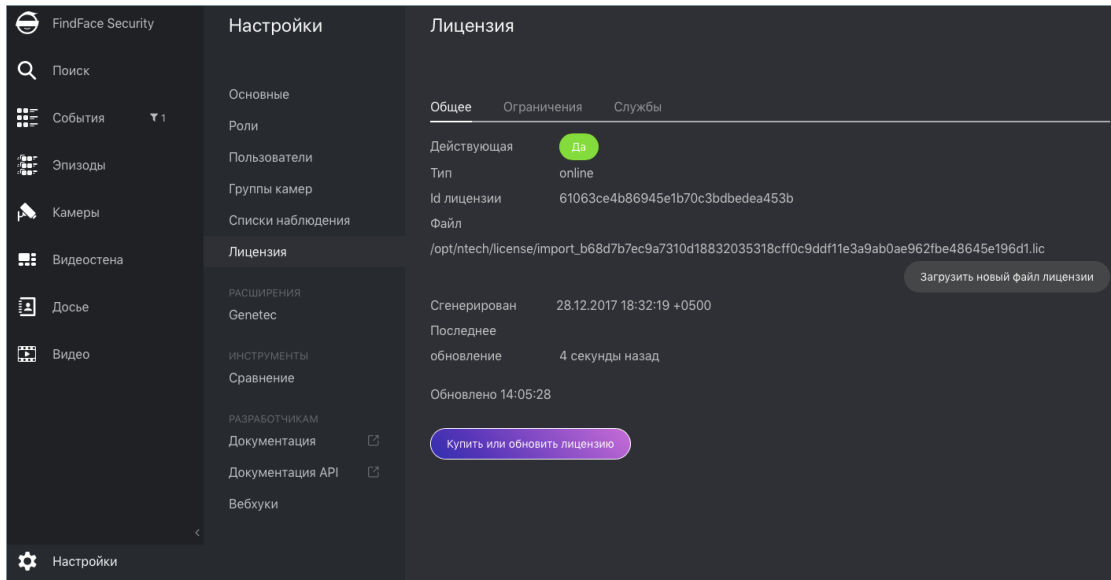
В следующем примере для досье с id=1879 запрашивается список лиц:

```
Fr 2017-12-22 17:53:32.475467 MSK [s=0b5566699751426983e13241301205e9;i=e26016;
↪ b=907c34cc1fde4398af63bb575587d9ba;m=246f6215d82;t=560eefaf634fe;x=b1374a144a46b5cd]
  PRIORITY=6
  _UID=123
  _GID=130
  _CAP_EFFECTIVE=0
  _BOOT_ID=907c34cc1fde4398af63bb575587d9ba
  _MACHINE_ID=a3eea61c03e041ef8e64d5c72f5fce40
  _HOSTNAME=ntechadmin
  SYSLOG_IDENTIFIER=ffsecurity
  THREAD_NAME=MainThread
  _TRANSPORT=journal
  _COMM=findface-securi
  _EXE=/opt/ffsecurity/bin/python3
  _CMDLINE=/opt/ffsecurity/bin/python /opt/ffsecurity/bin/findface-security runworker
  _SYSTEMD_SLICE=system-findface\x2dsecurity\x2dworker.slice
  _PID=6588
  _SYSTEMD_CGROUP=/system.slice/system-findface\x2dsecurity\x2dworker.slice/findface-security-
↪ worker@2.service
  _SYSTEMD_UNIT=findface-security-worker@2.service
  CODE_FILE=/opt/ffsecurity/lib/python3.5/site-packages/ffsecurity/mixins.py
  CODE_LINE=94
  CODE_FUNC=finalize_response
  REQUEST_USER=admin
  REQUEST_DATA={}
  LOGGER=ffsecurity.audit
  MESSAGE=Dee7Qvy4 GET /dossier-faces/?dossier=1879&limit=1000 200 by admin
  REQUEST_ID=Dee7Qvy4
  REQUEST_PATH=/dossier-faces/?dossier=1879&limit=1000
  _SOURCE_REALTIME_TIMESTAMP=1513954412475467
```

1.8.8 Лицензирование

Просмотр и обновление лицензии

Для просмотра текущей информации по лицензии или ее обновления, перейдите в *Настройки -> Лицензия*.



Устранение неполадок с лицензированием и findface-ntls

При устранении неполадок с лицензией и сервером `findface-ntls` (см. [Принцип лицензирования](#)) первым шагом является получение информации о лицензии и статусе сервера. Это можно сделать, отправив API-запрос в `findface-ntls`. Действия по устранению неполадок предпринимаются в учетном содержании API-ответа.

Совет: По вопросам устранения неполадок обращайтесь к нашим специалистам по адресу support@ntechlab.com.

Примечание: Онлайн-лицензирование выполняется через Глобальный менеджер лицензий NtechLab license.ntechlab.com. Проверьте его доступность. Для онлайн-лицензирования необходимы стабильное интернет-соединение и DNS.

Для получения информации о *лицензии* FindFace Security и статусе `findface-ntls`, выполните в консоли сервера `findface-ntls` следующую команду:

```
curl http://localhost:3185/license.json -s | jq
```

Ответ будет возвращен в формате JSON. Одним из наиболее значимых параметров в ответе является `last_updated`. Он показывает в секундах, как давно в последний раз проверялась локальная лицензия.

Интерпретируйте значение параметра `last_updated` следующим образом:

- `[0, 5]` — все работает отлично.
- `(5, 30]` — возможно имеют место быть какие-то проблемы со связью, либо с локальным накопителем, где хранятся файлы лицензий.
- `(30; 120]` — почти наверняка случилось что-то нехорошее.
- `(120; ∞)` — не удастся получить ответ от источника лицензирования в течение длительного времени. Необходимо вмешательство.

- "valid": false: связь с источником лицензирования так и не была установлена.

```
curl http://localhost:3185/license.json -s | jq
{
  "name": "NTLS",
  "time": 1565186356,
  "type": "online",
  "license_id": "61063ce4b86945e1b70c3bdbedea453b",
  "generated": 1514467939,
  "last_updated": 5,
  "valid": {
    "value": true,
    "description": ""
  },
  "source": "/opt/ntech/license/import_
↪b68d7b7ec9a7310d18832035318cff0c9ddf11e3a9ab0ae962fbe48645e196d1.lic",
  "limits": [
    {
      "type": "time",
      "name": "end",
      "value": 1609161621
    },
    {
      "type": "number",
      "name": "faces",
      "value": 9007199254740991,
      "current": 0
    },
    {
      "type": "number",
      "name": "cameras",
      "value": 4294967295,
      "current": 0
    },
    {
      "type": "number",
      "name": "extraction_api",
      "value": 256,
      "current": 0
    },
    {
      "type": "boolean",
      "name": "gender",
      "value": true
    },
    {
      "type": "boolean",
      "name": "age",
      "value": true
    },
    {
      "type": "boolean",
      "name": "emotions",
      "value": true
    },
    {
      "type": "boolean",
      "name": "fast-index",
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    "value": true
  },
  {
    "type": "boolean",
    "name": "sec-genetec",
    "value": false
  },
  {
    "type": "boolean",
    "name": "countries",
    "value": false
  },
  {
    "type": "boolean",
    "name": "beard",
    "value": false
  },
  {
    "type": "boolean",
    "name": "glasses",
    "value": false
  },
  {
    "type": "boolean",
    "name": "liveness",
    "value": false
  }
],
"services": [
  {
    "name": "video-worker",
    "ip": "127.0.0.1:53276"
  },
  {
    "name": "FindFace-tarantool",
    "ip": "127.0.0.1:53284"
  },
  {
    "name": "FindFace-tarantool",
    "ip": "127.0.0.1:53288"
  }
]
}

```

1.8.9 Автоматическое восстановление Tarantool

Если архитектура вашей системы не обеспечивает бесперебойную доступность серверов Tarantool, рекомендуется включить автоматическое восстановление базы данных. В этом случае каждый раз при возникновении ошибки во время чтения файла `.snap` или `.xlog`, Tarantool попытается прочитать как можно больше информации и восстановить файл, игнорируя битые записи.

Для включения автоматического восстановления базы данных выполните следующие действия:

1. Откройте файл конфигурации Tarantool.

```
sudo vi /etc/tarantool/instances.enabled/FindFace.lua
```

2. Раскомментируйте `force_recovery = true`.

```
box.cfg{
    force_recovery = true,
}
```

1.8.10 Ручная очистка базы данных от старых событий и эпизодов

Совет: Для настройки автоматического удаления событий и эпизодов см. *Автоматическое удаление старых событий и эпизодов*.

Для ручного удаления старых событий и эпизодов из базы данных FindFace Security используйте утилиту `cleanup_events`. Вы можете выбрать удаление только событий, для которых отсутствуют совпадения с досье, или только событий с совпадениями, а также выбрать удаление только полных кадров событий.

Справка по утилите `cleanup_events` вызывается следующей командой:

```
sudo findface-security cleanup_events --help
usage: findface-security cleanup_events [-h] [--matched-age MATCHED_AGE]
        [--unmatched-age UNMATCHED_AGE]
        [--fullframe-matched-age FULLFRAME_MATCHED_AGE]
        [--fullframe-unmatched-age FULLFRAME_UNMATCHED_AGE]
        [--version] [-v {0,1,2,3}]
        [--settings SETTINGS]
        [--pythonpath PYTHONPATH]
        [--traceback] [--no-color]

Delete old events
optional arguments:
  -h, --help            show this help message and exit
  --matched-age MATCHED_AGE
                        Minimum age in days of matched events to clean up
  --unmatched-age UNMATCHED_AGE
                        Minimum age in days of unmatched events to clean up
  --fullframe-matched-age FULLFRAME_MATCHED_AGE
                        Minimum age in days of matched events fullframes to
                        clean up
  --fullframe-unmatched-age FULLFRAME_UNMATCHED_AGE
                        Minimum age in days of unmatched events fullframes to
                        clean up
  --version            show program's version number and exit
  -v {0,1,2,3}, --verbosity {0,1,2,3}
                        Verbosity level; 0=minimal output, 1=normal output,
                        2=verbose output, 3=very verbose output
  --settings SETTINGS  The Python path to a settings module, e.g.
                        "myproject.settings.main". If this isn't provided, the
                        DJANGO_SETTINGS_MODULE environment variable will be
                        used.
  --pythonpath PYTHONPATH
                        A directory to add to the Python path, e.g.
                        "/home/djangoprojects/myproject".
```

(continues on next page)

(продолжение с предыдущей страницы)

```
--traceback      Raise on CommandError exceptions
--no-color       Don't colorize the command output.
```

Для удаления событий и эпизодов старше определенного количества дней используйте опцию `-matched-age/-unmatched-age`. Например, для удаления событий и эпизодов без совпадений старше 5 дней выполните команду:

```
sudo findface-security cleanup_events --unmatched-age 5
```

Для удаления событий с совпадениями старше 5 дней выполните команду:

```
sudo findface-security cleanup_events --matched-age 5
```

Следующие команды удаляют только полные кадры событий с совпадениями и без:

```
sudo findface-security cleanup_events --fullframe-matched-age 5
sudo findface-security cleanup_events --fullframe-unmatched-age 5
```

Важно: Должен быть задан хотя бы один из аргументов `-matched-age/-unmatched-age/-fullframe-matched-age, -fullframe-unmatched-age`.

1.8.11 Отключение сервисов

Существует возможность отключить следующие сервисы FindFace Security, если вы ими не пользуетесь:

- Эпизоды,
- Вебхуки,
- Персоны.

Для этого откройте файл конфигурации `findface-security` и внесите изменения в раздел `SERVICES`.

```
sudo vi /etc/ffsecurity/config.py

# disable unused services to increase
# overall system performance in some cases.
SERVICES = {
    "ffsecurity": {
        "episodes": False,
        "webhooks": False,
        "persons": False,
    }
}
```

В результате соответствующие вкладки исчезнут из веб-интерфейса.

Примечание: Вкладка останется, если на ней есть какие-либо объекты (например, на вкладке *Вебхуки* есть созданные вебхуки). Однако новые артефакты перестанут поступать.

1.9 Приложения

1.9.1 Настройка шифрования данных

Для обеспечения безопасности данных включите SSL-шифрование. Выполните следующие действия:

Важно: Использование самоподписанного сертификата не рекомендуется.

1. В директории с конфигурацией nginx создайте каталог для хранения информации о SSL-шифровании:

```
sudo mkdir /etc/nginx/ssl
```

2. Создайте ключ и сертификат SSL:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/nginx/ssl/my-example-  
domain.com.key -out /etc/nginx/ssl/my-example-domain.com.crt
```

Для заполнения полей сертификата вам будет предложено несколько вопросов. Ответьте на них, уделив особое внимание строке **Common Name**. В ней нужно ввести имя или публичный IP-адрес домена, связанного с сервером. Созданные файлы ключа `my-example-domain.com.key` и сертификата `my-example-domain.com.crt` будут сохранены в каталоге `/etc/nginx/ssl`.

3. Настройте nginx для использования SSL. Откройте файл конфигурации nginx `/etc/nginx/sites-available/ffsecurity-nginx.conf`. Внесите в файл следующие изменения:

1. Добавьте новый раздел `server {...}`, содержащий правило замены URL:

```
server {  
    listen 80;  
    server_name domain.ru www.domain.ru;  
    rewrite ^(.*) https://domain.ru$1 permanent;  
    access_log off;  
}
```

2. Закомментируйте следующие строки в существующем разделе `server {...}`:

```
# listen 80 default_server;  
# listen [::]:80 default_server;
```

3. Добавьте следующие строки, включая пути к сертификату и ключу, в существующий раздел `server {...}`:

```
listen 443 ssl;  
  
ssl_certificate      /etc/NGINX/ssl/domain.crt;  
ssl_certificate_key  /etc/nginx/ssl/domain.key;
```

4. В общем файле конфигурации `/etc/nginx/nginx.conf` найдите раздел **SSL Settings** и добавьте в его конец следующие строки:

```
ssl_session_cache    shared:SSL:10m;  
ssl_session_timeout  1h;
```

Пример файла конфигурации `/etc/nginx/sites-available/ffsecurity-nginx.conf` с корректно заданными настройками SSL показан в примере ниже:

```

upstream ffsecurity {
    server 127.0.0.1:8002;
}

upstream ffsecurity-ws {
    server 127.0.0.1:8003;
}

map $http_upgrade $ffsec_upstream {
    default "http://ffsecurity-ws";
    "" "http://ffsecurity";
}

server {
    listen 80;
    server_name domain.ru www.domain.ru;
    rewrite ^(.*) https://domain.ru$1 permanent;
    access_log off;
}

server {
    # listen 80 default_server;
    # listen [::]:80 default_server;
    listen 443 ssl;
    ssl_certificate /etc/nginx/ssl/domain.crt;
    ssl_certificate_key /etc/nginx/ssl/domain.key;

    root /var/lib/ffsecurity;

    autoindex off;

    server_name _;

    location = / {

        alias /usr/share/ffsecurity-ui/;
        try_files /index.html =404;
    }
    location /static/ {

    }
    location /uploads/ {
        add_header 'Access-Control-Allow-Origin' '*';
        add_header 'Access-Control-Allow-Methods' 'GET';
        add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-Requested-
        ↪With,If-Modified-Since,Cache-Control,Content-Type,Range,Authorization';
        add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range
        ↪';
        add_header 'Access-Control-Max-Age' 2592000;
    }
    location /ui-static/ {
        alias /usr/share/ffsecurity-ui/ui-static/;
    }
    location /doc/ {
        alias /opt/ffsecurity/doc/;
    }
    location ~ /videos/(?<video_id>[0-9]+)/upload/(.*)$ {

```

(continues on next page)

(продолжение с предыдущей страницы)

```

if ($request_method = 'OPTIONS') {
    add_header 'Content-Type' 'text/plain; charset=utf-8';
    add_header 'Content-Length' 0;
    return 204;
}
set $auth_request_uri "http://ffsecurity/videos/$video_id/auth-upload/";
auth_request /video-upload-auth/;

alias "/var/lib/ffsecurity/uploads/videos/$video_id.bin";
client_max_body_size 15g;

dav_access user:rw group:rw all:rw;
dav_methods PUT;

create_full_put_path on;
autoindex off;
autoindex_exact_size off;
autoindex_localtime on;
charset utf-8;

add_header 'Access-Control-Allow-Origin' '*';
add_header 'Access-Control-Allow-Methods' 'PUT, OPTIONS';
add_header 'Access-Control-Allow-Headers' 'authorization';
}
location = /video-upload-auth/ {
    internal;
    client_max_body_size 15g;
    proxy_set_header Content-Length "";
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_pass_request_body off;
    proxy_pass $auth_request_uri;
}

location / {
    client_max_body_size 300m;
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_pass $ffsec_upstream;
    proxy_read_timeout 5m;

    location ~ ^/(cameras|videos)/([0-9]+)/stream/?$ {
        proxy_set_header Host $http_host;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_pass http://ffsecurity;
    }

    location ~ ^/streams/(.*)$ {
        internal;
        proxy_pass $1;
    }
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    }
  }
}

```

4. Перезапустите nginx.

```
sudo service nginx restart
```

5. Внесите изменения в файл конфигурации `findface-security`. В параметрах `EXTERNAL_ADDRESS` и `ROUTER_URL` измените приставку `http://` на `https://`.

```

sudo vi /etc/ffsecurity/config.py

...
EXTERNAL_ADDRESS="https://my-example-domain.com"
...
ROUTER_URL="https://IP_address"

```

6. Если есть запущенные процессы `findface-video-worker`, нужно либо пересоздать камеры в веб-интерфейсе, либо изменить значение параметра `router_url` в job-заданиях, заменив приставку `http://` на `https://`. Это можно сделать с помощью команды, аналогичной следующей:

```

curl -s localhost:18810/jobs | jq -r '.[].["id"]' | xargs -I {} curl -X PATCH -d '{"router_url
↪": "https://domain.ru/video-detector/frame"}' http://localhost:18810/job/{}

```

1.9.2 Подробно о компонентах

`findface-extraction-api`

Компонент `findface-extraction-api` с помощью нейронных сетей обнаруживает лицо на изображении, извлекает из лица биометрический образец, а также распознает пол, возраст, эмоции и другие атрибуты лица.

Компонент взаимодействует с сервисом `findface-sf-api` следующим образом:

- Получает от него фотографию с лицом или нормализованное изображение лица.
- Возвращает координаты рамки с лицом, а также вектор признаков, данные о поле, возрасте, эмоциях и других атрибутах лица (если они были запрошены `findface-sf-api`).

Полный список функций:

- детекция (обнаружение) лица на исходном изображении с возвращением координат рамки с лицом,
- получение из исходного изображения нормализованного изображения лица,
- извлечение из нормализованного изображения лица вектора признаков (биометрического образца),
- распознавание атрибутов лица (пол, возраст, эмоции, борода, очки, медицинская маска и др.).

Сервис `findface-extraction-api` может работать с ускорением на CPU (устанавливается из пакета `findface-extraction-api`) или GPU (устанавливается из пакета `findface-extraction-api-gpu`).

Как для CPU-, так и для GPU-сервиса, настройка выполняется через файл конфигурации `/etc/findface-extraction-api.ini`, однако содержимое данного файла на CPU и GPU отличается.

Файл конфигурации сервиса на CPU:

```
detectors:
  max_batch_size: 1
  instances: 4
  models:
    cheetah:
      model: facedet/cheetah.cpu.fnk
      options:
        min_object_size: 32
        resolutions:
          - 256x256
          - 384x384
          - 512x512
          - 768x768
          - 1024x1024
          - 1536x1536
          - 2048x2048
      quality_estimator: true
normalizers:
  max_batch_size: 8
  instances: 4
  models:
    crop1x:
      model: ''
    crop2x:
      model: facenorm/crop2x.v2_maxsize400.cpu.fnk
    cropbbox:
      model: ''
    norm200:
      model: facenorm/bee.v2.cpu.fnk
extractors:
  max_batch_size: 8
  instances: 4
  models:
    age: ''
    beard: ''
    carattr_color: ''
    carattr_description: ''
    carattr_make: ''
    carattr_trash: ''
    countries47: ''
    emotions: ''
    face: face/ifruit_320.cpu.fnk
    gender: ''
    glasses3: ''
    headpose: ''
    liveness: ''
    luminance_overexposure: ''
    luminance_underexposure: ''
    medmask3: ''
    quality: faceattr/quality.v0.cpu.fnk
    sharpness: ''
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    validity: ''
gpu_device: 0
models_root: /usr/share/findface-data/models
cache_dir: /var/cache/findface/models_cache
listen: 127.0.0.1:18666
license_ntls_server: 127.0.0.1:3133
fetch:
    enabled: true
    size_limit: 10485760
max_dimension: 6000
allow_cors: false
ticker_interval: 5000
prometheus:
    timing_buckets: [0.001, 0.005, 0.01, 0.02, 0.03, 0.05, 0.1, 0.2, 0.3, 0.5, 0.75,
        0.9, 1, 1.1, 1.3, 1.5, 1.7, 2, 3, 5, 10, 20, 30, 50]
    resolution_buckets: [10000, 20000, 40000, 80000, 100000, 200000, 400000, 800000,
        1e+06, 2e+06, 3e+06, 4e+06, 5e+06, 6e+06, 8e+06, 1e+07, 1.2e+07, 1.5e+07, 1.8e+07,
        2e+07, 3e+07, 5e+07, 1e+08]
    faces_buckets: [0, 1, 2, 5, 10, 20, 50, 75, 100, 200, 300, 400, 500, 600, 700, 800,
        900, 1000]

```

Файл конфигурации сервиса на GPU:

```

detectors:
    max_batch_size: 1
    instances: 1
    models:
        cheetah:
            model: facedet/cheetah.gpu.fnk
            options:
                min_object_size: 32
                resolutions:
                    - 256x256
                    - 384x384
                    - 512x512
                    - 768x768
                    - 1024x1024
                    - 1536x1536
                    - 2048x2048
            quality_estimator: true
normalizers:
    max_batch_size: 8
    instances: 1
    models:
        crop1x:
            model: ''
        crop2x:
            model: facenorm/crop2x.v2_maxsize400.gpu.fnk
        cropbbox:
            model: ''
        norm200:
            model: facenorm/bee.v2.gpu.fnk
extractors:
    max_batch_size: 8
    instances: 1

```

(continues on next page)

(продолжение с предыдущей страницы)

```
models:
  age: ''
  beard: ''
  carattr_color: ''
  carattr_description: ''
  carattr_make: ''
  carattr_trash: ''
  countries47: ''
  emotions: ''
  face: face/ifruit_320.gpu.fnk
  gender: ''
  glasses3: ''
  headpose: ''
  liveness: ''
  luminance_overexposure: ''
  luminance_underexposure: ''
  medmask3: ''
  quality: faceattr/quality.v0.gpu.fnk
  sharpness: ''
  validity: ''
gpu_device: 0
models_root: /usr/share/findface-data/models
cache_dir: /var/cache/findface/models_cache
listen: 127.0.0.1:18666
license_ntls_server: 127.0.0.1:3133
fetch:
  enabled: true
  size_limit: 10485760
max_dimension: 6000
allow_cors: false
ticker_interval: 5000
prometheus:
  timing_buckets:
    - 0.001
    - 0.005
    - 0.01
    - 0.02
    - 0.03
    - 0.05
    - 0.1
    - 0.2
    - 0.3
    - 0.5
    - 0.75
    - 0.9
    - 1
    - 1.1
    - 1.3
    - 1.5
    - 1.7
    - 2
    - 3
    - 5
    - 10
    - 20
    - 30
```

(continues on next page)

(продолжение с предыдущей страницы)

```
- 50
resolution_buckets:
- 10000
- 20000
- 40000
- 80000
- 100000
- 200000
- 400000
- 800000
- 1e+06
- 2e+06
- 3e+06
- 4e+06
- 5e+06
- 6e+06
- 8e+06
- 1e+07
- 12000000.0
- 15000000.0
- 18000000.0
- 2e+07
- 3e+07
- 5e+07
- 1e+08
faces_buckets:
- 0
- 1
- 2
- 5
- 10
- 20
- 50
- 75
- 100
- 200
- 300
- 400
- 500
- 600
- 700
- 800
- 900
- 1000
```

Пользовательская настройка `findface-extraction-api` (как CPU, так и GPU) выполняется с использованием следующих параметров:

Параметр	Описание
<code>detector->quality_detector</code>	Включает/отключает оценку качества лица. В этом случае компонент <code>findface-extraction-api</code> будет возвращать показатель качества лица в параметре <code>quality_detector_score</code> . Показатель качества можно анализировать. Прямые изображения лиц анфас считаются наиболее качественными. Для них возвращаются значения вблизи 0, как правило, отрицательные (такие как -0.00067401276, например). Перевернутые лица и лица, повернутые под большими углами, оцениваются отрицательными значениями от -5 и меньше.
<code>cheetah->min_object_size</code>	Минимальный размер лица, которое будет гарантированно найдено. Определяется размером рамки с лицом (bbox). Чем больше значение, тем менее ресурсоемок процесс обнаружения лица.
<code>license_key</code>	Идентификатор сервера лицензирования <code>findface-ntls</code> .
<code>gpu_device</code>	Только для GPU) Номер GPU-устройства, используемого <code>findface-extraction-api-gpu</code> .

В зависимости от нужд вашего бизнеса, вам также может потребоваться включить модели распознавания атрибутов лица, таких как пол, возраст, эмоции, очки и/или борода. Удостоверьтесь, что для каждой модели вы указали правильный тип ускорения CPU или GPU: он должен совпадать с типом ускорения `findface-extraction-api`. Обратите внимание, что `findface-extraction-api` на CPU может работать только с CPU-моделями, в то время как `findface-extraction-api` на GPU поддерживает как GPU-, так и CPU-модели.

```
models:
  age: faceattr/age.v1.cpu.fnk
  emotions: faceattr/emotions.v1.cpu.fnk
  face: face/ifruit_320.cpu.fnk
  gender: faceattr/gender.v2.cpu.fnk
  beard: faceattr/beard.v0.cpu.fnk
  glasses3: faceattr/glasses3.v0.cpu.fnk
  medmask3: faceattr/medmask3.v1.cpu.fnk
```

Доступны следующие модели:

Атрибут лица	Ускорение	Параметр в файле конфигурации
биометрия лица	CPU	face: face/ifruit_320.cpu.fnk face: face/ifruit_160.cpu.fnk
	GPU	face: face/ifruit_320.gpu.fnk face: face/ifruit_160.gpu.fnk
возраст	CPU	age: faceattr/age.v1.cpu.fnk
	GPU	age: faceattr/age.v1.gpu.fnk
пол	CPU	gender: faceattr/gender.v2.cpu.fnk
	GPU	gender: faceattr/gender.v2.gpu.fnk
эмоции	CPU	emotions: faceattr/emotions.v1.cpu.fnk
	GPU	emotions: faceattr/emotions.v1.gpu.fnk
очки	CPU	glasses3: faceattr/glasses3.v0.cpu.fnk
	GPU	glasses3: faceattr/glasses3.v0.gpu.fnk
борода	CPU	beard: faceattr/beard.v0.cpu.fnk
	GPU	beard: faceattr/beard.v0.gpu.fnk
медицинская маска	CPU	medmask3: faceattr/medmask3.v1.cpu.fnk
	GPU	medmask3: faceattr/medmask3.v1.gpu.fnk

Совет: Для того чтобы отключить модель распознавания, передайте в соответствующий параметр пустое значение. Не удаляйте сам параметр, поскольку в этом случае будет выполняться поиск модели по умолчанию.

```
models:
  gender: ""
  age: ""
  emotions: ""
```

findface-sf-api

Компонент **findface-sf-api** представляет собой сервис, реализующий HTTP API основного функционала ядра FindFace (обнаружение и распознавание лиц, при этом сами операции обнаружения и распознавания лиц выполняются компонентом **findface-extraction-api**). Взаимодействует с базой биометрических данных Tarantool через компонент **findface-tarantool-server**, а также с компонентами **findface-extraction-api** (обнаружение и распознавание лиц) и **findface-upload** (хранилище исходных изображений и артефактов работы ядра FindFace).

Для обнаружения лица на фотографии в компонент **findface-sf-api** должен быть отправлен API-запрос, передающий данную фотографию в виде файла или URL. Данный запрос затем перенаправляется в компонент **findface-extraction-api**.

При наличии в системе видеодетектора лиц (присутствует в FindFace Security) компонент **findface-sf-api** получает данные об обнаруженных лицах вместе с правилами их обработки от компонента **findface-facerouter** и затем выполняет полученные директивы (например, сохраняет лица в определенную галерею базы данных).

Примечание: В FindFace Security, функции **findface-facerouter** выполняет сервис **findface-security**.

Полный список функций:

- реализация HTTP API по части обнаружения и распознавания лиц (операции выполняются **findface-extraction-api**).
- сохранение лиц в базу биометрических данных (через сервис **findface-tarantool-server**),
- сохранение исходных изображений, миниатюр и нормализованных изображений лиц на веб-сервере nginx (через сервис **findface-upload**).
- обеспечение взаимодействия всех компонентов системы.

Настройка компонента **findface-sf-api** выполняется через файл конфигурации **/etc/findface-sf-api.ini**.

```
listen: 127.0.0.1:18411
extraction-api:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  extraction-api: http://127.0.0.1:18666
storage-api:
```

(continues on next page)

(продолжение с предыдущей страницы)

```
timeouts:
  connect: 5s
  response_header: 30s
  overall: 35s
  idle_connection: 10s
max-idle-conns-per-host: 20
shards:
- master: http://127.0.0.1:8101/v2/
  slave: ''
- master: http://127.0.0.1:8102/v2/
  slave: ''
- master: http://127.0.0.1:8103/v2/
  slave: ''
- master: http://127.0.0.1:8104/v2/
  slave: ''
limits:
  url-length: 4096
  deny-networks: 127.0.0.0/8,192.168.0.0/16,10.0.0.0/8,::1/128,fe00::/8
  body-image-length: 33554432
  allow-return-facen: false
cache:
  type: memcache
  inmemory:
    size: 16384
  memcache:
    nodes:
    - 127.0.0.1:11211
    timeout: 100ms
  redis:
    network: tcp
    addr: localhost:6379
    password: ''
    db: 0
    timeout: 5s
normalized-storage:
  type: webdav
  enabled: true
  webdav:
    upload-url: http://127.0.0.1:3333/uploads/
    timeouts:
      connect: 5s
      response_header: 30s
      overall: 35s
      idle_connection: 10s
  s3:
    endpoint: ''
    bucket-name: ''
    access-key: ''
    secret-access-key: ''
    secure: true
    region: ''
    public-url: ''
    operation-timeout: 30
```

Пользовательская настройка `findface-sf-api` выполняется с использованием следующих параметров:

Параметр	Описание
extraction-api -> extraction-api	IP-адрес и порт сервера <code>findface-extraction-api</code> .
storage-api -> shards -> master	IP-адрес физического сервера с мастером шарда <code>findface-tarantool-server</code> .
storage-api -> shards -> slave	IP-адрес физического сервера с репликой шарда <code>findface-tarantool-server</code> .
limits -> body-image-length	Максимальный размер в байтах изображения, передаваемого через API-запрос.
normalized-storage -> webdav -> upload_url	Путь в WebDAV nginx, по которому в компонент <code>findface-upload</code> будут отправляться исходные изображения, миниатюры и нормализованные изображения лиц.

`findface-tarantool-server`

Сервис `findface-tarantool-server` обеспечивает взаимодействие между сервисом `findface-sf-api` и биометрической базой данных Tarantool следующим образом:

Совет: Подробнее см. [официальную документацию Tarantool](#).

- `findface-tarantool-server` получает от `findface-sf-api` данные для записи в базу данных (например, об обнаруженных лицах).
- По запросу от `findface-sf-api` `findface-tarantool-server` выполняет поиск по базе данных и возвращает его результат.

Для увеличения скорости поиска на каждом сервере с базой данных Tarantool могут быть созданы многочисленные сегменты («шарды») `findface-tarantool-server`. Их параллельное функционирование приводит к значительному увеличению производительности (в 70-100 раз).

Полный список функций:

- сохранение лиц в базу биометрических данных,
- выполнение поиска по базе биометрических данных,
- реализация прямых запросов в базу данных Tarantool (см. [Прямые API-запросы к базе данных Tarantool](#)).

Настройка компонента `findface-tarantool-server` выполняется через файл конфигурации `/etc/tarantool/instances.enabled/<shard-*>.lua`. В кластерной среде файл конфигурации настраивается отдельно для каждого шарда.

```
--
-- Please, read the tarantool cfg doc:
-- https://tarantool.org/doc/reference/configuration/index.html#box-cfg-params
--
box.cfg{
  --port to listen, direct tarantool access
  --Only need for admin operations
  --THIS IS NOT PORT YOU NEED FOR facenapi/sf-api
  listen = '127.0.0.1:32001',
  --Directory to store data
  vinyl_dir = '/opt/ntech/var/lib/tarantool/shard-001',
```

(continues on next page)

(продолжение с предыдущей страницы)

```

work_dir = '/opt/ntech/var/lib/tarantool/shard-001',
memtx_dir = '/opt/ntech/var/lib/tarantool/shard-001/snapshots',
wal_dir = '/opt/ntech/var/lib/tarantool/shard-001/xlogs',
--Maximum mem usage in bytes
memtx_memory = 200 * 1024 * 1024,
checkpoint_interval = 3600*4,
checkpoint_count = 3,
--uncomment only if you know what you are doing!!! and don't forget box.snapshot()
-- wal_mode = 'none',
--if true, tarantool tries to continue if there is an error while reading a snapshot/xlog
--files: skips invalid records, reads as much data as possible and re-builds the file
-- force_recovery = true,
}
pcall(function() box.schema.user.grant('guest', 'execute,read,write', 'universe') end)
dofile("/etc/ffsecurity/tnt_schema.lua")
-- host,port to bind for http server
FindFace = require("FindFace")
FindFace.start("127.0.0.1", 8101, {
    license_ntls_server="127.0.0.1:3133",
    meta_indexes=meta_indexes,
    meta_scheme = meta_scheme
})

```

Пользовательская настройка **findface-tarantool-server** выполняется с использованием следующих параметров:

Параметр	Описание
memtx_memory	Максимальный размер оперативной памяти в байтах, который может быть использован шардом Tarantool. Перед изменением данного параметра обратитесь к нашим экспертам за консультацией по адресу support@ntechlab.com .
force_recovery	Включает/отключает автоматическое восстановление базы данных Tarantool. Если автоматическое восстановление данных включено (true), каждый раз при возникновении ошибки во время чтения файла .snap или .xlog, Tarantool попытается прочитать как можно больше информации и восстановить файл, игнорируя битые записи.
license_ntls_server	Адрес сервера лицензирования findface-ntls .
meta_scheme	Структура базы данных для хранения биометрических параметров. Представляет собой набор полей, для каждого из которых указываются следующие параметры: id : id поля, name : название поля, должно совпадать с названием соответствующего параметра лица, field_type : тип данных, default : значение по умолчанию. Если значение по умолчанию для поля больше '1e14 - 1', то его следует записывать в виде строки, т. е. "123123" вместо 123123.

Структура базы данных передается из файла `/etc/ffsecurity/tnt_schema.lua` в параметр `meta_scheme`.

findface-upload

Компонент **findface-upload** представляет собой веб-сервер на базе WebDAV nginx, который используется для хранения исходных изображений, миниатюр и нормализованных изображений лиц (получает их от компонента **findface-sf-api**).

По умолчанию исходные изображения, миниатюры и нормализованные изображения лиц хранятся в каталоге `/var/lib/ffupload/uploads/`.

Компонент `findface-upload` автоматически настраивается при установке. Настройка не предусмотрена.

Видеодетекция лиц: `findface-video-manager` и `findface-video-worker`

Примечание: Компонент `findface-video-worker` поставляется в пакетах с ускорением на CPU (`findface-video-worker-cpu`) и GPU (`findface-video-worker-gpu`).

В этом разделе:

- Функции `findface-video-manager`
- Функции `findface-video-worker`
- Настройка видеодетекции лиц
- Job-задания

Функции `findface-video-manager`

Сервис `findface-video-manager` является частью модуля видеодетекции лиц и используется для непосредственного управления детекцией лиц на видео.

Сервис `findface-video-manager` взаимодействует с `findface-video-worker` следующим образом:

- Обеспечивает `findface-video-worker` настройками и списком видеопотоков для обработки. Для этого он выдает `findface-video-worker` так называемое *job-задание*, задачу на обработку видео, которая содержит параметры конфигурации и сведения о видеопотоке.
- В распределенной системе распределяет видеопотоки (job-задания) по свободным экземплярам `findface-video-worker`.

Примечание: Параметры конфигурации, передаваемые через job-задания, имеют больший приоритет, чем аналогичные параметры в файле конфигурации `findface-video-manager`.

Для работы `findface-video-manager` требуется установленный сервис ETCD. ETCD представляет собой стороннее программное обеспечение, реализующее распределенное хранилище ключей `findface-video-manager`. Используется в качестве координационной службы в распределенной системе, обеспечивая отказоустойчивость работы видеодетектора лиц.

Полный список функций `findface-video-manager`:

- конфигурирование параметров видеодетектора лиц,
- управление списком видеопотоков для обработки,
- управление видеодетекцией лиц.

Функции `findface-video-worker`

Компонент `findface-video-worker` (или `findface-video-worker-gpu`) является частью модуля видеодетекции лиц и служит для обнаружения лиц «на лету» в видеопотоке или видеофайле. Он поддерживает большинство видеоформатов и кодеков, которые могут быть декодированы [FFmpeg](#).

Сервис `findface-video-worker` взаимодействует с сервисами `findface-video-manager` и `findface-facerouter` следующим образом:

- По запросу `findface-video-worker` получает от `findface-video-manager` job-задание с настройками и списком видеопотоков для обработки.
- Сервис `findface-video-worker` отправляет полученные нормализованные изображения лиц вместе с полными кадрами и метаданными, такими как рамка вокруг лица, ID камеры и время детекции, в сервис `findface-facerouter` для дальнейшей обработки.

Примечание: В FindFace Security функции `findface-facerouter` выполняются сервисом `findface-security`.

Полный список функций `findface-video-manager`:

- обнаружение лиц на видео,
- извлечение нормализованных изображений лиц,
- поиск наилучшего изображения лица,
- дедупликация кадров с лицом (только один кадр на каждое событие распознавания лица).

При обработки видео `findface-video-worker` последовательно использует следующие алгоритмы:

- **Детектор движения.** Данный алгоритм позволяет снизить потребление ресурсов, поскольку детектор лиц включается только по движению в кадре.
- **Детектор лиц.** Алгоритм детектирует, отслеживает и захватывает лица на видео. Может работать одновременно с несколькими лицами в кадре. С помощью встроенной нейронной сети выполняет поиск кадра с лучшим изображением лица. Как только лучшее изображение найдено, отправляет его в компонент `findface-facerouter`.

Подбор лучшего изображения лица может быть выполнен в одном из следующих режимов:

- Режим реального времени
- Буферный режим

Режим реального времени

В режиме реального времени `findface-video-worker` начинает отправлять изображения лица в компонент `findface-facerouter` на лету после появления лица в поле зрения видеокамеры. Доступны следующие опции отправки (см. `/etc/findface-video-manager.conf` и [Добавление камеры](#)):

- Если параметр `realtime_post_every_interval: true`, детектор лиц выбирает лучший кадр в течение каждого из последовательных промежутков времени, равных `realtime_post_interval`, и отправляет его в `findface-facerouter`.
- Если `realtime_post_every_interval: false`, детектор лиц выбирает лучшее изображение лица динамически:
 1. Сначала оценивается качество изображения лица. Если оно превышает некое предустановленное пороговое значение, то лицо отправляется в `findface-facerouter`.

2. Порог повышается после каждой отправки изображения лица в `findface-facerouter`. Каждый раз, когда детектор лиц получает изображение того же лица лучшего качества, оно отправляется.
 3. При исчезновении лица из поля зрения видеокамеры снова устанавливается пороговое значение по умолчанию.
- Если `realtime_post_first_immediately: true`, детектор лиц отправляет самый первый кадр непосредственно после появления лица в поле зрения видеокамеры. Последующие отправки зависят от значения `realtime_post_every_interval`.

Буферный режим

Буферный режим требует меньший объем дискового пространства по сравнению с режимом реального времени, поскольку для каждого лица компонент `findface-video-worker` отправляет только одно изображение из трека, но наивысшего качества.

Буферный режим включается через параметр `overall_only` (см. `/etc/findface-video-manager.conf` и *Добавление камеры*).

Настройка видеодетекции лиц

Настройка видеодетектора лиц выполняется через следующие файлы конфигурации:

1. Файл конфигурации компонента `findface-video-manager` `/etc/findface-video-manager.conf`:

```
listen: 127.0.0.1:18810
etcd:
  endpoints: 127.0.0.1:2379
  dial_timeout: 3s
kafka:
  enabled: false
  endpoints: 127.0.0.1:9092
master:
  lease_ttl: 10
  self_url: 127.0.0.1:18811
  self_url_http: 127.0.0.1:18810
rpc:
  listen: 127.0.0.1:18811
  heart_beat_timeout: 4s
router_url: http://127.0.0.1:18820/v0/frame
exp_backoff:
  enabled: false
  min_delay: 1s
  max_delay: 1m0s
  factor: 2
  flush_interval: 2m0s
ntls:
  enabled: false
  url: http://127.0.0.1:3185/
  update_interval: 1m0s
prometheus:
  jobs_processed_duration_buckets:
    - 1
    - 30
    - 60
```

(continues on next page)

(продолжение с предыдущей страницы)

```
- 500
- 1800
- 3600
- 21600
- .inf
job_scheduler_script: ''
stream_settings:
  ffmpeg_params: []
  md_threshold: 0.002
  md_scale: 0.3
  fd_frame_height: -1
  uc_max_time_diff: 30
  uc_max_dup: 3
  uc_max_avg_shift: 10
  det_period: 8
  realtime: false
  npersons: 4
  disable_drops: false
  tracker_threads: 4
  parse_sei: false
  image_arg: photo
  additional_headers: []
  additional_body: []
  api_timeout: 15000
  api_ssl_verify: true
  post_uniq: true
  min_score: -2
  min_d_score: -1000
  realtime_dly: 500
  realtime_post_perm: false
  rot: ''
  roi: ''
  draw_track: false
  send_track: 0
  min_face_size: 0
  max_face_size: 0
  overall: true
  only_norm: false
  max_candidates: 0
  jpeg_quality: 95
  ffmpeg_format: ''
stream_settings_gpu:
  play_speed: -1
  filter_min_quality: -2
  filter_min_face_size: 1
  filter_max_face_size: 8192
  normalized_only: false
  jpeg_quality: 95
  overall_only: false
  use_stream_timestamp: false
  ffmpeg_params: []
  router_timeout_ms: 15000
  router_verify_ssl: true
  router_headers: []
  router_body: []
  start_stream_timestamp: 0
```

(continues on next page)

(продолжение с предыдущей страницы)

```

imotion_threshold: 0
rot: ''
roi: ''
realtime_post_interval: 1
realtime_post_every_interval: true
ffmpeg_format: ''
disable_drops: false
router_full_frame_png: false
router_disable_normalized: false
crop_fullframe_rot: false
realtime_post_first_immediately: false

```

Пользовательская настройка `findface-video-manager` выполняется с использованием следующих параметров:

Опция	Описание
<code>router_url</code>	IP-адрес и порт сервера <code>findface-facerouter</code> , который получает обнаруженные лица из <code>findface-video-worker</code> . В FindFace Security функции <code>findface-facerouter</code> выполняет компонент <code>findface-security</code> . Значение по умолчанию: <code>http://127.0.0.1:18820/v0/frame</code> .
<code>etcd -> endpoints</code>	IP-адрес и порт сервиса <code>etcd</code> . Значение по умолчанию: <code>127.0.0.1:2379</code> .
<code>ntls -> enabled</code>	Если <code>true</code> , компонент <code>findface-video-manager</code> отправляет в компонент <code>findface-video-worker</code> задания только на обработку того количества видеокамер, которое указано в лицензии. Значение по умолчанию: <code>false</code> .
<code>ntls -> url</code>	IP-адрес и порт сервера <code>findface-ntls</code> . Значение по умолчанию: <code>http://127.0.0.1:3185/</code> .

Вы также можете использовать следующие параметры:

Примечание: В разделе файла `stream_settings-gpu` вы найдете настройки, общие для всех видеопотоков. Настройки в данном разделе работают как в GPU, так и CPU-конфигурации. Настройки определенного потока, переданные в `job`-задании, имеют приоритет над настройками в файле конфигурации (см. *Job-задания*).

Примечание: Раздел `stream_settings` файла устарел и необходим только для обратной совместимости.

Опция	Описание
<code>play_speed</code>	Если меньше нуля, то скорость не ограничивается. В остальных случаях поток читается со скоростью <code>play_speed</code> . Не применимо для потоков с камер видеонаблюдения.
<code>filter_min_quality</code>	Минимальное значение качества изображения лица, отправляемого компонентом <code>findface-video-worker</code> в компонент <code>findface-facerouter</code> (<code>findface-security</code> в стандартной конфигурации FindFace Security). Значение определяется эмпирически: отрицательные рациональные числа до 0. Реперные точки: 0 = наиболее качественные лица, -1 = хорошее качество, -2 = удовлетворительное качество, -5 = последующее распознавание лиц может быть неэффективным. Значение по умолчанию: -2.
<code>filter_min_facesize</code>	Определяет минимальный размер лица в пикселях. Лица меньшего размера не отправляются. Значение по умолчанию: 0 (фильтр выключен).
<code>filter_max_facesize</code>	Определяет максимальный размер лица в пикселях. Лица большего размера не отправляются. Значение по умолчанию: 0 (фильтр выключен).
<code>normalized_only</code>	Включает/отключает отправку только нормализованных лиц без исходных кадров. Значение по умолчанию: false.
<code>jpeg_quality</code>	Качество сжатия исходного кадра в JPEG. Значение по умолчанию: 95 % от исходного размера.
<code>overall_only</code>	Буферный режим. Отправлять для лица один кадр наилучшего качества. Значение по умолчанию: true.
<code>use_stream_timestamp</code>	Включает/отключает отправку временных меток полученных из потока. Если false, отправлять текущие дату и время.
<code>ffmpeg_params</code>	Список ffmpeg-параметров видеопотока со значениями в виде массива ключ=значение: ["rtsp_transpotr=tcp", .., "ss=00:20:00"]. Полный список параметров на сайте FFMPEG . Значение по умолчанию: параметры не указаны.
<code>router_timeout</code>	Время ожидания в миллисекундах ответа от компонента <code>findface-facerouter</code> (<code>findface-security</code> в стандартной конфигурации FindFace Security) на API-запрос компонента <code>findface-video-worker</code> . Если время ожидания истекло, регистрируется ошибка. Значение по умолчанию: 15000.
<code>router_verify_ssl</code>	Включает/отключает проверку подписи SSL-сертификата при взаимодействии по <code>https</code> <code>findface-video-worker</code> с компонентом <code>findface-facerouter</code> (<code>findface-security</code> в стандартной конфигурации FindFace Security). Значение по умолчанию: true. Если false, может быть принят самоподписанный сертификат.
<code>router_headers</code>	Массив дополнительных заголовков в POST-запросе с изображением лица в формате ["ключ=значение"]. По умолчанию дополнительные заголовки не передаются.
<code>router_body</code>	Массив дополнительных полей в POST-запросе с изображением лица в формате ["ключ=значение"]. По умолчанию дополнительные поля не передаются.
<code>start_stream_timestamp</code>	Смещение указанного количества секунд к временным меткам из потока.
<code>imotion_threshold</code>	Минимальная интенсивность движения, которая будет регистрироваться детектором движения. Пороговое значение определяется эмпирически. Реперные точки: 0 = детектор выключен, 0.002 = значение по умолчанию, 0.05 = минимальная интенсивность слишком высока, чтобы зарегистрировать движение.
<code>rot</code>	Включает детектирование и отслеживание лиц только внутри заданной прямоугольной области $W \times H + X + Y$. Используйте данную опцию, чтобы уменьшить нагрузку на <code>findface-video-worker</code> . По умолчанию область не задана.
<code>roi</code>	Включает отправку на Сервер лиц, обнаруженных только внутри интересующей области $W \times H + X + Y$. По умолчанию область не задана.
<code>realtime_post_timeout</code>	Интервал в режиме реального времени. Период времени в секундах, в течение которого детектор лиц выбирает лучший кадр из трека и отправляет его в компонент <code>findface-facerouter</code> . Значение по умолчанию: 1.
<code>realtime_post_timeout_interval</code>	Интервал в режиме реального времени. Включает отправку лучшего кадра

1. Если вы выбрали пакет `findface-video-worker-cpu` с ускорением на CPU, используйте файл конфигурации `/etc/findface-video-worker-cpu.ini`:

```
## read streams from file, do not use VideoManager
## type:string env:CFG_INPUT longopt:--input
input =
## exit on first finished job, only when --input specified
## type:bool env:CFG_EXIT_ON_FIRST_FINISHED longopt:--exit-on-first-finished
exit_on_first_finished = false
## batch size
## type:number env:CFG_BATCH_SIZE longopt:--batch-size
batch_size = 4
## http server port for metrics, 0=do not start server
## type:number env:CFG_METRICS_PORT longopt:--metrics-port
metrics_port = 0
## resize scale, 1=do not resize
## type:double env:CFG_RESIZE_SCALE longopt:--resize-scale
resize_scale = 1.000000
## maximum number of streams
## type:number env:CFG_CAPACITY longopt:--capacity
capacity = 10
## command to obtain videomanager's grpc ip:port
## type:string env:CFG_MGR_CMD longopt:--mgr-cmd
mgr_cmd =
## videomanager grpc ip:port
## type:string env:CFG_MGR_STATIC longopt:--mgr-static
mgr_static = 127.0.0.1:18811
## ntlS server ip:port
## type:string env:CFG_NTLS_ADDR longopt:--ntls-addr
ntls_addr = 127.0.0.1:3133
## debug: save faces to dir
## type:string env:CFG_SAVE_DIR longopt:--save-dir
save_dir =
## minimum face size
## type:number env:CFG_MIN_FACE_SIZE longopt:--min-face-size
min_face_size = 60
## preinit detector for specified resolutions: "640x480;1920x1080"
## type:string env:CFG_RESOLUTIONS longopt:--resolutions
resolutions =
## worker labels: labels = k=v;group=enter
## type:string env:CFG_LABELS longopt:--labels
labels =
## use timestamps from SEI packet
## type:bool env:CFG_USE_TIME_FROM_SEI longopt:--use-time-from-sei
use_time_from_sei = false
## reader frame buffer size
## type:number env:CFG_FRAME_BUFFER_SIZE longopt:--frame-buffer-size
frame_buffer_size = 128
## skip count
## type:number env:CFG_SKIP_COUNT longopt:--skip-count
skip_count = 2
#-----
[streamer]
#-----
## streamer/shots webserver port, 0=disabled
## type:number env:CFG_STREAMER_PORT longopt:--streamer-port
port = 18999
## streamer url - how to access this worker on streamer_port
```

(continues on next page)

(продолжение с предыдущей страницы)

```

## type:string env:CFG_STREAMER_URL longopt:--streamer-url
url = 127.0.0.1:18999
#-----
[liveness]
#-----
## path to liveness fnk
## type:string env:CFG_LIVENESS_FNK longopt:--liveness-fnk
fnk =
## path to normalization for liveness
## type:string env:CFG_LIVENESS_NORM longopt:--liveness-norm
norm =
## liveness internal algo param
## type:double env:CFG_LIVENESS_INTERVAL longopt:--liveness-interval
interval = 1.000000
## liveness internal algo param
## type:number env:CFG_LIVENESS_STDEV_CNT longopt:--liveness-stdev-cnt
stdev_cnt = 1
#-----
[send]
#-----
## posting faces threads
## type:number env:CFG_SEND_THREADS longopt:--send-threads
threads = 8
## posting faces maximum queue size
## type:number env:CFG_SEND_QUEUE_LIMIT longopt:--send-queue-limit
queue_limit = 256
#-----
[tracker]
#-----
## interpolate undetected bboxes in track
## type:bool env:CFG_TRACKER_INTERPOLATE_BBOXES longopt:--tracker-interpolate-bboxes
interpolate_bboxes = true
## max face miss duration, sec
## type:double env:CFG_TRACKER_MISS_INTERVAL longopt:--tracker-miss-interval
miss_interval = 1.000000
## overlap threshold
## type:double env:CFG_TRACKER_OVERLAP_THRESHOLD longopt:--tracker-overlap-threshold
overlap_threshold = 0.250000
#-----
[models]
#-----
## path to pedestrian detector fnk
## type:string env:CFG_MODELS_BODY_DETECTOR longopt:--models-body-detector
body_detector =
## path to detector fnk
## type:string env:CFG_MODELS_DETECTOR longopt:--models-detector
detector = /usr/share/findface-data/models/facedet/cheetah_fast.cpu.fnk
## path to quality fnk
## type:string env:CFG_MODELS_QUALITY longopt:--models-quality
quality = /usr/share/findface-data/models/faceattr/quality.v0.cpu.fnk
## path to norm for quality fnk
## type:string env:CFG_MODELS_NORM_QUALITY longopt:--models-norm-quality
norm_quality = /usr/share/findface-data/models/facenorm/bee_fast.cpu.fnk
## path to norm200 fnk, for face send
## type:string env:CFG_MODELS_NORM_200 longopt:--models-norm-200
norm_200 = /usr/share/findface-data/models/facenorm/bee.v2.cpu.fnk

```

(continues on next page)

(продолжение с предыдущей страницы)

```
## path to norm_crop2x fnk, for face send
## type:string env:CFG_MODELS_NORM_CROP2X longopt:--models-norm-crop2x
norm_crop2x = /usr/share/findface-data/models/facenorm/crop2x.v2_maxsize400.cpu.fnk
## path to cache directory
## type:string env:CFG_MODELS_CACHE_DIR longopt:--models-cache-dir
cache_dir = /var/cache/findface/models_cache
```

Если вы выбрали пакет `findface-video-worker-gpu` с ускорением на GPU, используйте файл конфигурации `/etc/findface-video-worker-gpu.ini`.

```
## cuda device number
## type:number env:CFG_DEVICE_NUMBER longopt:--device-number
device_number = 0
## apply getprop patch
## type:bool env:CFG_PATCH_GETPROP longopt:--patch-getprop
patch_getprop = false
## read streams from file, do not use VideoManager
## type:string env:CFG_INPUT longopt:--input
input =
## exit on first finished job, only when --input specified
## type:bool env:CFG_EXIT_ON_FIRST_FINISHED longopt:--exit-on-first-finished
exit_on_first_finished = false
## batch size
## type:number env:CFG_BATCH_SIZE longopt:--batch-size
batch_size = 8
## http server port for metrics, 0=do not start server
## type:number env:CFG_METRICS_PORT longopt:--metrics-port
metrics_port = 0
## resize scale, 1=do not resize
## type:double env:CFG_RESIZE_SCALE longopt:--resize-scale
resize_scale = 1.000000
## maximum number of streams
## type:number env:CFG_CAPACITY longopt:--capacity
capacity = 30
## command to obtain videomanager's grpc ip:port
## type:string env:CFG_MGR_CMD longopt:--mgr-cmd
mgr_cmd =
## videomanager grpc ip:port
## type:string env:CFG_MGR_STATIC longopt:--mgr-static
mgr_static = 127.0.0.1:18811
## ntlS server ip:port
## type:string env:CFG_NTLS_ADDR longopt:--ntls-addr
ntls_addr = 127.0.0.1:3133
## debug: save faces to dir
## type:string env:CFG_SAVE_DIR longopt:--save-dir
save_dir =
## minimum face size
## type:number env:CFG_MIN_FACE_SIZE longopt:--min-face-size
min_face_size = 60
## preinit detector for specified resolutions: "640x480;1920x1080"
## type:string env:CFG_RESOLUTIONS longopt:--resolutions
resolutions =
## worker labels: labels = k=v;group=enter
## type:string env:CFG_LABELS longopt:--labels
labels =
## use timestamps from SEI packet
```

(continues on next page)

(продолжение с предыдущей страницы)

```

## type:bool env:CFG_USE_TIME_FROM_SEI longopt:--use-time-from-sei
use_time_from_sei = false
## reader frame buffer size
## type:number env:CFG_FRAME_BUFFER_SIZE longopt:--frame-buffer-size
frame_buffer_size = 128
## skip count
## type:number env:CFG_SKIP_COUNT longopt:--skip-count
skip_count = 0
#-----
[streamer]
#-----
## streamer/shots webserver port, 0=disabled
## type:number env:CFG_STREAMER_PORT longopt:--streamer-port
port = 0
## streamer url - how to access this worker on streamer_port
## type:string env:CFG_STREAMER_URL longopt:--streamer-url
url = 127.0.0.1:9999
#-----
[liveness]
#-----
## path to liveness fnk
## type:string env:CFG_LIVENESS_FNK longopt:--liveness-fnk
fnk =
## path to normalization for liveness
## type:string env:CFG_LIVENESS_NORM longopt:--liveness-norm
norm =
## liveness internal algo param
## type:double env:CFG_LIVENESS_INTERVAL longopt:--liveness-interval
interval = 1.000000
## liveness internal algo param
## type:number env:CFG_LIVENESS_STDEV_CNT longopt:--liveness-stdev-cnt
stdev_cnt = 1
#-----
[send]
#-----
## posting faces threads
## type:number env:CFG_SEND_THREADS longopt:--send-threads
threads = 8
## posting faces maximum queue size
## type:number env:CFG_SEND_QUEUE_LIMIT longopt:--send-queue-limit
queue_limit = 256
#-----
[tracker]
#-----
## interpolate undetected bboxes in track
## type:bool env:CFG_TRACKER_INTERPOLATE_BBOXES longopt:--tracker-interpolate-bboxes
interpolate_bboxes = true
## max face miss duration, sec
## type:double env:CFG_TRACKER_MISS_INTERVAL longopt:--tracker-miss-interval
miss_interval = 1.000000
## overlap threshold
## type:double env:CFG_TRACKER_OVERLAP_THRESHOLD longopt:--tracker-overlap-threshold
overlap_threshold = 0.250000
#-----
[models]
#-----

```

(continues on next page)

(продолжение с предыдущей страницы)

```

## path to pedestrian detector fnk
## type:string env:CFG_MODELS_BODY_DETECTOR longopt:--models-body-detector
body_detector =
## path to detector fnk
## type:string env:CFG_MODELS_DETECTOR longopt:--models-detector
detector = /usr/share/findface-data/models/facedet/cheetah_fast.gpu.fnk
## path to quality fnk
## type:string env:CFG_MODELS_QUALITY longopt:--models-quality
quality = /usr/share/findface-data/models/faceattr/quality.v0.gpu.fnk
## path to norm for quality fnk
## type:string env:CFG_MODELS_NORM_QUALITY longopt:--models-norm-quality
norm_quality = /usr/share/findface-data/models/facenorm/bee_fast.gpu.fnk
## path to norm200 fnk, for face send
## type:string env:CFG_MODELS_NORM_200 longopt:--models-norm-200
norm_200 = /usr/share/findface-data/models/facenorm/bee.v2.gpu.fnk
## path to norm_crop2x fnk, for face send
## type:string env:CFG_MODELS_NORM_CROP2X longopt:--models-norm-crop2x
norm_crop2x = /usr/share/findface-data/models/facenorm/crop2x.v2_maxsize400.gpu.fnk
## path to cache directory
## type:string env:CFG_MODELS_CACHE_DIR longopt:--models-cache-dir
cache_dir = /var/cache/findface/models_cache
#-----
[video_decoder]
#-----
## decode video on cpu
## type:bool env:CFG_VIDEO_DECODER_CPU longopt:--video-decoder-cpu
cpu = false

```

Пользовательская настройка findface-video-worker на CPU/GPU выполняется с использованием следующих параметров:

CPU	GPU	Описание
ntls-addr		IP-адрес и порт сервера <code>findface-ntls</code> .
mgr-static		IP-адрес сервера <code>findface-video-manager</code> , который обеспечивает <code>findface-video-worker</code> настройками и списком видеопотоков для обработки.
capacity		Максимальное количество видеопотоков, обрабатываемых <code>findface-video-worker</code> .
mgr-cmd		Команда для получения IP-адреса компонента <code>findface-video-manager</code> (опционально вместо <code>mgr-static</code>).
labels		Метки, используемые для привязки экземпляра <code>findface-video-worker</code> к определенной группе камер. См. <i>Привязка группы камер к экземпляру findface-video-worker</i> .
streamer -> port, url		IP-адрес и порт для доступа к <i>видеостене</i> .
liveness -> fnk		Путь к детектору <i>живых</i> лиц (Liveness).
liveness -> norm		Путь к нормализатору, используемому в детекторе <i>живых</i> лиц (Liveness).
input		Обрабатывать видеопотоки из файла, игнорируя данные потоков, поступающие от <code>findface-video-manager</code> .
exit_on_first_finished		(Если не указано) Выйти после завершения первого job-задания.
resize_scale		Масштабировать видеокадры с заданным коэффициентом.
save_dir		(Для отладки) Сохранять обнаруженные лица в заданный каталог.
min_face_size		Минимальный обнаруживаемый размер лица.
Н/п	device_id	Идентификатор используемого GPU-устройства.
Н/п	models_dir	Путь к каталогу с моделями GPU-детектора. В противном случае используйте данные из секции <code>[models]</code> .
Н/п	cpu	При необходимости декодировать видео на CPU.

Job-задания

Сервис `findface-video-manager` выдает `findface-video-worker` так называемое job-задание, задачу на обработку видео, которая содержит параметры конфигурации и сведения о видеопотоке.

Содержимое типичного job-задания показано в примере ниже.

```
curl http://127.0.0.1:18810/job/1 | jq
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
100 2045    100 2045    0     0  199k      0 --:--:-- --:--:-- --:--:-- 221k
[
  {
    "id": "ffsec_video_source",
    "enabled": false,
    "stream_url": "file:///var/lib/ffsecurity/uploads/videos/2.bin",
    "labels": {
      "camera_group_id": "1",
      "camera_id": "0",
      "video_source_id": "2"
    },
    "router_url": "http://127.0.0.1/video-detector/frame",
    "single_pass": true,
    "stream_settings": {
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    "ffmpeg_params": [],
    "md_threshold": 0.002,
    "md_scale": 0.3,
    "fd_frame_height": -1,
    "uc_max_time_diff": 30,
    "uc_max_dup": 3,
    "uc_max_avg_shift": 10,
    "det_period": 8,
    "realtime": false,
    "npersons": 4,
    "disable_drops": true,
    "tracker_threads": 4,
    "parse_sei": false,
    "image_arg": "photo",
    "additional_headers": [
      "Authorization=Token 08ad0f283812584b731c62f7d14cb7d0"
    ],
    "additional_body": [],
    "api_timeout": 15000,
    "api_ssl_verify": true,
    "post_uniq": true,
    "min_score": -2,
    "min_d_score": -1000,
    "realtime_dly": 500,
    "realtime_post_perm": false,
    "rot": "",
    "roi": "",
    "draw_track": false,
    "send_track": 0,
    "min_face_size": 0,
    "max_face_size": 0,
    "overall": true,
    "only_norm": false,
    "max_candidates": 0,
    "jpeg_quality": 95,
    "ffmpeg_format": ""
  },
  "stream_settings_gpu": {
    "play_speed": -1,
    "filter_min_quality": -2,
    "filter_min_face_size": 1,
    "filter_max_face_size": 8192,
    "normalized_only": false,
    "jpeg_quality": 95,
    "overall_only": false,
    "use_stream_timestamp": false,
    "ffmpeg_params": [],
    "router_timeout_ms": 15000,
    "router_verify_ssl": true,
    "router_headers": [
      "Authorization=Token 08ad0f283812584b731c62f7d14cb7d0"
    ],
    "router_body": [],
    "start_stream_timestamp": 0,
    "imotion_threshold": 0,
    "rot": "",

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    "roi": "",
    "realtime_post_interval": 1,
    "realtime_post_every_interval": true,
    "ffmpeg_format": "",
    "disable_drops": true,
    "router_full_frame_png": false,
    "router_disable_normalized": false,
    "crop_fullframe_rot": false,
    "realtime_post_first_immediately": false
  },
  "status": "DISABLED",
  "status_msg": "",
  "statistic": {
    "processed_duration": 0,
    "faces_posted": 0,
    "faces_failed": 0,
    "faces_not_posted": 0,
    "processing_fps": 0,
    "frames_dropped": 0,
    "frames_processed": 0,
    "frames_imotion_skipped": 0,
    "decoding_soft_errors": 0,
    "job_starts": 0
  },
  "restream_url": "ws://127.0.0.1:18999/stream/ffsec_video_source",
  "shots_url": "http://127.0.0.1:18999/shot/ffsec_video_source",
  "worker_id": "",
  "version": "brb4165r6c591lrq2a40"
}
]

```

Каждое job-задание имеет следующие параметры:

- **id:** id job-задания.
- **enabled:** статус активности.
- **stream_url:** URL/адрес видеопотока или файла для обработки.
- **labels:** метки, по которым будет осуществляться обработка обнаруженных лиц в компоненте **findface-facerouter** (**findface-security** в стандартной конфигурации FindFace Security).
- **single_pass:** если true (по умолчанию false), то не перезапускать обработку потока в случае ошибки.
- **router_url:** IP-адрес и порт компонента **findface-facerouter** (**findface-security** в стандартной конфигурации FindFace Security), в который компонент **findface-video-worker** отправляет обнаруженные лица для последующей обработки.
- **stream_settings:** используется только для обратной совместимости.
- **stream_settings_gpu:** параметры видеопотока, дублирующие *параметры* в файле конфигурации **findface-video-manager** (обладая при этом приоритетом).
- **status:** статус job-задания.
- **status_msg:** дополнительная информация о статусе job-задания.
- **statistic:** статистика выполнения job-задания (продолжительность процесса обработки, количество отправленных и неотправленных лиц, кадровая частота обработки, количество обработанных и пропущенных кадров, время начала обработки и т. д.).

- `worker_id`: id экземпляра `findface-video-worker`, выполняющего job-задание.

findface-ntls

Локальный сервер лицензирования `findface-ntls` – это сервис, который устанавливается на выбранном физическом сервере и служит для верификации лицензии FindFace. Для верификации используются следующие источники:

- Глобальный сервер лицензий NtechLab. Служит для лицензирования в сетях с доступом в интернет, в том числе с доступом через прокси-сервер.
- Ключ аппаратной защиты. Служит для лицензирования в закрытых сетях.

Для управления `findface-ntls` используйте основной веб-интерфейс FindFace Security. Доступны следующие возможности:

- просмотр списка приобретенных функций,
- просмотр списка текущих ограничений,
- загрузка файла лицензии,
- просмотр списка подключенных в данный момент компонентов.

Лицензируются следующие компоненты:

- `findface-tarantool-server`,
- `findface-extraction-api`,
- `findface-video-manager`,
- `findface-video-worker`.

Важно: После разрыва соединения между сервером лицензирования `findface-ntls` и лицензируемым компонентом или между сервером лицензирования `findface-ntls` и глобальным сервером лицензирования, время работы компонента составляет 6 часов.

Настройка компонента `findface-ntls` выполняется через файл конфигурации `/etc/findface-ntls.cfg`.

```
# Listen address of NTLS server where services will connect to.
# The format is IP:PORT
# Use 0.0.0.0:PORT to listen on all interfaces
# This parameter is mandatory and may occur multiple times
# if you need to listen on several specific interfaces or ports.
listen = 127.0.0.1:3133

# Directory with license files.
# NTLS use most recently generated one.
# Note: "recentness" of a license file is detected not by
#       mtime/ctime but from its internal structure.
#
# This parameter is mandatory and must occur exactly once.
license-dir = /opt/ntech/license

# You can specify proxy which NTLS will use to access
# global license server. The syntax is the same that is used by curl.
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# Proxy is optional
#proxy = http://192.168.1.1:12345

# This is bind address for NTLS web-interface.
# Note: there're no authorization or access restriction mechanisms
#       in NTLS UI. If you need one, consider using nginx as proxy
#       with .htaccess / ip-based ACLs.
# This parameter may be specified multiple times.
ui = 127.0.0.1:3185
```

Пользовательская настройка **findface-ntls** выполняется с использованием следующих параметров:

Параметр	Описание
listen	IP-адрес сервера, с которого осуществляется обращение лицензируемых компонентов в findface-ntls . Для того чтобы разрешить обращение с любого IP-адреса, установите значение 0.0.0.0:3133 .
license_data	Путь к каталогу для хранения файла лицензии.
proxy	IP-адрес и порт прокси-сервера (опционально).
ui	IP-адрес сервера, с которого будет доступен веб-интерфейс findface-ntls . Для того чтобы разрешить доступ со всех адресов, измените значение на “0.0.0.0” .

findface-security

Компонент **findface-security** обеспечивает доступ конечного пользователя к функциям ядра FindFace. Отвечает за взаимодействие между ядром FindFace Core и веб-интерфейсом, а также функционирование системы как единого целого, реализует HTTP- и веб-сокеты (вместе с Django), обновление базы данных и *вебхуки*.

Компонент **findface-security** также выполняет функции компонента **findface-facerouter** (часть ядра FindFace), задавая правила обработки обнаруженных лиц. Он получает рамку с лицом и нормализованное изображение лица вместе с исходным кадром и другими данными (например, датой и временем детекции) от сервиса **findface-video-worker** и перенаправляет их для дальнейшей обработки в сервис **findface-sf-api**.

Настройка компонента **findface-security** выполняется через файл конфигурации **/etc/ffsecurity/config.py**.

```
sudo vi /etc/ffsecurity/config.py

# =====
# FindFace Security configuration file
# =====
#
# This config file is written in Python's syntax and interpreted at FindFace Security
# service startup. You have to restart the service in order to apply changes.
#
# If you have any questions or suggestions, please contact us at support@ntechlab.com
# =====
# GENERAL SETTINGS
# =====
# enables additional logs
DEBUG = False
```

(continues on next page)

(продолжение с предыдущей страницы)

```

# media files directory
MEDIA_ROOT = "/var/lib/ffsecurity/uploads"
# static files directory
STATIC_ROOT = "/var/lib/ffsecurity/static"
# language code
LANGUAGE_CODE = 'en-us'
# time zone
TIME_ZONE = 'UTC'
# Database is used by FindFace Security to store cameras,
# camera groups, watchlists and so on. Only PostgreSQL is supported.
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'DISABLE_SERVER_SIDE_CURSORS': True,
        'NAME': 'ffsecurity',
        'PORT': 5439, 'USER': 'ntech', 'PASSWORD': '1Y4KrQbotHVFdZ0Ed5z8EBvDUzSsmw2o'
    }
}

# Signature key for session encryption
# Use pwgen -sncy 50 1/tr "" "." to generate your own unique key
SECRET_KEY = 'd1c22e3b368f2f8be4caa95b3540ce0c'
# =====
# FINDFACE SECURITY SETTINGS
# =====
# SERVICE_EXTERNAL_ADDRESS is prioritized for FFSecurity webhooks and Genetec plugin.
# EXTERNAL_ADDRESS is used instead if SERVICE_EXTERNAL_ADDRESS is not provided.
# You must provide either SERVICE_EXTERNAL_ADDRESS or EXTERNAL_ADDRESS in order
# to be able to work with FFSecurity webhooks and Genetec plugin.
SERVICE_EXTERNAL_ADDRESS = 'http://127.0.0.1'
# EXTERNAL_ADDRESS is used to access objects created inside FFSecurity via external
↳ links.
EXTERNAL_ADDRESS = ''
# - Base FFSecurity settings -
FFSECURITY = {
    # findface-video-worker authorization token
    'VIDEO_DETECTOR_TOKEN': 'fdc0fea20cf0a734c5e2de1886fc07b4',
    # base face matching confidence threshold
    'CONFIDENCE_THRESHOLD': 0.739,
    # episodes specific matching threshold that is used to join faces in an episode
    'EPISODES_THRESHOLD': 0.689,
    # minimum face quality sufficient to add it to a dossier
    'MINIMUM_DOSSIER_QUALITY': -2,
    # skip all unmatched faces
    'IGNORE_UNMATCHED': False,
    # matched events older than EVENTS_MAX_MATCHED_AGE will be automatically
    # deleted (every night at 1:17 am by default)
    'EVENTS_MAX_MATCHED_AGE': 30,
    # same as above but for unmatched events
    'EVENTS_MAX_UNMATCHED_AGE': 30,
    # same as EVENTS_MAX_MATCHED_AGE but for matched full frame images only (thumbnails
↳ won't be deleted)
    'EVENTS_MAX_FULLFRAME_UNMATCHED_AGE': 30,
    # same as above but for unmatched full frame images only (thumbnails won't be
↳ deleted)
    'EVENTS_MAX_FULLFRAME_MATCHED_AGE': 30,
    # NTLS licence server url

```

(continues on next page)

(продолжение с предыдущей страницы)

```

'NTLS_HTTP_URL': 'http://127.0.0.1:3185',
# findface-video-worker face posting address,
# it must be set to either FFSecurity EXTERNAL_ADDRESS (by default)
# or findface-facerouter url (in some specific cases)
'ROUTER_URL': 'http://127.0.0.1',
# send serialized dossiers, dossier-lists, camera and camera groups in webhooks
'VERBOSE_WEBHOOKS': False,
'THUMBNAIL_JPEG_QUALITY': 75,
# FFServer services urls
'VIDEO_MANAGER_ADDRESS': 'http://127.0.0.1:18810',
'SF_API_ADDRESS': 'http://127.0.0.1:18411',
'FFCOUNTER_ADDRESS': 'http://127.0.0.1:17300',
# additional events features.
# make sure that corresponding extractors
# are licensed and enabled at findface-extraction-api config file.
# available features are: gender, age, emotions, beard, glasses, medmask.
'EVENTS_FEATURES': [],
# feature specific confidence thresholds
'LIVENESS_THRESHOLD': 0.75,
'EMOTIONS_THRESHOLD': 0.25,
'BEARD_THRESHOLD': 0.7,
# -- Persons configuration --
# rrule (recurrence rule) for scheduling persons clusterization
'PERSONS_CLUSTERIZATION_SCHEDULE': 'RRULE:FREQ=DAILY;INTERVAL=1;WKST=MO;BYHOUR=0;
↳BYMINUTE=0',
# face to person matching confidence threshold
'PERSONS_CONFIDENCE_THRESHOLD': 0.739,
# minimum face quality for handle face in person service
'PERSON_EVENT_MIN_QUALITY': -2,
# counters full frame saving options:
# `always` - save always
# `detect` - save only if faces or silhouettes have been detected
# `never` - never save full frames
'COUNTERS_SAVE_FULLFRAME': 'always',
'COUNTERS_FULLFRAME_JPEG_QUALITY': 75,
'COUNTERS_THUMBNAIL_JPEG_QUALITY': 75,
# -- Optional parameters --
# Edit CUSTOM_FIELDS section to customize dossier content.
# Below is an example for integration FindFace Security with Sigur.
# 'CUSTOM_FIELDS': {
#     'dossier_meta': {
#         'items': [
#             {
#                 'name': 'personid',
#                 'default': '',
#                 'label': 'PersonID',
#                 'display': ['list', 'form'],
#                 'description': 'Sigur person ID'
#             },
#             {
#                 'name': 'firstname',
#                 'default': '',
#                 'label': 'First Name',
#                 'display': ['list', 'form'],
#                 'description': 'Sigur first name'
#             }
#         ]
#     }
# }

```

(continues on next page)

(продолжение с предыдущей страницы)

```

#         {
#             'name': 'lastname',
#             'default': '',
#             'label': 'Last Name',
#             'display': ['list', 'form'],
#             'description': 'Sigur last name'
#         },
#         {
#             'name': 'version',
#             'default': '',
#             'label': 'Version',
#             'display': ['list', 'form'],
#             'description': 'Sigur photo version'
#         }
#     ],
#     'filters': [
#         {
#             'name': 'personid',
#             'label': 'Sigur person ID filter',
#             'field': 'personid'
#         }
#     ]
# }
# },
# maximum event age in seconds than could be added to an episode.
# 'EPISODE_SEARCH_INTERVAL': 60,
# If none of these events matched, new episode is created.
# maximum episode duration (episode is closed after)
# 'EPISODE_MAX_DURATION': 300,
# if no new event added to an episode during this timeout, episode will be closed.
# 'EPISODE_EVENT_TIMEOUT': 30,
# maximum created thumbnail width
# 'THUMBNAIL_MAX_WIDTH': 320,
# social backend url. Contact support for additional information.
# 'SOCIAL_BACKEND': None,
# 'SOCIAL_HEADERS': {},
# unacknowledged events notification interval
# 'UNACKNOWLEDGED_NOTIFY_INTERVAL': 1,
}
# - FindFace Security user interface configuration dictionary -
FFSECURITY_UI_CONFIG = {
    "event": {
        "features": {
            "f_gender_class": ["male", "female"],
            "age": {
                "f_age_gte": "",
                "f_age_lte": ""
            },
        },
        "f_emotions_class": ["angry", "disgust", "fear", "happy", "sad", "surprise",
→ "neutral"],
        "f_glasses_class": ["none", "eye", "sun"],
        "f_beard_class": ["none", "beard"],
        "f_liveness_class": ["real", "fake"],
        "f_medmask_class": ["none", "correct"],
    }
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

}
# -- ASGI-server configuration --
# consult support before changing these settings.
# per worker thread pool size.
ASGI_THREADS = 32
UVICORN_SETTINGS = {
    # worker processes count, 'auto' sets it to logical cpu count
    'workers': 'auto',
    'host': 'localhost',
    'port': 8002,
    # websocket worker processes count,
    # 'auto' sets it to logical cpu count, but not more than 8.
    'ws-workers': 'auto',
    'ws-host': 'localhost',
    'ws-port': 8003,
}
# disable unused services to increase
# overall system performance in some cases.
SERVICES = {
    "ffsecurity": {
        "episodes": True,
        "webhooks": True,
        "persons": False,
        "counters": True,
    }
}
}
# =====
# FINDFACE SECURITY PLUGINS
# =====
# Uncomment lines below to enable plugins. Please consult documentation for
# a plugin specific settings.
# ===== Axon =====
# INSTALLED_APPS.append('ffsecurity_axon')
# FFSECURITY['AXXON'] = [
#     {
#         'name': 'server_name',
#         'api': 'http://example.com/',
#         'rtsp': 'rtsp://example.com:554/',
#         'user': 'user',
#         'password': 'password',
#     }
# ]
# FFSECURITY_UI_CONFIG['dossier'] = {
#     'video': True,
# }
# ===== Genetec =====
# INSTALLED_APPS.append('ffsecurity_genetec')
# ===== Sova =====
# INSTALLED_APPS.append('ffsecurity_sova')
# ===== CryptoPRO authentication =====
# INSTALLED_APPS.append('ffsecurity_cproauth')
# REST_FRAMEWORK['DEFAULT_AUTHENTICATION_CLASSES'] = [
#     'ffsecurity.auth.TokenAuthentication',
#     'ffsecurity_cproauth.auth.CryptoProOrTokenAuthentication'
# ]
# ===== DossierLists sync =====

```

(continues on next page)

(продолжение с предыдущей страницы)

```
# INSTALLED_APPS.append('ffsecurity_sync')
# token must be identical on master and slave
# use pwgen -s 64 1
# SYNC_TOKEN = 'change_me'
# SYNC_TIME = {
#     # 24 hour format
#     'hour': 3,
#     'minute': 0,
# }
```

Пользовательская настройка **findface-security** выполняется с использованием следующих параметров:

Параметр	Описание
EXTERNAL_ADDRESS	IP-адрес или URL, который будет использован для доступа к веб-интерфейсу FindFace Security.
SERVICE_EXTERNAL_ADDRESS	IP-адрес, приоритетно используемый в вебхуках и интеграции с Genetec.
VIDEO_DETECTION_TOKEN	Токен и укажите его в данном параметре, чтобы авторизовать модуль видеодетекции лиц.
VIDEO_MANAGER_ADDRESS	Адрес findface-video-manager.
EVENTS_MAX_MATCHED_AGE	Возраст совпадений с совпадением, при достижении которого должно происходить его автоматическое удаление из базы данных.
EVENTS_MAX_UNMATCHED_AGE	Возраст несовпадений без совпадения, при достижении которого должно происходить его автоматическое удаление из базы данных.
EVENTS_MAX_FULLFRAME_MATCHED_AGE	Возраст совпадений, только для полных кадров.
EVENTS_MAX_FULLFRAME_UNMATCHED_AGE	Возраст несовпадений, только для полных кадров.
NTLS_HTTP_URL	Адрес сервера findface-ntls.
ROUTER_URL	Адрес сервера findface-security, который будет получать обнаруженные на видео лица от экземпляров findface-video-worker. Адрес указывается внутренний или внешний, в зависимости от сети, в которой findface-video-worker взаимодействует с findface-security.
EXTRACTION_API	Адрес сервера findface-extraction-api.
SF_API_ADDRESS	Адрес сервера findface-sf-api.
IGNORE_UNMATCHED	Делать запись события в базу данных, если обнаруженное лицо отсутствует в списках наблюдения (верификация дала отрицательный результат). Данную настройку рекомендуется использовать при большом количестве посетителей. Пороговая степень схожести при верификации лиц определяется параметром CONFIDENCE_THRESHOLD.
CONFIDENCE_THRESHOLD	Степень схожести для верификации лиц в событиях.
EPISODES_THRESHOLD	Степень схожести для верификации лиц в эпизодах.
MINIMUM_DOSSIER_QUALITY	Минимальное количество лица на фотографии в досье. Если качество лица хуже минимального, пользователь не сможет загрузить такую фотографию в досье. Прямые изображения лиц анфас считаются наиболее качественными. Им соответствуют значения вблизи 0, как правило, отрицательные (такие как -0.00067401276, например). Перевернутые лица и лица, повернутые под большими углами, характеризуются отрицательными значениями от -5 и меньше. По умолчанию 'MINIMUM_DOSSIER_QUALITY': -2, что соответствует среднему качеству.
EVENTS_FEATURES	Укажите здесь модели для распознавания атрибутов лица, которые вы прописали в файле конфигурации findface-extraction-api.
LIVENESS_THRESHOLD	Порог живых лиц оценивает живость лица с определенной достоверностью. В зависимости от порогового значения достоверности, он возвращает бинарный результат Живой человек или Изображение .
EMOTIONS_THRESHOLD	Порог эмоций оценивается с определенной достоверностью. В зависимости от порогового значения достоверности, система различает эмоциональные лица от нейтральных.
BEARD_THRESHOLD	Наличие на лице бороды оценивается с определенной достоверностью. В зависимости от порогового значения достоверности, система возвращает бинарный результат нет или борода .
EPISODE_SEARCH_INTERVAL	Интервал (в секундах) для эпизодов) Период времени, предшествующий событию, в течение которого система ищет в биометрической базе данных события с похожими лицами. Если такого события не найдено, система создает новый эпизод. В противном случае она выбирает наиболее подходящее событие из открытого (LIVE) эпизода, отсортировав 100 последних похожих лиц. См. Настройка эпизодов .
EPISODE_MAX_DURATION	Максимальная продолжительность эпизода в секундах. По истечении этого времени эпизод автоматически закрывается.
EPISODE_MAX_LIVE_DURATION	Максимальное время в секундах с момента добавления последнего события в эпизод. По истечении этого времени эпизод автоматически закрывается.
CUSTOM_FIELDS	Добавьте этот раздел вручную, чтобы настроить содержание досье. Подробнее см. Пользовательские вкладки, поля и фильтры в досье .
VERBOSE_WEBHOOKS	Включить в вебхуках сериализованные досье, списки наблюдения, камеры и группы камер.
THUMBNAIL_CALLBACK_URL	Адрес для миниатюр.

Предупреждение: Секция FFSECURITY должна заканчиваться параметрами EVENTS_FEATURES/LIVENESS_THRESHOLD/ EMOTIONS_THRESHOLD/BEARD_THRESHOLD, которые должны быть расположены друг за другом в указанном порядке.

```
...
'SF_API_ADDRESS': 'http://127.0.0.1:18411',
'EVENTS_FEATURES': ['gender', 'age', 'emotions', 'beard', 'glasses', 'medmask'],
'LIVENESS_THRESHOLD': 0.945,
'EMOTIONS_THRESHOLD': 0.25
'BEARD_THRESHOLD': 0.7,
```

findface-facerouter

Важно: Компонент findface-facerouter не входит в состав стандартной конфигурации FindFace Security. При необходимости используйте его для интеграции. См. [Плагины](#).

Компонент findface-facerouter представляет собой сервис, через который задаются правила обработки обнаруженных на видео лиц. Правила задаются в виде пользовательских плагинов.

Компонент findface-facerouter принимает нормализованные изображения лиц вместе с исходным кадром и другой информацией (например, датой и временем детекции) от компонента findface-video-worker. В общем случае позволяет обрабатывать лица произвольным способом, в том числе отправлять их напрямую в партнерское приложение. В базовой реализации перенаправляет их в компонент findface-sf-api для дальнейшей обработки в соответствии с заданными правилами.

Полный список функций:

- задание правил обработки обнаруженных лиц на видео,
- перенаправление обнаруженных лиц в компонент findface-sf-api или другой сервис (в том числе стороннее приложение) для последующей обработки.

Настройка компонента findface-facerouter выполняется через файл конфигурации /etc/findface-facerouter.py.

```
# main.py options:

# debug                                = False
## debug - debug mode
# detector                             = ''
## detector - Detector to use if client fails to provide normalized face
## (nnd).Use "nnd" if you need to detect faces in such requests. Empty value
## rejects requests without face0.
# host                                 = ''
## host - host to listen
# port                                 = 18820
## port - port to listen
# sfapi_url                            = 'http://localhost:18411'
## sfapi_url - SF-API URL
# version                             = False
## version - print version

# plugin_dir.py options:
```

(continues on next page)

(продолжение с предыдущей страницы)

```

# plugin_dir = ''
## plugin_dir - Plugin directory for plugin_source='dir'

# abstract_define.py options:

# plugin_source = 'dir'
## plugin_source - Plugin source (dir)

# log.py options:

# log_file_max_size = 100000000
## log_file_max_size - max size of log files before rollover
# log_file_num_backups = 10
## log_file_num_backups - number of log files to keep
# log_file_prefix = None
## log_file_prefix - Path prefix for log files. Note that if you are running
## multiple tornado processes, log_file_prefix must be different for each of
## them (e.g. include the port number)
# log_rotate_interval = 1
## log_rotate_interval - The interval value of timed rotating
# log_rotate_mode = 'size'
## log_rotate_mode - The mode of rotating files(time or size)
# log_rotate_when = 'midnight'
## log_rotate_when - specify the type of TimedRotatingFileHandler interval other
## options:('S', 'M', 'H', 'D', 'W0'- 'W6')
# log_to_stderr = None
## log_to_stderr - Send log output to stderr (colorized if possible). By default
## use stderr if --log_file_prefix is not set and no other logging is
## configured.
# logging = 'info'
## logging - Set the Python log level. If 'none', tornado won't touch the
## logging configuration.

```

Пользовательская настройка `findface-facerouter` выполняется с использованием следующих параметров:

Параметр	Описание
<code>sfapi_url</code>	IP-адрес и порт сервера <code>findface-sf-api</code> .
<code>plugin_dir</code>	Каталог с плагинами, определяющими правила обработки обнаруженных на видео лиц.

1.9.3 Файл с параметрами установки

При установке FindFace Security из инсталлятора параметры установки автоматически сохраняются в файл `/tmp/<findface-installer-*.json`. Вы можете отредактировать данный файл и использовать его при установке FindFace Security на других серверах, не отвечая на вопросы инсталлятора повторно.

Совет: Подробная информация об инсталляторе приведена в разделе *Развертывание из консольного инсталлятора*.

Важно: Обязательно удалите поля `*.config`, `exp_ip` и `int_ip` перед установкой FindFace Security на

сервере с другим IP-адресом.

Пример файла с параметрами установки:

```
{
  "findface-tarantool-server.config": {
    "shard-003": {
      "TNT_FF_LISTEN_PORT": "8103",
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-003",
      "TNT_LISTEN": "127.0.0.1:32003",
      "TNT_META_INDEXES": "meta_indexes"
    },
    "shard-004": {
      "TNT_FF_LISTEN_PORT": "8104",
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-004",
      "TNT_LISTEN": "127.0.0.1:32004",
      "TNT_META_INDEXES": "meta_indexes"
    },
    "shard-008": {
      "TNT_FF_LISTEN_PORT": "8108",
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-008",
      "TNT_LISTEN": "127.0.0.1:32008",
      "TNT_META_INDEXES": "meta_indexes"
    },
    "shard-001": {
      "TNT_FF_LISTEN_PORT": "8101",
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-001",
      "TNT_LISTEN": "127.0.0.1:32001",
      "TNT_META_INDEXES": "meta_indexes"
    },
    "shard-006": {
      "TNT_FF_LISTEN_PORT": "8106",
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-006",
      "TNT_LISTEN": "127.0.0.1:32006",
      "TNT_META_INDEXES": "meta_indexes"
    },
    "shard-005": {
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    "TNT_FF_LISTEN_PORT": "8105",
    "TNT_META_SCHEME": "meta_scheme",
    "TNT_FF_LISTEN_IP": "127.0.0.1",
    "TNT_EXTRA_LUA": "\\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\\n",
    "TNT_FF_NTLS": "127.0.0.1:3133",
    "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-005",
    "TNT_LISTEN": "127.0.0.1:32005",
    "TNT_META_INDEXES": "meta_indexes"
  },
  "shard-002": {
    "TNT_FF_LISTEN_PORT": "8102",
    "TNT_META_SCHEME": "meta_scheme",
    "TNT_FF_LISTEN_IP": "127.0.0.1",
    "TNT_EXTRA_LUA": "\\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\\n",
    "TNT_FF_NTLS": "127.0.0.1:3133",
    "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-002",
    "TNT_LISTEN": "127.0.0.1:32002",
    "TNT_META_INDEXES": "meta_indexes"
  },
  "shard-007": {
    "TNT_FF_LISTEN_PORT": "8107",
    "TNT_META_SCHEME": "meta_scheme",
    "TNT_FF_LISTEN_IP": "127.0.0.1",
    "TNT_EXTRA_LUA": "\\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\\n",
    "TNT_FF_NTLS": "127.0.0.1:3133",
    "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-007",
    "TNT_LISTEN": "127.0.0.1:32007",
    "TNT_META_INDEXES": "meta_indexes"
  }
},
"findface-video-worker.variant": "cpu",
"product": "security",
"ignore_nolicense": true,
"findface-sf-api.config": {
  "extraction-api": {
    "extraction-api": "http://127.0.0.1:18666"
  },
  "limits": {
    "allow-return-facen": true
  },
  "listen": "127.0.0.1:18411",
  "storage-api": {
    "shards": [
      {
        "slave": "",
        "master": "http://127.0.0.1:8101/v2/"
      },
      {
        "slave": "",
        "master": "http://127.0.0.1:8102/v2/"
      },
      {
        "slave": "",
        "master": "http://127.0.0.1:8103/v2/"
      },
      {

```

(continues on next page)

(продолжение с предыдущей страницы)

```

        "slave": "",
        "master": "http://127.0.0.1:8104/v2/"
    },
    {
        "slave": "",
        "master": "http://127.0.0.1:8105/v2/"
    },
    {
        "slave": "",
        "master": "http://127.0.0.1:8106/v2/"
    },
    {
        "slave": "",
        "master": "http://127.0.0.1:8107/v2/"
    },
    {
        "slave": "",
        "master": "http://127.0.0.1:8108/v2/"
    }
]
}
},
"findface-extraction-api.variant": "cpu",
"memcached.config": {
    "listen_host": "127.0.0.1",
    "max_memory": 1024,
    "item_size": 16
},
"type": "stand-alone",
"findface-video-worker.config": {
    "streamer": [
        "port = 18999",
        "url = 127.0.0.1:18999"
    ],
    "FKVD_WRK_CAP": "10",
    "FKVD_MGR_ADDR": "127.0.0.1:18811",
    "FKVD_NTLS_ADDR": "127.0.0.1:3133"
},
"findface-security.config": {
    "SERVICE_EXTERNAL_ADDRESS": "http://172.17.46.98",
    "FFSECURITY": {
        "ROUTER_URL": "http://127.0.0.1"
    },
    "ntech_password": "dGP0Qc3yTAKbpa9jP5BueakzxJWcH61I"
},
"findface-counter.config": {
    "listen": "127.0.0.1:17300",
    "database": {
        "connection_string": "dbname=ffcounter host=/var/run/postgresql sslmode=disable"
    }
},
"findface-extraction-api.config": {
    "extractors": {
        "models": {
            "quality": "faceattr/quality.v0.cpu.fnk",
            "face": "face/ifruit_320.cpu.fnk",

```

(continues on next page)

```

    "gender": "",
    "emotions": "",
    "age": ""
  },
  "instances": 2
},
"normalizers": {
  "instances": 2
},
"listen": "127.0.0.1:18666",
"detectors": {
  "quality_estimator": true,
  "instances": 2
},
"license_ntls_server": "127.0.0.1:3133"
},
"inter_ip.advertised": "127.0.0.1",
"findface-ntls.config": {
  "NTLS_LISTEN_UI": "127.0.0.1:3185",
  "NTLS_LICENSE_DIR": "/opt/ntech/license",
  "NTLS_LISTEN": "127.0.0.1:3133"
},
"facen_model": "face/ifruit_320.cpu.fnk",
"ext_ip.advertised": "172.17.46.98",
"findface-data.models": [
  "./findface-data-age.v1-cpu_3.0.0_all.deb",
  "./findface-data-age.v1-gpu_3.0.0_all.deb",
  "./findface-data-beard.v0-cpu_3.0.0_all.deb",
  "./findface-data-beard.v0-gpu_3.0.0_all.deb",
  "./findface-data-detector-cheetah-cpu_3.0.0_all.deb",
  "./findface-data-detector-cheetah-fast-cpu_3.0.0_all.deb",
  "./findface-data-detector-cheetah-fast-gpu_3.0.0_all.deb",
  "./findface-data-detector-cheetah-gpu_3.0.0_all.deb",
  "./findface-data-detector-mtcnn-cpu_3.0.0_all.deb",
  "./findface-data-emotions.v1-cpu_3.0.0_all.deb",
  "./findface-data-emotions.v1-gpu_3.0.0_all.deb",
  "./findface-data-gender.v2-cpu_3.0.0_all.deb",
  "./findface-data-gender.v2-gpu_3.0.0_all.deb",
  "./findface-data-glasses3.v0-cpu_3.0.0_all.deb",
  "./findface-data-glasses3.v0-gpu_3.0.0_all.deb",
  "./findface-data-ifruit-160-cpu_3.0.0_all.deb",
  "./findface-data-ifruit-160-gpu_3.0.0_all.deb",
  "./findface-data-ifruit-320-cpu_3.0.0_all.deb",
  "./findface-data-ifruit-320-gpu_3.0.0_all.deb",
  "./findface-data-liveness.v3-cpu_3.0.0_all.deb",
  "./findface-data-liveness.v3-gpu_3.0.0_all.deb",
  "./findface-data-medmask3.v1-cpu_3.0.0_all.deb",
  "./findface-data-medmask3.v1-gpu_3.0.0_all.deb",
  "./findface-data-normalization-ant.v2-cpu_3.0.0_all.deb",
  "./findface-data-normalization-ant.v2-gpu_3.0.0_all.deb",
  "./findface-data-normalization-bee-fast-cpu_3.0.0_all.deb",
  "./findface-data-normalization-bee-fast-gpu_3.0.0_all.deb",
  "./findface-data-normalization-bee.v2-cpu_3.0.0_all.deb",
  "./findface-data-normalization-bee.v2-gpu_3.0.0_all.deb",
  "./findface-data-normalization-crop2x.v2-maxsize400-cpu_3.0.0_all.deb",
  "./findface-data-normalization-crop2x.v2-maxsize400-gpu_3.0.0_all.deb",

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    "./findface-data-pedet-edie-rc2-cpu_3.0.0_all.deb",
    "./findface-data-pedet-edie-rc2-gpu_3.0.0_all.deb",
    "./findface-data-quality.v0-cpu_3.0.0_all.deb",
    "./findface-data-quality.v0-gpu_3.0.0_all.deb"
],
"python3-pil.variant": "avx2",
"components": [
    "findface-data",
    "memcached",
    "etcd",
    "redis",
    "postgresql",
    "pgbouncer",
    "jq",
    "python3-pil",
    "findface-ntls",
    "findface-extraction-api",
    "findface-sf-api",
    "findface-counter",
    "findface-upload",
    "findface-video-manager",
    "findface-video-worker",
    "findface-security",
    "findface-tarantool-server"
],
"pgbouncer.config": {
    "ntech_password": "dGP0Qc3yTAKbpa9jP5BueakzxJWcH61I"
},
"findface-facerouter.config": {
    "host": "127.0.0.1",
    "plugin_source": "dir",
    "port": "18820",
    "sfapi_url": "http://127.0.0.1:18411",
    "plugin_dir": "/etc/findface-facerouter-plugins"
},
"tnt_instances": 8,
"postgresql.config": {
    "ntech_password": "dGP0Qc3yTAKbpa9jP5BueakzxJWcH61I"
},
"inter_ip.bind": "127.0.0.1",
"findface-video-manager.config": {
    "ntls": {
        "url": "http://127.0.0.1:3185/",
        "enabled": false
    },
    "rpc": {
        "listen": "127.0.0.1:18811"
    },
    "listen": "127.0.0.1:18810",
    "master": {
        "self_url_http": "127.0.0.1:18810",
        "self_url": "127.0.0.1:18811"
    }
},
"ext_ip.bind": "0.0.0.0"
}

```

1.9.4 Модели нейронных сетей

В этом разделе вы найдете сводную информацию по моделям нейронных сетей, созданным в нашей лаборатории и используемым в FindFace Security:

Примечание: Конфигурация для теста производительности на CPU и GPU:

- CPU - Intel® Core™ i7-5930K CPU @ 3.50GHz × 12
- GPU - GeForce GTX 1080

Важно: При чистой установке биометрической моделью по умолчанию является ifruit_320.

Модель	CPU, FPS	GPU, FPS	Тип нейронной сети
face/ifruit_160	Тестирование в процессе		Биометрия лица
face/ifruit_320			
faceattr/age.v1	14.99	529.35	Распознавание возраста
faceattr/beard.v0	15.03	532.05	Распознавание бороды
faceattr/emotions.v1	10.99	235.59	Распознавание эмоций
faceattr/gender.v2	15.01	523.22	Распознавание пола
faceattr/glasses3.v0	15.01	529.64	Распознавание очков
faceattr/medmask3.v1			Обнаружение медицинской маски

1.9.5 Хранилища данных FindFace Security

В этом разделе:

- [Список хранилищ](#)
- [Галереи биометрической базы данных](#)

Список хранилищ

FindFace Security использует следующие хранилища данных:

- Биометрическая база данных на основе Tarantool, в которой хранятся биометрические образцы (векторы признаков) и события идентификации лиц.
- Основная база данных системы на основе PostgreSQL, в которой хранятся внутренние данные системы, досье, учетные записи пользователей и настройки камер.
- Каталог `/var/lib/ffsecurity/uploads`, в котором хранятся загруженные в досье фотографии, видеофайлы и артефакты событий (полные кадры, миниатюры лиц и нормализованные изображения лиц).
- Каталог `/var/lib/ffupload/`, в котором хранятся только такие артефакты событий, как миниатюры лиц.

Галереи биометрической базы данных

В биометрической базе данных на основе Tarantool есть 3 галереи:

- `ffsec_dossier_face`: биометрические образцы, извлеченные из фотографий в досье.
- `ffsec_events`: биометрические образцы, извлеченные из лиц, обнаруженных на видео.
- `ffsec_persons`: центроиды персон (виртуальные биометрические образцы, усредненные по всем лицам персоны) и метаданные.

1.9.6 Опции резервного копирования базы данных

Для резервного копирования биометрической базы данных вам понадобится утилита `findface-storage-api-dump`. Данная утилита может быть запущена со следующими опциями:

Примечание: Вы можете найти подробную информацию по использованию `findface-storage-api-dump` в разделе *Резервное копирование и восстановление хранилищ данных*.

```
findface-storage-api-dump --help
```

Usage of findface-storage-api-dump:

```
-cache string
    Cache type: inmemory, redis or memcache (default "memcache")
-cache-inmemory-size int
    Maximum number of items in ARC cache (default 16384)
-cache-memcache-nodes value
    Comma-separated list of memcache shards (default 127.0.0.1:11211)
-cache-memcache-timeout duration
    Specifies read/write timeout (default 100ms)
-cache-redis-addr string
    Host:Port address (default "localhost:6379")
-cache-redis-db int
    Database to be selected after connecting to the server.
-cache-redis-network string
    Network type, either tcp or unix (default "tcp")
-cache-redis-password string
    Optional password. Must match the password specified in the requirepass server_
↪ configuration option.
-cache-redis-timeout duration
    Specifies dial/read/write timeout (default 5s)
-config string
    Path to config file
-config-template
    Output config template and exit
-continue-on-errors
    Continue on errors instead of exiting
-cpu-profile string
    Enable CPU profile and set output file
-extraction-api-extraction-api string
    Extraction API address (default "http://127.0.0.1:18666")
-extraction-api-timeouts-connect duration
    extraction-api-timeouts-connect (default 5s)
-extraction-api-timeouts-idle-connection duration
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    extraction-api-timeouts-idle-connection (default 10s)
-extraction-api-timeouts-overall duration
    extraction-api-timeouts-overall (default 35s)
-extraction-api-timeouts-response-header duration
    extraction-api-timeouts-response-header (default 30s)
-limits-allow-return-facen
    Allow returning raw feature vectors to detect responses if ?return_facen=true
-limits-body-image-length int
    Maximum length of image supplied in request body (default 33554432)
-limits-deny-networks string
    Comma-separated list of subnets that are not allowed to fetch from (default "127.0.0.0/8,
↪192.168.0.0/16,10.0.0.0/8,::1/128,fe00::/8")
-limits-url-length int
    Maximum supported url length in bytes (default 4096)
-listen string
    IP:port to listen on (default ":18411")
-normalized-storage-enabled
    Enables normalize saving (default true)
-normalized-storage-s3-access-key string
    Access key for the object storage
-normalized-storage-s3-bucket-name string
    S3 storage bucket name
-normalized-storage-s3-endpoint string
    S3 compatible object storage endpoint
-normalized-storage-s3-operation-timeout int
    Storage operations (Get,Put,Delete) timeout in seconds (default 30)
-normalized-storage-s3-public-url string
    Storage public url
-normalized-storage-s3-region string
    Storage region
-normalized-storage-s3-secret-access-key string
    Secret key for the object storage
-normalized-storage-s3-secure
    If 'true' API requests will be secure (HTTPS), and insecure (HTTP) otherwise (default true)
-normalized-storage-webdav-timeouts-connect duration
    normalized-storage-webdav-timeouts-connect (default 5s)
-normalized-storage-webdav-timeouts-idle-connection duration
    normalized-storage-webdav-timeouts-idle-connection (default 10s)
-normalized-storage-webdav-timeouts-overall duration
    normalized-storage-webdav-timeouts-overall (default 35s)
-normalized-storage-webdav-timeouts-response-header duration
    normalized-storage-webdav-timeouts-response-header (default 30s)
-normalized-storage-webdav-upload-url string
    webdav storage for normalized, disable normalized if empty string (default "http://127.0.0.
↪1:3333/uploads/")
-normalized_storage string
    Normalized storage type: webdav, s3 (default "webdav")
-output-dir string
    Output directory (default ".")
-storage-api-max-idle-conns-per-host int
    storage-api-max-idle-conns-per-host (default 20)
-storage-api-timeouts-connect duration
    storage-api-timeouts-connect (default 5s)
-storage-api-timeouts-idle-connection duration
    storage-api-timeouts-idle-connection (default 10s)
-storage-api-timeouts-overall duration

```

(continues on next page)

(продолжение с предыдущей страницы)

```

storage-api-timeouts-overall (default 35s)
-storage-api-timeouts-response-header duration
storage-api-timeouts-response-header (default 30s)

```

1.9.7 Опции восстановления базы данных

Для восстановления биометрической базы данных из резервной копии вам понадобится утилита `findface-storage-api-restore`. Данная утилита может быть запущена со следующими опциями:

Примечание: Вы можете найти подробную информацию по использованию `findface-storage-api-restore` в разделе *Резервное копирование и восстановление галереи данных*.

```

findface-storage-api-restore --help

Usage of findface-storage-api-restore:
-cache string
    Cache type: inmemory, redis or memcache (default "memcache")
-cache-inmemory-size int
    Maximum number of items in ARC cache (default 16384)
-cache-memcache-nodes value
    Comma-separated list of memcache shards (default 127.0.0.1:11211)
-cache-memcache-timeout duration
    Specifies read/write timeout (default 100ms)
-cache-redis-addr string
    Host:Port address (default "localhost:6379")
-cache-redis-db int
    Database to be selected after connecting to the server.
-cache-redis-network string
    Network type, either tcp or unix (default "tcp")
-cache-redis-password string
    Optional password. Must match the password specified in the requirepass server.
↪ configuration option.
-cache-redis-timeout duration
    Specifies dial/read/write timeout (default 5s)
-config string
    Path to config file
-config-template
    Output config template and exit
-cpu-profile string
    Enable CPU profile and set output file
-dont-create-gallery
    Don't create gallery, fail if doesn't exist
-extraction-api-extraction-api string
    Extraction API address (default "http://127.0.0.1:18666")
-extraction-api-timeouts-connect duration
    extraction-api-timeouts-connect (default 5s)
-extraction-api-timeouts-idle-connection duration
    extraction-api-timeouts-idle-connection (default 10s)
-extraction-api-timeouts-overall duration
    extraction-api-timeouts-overall (default 35s)
-extraction-api-timeouts-response-header duration
    extraction-api-timeouts-response-header (default 30s)

```

(continues on next page)

```

-limits-allow-return-facen
    Allow returning raw feature vectors to detect responses if ?return_facen=true
-limits-body-image-length int
    Maximum length of image supplied in request body (default 33554432)
-limits-deny-networks string
    Comma-separated list of subnets that are not allowed to fetch from (default "127.0.0.0/8,
↪192.168.0.0/16,10.0.0.0/8,::1/128,fe00::/8")
-limits-url-length int
    Maximum supported url length in bytes (default 4096)
-listen string
    IP:port to listen on (default ":18411")
-normalized-storage-enabled
    Enables normalize saving (default true)
-normalized-storage-s3-access-key string
    Access key for the object storage
-normalized-storage-s3-bucket-name string
    S3 storage bucket name
-normalized-storage-s3-endpoint string
    S3 compatible object storage endpoint
-normalized-storage-s3-operation-timeout int
    Storage operations (Get,Put,Delete) timeout in seconds (default 30)
-normalized-storage-s3-public-url string
    Storage public url
-normalized-storage-s3-region string
    Storage region
-normalized-storage-s3-secret-access-key string
    Secret key for the object storage
-normalized-storage-s3-secure
    If 'true' API requests will be secure (HTTPS), and insecure (HTTP) otherwise (default true)
-normalized-storage-webdav-timeouts-connect duration
    normalized-storage-webdav-timeouts-connect (default 5s)
-normalized-storage-webdav-timeouts-idle-connection duration
    normalized-storage-webdav-timeouts-idle-connection (default 10s)
-normalized-storage-webdav-timeouts-overall duration
    normalized-storage-webdav-timeouts-overall (default 35s)
-normalized-storage-webdav-timeouts-response-header duration
    normalized-storage-webdav-timeouts-response-header (default 30s)
-normalized-storage-webdav-upload-url string
    webdav storage for normalized, disable normalized if empty string (default "http://127.0.0.
↪1:3333/uploads/")
-normalized_storage string
    Normalized storage type: webdav, s3 (default "webdav")
-rename string
    Ignore dump header and use this string as gallery name
-storage-api-max-idle-conns-per-host int
    storage-api-max-idle-conns-per-host (default 20)
-storage-api-timeouts-connect duration
    storage-api-timeouts-connect (default 5s)
-storage-api-timeouts-idle-connection duration
    storage-api-timeouts-idle-connection (default 10s)
-storage-api-timeouts-overall duration
    storage-api-timeouts-overall (default 35s)
-storage-api-timeouts-response-header duration
    storage-api-timeouts-response-header (default 30s)

```

2.1 Веб-интерфейс

Работа с FindFace Security выполняется через веб-интерфейс. Для того чтобы отобразить веб-интерфейс, введите его адрес в адресной строке браузера и пройдите авторизацию.

Примечание: Логин и пароль для авторизации выдаются администратором.

Веб-интерфейс имеет удобный и интуитивный дизайн и обеспечивает доступ к следующим функциям:

- *Идентификация по базам данных.*
- *Идентификация лиц в режиме реального времени.*
- *Работа с досье* (только для пользователей с правами оператора).
- *Видеостена.*

2.2 Идентификация по базам данных

FindFace Security позволяет выполнять идентификацию (поиск) лиц по следующим базам данных:

- База данных обнаруженных на видео лиц (вкладка *События*).
- База данных досье (вкладка *Досье*). Содержит эталонные изображения лиц.

Поиск лиц выполняется на вкладке *Поиск*.

В этой главе:

- Идентификация лица по базе данных обнаруженных лиц
- Идентификация лица по базе данных досье

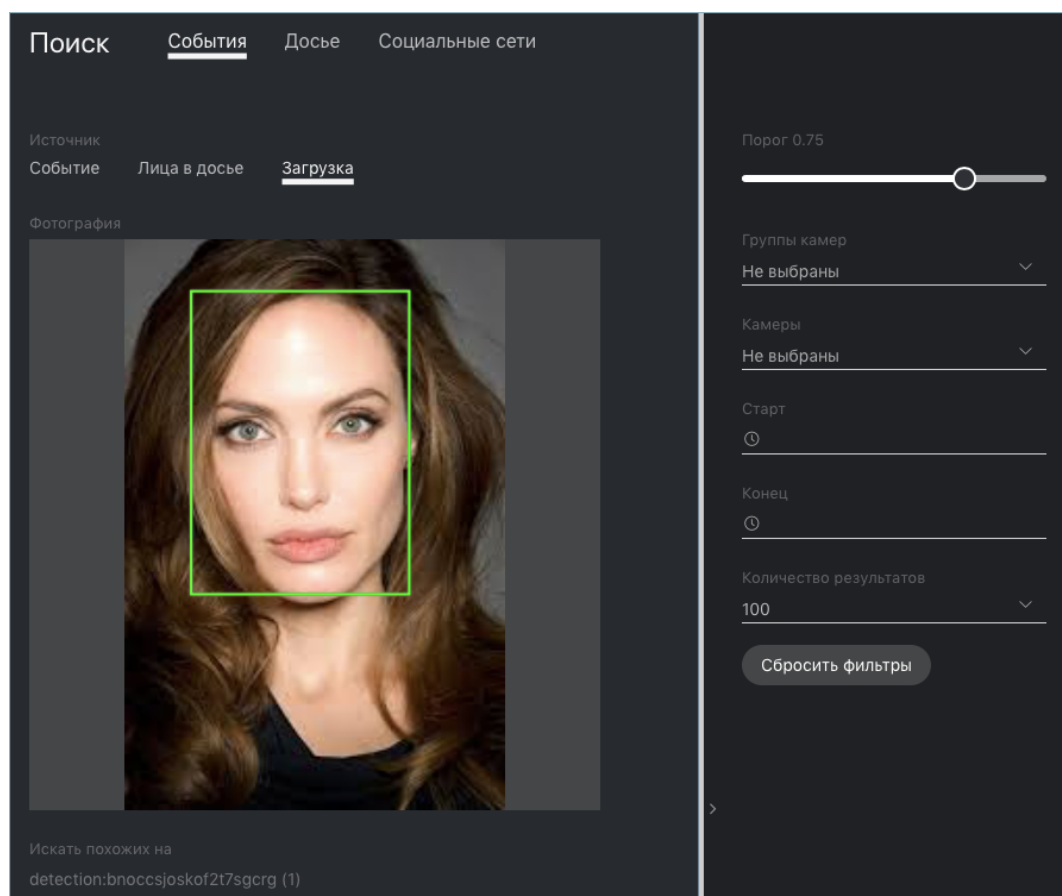
2.2.1 Идентификация лица по базе данных обнаруженных лиц

FindFace Security позволяет выполнять идентификацию лица по базе данных обнаруженных на видео лиц.

Примечание: В интерфейсе база данных представлена списком событий (вкладка *События*).

Для идентификации лица по базе данных выполните следующие действия:

1. Перейдите на вкладку *Поиск*.



2. Укажите место поиска: *События*.
3. Задайте лицо, которое требуется найти, одним из следующих способов:
 - Задав ID события с искомым лицом.
 - Задав ID досье с искомым лицом. Если досье содержит несколько фотографий, выберите те, которые будут использованы для поиска.

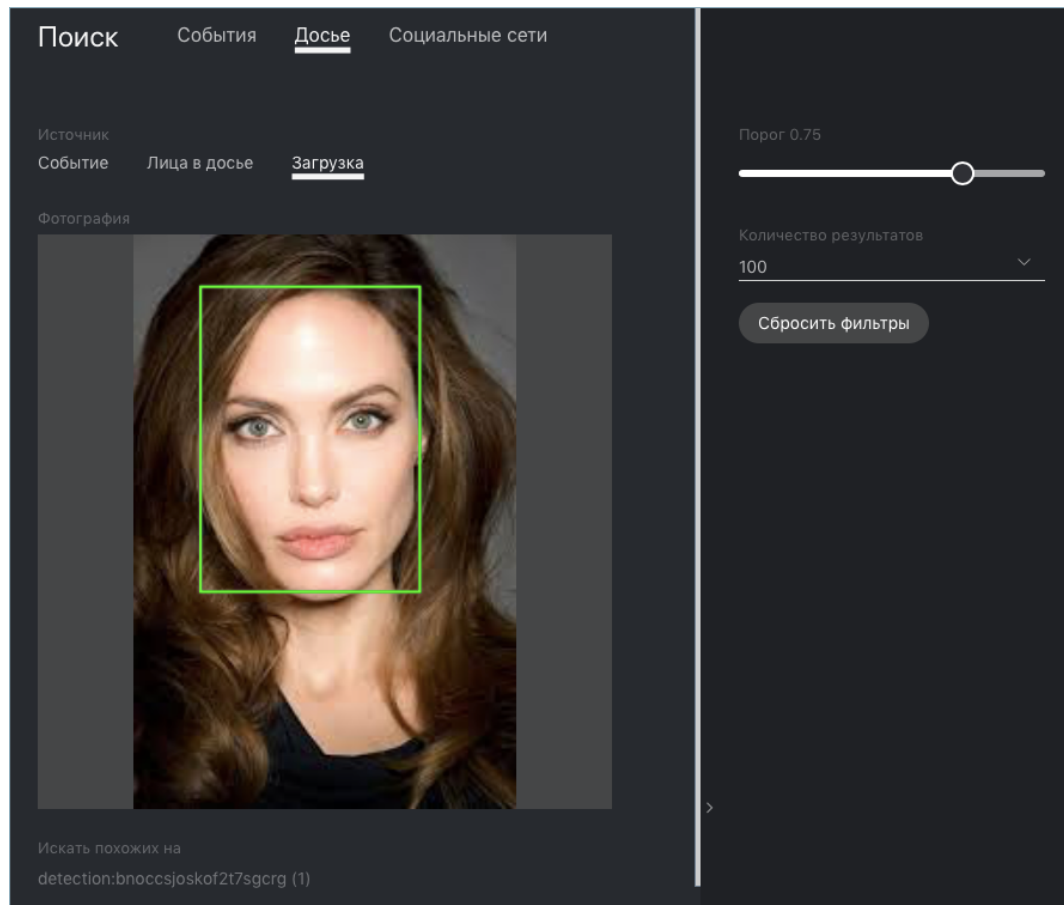
- Загрузив фотографию. Фотография будет отображена в одноименном поле. Если на фотографии присутствует несколько лиц, выберите нужное.
4. По умолчанию в результатах поиска отображаются лица, степень схожести которых с искомым равна или превышает 0.75. При необходимости измените данное значение.
 5. При необходимости укажите группу камер и период времени, в течение которого произошло событие.
 6. Выберите метод сортировки результатов поиска: по степени уверенности алгоритма в совпадении лиц (степени схожести между лицами) или дате.
 7. Укажите максимальное количество событий в результатах поиска.
 8. Нажмите *Поиск*. Результаты поиска будут отображены ниже. Для каждого найденного лица будет указана вероятность его совпадения с лицом на фотографии.

2.2.2 Идентификация лица по базе данных досье

FindFace Security позволяет выполнять идентификацию лица по базе данных, содержащей досье с эталонными изображениями лиц.

Для идентификации лица по базе данных выполните следующие действия:

1. Перейдите на вкладку *Поиск*.



2. Укажите место поиска: *Досье*.

3. Задайте лицо, которое требуется найти, одним из следующих способов:
 - Задав ID события с искомым лицом.
 - Задав ID досье с искомым лицом. Если досье содержит несколько фотографий, выберите те, которые будут использованы для поиска.
 - Загрузив фотографию. Фотография будет отображена в одноименном поле. Если на фотографии присутствует несколько лиц, выберите нужное.
4. По умолчанию в результатах поиска отображаются лица, степень схожести которых с искомым равна или превышает 0.75. При необходимости измените данное значение.
5. Выберите метод сортировки результатов поиска: по степени уверенности алгоритма в совпадении лиц (степени схожести между лицами) или дате.
6. Укажите максимальное количество событий в результатах поиска.
7. Нажмите *Поиск*. Результаты поиска будут отображены ниже. Для каждого найденного лица будет указана вероятность его совпадения с лицом на фотографии.

2.3 Идентификация лиц в режиме реального времени

Результат работы системы по части идентификации лиц на видеоизображении в режиме реального времени отображается на вкладках *События* и *Эпизоды*. Помимо работы с текущими событиями идентификации, данные вкладки также предоставляют доступ к истории событий. Данный раздел полностью посвящен работе с вкладкой *События*.

Совет: *Эпизоды* позволяют поднять безопасность на новый уровень.

Совет: Поиск лица в списке событий и базе данных досье с эталонными изображениями лиц выполняется на вкладке *Поиск*.

Совет: Для идентификации лиц в архивных видео см. *Идентификация лиц в офлайн видео*.

Важно: Вы можете *включить звуковые уведомления* для событий, связанных с конкретными списками наблюдения. В некоторых браузерах для того чтобы воспроизводился звук, вкладка с событиями должна оставаться в фокусе. Чтобы выделить вкладку, откройте ее и щелкните в любом месте страницы.

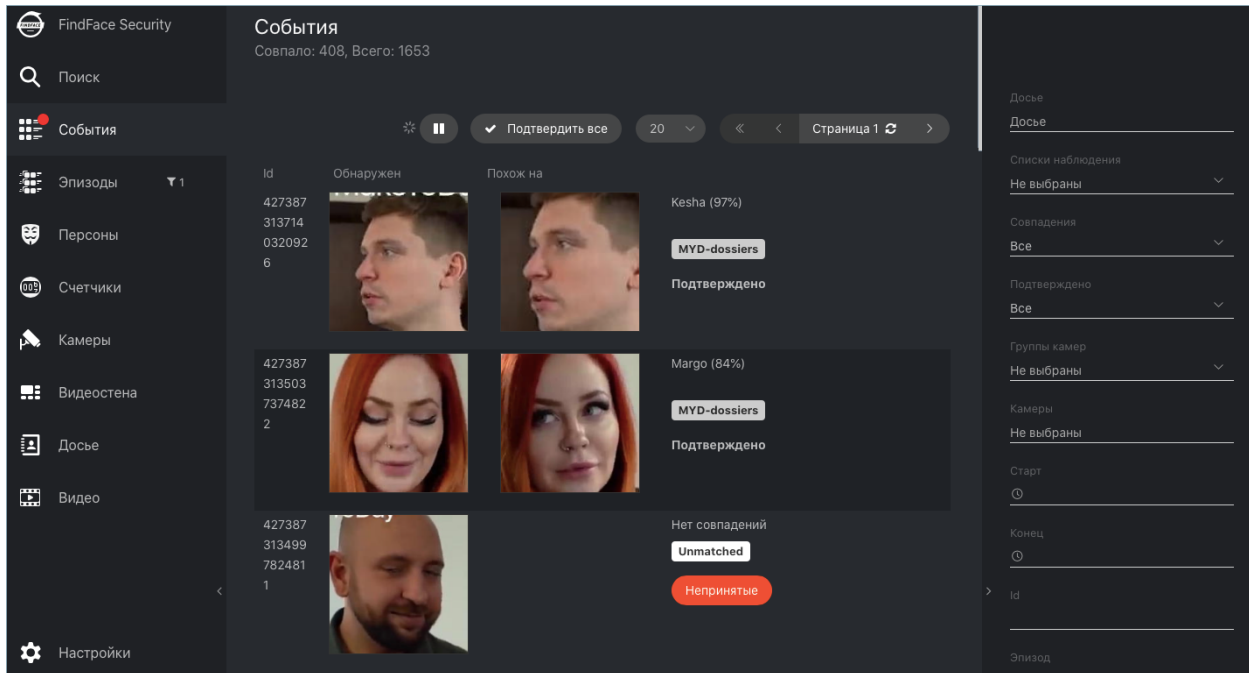
В этой главе:

- *Просмотр событий идентификации в режиме реального времени*
- *Распознавание живых лиц и атрибутов лица*
- *Карточка события. Принятие события*

- Карточка события. Поиск лица

2.3.1 Просмотр событий идентификации в режиме реального времени

При обнаружении лица в списке событий выводится уведомление.



Уведомление содержит следующую информацию:

- Если на лицо отсутствует досье: нормализованное изображение лица, дата и время обнаружения лица, группа камер.
- Если на лицо заведено досье: нормализованное изображение лица, фотография из досье, имя персоны, степень схожести лиц, комментарий из досье, список досье, дата и время обнаружения лица, группа камер.

Примечание: Система может быть настроена таким образом, что уведомления будут выводиться только для лиц с досье.

Важно: Для того чтобы остановить вывод новых уведомлений, нажмите на кнопку  над списком событий.

К событиям (уведомлениям) в списке можно применить следующие фильтры:

- *Досье:* отображать только события по определенному досье.
- *Списки наблюдения:* отображать только события по определенному списку наблюдения.

Примечание: Для просмотра только лиц без совпадений в списке событий, установите в филь-

тре *Без совпадений*.

- *Совпадения*: отображать только события с совпадением лиц/без совпадения лиц или все события.
- *Подтверждено*: отображать только принятые/непринятые или все события.
- *Камеры*: отображать только события по определенной камере.
- *Группы камер*: отображать только события по определенной группе камер.
- *Старт, Конец*: отображать только события, случившиеся в определенный период времени.
- *id*: отобразить событие с определенным ID.
- *Episode*: отобразить события из эпизода с определенным ID.

2.3.2 Распознавание живых лиц и атрибутов лица

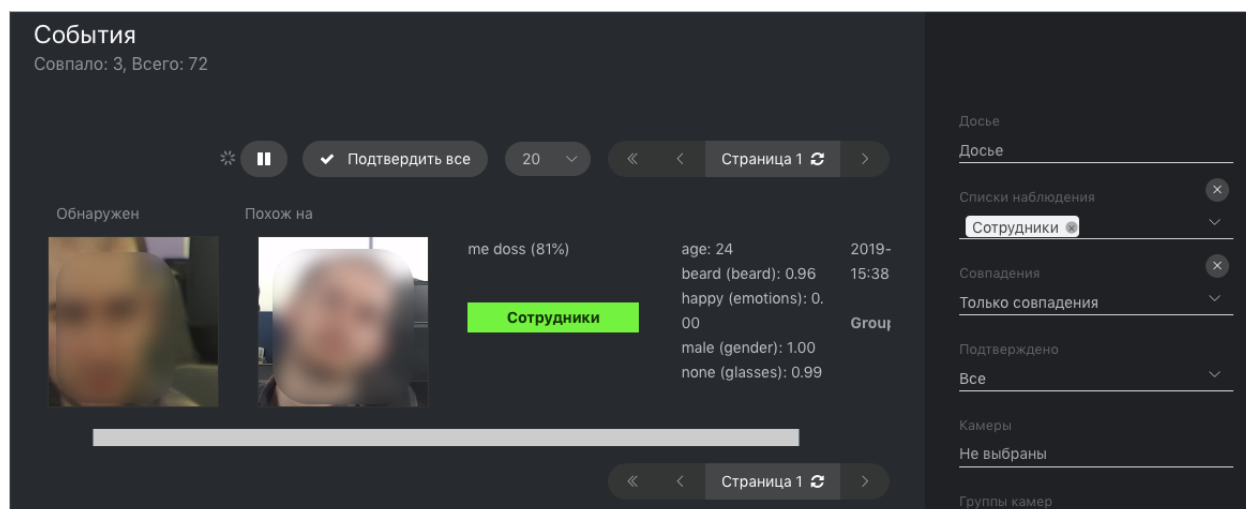
В зависимости от настроек системы, вы можете видеть оценку Liveness лица, а также результат распознавания атрибутов лица, таких как пол, возраст, эмоции, наличие очков и бороды.

Детектор живых лиц (Liveness) в автоматическом режиме отличает настоящее лицо от лица на фото- или видеоизображении, предотвращая мошенничество.

Примечание: Результат Liveness может быть null. Так происходит, когда детектор живых лиц отключен или не может достоверно оценить Liveness на предоставленном изображении.

Результат распознавания атрибутов лица возвращается в следующем формате:

Атрибут лица	Формат результата	Пример
Возраст	Атрибут: возраст : число лет	возраст: 33
Пол	Результат: мужской/женский (атрибут: пол): уверенность алгоритма в результате	женский (пол): 0.95
Эмоции	Результат: злость/отвращение/страх/счастье/грусть/удивление (атрибут: эмоции): уверенность алгоритма в результате	счастье (эмоции): 0.99
Очки	Результат: медицинские/солнечные/нет (атрибут: очки): уверенность алгоритма в результате	нет (очки): 0.87
Борода	Результат: борода/нет (атрибут: борода): уверенность алгоритма в результате	нет (борода): 0.91



При необходимости выполните фильтрацию событий по атрибутам лиц, отобразите только живые лица/попытки спуфинга.

Возраст

От До

Борода

☐ Нет ☐ Борода

Эмоции

☐ Злость ☐ Отвращение
☐ Страх ☐ Счастье
☐ Грусть ☐ Удивление
☐ Нейтральное состояние

Пол

☐ Мужской ☐ Женский

Очки

☐ Нет ☐ Медицинские
☐ Солнечные

Liveness

☐ Живой человек
☐ Изображение

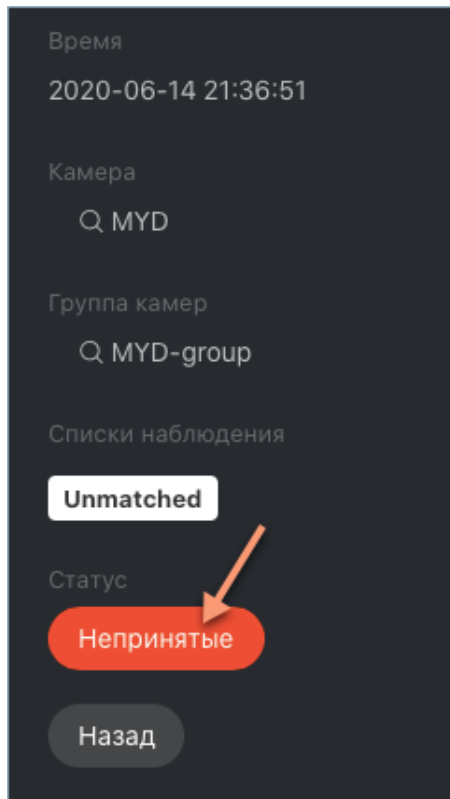
Медицинская маска

☐ Нет
☐ Медицинская маска

2.3.3 Карточка события. Принятие события

Для того чтобы перейти в карточку события из списка событий, щелкните в уведомлении по результату распознавания (*Нет совпадений* или имя из досье).

Карточка содержит ту же информацию, что и *уведомление*, а также предоставляет возможность принять событие. Для того чтобы это сделать, поставьте флажок *Подтверждение события*. Нажмите на кнопку *Сохранить*.



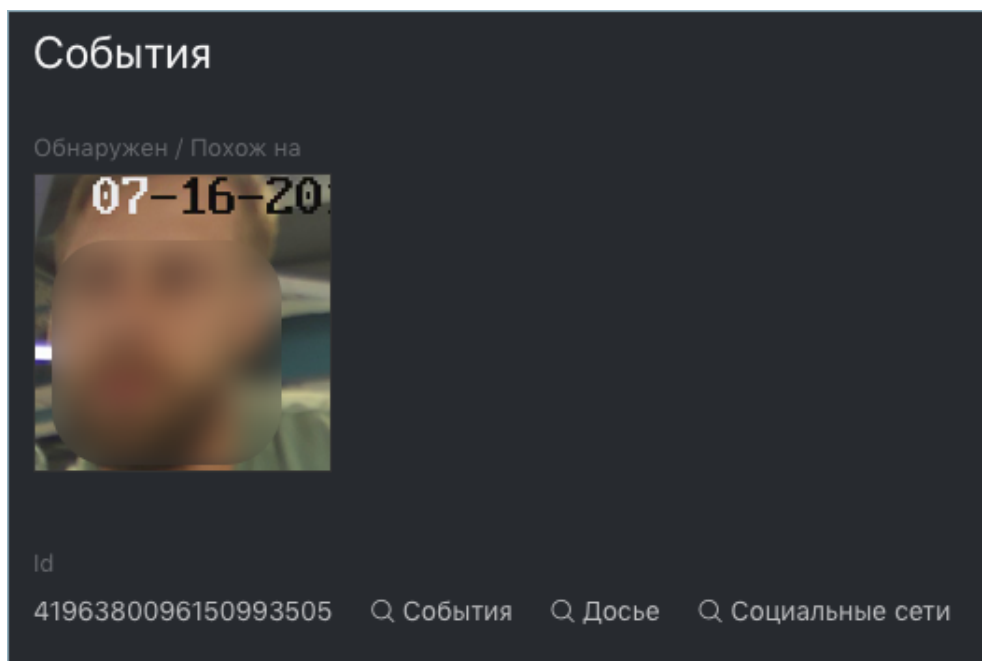
Совет: Если на обнаруженное лицо заведено досье, в него можно перейти, щелкнув по имени персоны в карточке события.

Совет: Для того чтобы принять все события, нажмите на кнопку  над списком событий.

Примечание: Принятие события может быть автоматизировано для выбранных списков наблюдения.

2.3.4 Карточка события. Поиск лица

FindFace Security позволяет искать обнаруженные лица в базе данных обнаруженных лиц и в базе данных досье с эталонными изображениями лиц. Для перехода на вкладку поиска из карточки события нажмите *События* и *Досье*.



См.также:

- *Идентификация по базам данных.*

2.4 Эпизоды событий

Результат работы системы по части идентификации лиц на видеоизображении в режиме реального времени отображается на вкладках *События* и *Эпизоды*. Помимо работы с текущими событиями идентификации, данные вкладки также предоставляют доступ к истории событий. Данный раздел полностью посвящен работе с вкладкой *Эпизоды*.

См.также:

- *Идентификация лиц в режиме реального времени*
- *Настройка эпизодов*

Эпизод — это набор событий идентификации, в которых фигурируют лица одного и того же человека, обнаруженные в течение определенного периода времени. Поскольку события на вкладке *События* отображаются в произвольном порядке, обработка большого количества разнородных событий может быть делом затруднительным и неэффективным. Функция Эпизоды позволяет автоматически объединять входящие события на основе времени обнаружения и схожести лиц. Это позволяет с легкостью обрабатывать разнородные события даже в больших количествах.

Совет: Поиск лица в списке событий и базе данных досье с эталонными изображениями лиц выполняется на вкладке *Поиск*.

Совет: Для идентификации лиц в архивных видео см. *Идентификация лиц в офлайн видео*.

В этой главе:

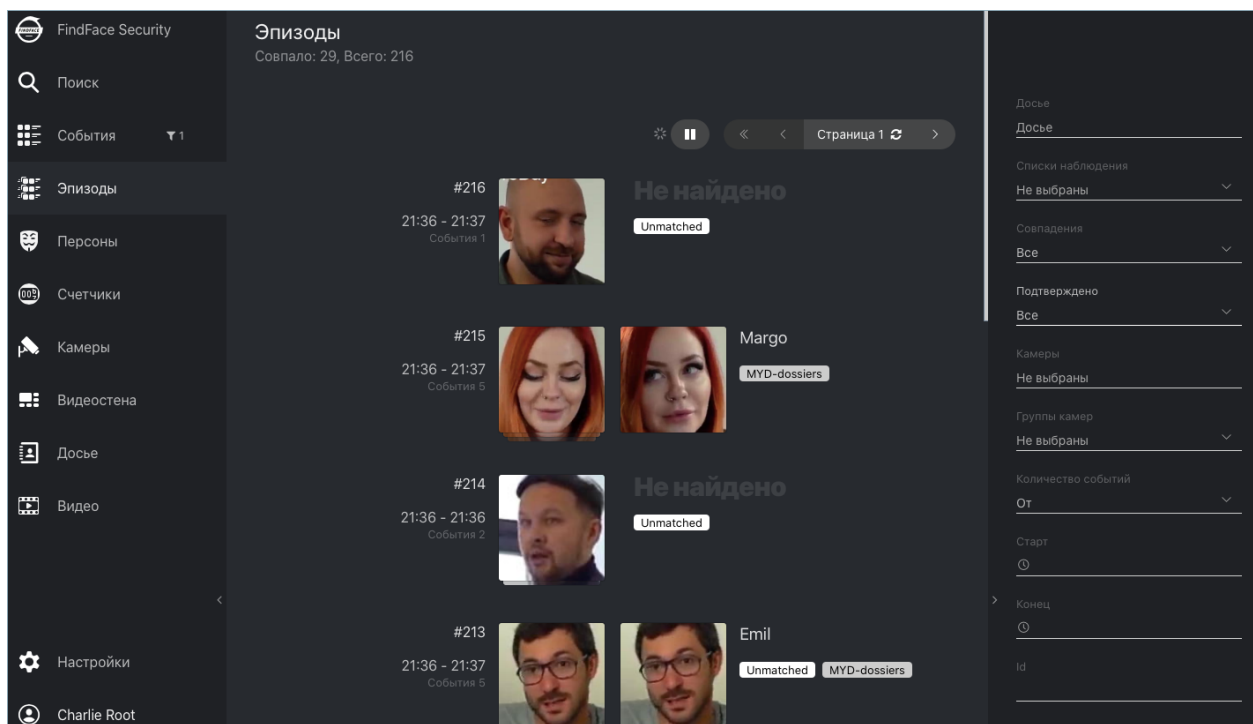
- Просмотр эпизодов
- Принятие события и эпизода
- Фильтрация событий по ID эпизода

2.4.1 Просмотр эпизодов

Эпизоды бывают двух типов:

- LIVE: открытый на данный момент эпизод, в который могут добавлены новые события.
- Закрытый: закрытый эпизод, добавление событий невозможно.

Список эпизодов с фильтрами и статистикой отображается на вкладке *Эпизоды*. Событие обнаружения лица либо добавляется в существующий LIVE-эпизод, либо инициирует создание нового эпизода. Каждому эпизоду присваивается id, который впоследствии можно использовать для фильтрации событий и эпизодов.



К эпизодам в списке можно применить следующие фильтры:

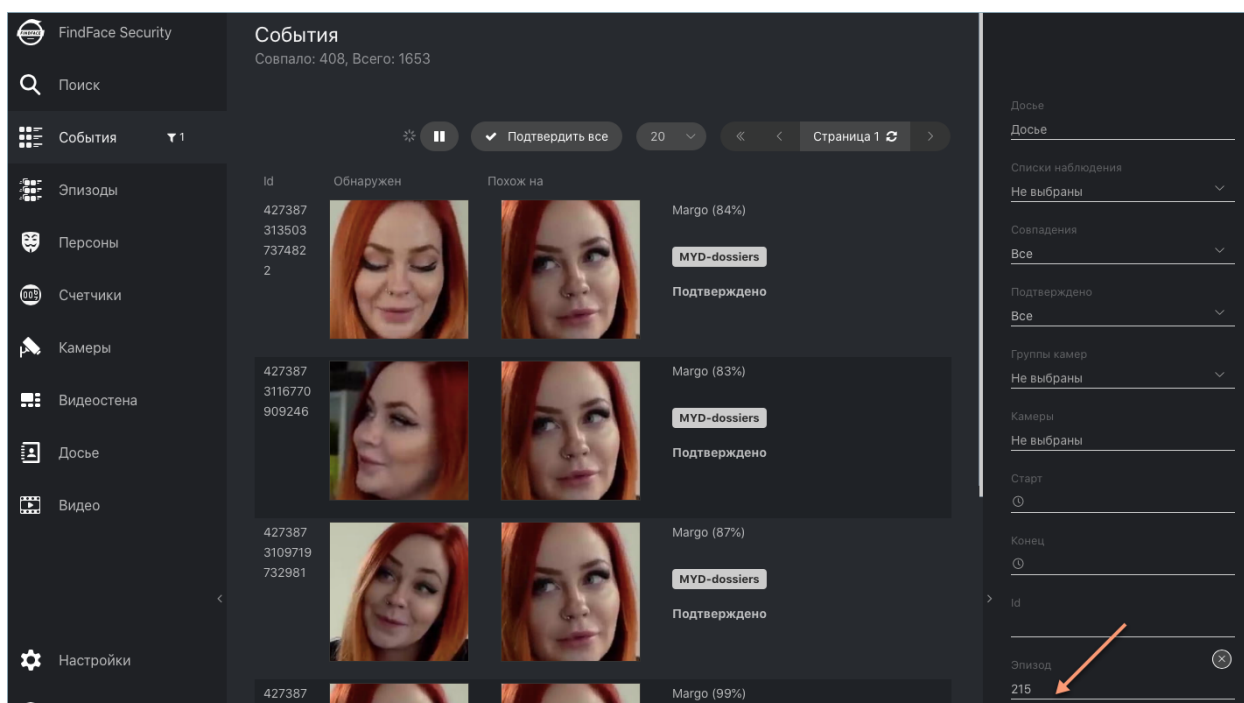
- *Досье*: отображать только эпизоды по определенному досье.
- *Списки наблюдения*: отображать только эпизоды по определенному списку наблюдения.

Примечание: Для просмотра только лиц без совпадений в списке эпизодов, установите в фильтре *Без совпадений*.

- *Совпадения*: отображать только эпизоды с совпадением лиц/без совпадения лиц или все эпизоды.
- *Подтверждено*: отображать только принятые/непринятые или все эпизоды.
- *Камеры*: отображать только эпизоды по определенной камере.
- *Группы камер*: отображать только эпизоды по определенной группе камер.
- *Старт, Конец*: отображать только эпизоды, случившиеся в определенный период времени.
- *id*: отобразить эпизод с определенным ID.

Вы также можете отфильтровать эпизоды по Liveness и атрибутам лица (если это применимо к вашей системе).

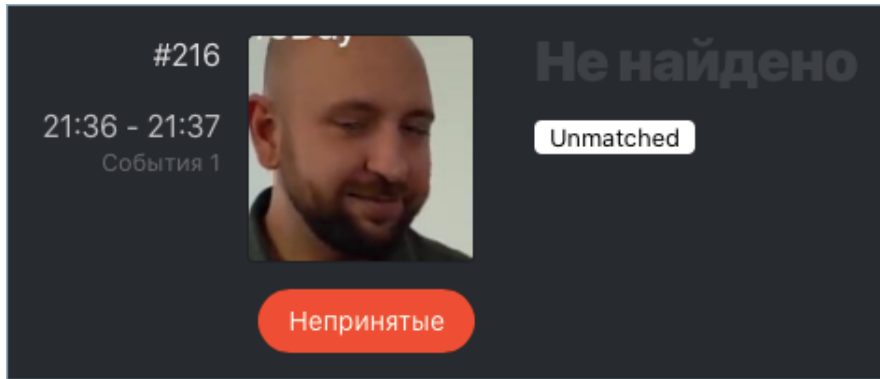
Для просмотра событий щелкните по нужному эпизоду в списке. Вы будете переправлены на вкладку *События* с соответствующим ID эпизода в фильтре *Эпизод*:



Работа с вкладкой *События* описана в разделе *Идентификация лиц в режиме реального времени*.

2.4.2 Принятие события и эпизода

Для того чтобы подтвердить эпизод целиком, нажмите *Непринятые* в списке эпизодов. В результате все события в эпизоде будут автоматически подтверждены, включая события, которые еще не добавлены (в случае эпизода LIVE).



Эпизод также автоматически подтверждается, если вы приняли все события по отдельности.

2.4.3 Фильтрация событий по ID эпизода

Для того чтобы отобразить события по ID эпизода, используйте фильтр *id* на вкладке *Эпизоды* или фильтр *ID эпизода* на вкладке *События*.

2.5 Работа с досье

FindFace Security позволяет создавать досье на интересующих лиц, содержащее одну или несколько фотографий. Досье классифицируется по принадлежности к тому или иному списку наблюдения.

В этом разделе:

- *Создание досье*
- *Просмотр досье из списка наблюдения*

2.5.1 Создание досье

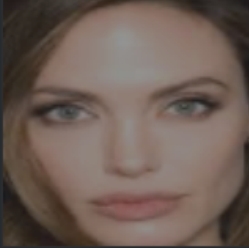
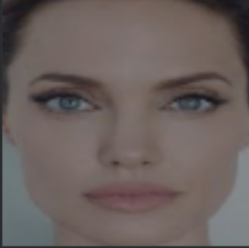
Для создания досье выполните следующие действия:

1. В веб-интерфейсе перейдите на вкладку *Досье*.
2. Нажмите *+*.

Создать досье

У вас много досье для загрузки? Попробуйте [Пакетную загрузку досье](#)

Фотографии



Файлы

 Проверки_2.docx, 9
8.97KB

Имя

Лара Крофт

Комментарий

Списки наблюдения

Default Watch List   

☒ Активное

Сохранить

Назад

- Добавьте одну или несколько фотографий и введите имя человека. При необходимости добавьте комментарий.

Важно: Лицо на фотографии должно быть надлежащего качества, т. е. в близком к анфас положении. При несоответствии фотографии данному требованию будет выведено сообщение с описанием ошибки.

Создать досье

У вас много досье для загрузки? Попробуйте **Пакетную загрузку досье**

Фотографии




Имя

Вито Корлеоне

Комментарий

Крестный отец

Списки наблюдения

Красный список

☒ Активное

Сохранить Назад

- Из раскрывающегося списка *Списки наблюдения* выберите список (или несколько списков, по очереди), в который следует добавить досье.

Примечание: Если ни один список наблюдения не подходит, *создайте* новый или попросите администратора сделать это.

- Убедитесь, что поставлен флажок *Активное*. Если досье неактивно, оно не будет использоваться для *идентификации лица* в режиме реального времени.
- Нажмите *Сохранить*.

2.5.2 Просмотр досье из списка наблюдения

Все созданные в FindFace Security досье отображаются на вкладке *Досье*. Используйте фильтр *Списки наблюдения*, чтобы отфильтровать досье по спискам.

2.6 Видеостена

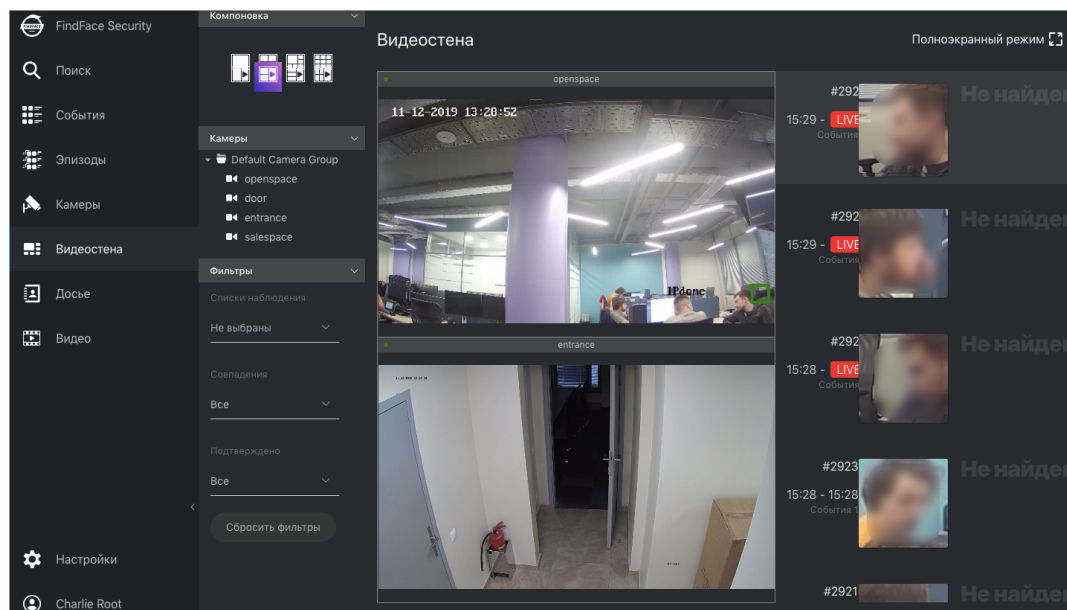
В FindFace Security встроен базовый функционал видеонаблюдения. Используйте видеостену для отображения видео с камер и видеофайлов.

Видеостена может работать в двух режимах (по 4 раскладки в каждом):

- видеотрансляция,
- видеотрансляция с детектированием лиц и лентой эпизодов.

Для отображения на видеостене видеоизображения выполните следующие действия:

1. Перейдите на вкладку *Видеостена*.
2. Выберите режим работы видеостены и раскладку камер.



3. Перетащите на видеостену выбранные камеры.

Вы можете работать с лентой эпизодов на видеостене *по аналогии* с вкладкой *Эпизоды*, включая следующие основные фильтры:

- *Списки наблюдения*
- *Совпадения*.
- *Подтверждено*.

2.7 Мобильный веб-интерфейс

Для работы с FindFace Security также можно использовать упрощенную мобильную версию системы. Мобильное приложение FindFace Security поставляется по запросу для Android.

В этом разделе:

- *Рекомендуемые системные требования*
- *Функции*

2.7.1 Рекомендуемые системные требования

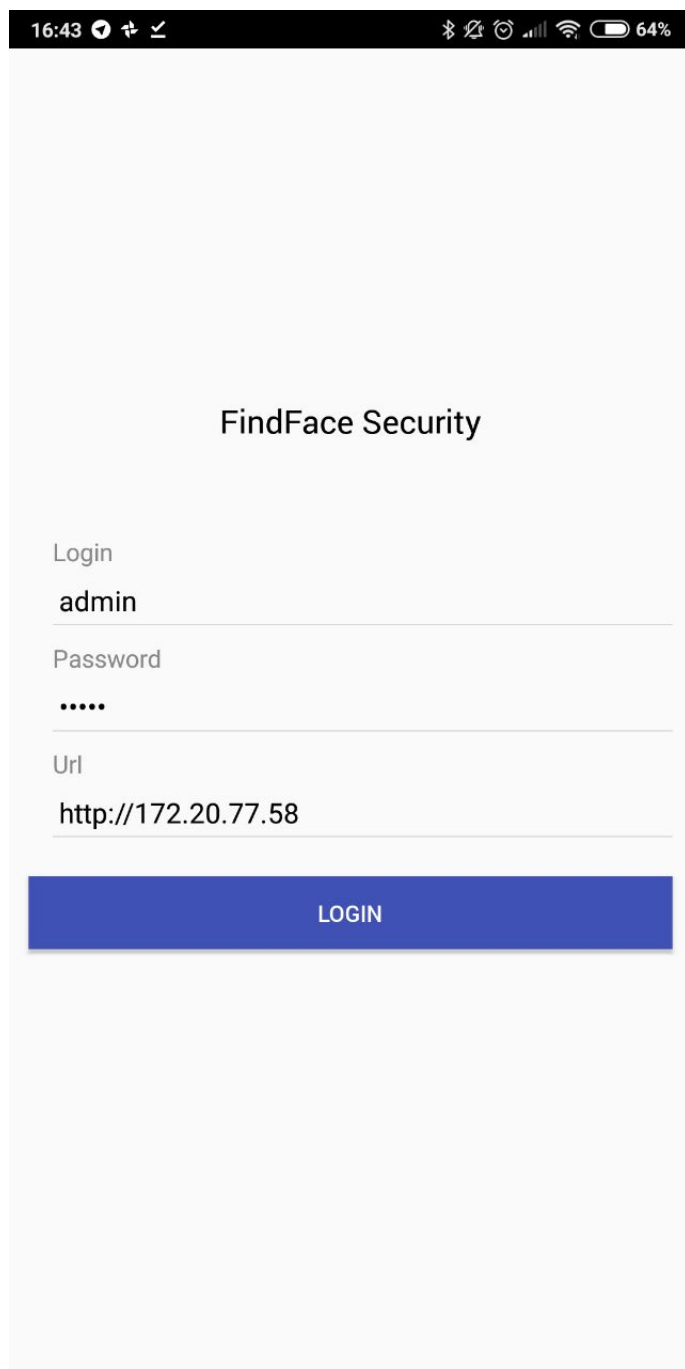
- ОС: Android 6.

Примечание: Функционирование на версиях 4.4-5.x возможно, но не гарантируется.

- CPU: 2 ядра.
- RAM: 2 GB.
- Емкость аккумулятора: не менее 4000 мАч для 8-часовой рабочей смены.

2.7.2 Функции

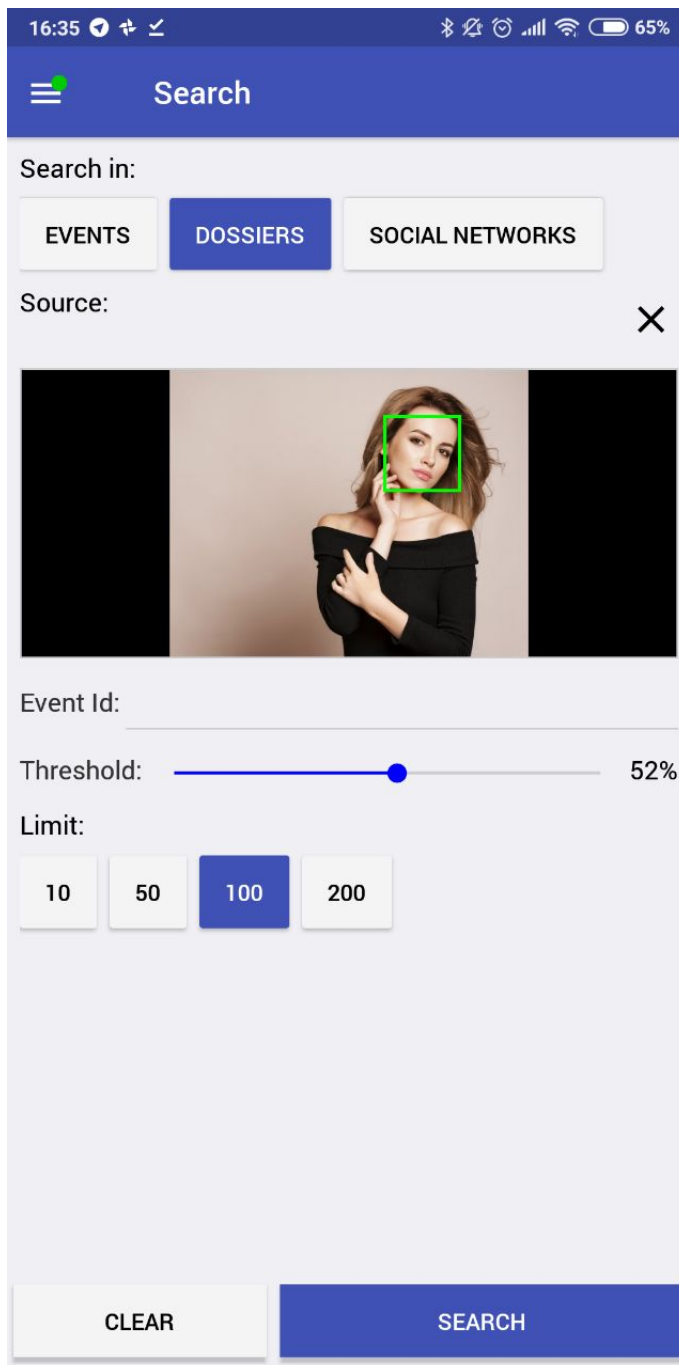
В приложении введите свой логин и пароль FindFace Security, а также адрес сервера FindFace Security и пройдите авторизацию.



The screenshot shows a mobile web interface for 'FindFace Security'. At the top is a black status bar with the time '16:43', various system icons, and a battery level of '64%'. The main content area has a light gray background. In the center, the text 'FindFace Security' is displayed. Below this, there are three input fields: 'Login' with the value 'admin', 'Password' with masked characters '.....', and 'Url' with the value 'http://172.20.77.58'. At the bottom of the form is a blue button labeled 'LOGIN'.

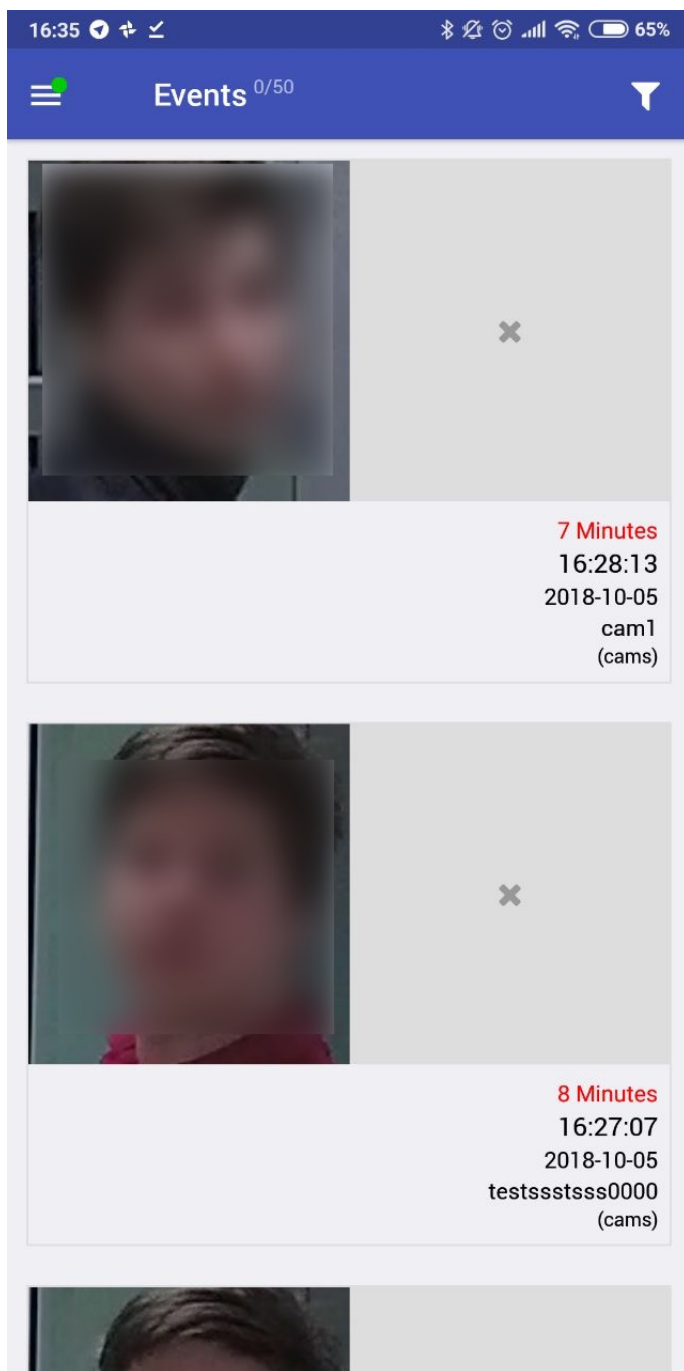
Мобильный веб-интерфейс имеет удобный и интуитивный дизайн и обеспечивает доступ к следующим функциям:

- Поиск лиц в базах данных.



- Идентификация лиц по базам данных в режиме реального времени

Важно: Для того чтобы в мобильной версии получать события в формате push-оповещений, для соответствующего списка наблюдения нужно поставить флажки *Требовать подтверждение события* и *Включить звуковое оповещение* (через полноформатный веб-интерфейс).



- Работа с досье на персону.

16:33 65%

← Edit Dossier

Photos +

Id: 1

Name: Sample

Comment: Comment

Watch lists:

☐ allow ✓

Active: ✓

UPDATE

Работа с данными функциями аналогична полноформатной версии.

Важно: Для доступа в *Настройки*, вам потребуется ввести PIN-код, по умолчанию 1234.

Руководство по интеграции

Данная глава посвящена возможностям интеграции с FindFace Security. Для интеграции своей системы используйте HTTP API, веб-хуки и плагины. Также обратите внимание на список наших готовых интеграций с партнерами.

3.1 HTTP API

Подробная интерактивная документация HTTP API FindFace Security доступна после установки по адресу http://<ffsecurity_ip:port>/api-docs. Изучайте и пробуйте.

Совет: Документацию также можно найти в веб-интерфейсе, перейдя в меню по пунктам *Настройки* -> *Документация API*.

 FindFace Security API doc

Internal API documentation

[Base URL: 172.17.46.22 /]
/swagger.json

Authentication
All API methods require a simple token-based HTTP Authentication. In order to authenticate, you should put word "Token" and a key into the Authorization HTTP header, separated by a whitespace:
"Authorization: Token be94403fb59c305b8d6db7ea1f90e019bef3ac85389cf2b10e04b8cf495b31a3"
All requests that fail to provide a valid authentication token will result in a HTTP 401 Unauthorized response.

Parameters Format
There are two ways to pass parameters to the API methods:

- application/json: parameters are represented by a JSON contained in the body.
- multipart/form-data: parameters are encoded into separate parts. This way supports uploading a photo image file in the same request.

Additional Information
Standard extraction limits:

- Image formats: JPEG, PNG, WEBP
- Maximum photo file size: 10 MB
- Maximum photo resolution: 6000 pixels on the biggest side
- Minimal size of a face: 50x50 pixels

Check /etc/findface-extraction-api.ini for custom definition
[Contact the developer](#)

Schemes

HTTP

Authorize

auth

System Preferences

3.2 Вебхуки

Вы можете настроить FindFace Security для автоматической отправки уведомлений об определенных событиях, эпизодах, записях счетчика на заданный URL-адрес. Для этого создайте и настройте вебхук. При наступлении нужного события/эпизода/записи счетчика FindFace Security отправит HTTP-запрос на URL-адрес, указанный в настройках вебхука.

Вебхуки можно использовать для решения разнообразных задач, например, для уведомления пользователя об определенном событии, вызова определенных действий на целевом веб-сайте, при решении задач безопасности, таких как удаленное автоматическое управление доступом и др.

В этом разделе:

- *Настройка вебхука*
- *Как работает вебхук*

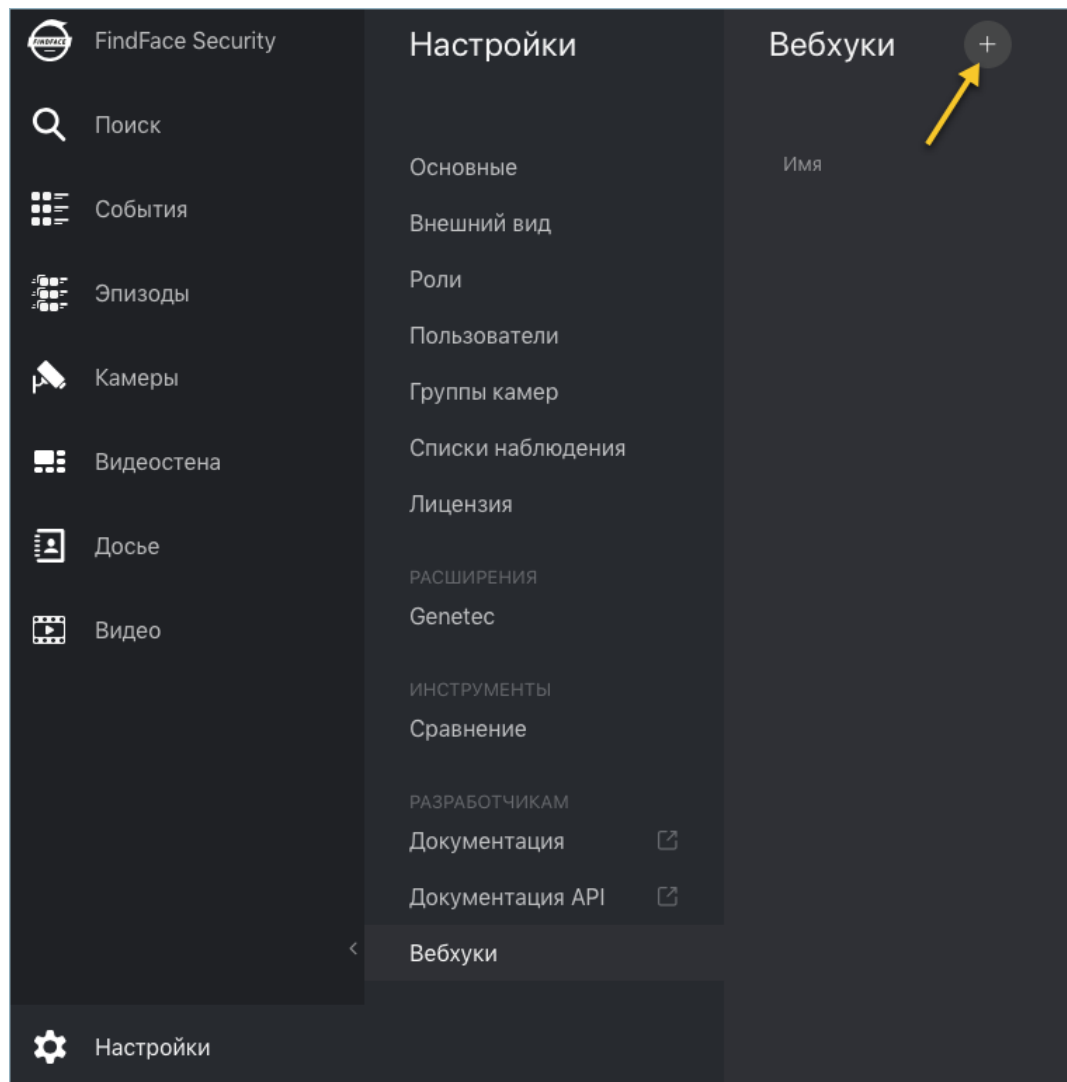
3.2.1 Настройка вебхука

Важно: Для создания вебхука необходимы права администратора.

Примечание: Для того чтобы использовать вебхуки, обязательно укажите по крайней мере один из параметров SERVICE_EXTERNAL_ADDRESS/EXTERNAL_ADDRESS в файле /etc/ffsecurity/config.py.

Для создания вебхука выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Вебхуки*.
2. Нажмите *+*.



3. Введите имя вебхука.

Создать вебхук

• Заголовок вебхука

Вебхук

• Url

https://mywebhook.com

Размер пачки сообщений

4

Количество попыток отправки (0 - не ограничено)

10

Фильтры

```
{
  "events": {
    "allowed_bs_types": [
      "overall",
      "realtime"
    ],
    "camera_group_in": [],
    "camera_in": [],
  }
}
```

☒ Активный

Сохранить Назад

4. Укажите адрес, на который будут отправляться оповещения.
5. Вы можете отправлять уведомления в пакетном режиме. Укажите максимальное количество уведомлений в пакете. Фактическое количество может быть меньшим.

6. Укажите максимальное количество попыток отправить уведомление. Интервал между попытками увеличивается экспоненциально с максимумом 100 секунд.
7. FindFace Security будет автоматически отправлять оповещения о событиях, эпизодах и показаниях счетчика, удовлетворяющих заданным фильтрам. Фильтровать события можно по следующим параметрам:

События:

- `allowed_bs_types`: *режим отслеживания лиц на видео*, возможные значения: `overall`, `realtime`.
- `camera_group_in`: id группы камер, число.
- `camera_in`: id камеры, число.
- `matched_lists_in`: id списка наблюдения, число.
- `matched_dossier_in`: id совпавшего досье, число.
- `matched`: статус события *совпадение* (`true` или `false`), логический.
- `confidence_gte`: минимальное значение уверенности алгоритма, число.

Эпизоды:

- `allowed_types`: статус эпизода, возможные значения: открытие эпизода (`episode_open`), добавление нового события в эпизод (`episode_event`), закрытие эпизода (`episode_close`).
- `camera_group_in`: id группы камер, число.
- `camera_in`: id камеры, число.
- `matched_lists_in`: id списка наблюдения, число.
- `matched`: статус события *совпадение* (`true` или `false`), логический.
- `events_count_gte`: минимальное количество событий в эпизоде, число.
- `events_count_lte`: максимальное количество событий в эпизоде, число.

Счетчики:

- `counter_in`: id счетчика, число.
- `camera_group_in`: id группы камер, число.
- `camera_in`: id камеры, число.
- `faces_gte`: минимальное количество лиц в записи счетчика, число.
- `faces_lte`: максимальное количество лиц в записи счетчика, число.
- `silhouettes_gte`: минимальное количество силуэтов в записи счетчика, число.
- `silhouettes_lte`: максимальное количество силуэтов в записи счетчика, число.

```
{
  "events": {
    "allowed_bs_types": [
      "overall",
```

(continues on next page)

(продолжение с предыдущей страницы)

```

        "realtime"
      ],
      "camera_group_in": [],
      "camera_in": [],
      "matched_lists_in": [],
      "matched_dossier_in": [],
      "matched": true,
      "confidence_gte": 0.75
    },
    "episodes": {
      "allowed_types": [
        "episode_open",
        "episode_event",
        "episode_close"
      ],
      "camera_group_in": [],
      "camera_in": [],
      "matched_lists_in": [],
      "matched": true,
      "events_count_gte": 0,
      "events_count_lte": 999
    },
    "counters": {
      "counter_in": [],
      "camera_group_in": [],
      "camera_in": [],
      "faces_gte": 0,
      "faces_lte": 0,
      "silhouettes_gte": 0,
      "silhouettes_lte": 0
    }
  }
}

```

Важно: Используйте только фильтры, соответствующие цели поиска. Для выключения фильтра удалите его из вебхука. Не оставляйте фильтр пустым ({}), поскольку в этом случае фильтр вернет пустой результат.

Примечание: Для получения всех оповещений передайте скобки без вложенных фильтров:

```
{}
```

Совет: Пример №1. Получение оповещений обо всех событиях:

```
{ "events": {} }
```

Пример №2. Получение оповещений об открытии эпизодов с совпадениями в досье:

```
{ "episodes": { "allowed_types": ["episode_open"], "matched": true } }
```

Примечание: Вы можете задать несколько значений в фильтрах с квадратными скобками. В

этом случае вебхук будет активироваться при совпадении одного из значений фильтра. В примере ниже вы будете оповещены о событии с группы камер 1 или 3, если совпало досье с id 12 или 25.

```
{
  "events": {
    "camera_group_in": [1, 3],
    "matched_dossier_in": [12, 25],
  },
}
```

8. Поставьте флажок *Активный*.

9. Нажмите *Сохранить*.

3.2.2 Как работает вебхук

Для тестирования работы вебхука можно использовать следующий простой веб-сервер на Python, обеспечивающий захват отправленных вебхуком оповещений о событиях:

```
from pprint import pprint
from aiohttp import web

async def handle(request):
    pprint(await request.json())
    return web.Response(status=200)

app = web.Application()
# for aiohttp v 3.x
# app.add_routes([web.post('/', handle)])

# for aiohttp v 2.x
app.router.add_post('/', handle)

web.run_app(app, port=8888)
```

Важно: Используемый вами сервер-ловушка оповещений по вебхуку должен возвращать ответ HTTP 200 после получения вебхука от FindFace Security, по аналогии с примером выше.

Если для вебхука не задано ни одного фильтра, данный веб-сервер будет получать оповещения о каждом произошедшем в системе событии, эпизоде и записи счетчика. Оповещения отправляются в следующем формате:

Событие

```
[{'acknowledged': True,
  'acknowledged_by': None,
  'acknowledged_date': '2020-05-18T15:08:38+00:00',
  'acknowledged_reaction': '',
  'bs_type': 'overall',
  'camera': None,
```

(continues on next page)

(продолжение с предыдущей страницы)

```

'camera_group': 1,
'confidence': 0.0,
'created_date': '2020-05-18T15:08:38+00:00',
'episode': None,
'event_type': 'event_created',
'face': 'http://172.17.46.134/uploads/2020/05/18/event/150842_face_AgohWm.jpg',
'features': {'age': None,
             'beard': None,
             'emotions': None,
             'gender': None,
             'glasses': None,
             'liveness': None,
             'medmask': None},
'frame': 'http://172.17.46.134/uploads/2020/05/18/event/150842_full_frame_Y3vtGe.jpg',
'frame_coords_bottom': 320,
'frame_coords_left': 117,
'frame_coords_right': 170,
'frame_coords_top': 242,
'id': '4267625862518432158',
'looks_like_confidence': None,
'matched': False,
'matched_dossier': None,
'matched_face': '',
'matched_lists': [-1],
'quality': -0.000766,
'scores': {'liveness_score': None,
           'quality': -0.000766480341553,
           'track': {'first_timestamp': '2020-05-18T15:08:38',
                     'id': '43277e17b1c2-44',
                     'last_timestamp': '2020-05-18T15:08:39'},
           'track_duration_seconds': 2.502499999999997},
'video_source': 1,
'webhook_type': 'events']]

```

Открытие эпизода

```

[{'acknowledged': True,
  'acknowledged_by': None,
  'acknowledged_date': None,
  'acknowledged_reaction': '',
  'best_event': '4267637154774219594',
  'camera_groups': [1],
  'cameras': [],
  'closed_date': None,
  'created_date': '2020-05-18T16:18:49.111880Z',
  'event_type': 'episode_open',
  'events_count': 1,
  'features': None,
  'id': 2118,
  'last_event': {'acknowledged': True,
                 'acknowledged_by': None,
                 'acknowledged_date': '2020-05-18T16:18:46+00:00',
                 'acknowledged_reaction': '',
                 'camera': None,
                 'camera_group': 1,

```

(continues on next page)

(продолжение с предыдущей страницы)

```

'confidence': 0.0,
'created_date': '2020-05-18T16:18:46+00:00',
'episode': 2118,
'face': 'http://172.17.46.134/uploads/2020/05/18/event/161849_face_j2TQwk.jpg',
'features': {'age': None,
             'beard': None,
             'emotions': None,
             'gender': None,
             'glasses': None,
             'liveness': None,
             'medmask': None},
'frame': 'http://172.17.46.134/uploads/2020/05/18/event/161849_full_frame_vTfuH9.
↪jpg',
'frame_coords_bottom': 327,
'frame_coords_left': 778,
'frame_coords_right': 901,
'frame_coords_top': 161,
'id': '4267637154774219594',
'looks_like_confidence': None,
'matched': False,
'matched_dossier': None,
'matched_face': '',
'matched_lists': [-1],
'quality': -0.000311,
'scores': {'liveness_score': None,
           'quality': -0.000311948591843,
           'track': {'first_timestamp': '2020-05-18T16:18:46',
                     'id': '1ee9a3612af3-9',
                     'last_timestamp': '2020-05-18T16:18:47'},
           'track_duration_seconds': 2.0399999999999999},
'video_source': 2},
'matched_event': None,
'matched_lists': [-1],
'open': True,
'webhook_type': 'episodes']]

```

Заккрытие эпизода

```

[{'acknowledged': True,
  'acknowledged_by': None,
  'acknowledged_date': None,
  'acknowledged_reaction': '',
  'best_event': {'acknowledged': True,
                 'acknowledged_by': None,
                 'acknowledged_date': '2020-05-18T15:09:57+00:00',
                 'acknowledged_reaction': '',
                 'camera': None,
                 'camera_group': 1,
                 'confidence': 0.0,
                 'created_date': '2020-05-18T15:09:57+00:00',
                 'episode': 518,
                 'face': 'http://172.17.46.134/uploads/2020/05/18/event/151012_face_5L1HQL.jpg',
                 'features': {'age': None,
                             'beard': None,
                             'emotions': None,

```

(continues on next page)

(продолжение с предыдущей страницы)

```

        'gender': None,
        'glasses': None,
        'liveness': None,
        'medmask': None},
    'frame': 'http://172.17.46.134/uploads/2020/05/18/event/151012_full_frame_CdNn2N.
↪jpg',
    'frame_coords_bottom': 299,
    'frame_coords_left': 917,
    'frame_coords_right': 1005,
    'frame_coords_top': 179,
    'id': '4267626103667833809',
    'looks_like_confidence': None,
    'matched': False,
    'matched_dossier': None,
    'matched_face': '',
    'matched_lists': [-1],
    'quality': -0.653877,
    'scores': {'liveness_score': None,
               'quality': -0.653877139091491,
               'track': {'first_timestamp': '2020-05-18T15:09:57',
                          'id': '43277e17b1c2-231',
                          'last_timestamp': '2020-05-18T15:09:57'},
               'track_duration_seconds': 0.2502555555555554},
    'video_source': 1},
'camera_groups': [1],
'cameras': [],
'closed_date': '2020-05-18T15:10:42.870851Z',
'created_date': '2020-05-18T15:10:12.201230Z',
'event_type': 'episode_close',
'events_count': 1,
'features': None,
'id': 518,
'last_event': {'acknowledged': True,
               'acknowledged_by': None,
               'acknowledged_date': '2020-05-18T15:09:57+00:00',
               'acknowledged_reaction': '',
               'camera': None,
               'camera_group': 1,
               'confidence': 0.0,
               'created_date': '2020-05-18T15:09:57+00:00',
               'episode': 518,
               'face': 'http://172.17.46.134/uploads/2020/05/18/event/151012_face_5LlHQL.jpg',
               'features': {'age': None,
                            'beard': None,
                            'emotions': None,
                            'gender': None,
                            'glasses': None,
                            'liveness': None,
                            'medmask': None},
               'frame': 'http://172.17.46.134/uploads/2020/05/18/event/151012_full_frame_CdNn2N.
↪jpg',
               'frame_coords_bottom': 299,
               'frame_coords_left': 917,
               'frame_coords_right': 1005,
               'frame_coords_top': 179,
               'id': '4267626103667833809',

```

(continues on next page)

(продолжение с предыдущей страницы)

```

        'looks_like_confidence': None,
        'matched': False,
        'matched_dossier': None,
        'matched_face': '',
        'matched_lists': [-1],
        'quality': -0.653877,
        'scores': {'liveness_score': None,
                    'quality': -0.653877139091491,
                    'track': {'first_timestamp': '2020-05-18T15:09:57',
                               'id': '43277e17b1c2-231',
                               'last_timestamp': '2020-05-18T15:09:57'},
                    'track_duration_seconds': 0.250255555555554},
        'video_source': 1},
    'matched_event': None,
    'matched_lists': [-1],
    'open': False,
    'webhook_type': 'episodes']]

```

Запись счетчика

```

[{'camera': 3,
  'camera_group': 1,
  'counter': 2,
  'counter_name': 'smosh',
  'created_date': '2020-05-18T16:15:06.679592Z',
  'event_type': 'counter_record',
  'faces_bbox': [[[700, 210], [894, 210], [894, 464], [700, 464]],
                  [[160, 190], [304, 190], [304, 394], [160, 394]]],
  'faces_count': 2,
  'fullframe': 'http://172.17.46.134/uploads/2020/05/18/counters/161506_fullframe_7Z8n7X.jpg',
  'id': 16,
  'silhouettes_bbox': [[[15, 135], [584, 135], [584, 709], [15, 709]],
                        [[585, 80], [1194, 80], [1194, 684], [585, 684]],
                        [[0, 380], [69, 380], [69, 714], [0, 714]]],
  'silhouettes_count': 3,
  'thumbnail': 'http://172.17.46.134/uploads/2020/05/18/counters/161506_thumb_XLMFwE.jpg',
  'webhook_type': 'counters'}]

```

Для просмотра статуса отправки вебхука в FindFace Security, выполните следующую команду:

```
sudo journalctl -u findface-security.service | grep 'WebhooksManager'
```

Ответ в случае успеха:

```

May 18 21:21:38 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] Updating "events" workers for ↵
↵webhooks: {2}
May 18 21:21:52 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:9KHqkQg7-VW:aa3af58f] ↵
↵Webhook updater processing message(type="events:event_created"). Consumer reception delta: 0.
↵002617
May 18 21:21:52 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↵2, type="events") sent batch(len=1, type="events"): ['4267685965791894192']
May 18 21:21:53 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:v59UsC1V-VW:75c4a9ec] ↵
↵Webhook updater processing message(type="events:event_created"). Consumer reception delta: 0.
↵002386

```

(continues on next page)

(продолжение с предыдущей страницы)

```

May 18 21:21:53 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685968207813297']
May 18 21:21:53 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:vKN1XiIn-VW:c0219d31]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳004499
May 18 21:21:53 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685968837561053']
May 18 21:21:55 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:zZ08v4LJ-VW:feff75dd]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳001905
May 18 21:21:55 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685973269790230']
May 18 21:21:57 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:EbpDel24-VW:083688e2]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳002017
May 18 21:21:57 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685977917324748']
May 18 21:21:57 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:L5XoQTdq-VW:6fle397f]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳009237
May 18 21:21:57 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685979796372941']
May 18 21:21:58 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:ZZ33mwuv-VW:a4cad3a2]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳008542
May 18 21:21:58 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685980899116054']
May 18 21:21:58 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:BfAQRgp0-VW:4c19b207]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳003183
May 18 21:21:58 qa-2 ffsecurity[17851]: INFO      [WebhooksManager:2] <queue: 0> Webhook worker(id-
↳2, type-"events") sent batch(len-1, type-"events"): ['4267685982215838395']

```

Ответ, если попытка была неудачной:

```

May 18 21:29:09 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:jrGdiC7e-VW:2def51cf]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳003909
May 18 21:29:09 qa-2 ffsecurity[17851]: WARNING [WebhooksManager:2] <queue: 1> Webhook worker(id-
↳2, type-"events") Error sending webhook: 405, message='Not Allowed'. Attempt 2 out of 10. Next↳
↳attempt in 0.729 seconds.
May 18 21:29:10 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:jgqLszI7-VW:6a7fea19]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳002402
May 18 21:29:10 qa-2 ffsecurity[17851]: WARNING [WebhooksManager:2] <queue: 2> Webhook worker(id-
↳2, type-"events") Error sending webhook: 405, message='Not Allowed'. Attempt 3 out of 10. Next↳
↳attempt in 1.968 seconds.
May 18 21:29:10 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:LLGB1RRR-VW:053d7c7d]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳003794
May 18 21:29:11 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:4V123NQD-VW:a4640479]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳037162
May 18 21:29:11 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:QKY577ed-VW:41cd531a]↳
↳Webhook updater processing message(type-"events:event_created"). Consumer reception delta: 0.
↳005274

```

(continues on next page)

(продолжение с предыдущей страницы)

```

May 18 21:29:12 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:SVSrljin-VW:973ae0dd]
↳Webhook updater processing message(type="events:event_created"). Consumer reception delta: 0.
↳004273
May 18 21:29:12 qa-2 ffsecurity[17851]: WARNING  [WebhooksManager:2] <queue: 6> Webhook worker(id-
↳2, type="events") Error sending webhook: 405, message='Not Allowed'. Attempt 4 out of 10. Next
↳attempt in 5.314 seconds.
May 18 21:29:12 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:pho03HFd-VW:9c6812d1]
↳Webhook updater processing message(type="events:event_created"). Consumer reception delta: 0.
↳019604
May 18 21:29:13 qa-2 ffsecurity[17851]: INFO      [WebhooksManager] [SC:WDMmZ5MO-VW:842b3397]
↳Webhook updater processing message(type="events:event_created"). Consumer reception delta: 0.
↳231164

```

3.3 Интеграции с партнерами

3.3.1 Genetec Security Center

Интеграция FindFace Security с программным комплексом Genetec Security Center позволяет добавлять функционал распознавания лиц в системы безопасности на базе Genetec.

Настройка интеграции

Интеграция с Genetec Security Center реализуется через плагин `findface-genetec`. По умолчанию плагин отключен.

Перед настройкой интеграции на стороне FindFace Security разверните программное обеспечение Genetec Web SDK и Media Gateway и создайте в Genetec Security Center оповещение **Alarm**, которое будет отображаться при наступлении в FindFace Security события распознавания лица.

Важно: Для того чтобы интеграция Genetec-FindFace работала, вам также понадобится приобрести соответствующую лицензию у Genetec (номер по каталогу лицензий `GSC-1SDK-Ntech-FindFace`) и активировать ее в Genetec Security Center.

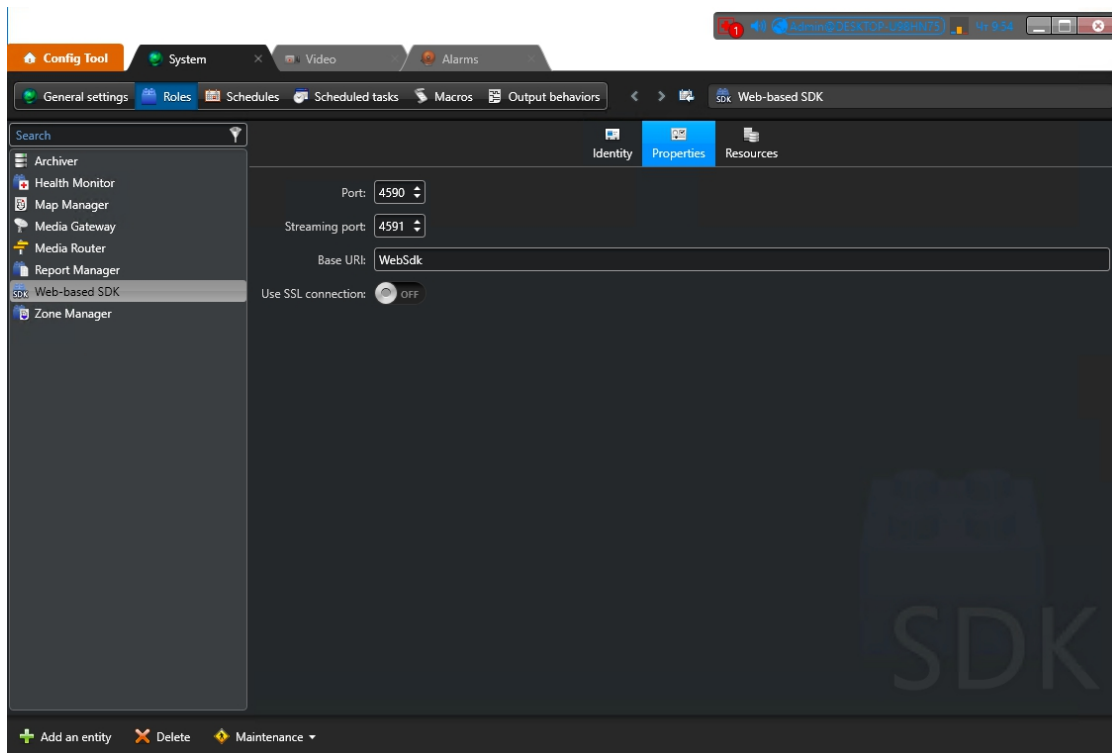
Purchase order			✕
Part #	Description	Quantity	
GSC-5.8	Version 5.8	1	
GSC-Om-E	GSC Omnicast Enterprise Package which includes: Archiving and Auxiliary Archiving support, Media Router, Audio, Remote Security Desk, Camera Sequences, Camera Blocking, Camera Dewarping, Hardware Matrix Support, Time Zone, Edge recording and trickling, Keyboard and Joystick Support, Max. 300 cameras per Archiver / 100 cameras on the Directory machine	1	
GSC-Base-5.8	Genetec Security Center (GSC) Base Package - Version 5.8 which includes: 1 Directory, 5 Security Desk client connections (incl. Web Client), Plan Manager Basic, Alarm Management, Advanced Reporting, System Partitioning, Zone Monitoring, Ю Modules Support, Email Support, Macros Support (actual macro	1	
GSC-Om-E-1C	1 camera connection	20	
GSC-1MobileU	One (1) Genetec Security Mobile app connection Supported only with GSC Mobile	1	
GSC-1SDK-NTech-FindFace	One (1) Genetec SDK connection for Ntech with FindFace	20	
			OK

В этой главе:

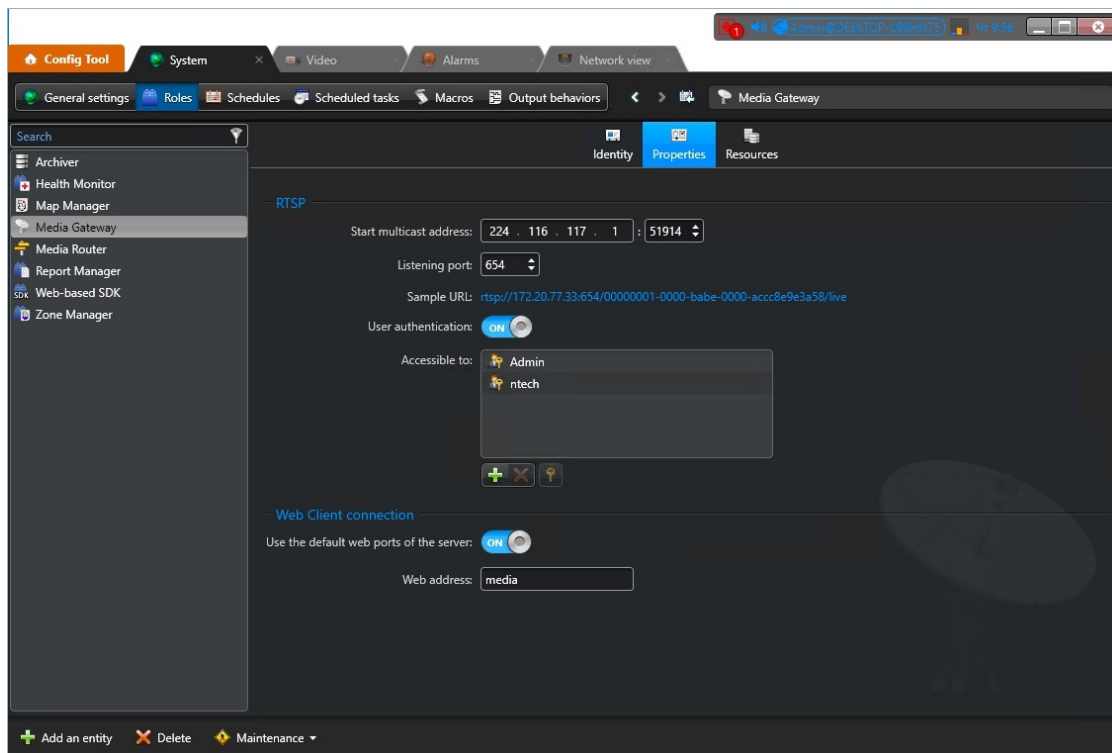
- *Настройка Genetec Web SDK и Media Gateway*
- *Создание оповещения в Genetec Security Center*
- *Включение интеграции Genetec в FindFace Security*
- *Настройка точек доступа в FindFace Security*
- *Импорт камер из Genetec Security Center*
- *Создание списков наблюдения и досье в FindFace Security*

Настройка Genetec Web SDK и Media Gateway

Для того чтобы развернуть Web SDK, используйте ПО Genetec Config Tool. Детали настройки приведены в официальной справочной документации *Security Center Administrator Guide -> Chapter 52: Role Types -> Web-based SDK configuration tabs*.



Для того чтобы развернуть Media Gateway в Genetec Config Tool, ознакомьтесь с содержанием главы *Security Center Administrator Guide -> Chapter 24: Video Deployment*.

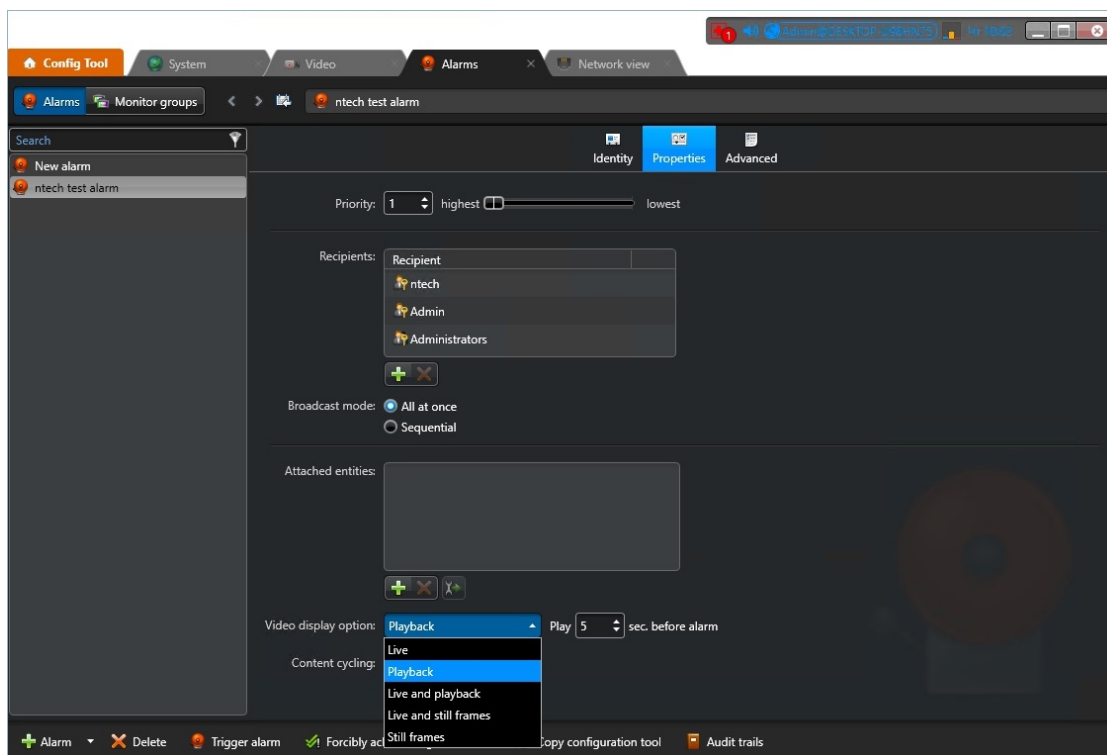


Важно: Убедитесь, что фаервол настроен таким образом, что порты WebSDK и Media Gateway

остаются открытыми.

Создание оповещения в Genetec Security Center

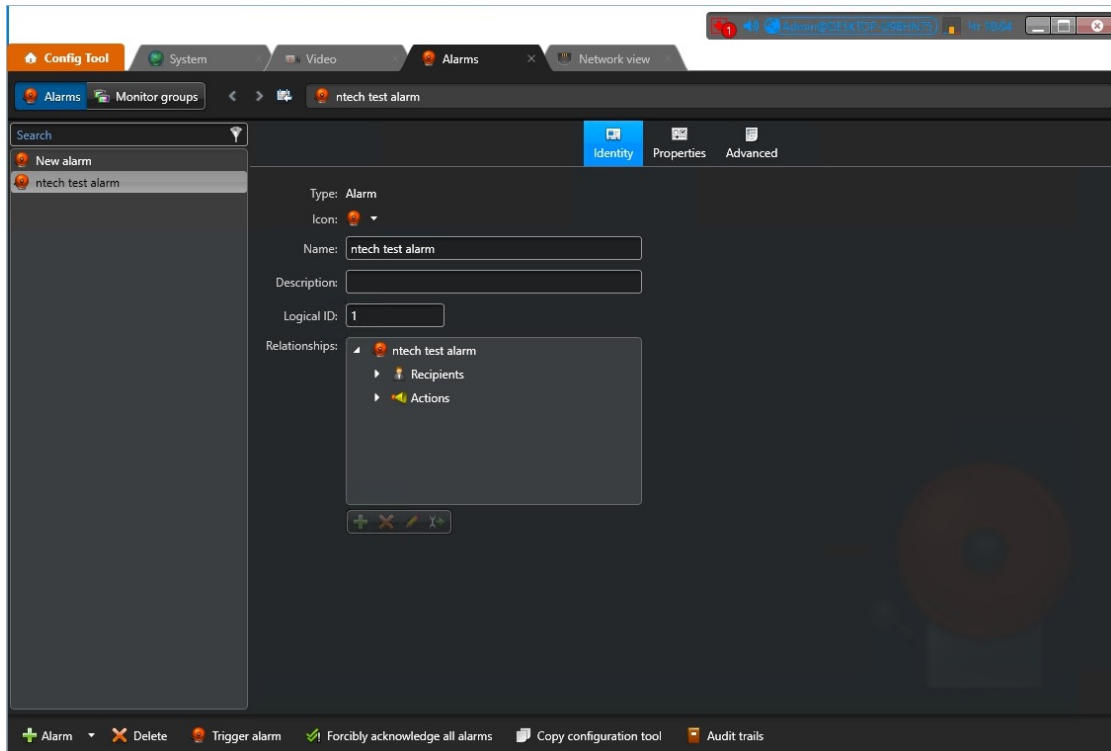
Создайте и настройте новое оповещение **Alarm** в Genetec Config Tool, руководствуясь разделом *Security Center Administrator Guide -> Chapter 48: Alarms -> Creating Alarms*.



Совет: На вкладке *Properties* выберите ту опцию отображения видео *Video display option*, которая в наибольшей степени соответствует вашим нуждам. Доступные опции *Live*, *Playback*, и т. д.

Совет: Для того чтобы активировать операции с оповещением *Alarm Procedures* и автоповорот видео непосредственно во всплывающем окне оповещения, включите *Content cycling*.

При настройке интеграции на стороне FindFace Security вам потребуется ввести логическое id оповещения, которое задается на вкладке *Identity*.



Включение интеграции Genetec в FindFace Security

Для включения плагина `findface-genetec` откройте файл конфигурации `findface-security` и прокомментируйте строку `INSTALLED_APPS.append('ffsecurity_genetec')`.

Также убедитесь, что по крайней мере один из следующих параметров задан: `SERVICE_EXTERNAL_ADDRESS` или `EXTERNAL_ADDRESS`.

```
sudo vi /etc/ffsecurity/config.py

...
# SERVICE_EXTERNAL_ADDRESS is prioritized for FFSecurity webhooks and Genetec plugin.
# EXTERNAL_ADDRESS is used instead if SERVICE_EXTERNAL_ADDRESS is not provided.
# You must provide either SERVICE_EXTERNAL_ADDRESS or EXTERNAL_ADDRESS in order
# to be able to work with FFSecurity webhooks and Genetec plugin.
SERVICE_EXTERNAL_ADDRESS = 'http://127.0.0.1'
# EXTERNAL_ADDRESS is used to access objects created inside FFSecurity via external
↪ links.
EXTERNAL_ADDRESS = ''

...
# FINDFACE SECURITY PLUGINS
# =====
# Uncomment lines below to enable plugins. Please consult documentation for
# a plugin specific settings.
...
# ===== Genetec =====
INSTALLED_APPS.append('ffsecurity_genetec')
```

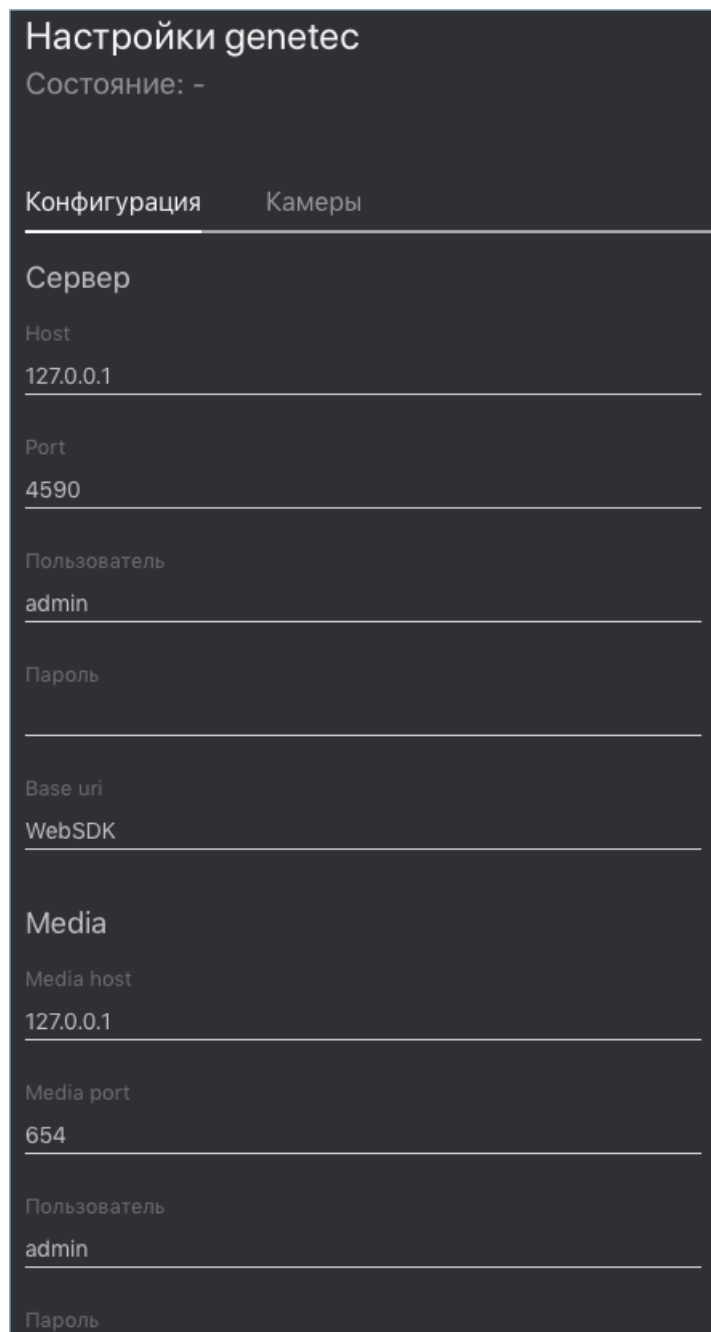
Перезапустите `findface-security`.

```
sudo systemctl restart findface-security.service
```

Настройка точек доступа в FindFace Security

Для того чтобы установить соединение между FindFace Security и Genetec Security Center, выполните следующие действия:

1. Перейдите на вкладку *Настройки*. Выберите *Genetec*.



Настройки genetec

Состояние: -

Конфигурация Камеры

Сервер

Host
127.0.0.1

Port
4590

Пользователь
admin

Пароль

Base uri
WebSDK

Media

Media host
127.0.0.1

Media port
654

Пользователь
admin

Пароль

2. В секциях *Сервер* и *Media*, укажите *настройки* точек доступа Web SDK и Media Gateway.

Важно: Порты WebSDK и Media Gateway должны быть открыты.

- В секции `guiabel:Ids` укажите *логический id* оповещения **Alarm**, которое будет отображаться в Genetec Security Center при наступлении события распознавания лица в FindFace Security.

The screenshot shows a dark-themed interface with the title 'Ids'. Below it is a label 'Alarm id' followed by a text input field containing the number '1'. At the bottom is a purple rounded button labeled 'Сохранить'.

- Нажмите *Сохранить*. Если соединение с Genetec Security Center успешно установлено, статус будет автоматически изменен на *Сконфигурирован*.

Импорт камер из Genetec Security Center

Как только соединение с Genetec Security Center установлено, можно импортировать камеры. Для этого выберите *Камеры* на вкладке *Genetec* и нажмите *Импорт*.

The screenshot shows the 'Genetec Settings' screen. At the top, it says 'State: Configured' and 'English' is selected in a dropdown. There are two tabs: 'Config' and 'Cameras', with 'Cameras' being the active one. Below the tabs is the 'Import Cameras' section, which contains a purple rounded button labeled 'Import'.

Данное действие создаст *группу камер Genetec*, включающую в себя все камеры из Genetec Security Center.

The screenshot shows the 'Camera Groups' screen. It features a table with the following data:

Id	Name	Active
1	1	☒
2	Genetec Imported from Genetec Security Center	☒

Для того чтобы посмотреть список камер, на панели навигации FindFace Security перейдите на вкладку *Камеры*. Для того чтобы исключить камеру из распознавания лиц, просто деактивируйте ее в этом списке.

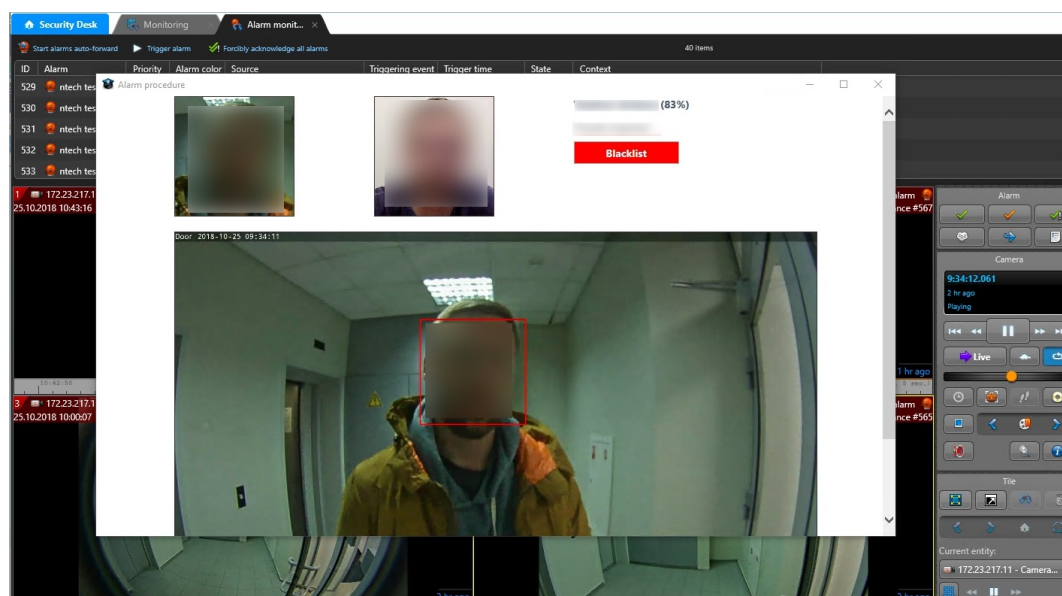
Создание списков наблюдения и досье в FindFace Security

После настройки точек доступа и импорта камер завершите интеграцию, создав *базу данных досье*. После этого оповещения о событиях распознавания лиц будут автоматически отправляться в Genetec Security Center. См. *Оповещения в Genetec Security Center*.

Оповещения в Genetec Security Center

Каждое событие распознавания лица с камеры Genetec, для которого найдено досье, активирует соответствующее оповещение **alarm** в Genetec Security Center. Каждое оповещение, отправленное FindFace Security, связывается с камерой-источником события распознавания лица, поэтому вы можете сразу же просматривать живое или архивное видео в задаче Alarm Monitoring в Genetec Security Desk. FindFace Security также использует операции с оповещением Alarm Procedures для обеспечения пользователя дополнительными данными по событию, такими как:

- обнаруженное на видео лицо
- найденное похожее лицо из базы данных досье
- имя человека и комментарий из досье
- степень схожести лиц (уверенность алгоритма в совпадении)
- название списка наблюдения
- полный кадр



Обработка полученного оповещения о распознавания лица выполняется аналогично другим оповещениям в Genetec Security Center.

3.3.2 Axxon Next

Интеграция FindFace Security с программным комплексом Axxon Next позволяет обрабатывать видеопотоки из системы безопасности на базе Axxon и анализировать их на предмет наличия лиц из досье.

Важно: Один экземпляр FindFace Security может взаимодействовать только с одним сервером Аххон Next.

Интеграция с Аххон Next выполняется с использованием плагина `ffsecurity_axhon`.

Для того чтобы настроить интеграцию с Аххон Next в ОС Ubuntu, выполните следующие действия:

1. Активируйте плагин, раскомментировав в файле конфигурации `/etc/ffsecurity/config.py` строку `INSTALLED_APPS.append('ffsecurity_axhon')`.

```
sudo vi /etc/ffsecurity/config.py

...

# =====
# FINDFACE SECURITY PLUGINS
# =====
# Uncomment lines below to enable plugins. Please consult documentation for
# a plugin specific settings.
# ===== Axhon =====
INSTALLED_APPS.append('ffsecurity_axhon')
```

2. В файле конфигурации раскомментируйте секцию `FFSECURITY->AXXON`. Заполните ее так, как показано в примере ниже. В параметре `api` укажите адрес сервера Аххон Next, по которому FindFace Security будет обращаться к API Аххон и за HLS-потоками архива. В параметре `rtsp` укажите общий адрес видеопотоков на сервере Аххон Next. `name`, `user`, `password`: имя сервера Аххон Next и учетные данные для доступа к нему.

```
FFSECURITY['AXXON'] = [
    {
        'name': 'server_name',
        'api': 'http://example.com/',
        'rtsp': 'rtsp://example.com:554/',
        'user': 'user',
        'password': 'password',
    }
]
```

3. (Опционально). Если в событиях распознавания лиц требуется отображать клипы видео из Аххон Next, раскомментируйте секцию `FFSECURITY_UI_CONFIG['dossier']`.

```
FFSECURITY_UI_CONFIG['dossier'] = {
    'video': True,
}
```

4. Создайте камеры в FindFace Security (см. [Управление видеокамерами](#)). При создании камер вам потребуется ввести их URL в формате `axhon:<friendlyNameLong>`, где `friendlyNameLong` - имя камеры на сервере Аххон Next. Данное имя можно посмотреть в интерфейсе Аххон, или через API Аххон, выполнив команду:

```
curl http://user:password@127.0.0.1/video-origins/

{
  "OLOLOE-DEV/DeviceIpint.vhod_1/SourceEndpoint.video:0:0" : {
    "friendlyNameLong" : "vhod_1.Vhod_1",
    "friendlyNameShort" : "Vhod_1",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
"origin" : "OLOLOE-DEV/DeviceIpint.vhod_1/SourceEndpoint.video:0:0",
"state" : "signal_restored"
}
}
```

Для единственной камеры из примера выше URL должен быть задан как `axxon:vhod_1.Vhod_1`.

На этом настройка интеграции будет завершена. Если интеграция настроена корректно, FindFace Security будет выполнять проверку наличия лиц из досье в видеопотоках Axxon Next, а в событиях распознавания лиц будут отображаться клипы видео из Axxon Next (при соответствующих настройках).

3.4 Плагины

В ходе настройки системы вы можете установить собственные правила обработки лиц, обнаруженных на видео. Для этого понадобится написать плагин на Python.

Плагины подключаются через файл конфигурации компонента `findface-facerouter`. Использование плагинов позволяет индивидуально и очень точно выполнить настройку видеодетекции лиц в том, что касается операций с обнаруженными лицами.

Совет: В качестве примера см. плагин по умолчанию `/opt/ffsecurity/fr_plugin/ffsec_fr_plugin.py`.

3.4.1 Развертывание findface-facerouter в FindFace Security

Для развертывания компонента `findface-facerouter` выполните следующие действия:

1. Установите `findface-facerouter` либо из *консольного инсталлятора*, либо из apt-репозитория следующим образом:

```
sudo apt update
sudo apt install -y findface-facerouter
```

2. Откройте файл конфигурации `/etc/findface-facerouter.py`.

```
sudo vi /etc/findface-facerouter.py
```

3. Если компоненты `findface-facerouter` и `findface-sf-api` установлены на разных физических серверах, раскомментируйте параметр `sfapi_url` и укажите в нем IP-адрес сервера `findface-sf-api`.

```
sfapi_url = 'http://localhost:18411'
```

4. Откройте файл конфигурации `/etc/ffsecurity/config.py`. В параметре `ROUTER_URL` актуализируйте IP-адрес и порт `findface-facerouter` (по умолчанию порт 18820). IP-адрес указывается внешний или внутренний в зависимости от сети, по которой `findface-video-worker` взаимодействует с `findface-facerouter`.

```
sudo vi /etc/ffsecurity/config.py

...
FFSECURITY = {
    'ROUTER_URL': 'http://127.0.0.1:18820/v0/frame?',
```

5. Добавьте сервис `findface-facerouter` в автозагрузку Ubuntu и запустите сервис.

```
sudo systemctl enable findface-facerouter.service && sudo systemctl start findface-facerouter.
↪service
```

6. Перезапустите сервис `findface-security`.

```
sudo systemctl restart findface-security.service
```

3.4.2 Настройка `findface-facerouter` на использование плагинов

Совет: Для начала попробуйте плагин по умолчанию `/opt/ffsecurity/fr_plugin/ffsec_fr_plugin.py`. Или *создайте свой собственный*.

Важно: Обязательно предварительно *измените* структуру биометрической базы Tarantool в соответствии с правилами обработки, заданными в плагинах.

Важно: Компонент `findface-facerouter` должен быть *установлен и настроен*.

Для настройки компонента `findface-facerouter` на использование плагинов выполните следующие действия:

1. Поместите плагин в каталог по вашему выбору. Все используемые плагины должны находиться в одном каталоге.
2. Откройте файл конфигурации `findface-facerouter`. Раскомментируйте параметр `plugin_dir` и укажите каталог с плагином.

Предупреждение: Содержимое `findface-facerouter.py` должно представлять собой синтаксически корректный код Python.

```
sudo vi /etc/findface-facerouter.py

plugin_dir = '/etc/findface/plugins/'
```

3. Перезапустите `findface-facerouter`.

```
sudo systemctl restart findface-facerouter.service
```

3.4.3 Принципы написания плагина

В этом разделе:

- *Архитектура плагина*
- *Метод preprocess*
- *Метод process*
- *Метод shutdown*

Архитектура плагина

После того как компонент `findface-video-worker` обнаруживает лицо, он отправляет его в компонент `findface-facerouter` в виде HTTP API запроса. Для обработки запроса каждый плагин должен экспортировать функцию `activate(app, ctx, plugin_name, plugin_source)`.

Параметры функции `activate`:

- `app`: сущность `tornado.web.Application` компонента `findface-facerouter`.
- `ctx`: контекст, передаваемый плагину при активации.
- `plugin_name`: имя активируемого плагина.
- `plugin_source`: объект источника, из которого загружается плагин.

При активации плагину передается следующий контекст:

1. `request.ctx.sfapi`: настроенный экземпляр `ntech.sfapi_client.Client`, к которому можно обращаться напрямую для обработки результата видеодетекции (создание новой галереи, добавление лица в галерею и т. д.).
2. `plugins`: `OrderedDict` со всеми плагинами (`key`: имя плагина, `value`: результат, возвращенный функцией `activate`).
3. `idgen`: генератор id, который может вызываться как `ctx.idgen()`.

Функция `activate(app, ctx, plugin_name, plugin_source)` должна вернуть объект со следующими методами:

1. `preprocess`,
2. `process`,
3. `shutdown` (опционально).

Метод preprocess

В данном методе плагин решает, интересует ли его полученное лицо, и если да, возвращает кортеж или список, содержащий одну или несколько строк: `'facen'`, `'gender'`, `'age'`, `'emotions'`, что соответственно означает, что нужно извлечь вектор признаков, распознать пол, возраст и/или эмоции. Если возвращенные кортеж или список непусты, компонент `findface-facerouter` перенаправляет обнаруженное лицо в компонент `findface-sf-api` в запросе `/detect POST` с соответствующими параметрами в query string (`facen=on`, `gender=on`, `age=on`, `emotions=on`).

Синтаксис базового метода `preprocess`, от которого следует наследоваться (см. класс `Plugin`):

```
preprocess(self, request: FrHTTPRequest, labels: typing.Mapping[str, str]) → typing.Tuple[str]
```


Параметры

- `FrHTTPRequest` (*tornado.httpserver.HTTPRequest*) – HTTP API запрос, который включает в себя дополнительный аргумент `params`
- `labels` (*dictionary*) – пользовательский набор меток кадра, который задается в параметрах задания для компонента `findface-video-worker` и затем присваивается кадру

Аргумент `params` `FrHTTPRequest` содержит следующие поля:

Параметры

- `photo` (*bytes*) – кадр с лицом в формате JPEG
- `face0` (*bytes*) – нормализованное изображение лица
- `bbox` (list of integers `[[x1,y1,x2,y2]]`, where `x1`: x coordinate of the top-left corner, `y1`: y coordinate of the top-left corner, `x2`: x coordinate of the bottom-right corner, `y2`: y coordinate of the bottom-right corner) – координаты рамки с лицом в кадре
- `cam_id` (*string*) – id видеокамеры
- `timestamp` (*datetime.datetime*) – временная метка кадра
- `detectorParams` (*dictionary*) – словарь со служебно-отладочной информацией от детектора
- `bs_type` (*string*) – режим поиска лучшего кадра. Доступные опции: `overall` (буферный режим: на сервер отправляется кадр, который был лучшим за весь период нахождения лица в поле зрения видеокамеры), `realtime` (режим реального времени: на сервер отправляются кадры, считающиеся лучшими в последовательных интервалах).
- `labels` (*dictionary*) – (дублирует `params.labels`) пользовательский набор меток кадра, который задается в параметрах задания для компонента `findface-video-worker` и затем присваивается кадру

Решение об обработке лица принимается на основании данных из `request.params`, в том числе пользовательского набора меток, а также из любых других соображений.

Метод `process`

Данный метод вызывается, если метод `preprocess` вернул непустой кортеж или список (`'facen'`, `'gender'`, `'age'`, `'emotions'`). После того, как компонент `findface-sf-api` вернул ответ с результатом детекции (см. запрос `/detect POST`) со всеми запрошенными параметрами лица, компонент `findface-facerouter` вызывает метод `process` плагина для выполнения собственно обработки лица.

Для выполнения обработки лица плагин использует `request.ctx.sfapi`.

Синтаксис базового метода `process`, от которого следует наследоваться (см. класс `Plugin`):

```
process(self, request: FrHTTPRequest, photo: bytes, bbox: typing.List[int], event_id: int,
        detection: DetectFace)
```

Метод `shutdown`

Данный метод вызывается только перед завершением работы компонента `findface-facerouter`.

Синтаксис базового метода `shutdown`, от которого следует наследоваться (см. класс `Plugin`):

```
shutdown(self)
```

3.4.4 Классы и методы

В этом разделе:

- Базовые классы
- Классы объектов
- Обнаружение лица и работа с галереями
- Фильтры для поиска по базе данных
- Отображение сообщений об ошибках

Базовые классы

```
class facerouter.plugin.Plugin
```

Данный класс предоставляет базовые методы для написания плагина, описанные в разделе *Принципы написания плагина*. Пользовательский класс, выполняющий роль оболочки для плагина, должен наследовать от класса `Plugin`.

```
preprocess(self, request: FrHTTPRequest, labels: typing.Mapping[str, str]) → typing.Tuple[str]
```

Возвращает кортеж, включающий в себя одну или несколько строк: 'facen', 'gender', 'age', 'emotions', что соответственно означает, что компонент `findface-facerouter` должен запросить у компонента `findface-sf-api` извлечение вектора признаков, распознать пол, возраст и/или эмоции.

Параметры

- `FrHTTPRequest` (*tornado.httserver.HTTPRequest*) – HTTP API запрос, который включает в себя дополнительный аргумент `params`
- `labels` (*dictionary*) – пользовательские метки из `request.params`

Результат одна или несколько строк 'facen', 'gender', 'age', 'emotions'

Тип результата `tuple`

Аргумент `params` `FrHTTPRequest` содержит следующие поля:

Параметры

- `photo` (*bytes*) – кадр с лицом в формате JPEG
- `face0` (*bytes*) – нормализованное изображение лица
- `bbox` (list of integers `[[x1,y1,x2,y2]]`, where `x1`: x coordinate of the top-left corner, `y1`: y coordinate of the top-left corner, `x2`: x coordinate of the bottom-right corner, `y2`: y coordinate of the bottom-right corner) – координаты рамки с лицом в кадре
- `cam_id` (*string*) – id камеры
- `timestamp` (*datetime.datetime*) – временная метка кадра
- `detectorParams` (*dictionary*) – словарь со служебно-отладочной информацией от детектора

- **bs_type** (*string*) – режим поиска лучшего кадра. Доступные опции: overall (буферный режим: на сервер отправляется кадр, который был лучшим за весь период нахождения лица в поле зрения видеокамеры), realtime (режим реального времени: на сервер отправляются кадры, считающиеся лучшими в последовательных интервалах).
- **labels** (*dictionary*) – (дублирует `params.labels`) пользовательский набор меток кадра, который задается в параметрах задания для компонента `findface-video-worker` и затем присваивается кадру

`process(self, request: FrHTTPRequest, photo: bytes, bbox: typing.List[int], event_id: int, detection: DetectFace)`

Принимает атрибуты обнаруженного лица.

Параметры

- **request** (*tornado.httppserver.HTTPRequest*) – HTTP API-запрос от `findface-video-worker`
- **photo** (*bytes*) – кадр с лицом в формате JPEG из `request.params`
- **bbox** (list of integers `[[x1,y1,x2,y2]]`, where `x1`: x coordinate of the top-left corner, `y1`: y coordinate of the top-left corner, `x2`: x coordinate of the bottom-right corner, `y2`: y coordinate of the bottom-right corner) – координаты рамки с лицом в кадре из `request.params`
- **event_id** (*uint64*) – id обнаруженного на видео лица (автоматически задается компонентом `findface-facerouter` при получении лица от `findface-video-worker`). Может использоваться в качестве пользовательского идентификатора лица в базе данных.
- **detection** (*objects.DetectFace*) – результат детекции, полученный от компонента `findface-sf-api`, включающий в себя запрошенные параметры лица, такие как вектор признаков, пол, возраст, эмоции.

Результат *н/п*

Тип результата *н/а*

`shutdown(self)`

Данный метод вызывается только перед завершением работы компонента `findface-facerouter`.

Параметры *н/п*

Результат *н/п*

Классы объектов

`class objects.BBox`

Представляет собой координаты рамки с лицом.

`class objects.DetectFace`

Представляет собой результат детекции со следующими полями:

Параметры

- **id** (*string*) – id результата детекции в `memcached`
- **bbox** (*objects.Bbox*) – координаты рамки с лицом

- `features` (*dictionary*) – информация о поле (`gender`), возрасте (`age`) и эмоциях (`emotions`) (опционально)

`class objects.DetectResponse`

Представляет собой список объектов `objects.DetectionFace` с дополнительным полем `orientation`, содержащим информацию об ориентации EXIF лица.

Параметры `orientation` (*EXIF orientation*) – ориентация обнаруженного лица

`class objects.FaceId(namedtuple('FaceId', ('gallery', 'face')))`

Представляет собой объект пользовательского идентификатора лица в галерее.

Параметры

- `gallery` (*string*) – имя галереи
- `face` (*integer*) – пользовательский идентификатор лица в галерее

`class objects.Face`

Представляет собой результат поиска лица в базе данных по биометрическому образцу

Параметры

- `id` (`objects.FaceId`) – объект `FaceId`
- `features` (*dictionary*) – информация о поле, возрасте и эмоциях
- `meta` (*dictionary*) – метаданные лица
- `confidence` (*float*) – степень схожести лица с заданным биометрическим образом

`class objects.ListResponse`

Представляет собой список объектов `objects.Face` (т. е. список результатов поиска по биометрическому образцу) с дополнительным полем `next_page`, содержащим информацию о следующей странице с результатами.

Параметры `next_page` (*string*) – курсор следующей страницы с результатами поиска

Обнаружение лица и работа с галереями

`class ntech.sfapi_client.client.Client`

Предоставляет базовые методы для обнаружения лиц на изображении и работы с галереями.

`detect(self, *, url=None, image=None, facen=False, gender=False, age=False, emotions=False, return_facen=False, autorotate=False, detector: str = None, timeout=None) → DetectResponse`

Обнаруживает лица на изображении и возвращает обнаруженные лица.

Параметры

- `url` (*URL*) – URL изображения, если вы передаете общедоступное изображение из интернета
- `image` (*bytes*) – файл PNG/JPG/WEBP, если вы передаете изображение в виде файла
- `facen` (*boolean*) – извлечь вектор признаков из обнаруженного лица. Для сохранения результата детекции в `memcached` передайте `facen=True`.
- `gender` (*boolean*) – извлечь и вернуть информацию о поле
- `age` (*boolean*) – извлечь и вернуть информацию о возрасте

- `emotions (boolean)` – извлечь и вернуть информацию об эмоциях
- `return_facen (boolean)` – вернуть вектор признаков в результате работы метода
- `autorotate (boolean)` – автоматически повернуть изображение в 4-х разных ориентациях для обнаружения лиц в каждой их них. Пересекающиеся направления с $\text{IOU} > 0.5$ будут объединены
- `detector (boolean)` – может принимать значение `nnd` или `normalized`. Детектор `normalized` используется для обработки нормализованных изображений, например, поступающих от видеодетектора лиц
- `timeout (number)` – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат Результат детекции

Тип результата Объект `DetectorResponse`

`gallery(self, name)`

Возвращает объект `sfapi_client.Gallery` для последующей с ним работы (например, получения списка лиц).

Параметры `name (string)` – имя галереи

Результат объект типа «галерея»

Тип результата `sfapi_client.Gallery`

`list_galleries(self, timeout=None):`

Возвращает список галерей.

Параметры `timeout (number)` – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат список галерей со свойствами `name` (имя галереи, строка) и `number` (количество лиц в галереи, число)

Тип результата `list of GalleryListItem`

`class ntech.sfapi_client.gallery.Gallery`

Предоставляет методы для работы с галереями и лицами в них.

`list(self, *, filters: typing.Iterable[filters.Filter] = None, limit: int = 1000, sort: str = "", page=None, ignore_errors=False, timeout=None) → ListResponse`

Возвращает объект типа список с лицами из галереи, удовлетворяющими заданным фильтрам. Возвращаемый объект типа список содержит дополнительное свойство `next_page`, которое может использоваться как значение параметра `page` в последующих запросах.

Параметры

- `filters (sfapi_client.filters.Filter)` – список фильтров
- `limit (integer)` – максимальное количество лиц в ответе
- `sort (string)` – метод сортировки лиц. Возможные значения: `id` (по возрастанию `id`), `-id` (по убыванию `id`), `-confidence` (по убыванию степени схожести лиц). Сортировка по `id` возможна только при отключенном фильтре `facen`, который задает вектор признаков для поиска в базе данных (т.н. идентификация лица). Наоборот, сортировка по степени схожести лиц (`confidence`) возможна

только при включенном фильтре `facen`. По умолчанию метод использует сортировку по возрастанию `id` (вектор признаков не задан) и по убыванию степени схожести лиц (вектор признаков задан).

- `page` – вернуть результаты, начиная с указанной страницы. Номер следующей страницы с результатами возвращается в ответе сервера в виде `next_page`.
- `ignore_errors` (*boolean*) – Игнорировать ошибку обращения к базе данных, если поиск по галерее выполняется, когда некоторые сервера базы данных недоступны. В этом случае поиск будет выполнен с использованием доступных серверов.
- `timeout` (*number*) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат список с лицами из галереи, удовлетворяющими заданным фильтрам.

Тип результата `ListResponse` object

```
add(self, new_id: typing.Union[int, typing.Callable], source: typing.Union[DetectFace, Face, str], *, meta: typing.Dict[str, typing.Union[int, str, typing.List[str]]] = None, regenerate_attempts=None, timeout=None) → Face
```

Создает лицо в галерее.

Параметры

- `new_id` (*integer or callable*) – пользовательский идентификатор лица в базе данных. Может быть (`async`) `callable`, который возвращает `id`. Для генерации `id` может использоваться функция `ctx.idgen()` из контекста.
- `source` (*sfapi_client.DetectFace, sfapi_client.Face, sfapi_client.FaceId, or string*) – источник, из которого лицо добавляется в базу данных, может представлять собой лицо в базе данных или результат детекции.
- `meta` (*dictionary*) – метаданные лица. Ключи могут быть строками, а значения – целыми числами, строками или списками строк. Перед добавлением метаданных в базу данных должна быть создана соответствующая структура.
- `regenerate_attempts` – количество попыток генерации уникального `id` функцией `ctx.idgen()`, если `new_id` `callable`.
- `timeout` (*number*) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат представление созданного лица

Тип результата `Face` object

```
delete(self, face: typing.Union[Face, int], timeout=None) → None
```

Удаляет лицо из галереи.

Параметры

- `face` (*sfapi_client.Face, sfapi_client.FaceId or id in integer*) – лицо, которое нужно удалить из базы данных
- `timeout` (*number*) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат `None`

`get(self, face: typing.Union[Face, int], timeout=None) → Face`

Возвращает лицо из галереи.

Параметры

- `face` (`sfapi_client.Face`, `sfapi_client.FaceId` or `id in integer`) – лицо, которое нужно извлечь из базы данных
- `timeout` (`number`) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат представление лица

Тип результата Face object

`create(self, timeout=None) → None`

Создает галерею в `findface-sf-api` в виде объекта `sfapi_client.Gallery`. Объект `sfapi_client.Gallery` представляет собой промежуточный объект, и для работы с ним не требуется наличия галереи на сервере.

Параметры `timeout` (`number`) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат None

`drop(self, timeout=None) → None:`

Удаляет галерею из `findface-sf-api`.

Параметры `timeout` (`number`) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат None

`update(self, face: typing.Union[Face, str], *, meta: typing.Dict[str, typing.Union[int, str, typing.List[str]]] = None, timeout=None) → Face`

Редактирует метаданные лица в галерее.

Параметры

- `face` (`sfapi_client.Face`, `sfapi_client.FaceId` or `id in integer`) – лицо, метаданные которого нужно заменить в базе данных
- `meta` (`dictionary`) – метаданные лица, которые нужно заменить. Ключи могут быть строками, а значения – целыми числами, строками или списками строк. Если поля `meta` не передаются или `null`, они не изменяются в базе данных.
- `timeout` (`number`) – максимальное время ожидания ответа от ядра FindFace в секундах (если `none`, используется время ожидания ответа, заданное по умолчанию)

Результат представление измененного лица

Тип результата Face object

Фильтры для поиска по базе данных

`class ntech.sfapi_client.filters.Filter`

Общий класс. Представляет собой сводный список фильтров (со значениями), которые должны быть применены к содержимому галереи.

`serialize(self)`

Метод для передачи списка фильтров со значениями в компонент `findface-sf-api`.

Результат имена и значения заданных фильтров

Тип результата `tuple („filtername“, [«value1», «value2»])`

`class ntech.sfapi_client.filters.Id`

Предоставляет методы для фильтрации содержимого галереи по id. Для использования фильтра нужно напрямую вызвать соответствующий classmethod, не создавая экземпляр класса.

`classmethod lte(cls, value: int) → Filter`

Фильтр LTE. Выбрать все лица с id, меньшим или равным указанному.

Параметры `value (integer)` – значение id

Результат имя фильтра (LTE) и его значение.

Тип результата объект класса Filter.

Пример: `Id.lte(1234)` выбирает лица с id, меньшим или равным 1234.

`classmethod gte(cls, value: int) → Filter`

Фильтр GTE. Выбрать все лица с id, большим или равным указанному.

Параметры `value (integer)` – значение id

Результат имя фильтра (GTE) и его значение.

Тип результата объект класса Filter.

Пример: `Id.gte(1234)` выбирает лица с id, большим или равным 1234.

`classmethod oneof(cls, *value: typing.Union[int]) → Filter`

Фильтр IN. Выбрать лица с id из заданной последовательности.

Параметры `value (list of integers)` – список значений id

Результат имя фильтра (IN) и его значение.

Тип результата объект класса Filter.

Пример: `Id.oneof(1234, 5678)` выбирает лицо с id 1234 и/или 5678.

`class ntech.sfapi_client.filters.Meta`

Предоставляет методы для фильтрации содержимого галереи по метаданным. Для использования фильтра нужно напрямую вызвать соответствующий метод, не создавая экземпляр класса.

`classmethod lte(self, value: typing.Union[str, int]) → Filter`

Фильтр LTE. Выбрать все лица, у которых определенная строка в метаданных меньше или равна указанному значению.

Параметры `value (string or integer)` – значение строки с метаданными

Результат имя фильтра (LTE) и его значение.

Тип результата объект класса Filter.

Пример: `Meta('foo').lte(1234)` выбирает лица с мета-строкой `foo`, имеющей значение меньшее или равное 1234.

`classmethod gte(self, value: typing.Union[str, int]) → Filter`

Фильтр GTE. Выбрать все лица, у которых определенная строка в метаданных больше или равна указанному значению.

Параметры `value (string or integer)` – значение строки с метаданными

Результат имя фильтра (GTE) и его значение.

Тип результата объект класса Filter.

Пример: `Meta('foo').gte(1234)` выбирает лица с мета-строкой `foo`, имеющей значение большее или равное 1234.

`classmethod oneof(self, *value: typing.Union[str, int]) → Filter`

Фильтр IN. Выбрать все лица, у которых определенная строка в метаданных совпадает с одним из значений из заданной последовательности.

Параметры `value` (*list of strings or integers*) – список строк с метаданными

Результат имя фильтра (IN) и его значение.

Тип результата объект класса Filter.

Пример: `Meta.oneof(1234, 5678)` выбирает лица с мета-строкой, имеющей значение 1234 и/или 5678.

`classmethod subset(self, *value: str) → Filter`

Фильтр SUBSET. Выбрать все лица, у которых определенная строка содержит все значения из указанной последовательности.

Параметры `value` (*list of strings or integers*) – список строк с метаданными

Результат имя фильтра (SUBSET) и его значение.

Тип результата объект класса Filter.

Пример: `Meta('foo').subset('male', 'angry')` выбирает лица с мета-строкой `foo`, содержащей все значения из последовательности `["male", "angry"]`.

`class ntech.sfapi_client.filters.Detection(Filter)`

Предоставляет метод для идентификации (поиска похожих лиц в базе данных) обнаруженного лица.

`__init__(self, id: typing.Union[str, objects.DetectFace], threshold: float)`

Параметры

- `id` (`objects.DetectFace` or temporary face id in memcached returned by `sfapi_client.Client.detect()`, string) – лицо (результат детекции), которое нужно найти в базе данных
- `threshold` (*float*) – минимальная степень схожести лиц от 0 до 1

Пример: `Detection(det1, 0.77)` выбирает лица, похожие на результат детекции `det1` со степенью схожести, большей или равной 0.77.

`class ntech.sfapi_client.filters.Face(Filter)`

Предоставляет метод для поиска лиц в базе данных, похожих на лицо из галереи.

`__init__(self, id: typing.Union[str, objects.Face], threshold: float)`

Параметры

- `id` (`objects.Face`, `objects.FaceId` or custom face id in the gallery, string) – лицо из базы данных, которое нужно найти
- `threshold` (*float*) – минимальная степень схожести лиц от 0 до 1

Пример: `Detection(FaceId("gal1", 1234), 0.77)` выбирает лица, похожие на лицо с пользовательским идентификатором `face 1234` из галереи `gal1` со степенью схожести, большей или равной `0.77`.

Пример использования нескольких фильтров

```
filters=[filters.Id.gte(123456), filters.Meta('age').gte(45), filters.Meta('camera').oneof('abc',
↪ 'def')]
```

Отображение сообщений об ошибках

`class sfapi_client.SFApiRemoteError`

Данное сообщение об ошибке появляется, если ошибка произошла по причине, отличной от сетевого сбоя.

Сообщение об ошибке содержит как минимум два поля:

- `code` — это код ошибки в виде `CAPS_AND_UNDERSCORES`, который может быть использован для автоматического преобразования.
- `reason` — это описание ошибки, предназначенное для прочтения человеком.

Полный список ошибок

Код ошибки	Описание
<code>UNKNOWN_ERROR</code>	Ошибка неизвестного происхождения.
<code>BAD_PARAM</code>	Запрос может быть прочитан, однако некоторые параметры метода недействительны. Данный тип ответа содержит дополнительные атрибуты <code>param</code> и <code>value</code> для описания ошибочных параметров.
<code>CONFLICT</code>	Конфликт.
<code>EXTRACTION_ERROR</code>	Ошибка при извлечении из лица вектора признаков.
<code>LICENSE_ERROR</code>	Конфигурация системы не соответствует лицензии.
<code>MALFORMED_REQUEST</code>	Запрос неправильно сформирован и не может быть прочитан.
<code>OVER_CAPACITY</code>	Превышен размер очередей в компоненте <code>findface-extraction-api</code> .
<code>SOURCE_NOT_FOUND</code>	Параметре <code>from</code> задано несуществующее лицо.
<code>SOURCE_GALLERY_NOT_FOUND</code>	Параметру <code>from</code> задана несуществующая галерея.
<code>STORAGE_ERROR</code>	Биометрическая база данных недоступна.
<code>CACHE_ERROR</code>	Хранилище memcached недоступно.
<code>NOT_FOUND</code>	Подходящие лица не найдены.
<code>NOT_IMPLEMENTED</code>	Функционал не реализован.
<code>GALLERY_NOT_FOUND</code>	Подходящие галереи не найдены.

`class sfapi_client.SFApiMalformedResponseError`

Это сообщение об ошибке появляется, если ошибка произошла из-за сбоя в сети, или если Клиент не смог прочитать API-ответ от `findface-sf-api`.

3.4.5 Пример

Следующий пример иллюстрирует основы написания плагина и использование классов и методов. Данный плагин запрашивает атрибуты лица у `findface-sf-api`, а затем отправляет запрос в `<FFSEC_URL>/video-detector/process` для создания события с данными, полученными из `findface-sf-api`.

Вы можете найти данный плагин в каталоге `/opt/ffsecurity/fr_plugin/ffsec_fr_plugin.py`. *Настройте* систему на его использование и попробуйте.

Важно: Убедитесь, что в переменной `FFSEC_URL` указаны актуальный IP-адрес и порт сервера `findface-security`.

```
import datetime
import logging
import aiohttp
from dateutil.tz import tzutc
from facerouter.plugin import Plugin
from ntech import sfapi_client
from ntech.asyncio_utils import wrap_future
from ntech.asyncio_utils.noop_cookie import NoopCookieJar
from ntech.tornado_utils import asyncio_to_tornado
# change this if your ffsecurity is located on another host or listens on a non-default port
FFSEC_URL = 'http://127.0.0.1:8002'
logger = logging.getLogger(__name__)
class FFSecurityPlugin(Plugin):
    def __init__(self, ctx, ffsec_url):
        super().__init__(ctx)
        self.ffsec_url = ffsec_url.rstrip('/')
        self.session = aiohttp.ClientSession(cookie_jar=NoopCookieJar())
        self.future_wrapper = asyncio_to_tornado
    def deactivate(self, *args):
        self.session.close()
    def request_headers(self, request):
        return {
            "Authorization": request.headers['Authorization'],
            'X-Request-ID': request.request_id,
        }
    @wrap_futures
    async def preprocess(self, request, labels):
        # somewhat hacky way to pass data between preprocess and process:
        request.ffsec_reception_timestamp = datetime.datetime.now(tzutc())
        headers = self.request_headers(request)
        async with self.session.post(self.ffsec_url + '/video-detector/preprocess',
        ↪headers=headers) as resp:
            resp.raise_for_status()
            resp_json = await resp.json()
            logger.debug("request_id=%r preprocess: ffsecurity response: %r", request.request_id,
            ↪resp_json)
            plugin_wants = resp_json['plugin_wants']
            request.ffsec_plugin_wants = plugin_wants
            logger.info("request_id=%r preprocess: ffsecurity requested features: %r", request.
            ↪request_id, plugin_wants)
            return plugin_wants
    @wrap_futures
    async def process(self, request, photo, bbox, event_id, detection: sfapi_client.DetectFace):
        headers = self.request_headers(request)
        with aiohttp.MultipartWriter('form-data') as mpwriter:
            part = aiohttp.payload.BytesPayload(request.params.photo)
            part.set_content_disposition('form-data', name='photo', filename='photo.jpg')
            mpwriter.append(part)
            part = aiohttp.payload.BytesPayload(b'')
            part.set_content_disposition('form-data', name='face0', filename='norm.png')
```

(continues on next page)

(продолжение с предыдущей страницы)

```

mpwriter.append(part)
part = aiohttp.payload.JsonPayload(request.params.detectorParams)
part.set_content_disposition('form-data', name='detectorParams')
mpwriter.append(part)
part = aiohttp.payload.JsonPayload([list(bbox)])
part.set_content_disposition('form-data', name='bbox')
mpwriter.append(part)
part = aiohttp.payload.StringPayload(request.params.cam_id)
part.set_content_disposition('form-data', name='cam_id')
mpwriter.append(part)
part = aiohttp.payload.StringPayload(request.params.timestamp.isoformat())
part.set_content_disposition('form-data', name='timestamp')
mpwriter.append(part)
part = aiohttp.payload.StringPayload(request.ffsec_reception_timestamp.isoformat())
part.set_content_disposition('form-data', name='reception_timestamp')
mpwriter.append(part)
part = aiohttp.payload.JsonPayload(request.ffsec_plugin_wants)
part.set_content_disposition('form-data', name='plugin_wants')
mpwriter.append(part)
if request.params.bs_type is not None:
    part = aiohttp.payload.StringPayload(request.params.bs_type)
    part.set_content_disposition('form-data', name='bs_type')
    mpwriter.append(part)
part = aiohttp.payload.JsonPayload({
    'id': getattr(detection, 'id', None),
    'features': detection.features,
    'bbox': detection.bbox._asdict(),
    'facen': getattr(detection, 'facen', None),
    'attributes': detection.attributes,
})
part.set_content_disposition('form-data', name='detection')
mpwriter.append(part)
async with self.session.post(
    self.ffsec_url + '/video-detector/process',
    data=mpwriter,
    headers=headers
) as resp:
    await resp.read()
    resp.raise_for_status()
    logger.info("request_id=%r process: ffsecurity accepted event", request.request_id)
async def activate(app, ctx, plugin_name, plugin_source):
    plugin = FFSecurityPlugin(ctx=ctx, ffsec_url=FFSEC_URL)
    return plugin

```

f

`facrouter.plugin`, [222](#)

n

`ntech.sfapi_client.client`, [224](#)

`ntech.sfapi_client.filters`, [227](#)

`ntech.sfapi_client.gallery`, [225](#)

O

`objects`, [223](#)

СИМВОЛЫ

- `__init__()` (метод `ntech.sfapi_client.filters.Detection`), 229
- `__init__()` (метод `ntech.sfapi_client.filters.Face`), 229
- ### A
- `add()` (метод `ntech.sfapi_client.gallery.Gallery`), 226
- ### B
- `BBox` (класс в `objects`), 223
- ### C
- `Client` (класс в `ntech.sfapi_client.client`), 224
- `create()` (метод `ntech.sfapi_client.gallery.Gallery`), 227
- ### D
- `delete()` (метод `ntech.sfapi_client.gallery.Gallery`), 226
- `detect()` (метод `ntech.sfapi_client.client.Client`), 224
- `Detection` (класс в `ntech.sfapi_client.filters`), 229
- `drop()` (метод `ntech.sfapi_client.gallery.Gallery`), 227
- ### F
- `Face` (класс в `ntech.sfapi_client.filters`), 229
- `facrouter.plugin` (модуль), 222
- `Filter` (класс в `ntech.sfapi_client.filters`), 227
- ### G
- `Gallery` (класс в `ntech.sfapi_client.gallery`), 225
- `gallery()` (метод `ntech.sfapi_client.client.Client`), 225
- `get()` (метод `ntech.sfapi_client.gallery.Gallery`), 227
- `gte()` (метод класса `ntech.sfapi_client.filters.Id`), 228
- `gte()` (метод класса `ntech.sfapi_client.filters.Meta`), 228
- ### I
- `Id` (класс в `ntech.sfapi_client.filters`), 228
- ### L
- `list()` (метод `ntech.sfapi_client.gallery.Gallery`), 225
- `lte()` (метод класса `ntech.sfapi_client.filters.Id`), 228
- `lte()` (метод класса `ntech.sfapi_client.filters.Meta`), 228
- ### M
- `Meta` (класс в `ntech.sfapi_client.filters`), 228
- ### N
- `ntech.sfapi_client.client` (модуль), 224
- `ntech.sfapi_client.filters` (модуль), 227
- `ntech.sfapi_client.gallery` (модуль), 225
- ### O
- `objects` (модуль), 223
- `objects.DetectFace` (класс в `objects`), 223
- `objects.DetectResponse` (класс в `objects`), 224
- `objects.Face` (класс в `objects`), 224
- `objects.FaceId` (класс в `objects`), 224
- `objects.ListResponse` (класс в `objects`), 224
- `oneof()` (метод класса `ntech.sfapi_client.filters.Id`), 228
- `oneof()` (метод класса `ntech.sfapi_client.filters.Meta`), 229
- ### P
- `Plugin` (класс в `facrouter.plugin`), 222
- `preprocess()`, 220

`preprocess()` (*метод* `facrouter.plugin.Plugin`), 222
`process()`, 221
`process()` (*метод* `facrouter.plugin.Plugin`), 223

S

`serialize()` (*метод*
 `ntech.sfapi_client.filters.Filter`), 227
`sfapi_client.SFApiMalformedResponseError`
 (*класс в* `ntech.sfapi_client.filters`), 230
`sfapi_client.SFApiRemoteError` (*класс в*
 `ntech.sfapi_client.filters`), 230
`shutdown()`, 222
`shutdown()` (*метод* `facrouter.plugin.Plugin`), 223
`subset()` (*метод* *класса*
 `ntech.sfapi_client.filters.Meta`), 229

U

`update()` (*метод* `ntech.sfapi_client.gallery.Gallery`),
 227