
FindFace

Выпуск 2.1.2

NtechLab

апр. 24, 2024

1	Новое в FindFace CIBR 2.1.2	3
2	Руководство системного администратора	5
2.1	Архитектура	5
2.2	Требования	11
2.3	Развертывание и удаление FindFace CIBR	13
2.4	Администрирование и конфигурирование	39
2.5	Обслуживание и устранение неисправностей	88
2.6	Приложения	121
3	Руководство пользователя	125
3.1	Приступая к работе	125
3.2	Основы веб-интерфейса	126
3.3	Картотека	127
3.4	Происшествия	133
3.5	Поиск лиц в системе	149
3.6	Сравнение двух лиц	152
3.7	Отчеты	153
3.8	Журнал действий пользователей	155
3.9	Удаленный мониторинг и удаленный поиск	157
4	Руководство по интеграции	163
4.1	HTTP API	163

Программный комплекс анализа и распознавания биометрических данных FindFace в специализированной комплектации CIBR (Criminal Investigation Biometric Registry) — это программная платформа для криминалистов, основанная на передовой технологии распознавания лиц. Платформа предназначена для ведения уголовных расследований посредством анализа связанных видеоматериалов и поиска лиц в системах общественной и транспортной безопасности.

Функциональность FindFace CIBR для криминалистов

- **Биометрические криминалистические базы данных.** Для загрузки баз данных в FindFace CIBR создайте картотеку и поместите карточки людей в соответствующие списки наблюдения (Розыск, Беглецы и т. д.). Карточки могут быть загружены пакетно. В карточке содержатся агрегированные данные о человеке: биометрический образец лица, сканы документов, сведения о судимости и другая информация.
- **Происшествия.** Загрузите видеозапись инцидента или фото в FindFace CIBR, чтобы обнаружить и идентифицировать присутствующие на них человеческие лица. При наличии биометрических данных обнаруженных людей в криминалистических базах, FindFace CIBR сможет распознать их на данном этапе.

После предварительного анализа вы сможете обработать результаты распознавания лиц, используя встроенный инструмент. Отделите участников происшествия от посторонних лиц и установите подозреваемых и потерпевших.

- **Поиск.** Поиск определенных людей в системе.
- **Удаленный мониторинг.** Объедините FindFace CIBR с системами общественной и транспортной безопасности. Получайте в режиме реального времени оповещения о появлении определенных людей от удаленных систем распознавания лиц. Это поможет отследить местонахождение и маршрут преступника, определить предполагаемых сообщников, найти пропавших людей.
- **Удаленный поиск.** Поиск определенных людей в удаленных системах распознавания лиц.
- **Верификация лица.** Убедитесь, что два заданных лица принадлежат одному и тому же человеку.
- **Отчеты.** Подробные отчеты по результатам поиска и карточкам.

Технические характеристики

- Платформа на основе AI.
- Удобный консольный инсталлятор и дружелюбный интерфейс.
- Развертывание на одном или нескольких серверах.
- Повышенная производительность и отказоустойчивость в высоконагруженных системах с большим количеством подключенных камер и клиентов.
- Возможность лицензирования в открытых и закрытых системах.
- CPU- и GPU-ускорение на ваш выбор.

Безопасность системы

- Прогрессивное управление пользователями.
- Комплексный, удобный журнал действий пользователя с возможностью поиска.
- Утилиты резервного копирования и восстановления.
- Возможность мониторинга сессий пользователей и блокировки устройств без деактивации учетных записей пользователей.

Полезные мелочи

- Быстрое создание картотеки.
- Расширенный набор поисковых фильтров.

Интеграция

- Интеграция по HTTP API.

Новое в FindFace CIBR 2.1.2

Значительные внутренние изменения продукта, представленные еще в [предыдущей версии](#), позволили реализовать в FindFace CIBR 2.1.2 важные функции и улучшения.

Новые функции:

- Функции, помогающие исключить злоупотребление полномочиями и утечку данных:
 - Защита медиафайлов системы всегда включена, и отключить ее невозможно.
 - Если действие пользователя связано с использованием фотографии, в записи журнала теперь *отображается эта фотография*.
 - Другие функции:
 - Функция очистки событий ежедневного поиска: позволяет экономить ресурсы хранения данных.
- См. *Интеграция с удаленными системами распознавания лиц*

Улучшенные алгоритмы, интерфейс, удобство использования:

- Улучшения поиска: теперь поиск объекта интереса выполняется не только среди участников происшествия, но и по всем распознанным лицам.
- См. *Поиск лиц в системе*
- Одно событие на сервере-марионетке порождает не более одного события в списке удалённого мониторинга — того, у которого степень схожести выше.
- См. *Удаленный мониторинг и удаленный поиск*
- Происшествия, находящиеся в архиве, не расходуют единицы лицензии.
- См.:
- *Архив происшествий*

– *Лицензионные лимиты*

- В происшествиях следите за фактическим ходом выполнения обработки видео на шкале прогресса.

См. *Происшествия*

Руководство системного администратора

Эта глава посвящена развертыванию и последующему обновлению и техническому обслуживанию FindFace CIBR во время эксплуатации.

2.1 Архитектура

Хотя взаимодействие с FindFace CIBR 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 CIBR.

В этой главе:

- *Процесс распознавания*
- *Платформа на основе Docker*
- *Архитектурные элементы*
 - *Схема архитектуры*
 - *Ядро FindFace*
 - *Прикладной модуль (FindFace CIBR)*
- *Аппаратное ускорение на CPU и GPU*

2.1.1 Процесс распознавания

FindFace CIBR обнаруживает изображение человеческого лица на фотографии или видеоизображении и подготавливает его посредством алгоритма нормализации. Далее нормализованное изображение используется для извлечения вектора признаков лица (n-мерного вектора числовых признаков, которые характеризуют лицо). Векторы признаков объектов хранятся в базе данных и в дальнейшем используются для верификации и идентификации.

2.1.2 Платформа на основе Docker

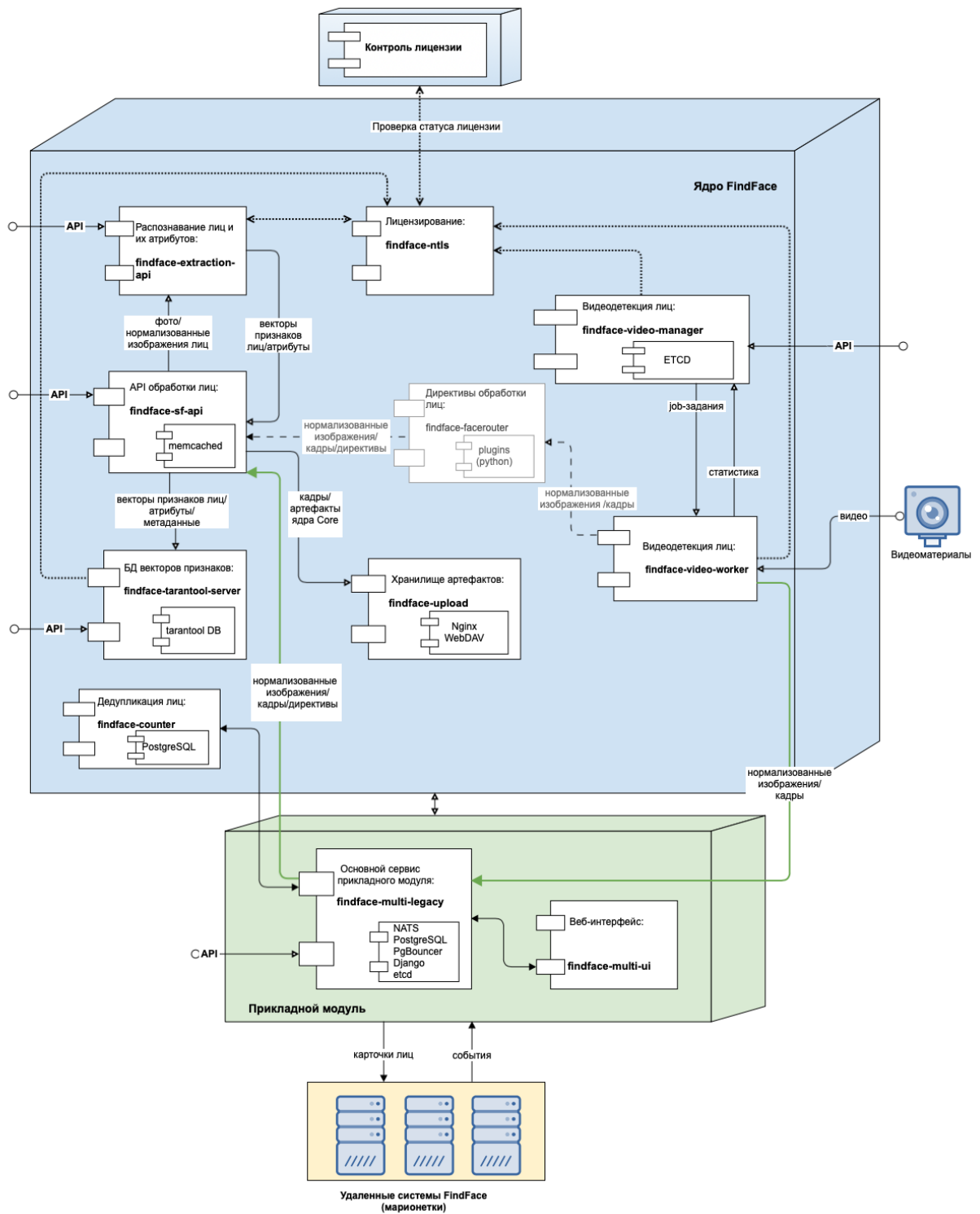
FindFace CIBR разворачивается в Docker, наборе продуктов формата платформа как услуга (PaaS), которые используют виртуализацию на уровне операционной системы, чтобы поставлять программное обеспечение в пакетах, называемых контейнерами. Каждый сервис FindFace CIBR работает в контейнере Docker.

2.1.3 Архитектурные элементы

FindFace CIBR состоит из следующих основных архитектурных элементов:

- Ядро FindFace, передовая технология распознавания объектов на базе искусственного интеллекта, которая может использоваться в качестве отдельного продукта [FindFace Enterprise Server](#).
- Прикладной модуль (FindFace CIBR), реализующий набор инструментов для проведения уголовных расследований с использованием видео- и фотоматериалов.

Схема архитектуры



Ядро FindFace

Внутри ядра FindFace – отдельный продукт [FindFace Enterprise Server](#). Он включает в себя следующие компоненты:

Ком- по- нент	Исполь- зуемые порты	Описание	По- став- щик
findface-extraction-api	18666	Сервис, использующий нейронные сети для обнаружения лица на изображении и извлечения его вектора признаков. Он также распознает атрибуты лиц, например, пол, возраст, эмоции, борода, очки. CPU- или GPU-ускорение.	Соб- ствен- ная разви- тка NtechLab
findface-sf-api	18411	Сервис, реализующий HTTP API обнаружения и распознавания лиц.	
findface-tarantool-server	32001, 330xx, 81xx)	Сервис, обеспечивающий взаимодействие между сервисом findface-sf-api и базой данных векторов признаков лиц на основе Tarantool.	
findface-upload	3333	Веб-сервер на базе NginX, используемый как хранилище исходных изображений, миниатюр и нормализованных изображений лиц.	
findface-facerouter	18820	Сервис, который используется для задания правил обработки обнаруженных на видео лиц. В FindFace CIBR функции findface-facerouter выполняются findface-multi-legacy (см. <i>Прикладной модуль (FindFace CIBR)</i>).	
findface-video-manager	18810, 18811	Сервис, являющийся частью модуля видеодетекции лиц, через который осуществляется управление детекцией лиц на видео, а именно задаются настройки и список видеофайлов для обработки.	
findface-video-worker	18999	Сервис, часть модуля видеодетекции лиц, который распознает лица на видео и отправляет их нормализованные изображения, соответствующие видеокadres и метаданные (такие как время обнаружения) в сервис findface-facerouter для дальнейшей обработки в соответствии с заданными правилами. Поддерживается CPU- и GPU-ускорение.	
findface-ntls	443 (TCP), 3133, 3185	Локальный сервер лицензий, который проверяет подлинность <i>лицензии</i> FindFace CIBR, взаимодействуя с глобальным сервером лицензий NtechLab. Для закрытых систем поддерживается лицензирование через ключ аппаратной защиты и по слепку с оборудования.	
findface-counter	18300	Сервис, используемый для дедупликации событий.	
Tarantool	Порты шардов (по умолчанию 330xx, 81xx)	Стороннее программное обеспечение, на основе которого реализована база данных, хранящая извлеченные векторы признаков и события распознавания лиц. Системные данные, карточки, пользовательские аккаунты камер хранятся в PostgreSQL (часть прикладного модуля FindFace CIBR).	Tarantool
etcd	2379	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	80; SSL: 8002, 8003, 443, 80	Стороннее программное обеспечение, которое реализует веб-интерфейсы системы.	nginx
memcached	11211	Стороннее программное обеспечение, реализующее сервис кэширования данных в оперативной памяти на основе хеш-таблицы. Используется компонентом findface-sf-api для временного хранения извлеченных векторов признаков лиц перед их записью в базу данных Tarantool.	memcached
2.1. Архитектура			9

Прикладной модуль (FindFace CIBR)

Прикладной модуль FindFace CIBR включает в себя следующие компоненты:

Ком- по- нент	Ис- поль- зу- е- мые пор- ты	Описание	По- став- щик
findface-multi-legacy	На- стра- и- ва- ет- ся	Сервис, который служит шлюзом доступа к ядру FindFace. Обеспечивает взаимодействие между ядром FindFace и веб-интерфейсом, HTTP и веб-сокетом, обеспечивает мониторинг лиц, уведомления о событиях и т.д., работу системы в целом.	Соб- ствен- ная раз- ра- ботка
findface-multi-pause	н/п	Внутренние сервисы, помогающие findface-multi-legacy . Сервис findface-multi-audit является заделом на будущее. Он станет полностью функциональными в следующих версиях. На данный момент он частично дублирует функциональность findface-multi-legacy . Используйте findface-multi-legacy для работы с ним. Сервис findface-multi-identity-provider служит для аутентификации и управления пользователями и ролевой моделью доступов.	NtechLab
findface-multi-audit	8012, 8013, 8014		
findface-multi-identity-provider	8022, 8023, 8024		
cleaner	н/п	Сервис отвечает за очистку данных.	
findface-multi-ui	На- стра- и- ва- ет- ся	Основной веб-интерфейс, использующийся для взаимодействия с FindFace CIBR. Функционирует на основе фреймворка Django . Позволяет работать с событиями распознавания лиц, искать лица в базах данных, управлять происшествиями, пользователями, карточками и списками наблюдения и многое другое.	
NATS	4222	Стороннее программное обеспечение, которое реализует брокер сообщений внутри findface-multi-legacy .	NATS
etcd	2379	Стороннее программное обеспечение, реализующее блокировки в сервисе findface-multi-legacy , например блокировки в механизме проверки лицензий, отчетах, обработке видео и т. д.	etcd
Pgbouncer	5439	Стороннее программное обеспечение, обеспечивающее облегченный пул соединений для PostgreSQL. Необязательный компонент, используется для увеличения производительности базы данных при высокой нагрузке.	PgBouncer
PostgreSQL	5432	Стороннее программное обеспечение, реализующее основную базу данных системы. В базе данных хранятся карточки людей и данные для внутреннего использования. Векторы признаков лиц и события распознавания лиц хранятся в Tarantool (часть ядра FindFace).	PostgreSQL

См. также:

- *Хранилища данных FindFace CIBR*

2.1.4 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 *развертывания*.

Важно: Для выбора конфигурации оборудования см. *Требования*.

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

2.2 Требования

В этой главе:

- Системные требования для базовой конфигурации
- Необходимые навыки администратора
- Форматы видеофайлов

2.2.1 Системные требования для базовой конфигурации

Для расчета характеристик серверов FindFace CIBR используйте приведенные ниже требования.

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

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

Важно: На серверах с процессором AMD полная функциональность CPU-сервиса `findface-extraction-api` не гарантируется. Вместо него используйте GPU-сервис `findface-extract-api-gpu` вместе с версией нейронных сетей для GPU.

Примечание: В случае высоконагруженной системы обязательно используйте SSD.

	Минимальная	Рекомендуемая
CPU	Intel Core i5 CPU с 4+ физическими ядрами 3+ ГГц. Поддержка AVX2	Intel Xeon Silver/Gold с 6+ физическими ядрами
	На собственные нужды FindFace CIBR требуется 2 ядра HT > 2.5 ГГц. Характеристики также зависят от количества одновременно обрабатываемых видеофайлов. Для одного файла 720p@25FPS требуется 2 ядра >2.5 ГГц. Поддержка AVX2.	
GPU (опционально)	GeForce® RTX 3060 12 Гб	NVIDIA A10
	Поддерживаемые устройства: NVIDIA, архитектура Pascal и выше. <i>Примечание: видеокарты NVIDIA GeForce RTX 40 Series в настоящее время не поддерживаются.</i>	
RAM	16 Гб	32+ Гб
	Потребление оперативной памяти зависит от: <ul style="list-style-type: none"> • количества используемых алгоритмов, • количества выбранных атрибутов, • количества обрабатываемых лиц и т.д. Для одного видеофайла 720p@25FPS требуется 2 Гб RAM. Пожалуйста, свяжитесь с нашей службой поддержки для получения более подробной информации (support@ntechlab.com).	
HDD (SSD для лучшей производительности)	65 Гб	65+ Гб
	На собственные нужды FindFace CIBR требуется 65 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 Server / Desktop от 18 до 22, только x64, RHEL, CentOS 7, Debian 11.	

Примечание: Вы также можете использовать виртуальную машину на базе процессора Intel, если поддерживаются инструкции AVX2, а 8 физических ядер выделены исключительно виртуальной машине.

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

2.2.2 Необходимые навыки администратора

Администратор FindFace CIBR должен обладать навыками работы с ОС, на которой развернут экземпляр продукта, на уровне продвинутого пользователя.

2.2.3 Форматы видеофайлов

FindFace CIBR поддерживает широкий спектр форматов файлов в зависимости от типа ускорения, CPU или GPU.

Инсталляции с ускорением CPU и GPU поддерживают все кодеки FFmpeg. Кроме того, поддерживаются следующие кодеки:

- *CPU-ускорение*: FLV (как кодек, так и как контейнер), H263, H264, H265, MJPEG, VP8, VP9, MPEG1VIDEO, MPEG2VIDEO, MSMPEG4v2, MSMPEG4v3.
- *GPU-ускорение*: MJPEG, H264, H265, VP9 и другие, в зависимости от списка кодеков, поддерживаемых используемой видеокартой. Кроме того, для инсталляций с `video-worker-gpu`, можно расширить количество поддерживаемых кодеков, включив декодирование видео на CPU, которое недоступно по умолчанию.

Чтобы включить декодирование видео на CPU для инсталляций на GPU-ускорении, выполните следующее:

1. Откройте файл `/opt/findface-cibr/configs/findface-video-worker/findface-video-worker.yaml`.

```
sudo vi /opt/findface-cibr/configs/findface-video-worker/findface-video-worker.yaml
```

2. Установите `cpu: true` в разделе `video_decoder`.

```
...
video_decoder:
  cpu: true
...
```

3. Перезапустите контейнер `findface-cibr-findface-video-worker-1`.

```
sudo docker container restart findface-cibr-findface-video-worker-1
```

2.3 Развертывание и удаление FindFace CIBR

Платформа на основе Docker

Начиная с версии 2.1.1, FindFace CIBR использует для работы платформу Docker. Прежде чем приступить к развертыванию FindFace CIBR, вам потребуется установить и запустить продукты Docker. Для вашего удобства данная глава содержит разделы по подготовке серверов *Ubuntu*, *CentOS*, и *Debian*, в которых описаны тонкости установки Docker на вышеуказанные операционные системы. Для других платформ см. документацию [Docker](#).

Драйвер NVIDIA и NVIDIA Container Runtime (только для GPU)

Если вы собираетесь развернуть FindFace CIBR с GPU-ускорением, вам потребуется установить драйвер NVIDIA и среду NVIDIA Container Runtime. Информацию об этом вы найдете в разделах по подготовке серверов.

Варианты развертывания

После завершения подготовки сервера настало время развернуть FindFace CIBR. Здесь предоставляются следующие возможности:

1. Автоматическая установка односерверной конфигурации FindFace CIBR. Будучи самым простым, данный тип установки отлично подходит для начала работы с FindFace CIBR. Мы рекомендуем выбрать его, если вы только знакомитесь с системой. Подробнее см. [Автоматическое односерверное развертывание FindFace CIBR](#).
2. Полностью настраиваемая установка, которая требует фундаментального понимания архитектуры продукта. См. [Полностью настраиваемая установка](#).

Примечание: Если вы выберете тип установки № 2, не забудьте [установить необходимые модели нейронных сетей](#) вместе с компонентом `findface-extraction-api`.

Вопросы инсталлятора и автоматическое развертывание продукта из файла с параметрами установки

Перед началом активной фазы установки инсталлятор задаст вам несколько вопросов и выполнит ряд автоматических проверок, чтобы убедиться, что сервер соответствует системным требованиям. После ответа на каждый вопрос нужно нажать **Enter**.

Для установки той же конфигурации FindFace CIBR на другом сервере используйте автоматическое развертывание из файла с параметрами установки. В этом случае вам не придется снова отвечать на вопросы инсталлятора. Точный путь к файлу с параметрами установки отображается сразу после последнего вопроса инсталлятора, до начала активной фазы установки:

```
[I 2023-10-06 13:30:48,766 main:142] Your answers were saved to /tmp/findface-installer-  
↪p01n9sn3.json
```

Важно: Обязательно удалите поля `*.config`, `exp_ip` и `int_ip` перед установкой FindFace CIBR на сервере с другим IP-адресом.

Для запуска автоматического развертывания из файла `/tmp/<findface-installer-*>.json`, выполните следующую команду:

```
sudo ./<findface-*>.run -f /tmp/<findface-installer-*>.json
```

Процедуры и первичные инструкции после развертывания

Ознакомьтесь с разделом *Процедуры и первичные инструкции после развертывания*, чтобы узнать, как установить часовой пояс, активировать лицензию и настроить ведение логов. Данный раздел также обучит вас нескольким основным командам, которые помогут вам начать работу с контейнерами FindFace CIBR, если вы новичок в Docker.

Важно: Первый после развертывания запуск сервисов с GPU-ускорением `findface-extraction-api` и `findface-video-worker` может занять продолжительное время из-за процесса кэширования (до 45 минут).

Важно: Хотя FindFace CIBR предоставляет инструменты для защиты от несанкционированного доступа, они не заменяют правильно настроенный межсетевой экран. Обязательно используйте брандмауэр, чтобы усилить защиту сети FindFace CIBR.

Удаление экземпляра продукта

Для удаления экземпляра FindFace CIBR вам потребуется выполнить набор команд. См. раздел *Удаление экземпляра продукта*.

2.3.1 Подготовка сервера Ubuntu

Для того чтобы подготовить сервер на Ubuntu для развертывания FindFace CIBR, следуйте приведенным ниже инструкциям, соблюдая последовательность.

Примечание: Для других платформ обратитесь к следующим ресурсам:

- [Драйверы NVIDIA](#)
- [Docker Engine](#)
- [Docker Compose](#)
- [NVIDIA Container Toolkit](#)

В этом разделе:

- *GPU: Установка драйверов NVIDIA*
- *Установка продуктов Docker*
- *GPU: Установка NVIDIA Container Runtime*

GPU: Установка драйверов NVIDIA

Первым шагом подготовки сервера является установка драйвера NVIDIA. Данный шаг актуален только для конфигурации GPU. Если ваша конфигурация использует CPU-ускорение, можно переходить к *установке Docker*.

Для работы FindFace CIBR с ускорением на GPU потребуется драйвер NVIDIA 530 или более поздней версии. Добавьте репозиторий NVIDIA и установите из него подходящий драйвер.

Предупреждение: Не рекомендуется использовать в качестве альтернативы установщик `.run` с сайта [NVIDIA Driver Downloads](https://developer.download.nvidia.com/compute/cuda/repos/IDversion/$arch/3bf863cc.pub), так как драйверы, установленные посредством запуска `.run` файла, могут конфликтовать с драйверами, установленными посредством пакетов.

Для установки драйвера 530 из репозитория выполните следующие действия:

1. Установите ключ подписи репозитория:

```
arch=$(uname -m); version=$(. /etc/os-release; echo $ID$VERSION_ID | sed -r 's/\././g' \
↪'); sudo bash -c \
"sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/compute/cuda/
↪repos/$ID$version/$arch/3bf863cc.pub \
&& apt update"
```

2. Установите aptitude:

```
sudo apt-get install aptitude
```

3. Выполните установку nvidia-driver-530:

```
sudo aptitude install nvidia-driver-530
```

4. Выполните перезагрузку:

```
sudo reboot
```

Установка продуктов Docker

Продукты Docker должны быть установлены как на CPU-, так и на GPU-серверах. Выполните следующие действия:

1. Обновите индекс пакетов apt и установите пакеты, чтобы разрешить apt использовать репозиторий через HTTPS.

```
sudo apt-get update

sudo apt-get install \
  ca-certificates \
  curl \
  gnupg \
  lsb-release
```

Совет: При выполнении группы команд `sudo apt-get install \` возможны следующие ошибки:


```
E: Could not get lock /var/lib/dpkg/lock-frontend - open (11: Resource temporarily
↳ unavailable)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), is
↳ another process using it?
```

Существуют два способа разрешить их:

1. Принудительно завершите все процессы `apt-get`, запущенные в данный момент в системе.

```
sudo killall apt apt-get
```

2. Если предыдущая команда не помогла, выполните приведенный ниже набор команд. Если некоторые из подлежащих удалению каталогов не существуют, просто перейдите к следующей команде.

```
sudo rm /var/lib/apt/lists/lock
sudo rm /var/cache/apt/archives/lock
sudo rm /var/lib/dpkg/lock
sudo rm /var/lib/dpkg/lock-frontend
sudo dpkg --configure -a
```

2. Добавьте на сервер официальный ключ GPG для Docker (ключ GNU Privacy Guard).

```
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /
↳ etc/apt/keyrings/docker.gpg
```

3. Настройте репозиторий Docker.

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
↳ https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/
↳ null
```

4. Повторно обновите индекс пакетов `apt`.

```
sudo apt-get update
```

Совет: Если вы получили ошибку GPG при выполнении данной команды, попробуйте предоставить разрешение на чтение для файла открытого ключа Docker перед обновлением индекса пакетов.

```
sudo chmod a+r /etc/apt/keyrings/docker.gpg
sudo apt-get update
```

5. Установите последние версии продуктов Docker.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
```

Примечание: Последними протестированными версиями являются версии 26.*. Использование

более поздних версий пакетов Docker, если таковые имеются, остается на ваше усмотрение.

6. Проверьте успешность установки Docker. Следующая команда загружает тестовый образ и запускает его в контейнере. Когда контейнер запускается, он печатает подтверждающее сообщение и завершает работу.

```
sudo docker run hello-world

Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

7. Установите docker-compose.

```
sudo curl -SL https://github.com/docker/compose/releases/download/v2.15.1/docker-
compose-linux-x86_64 -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

8. Выполните [процедуры после установки Docker Engine](#), чтобы упростить свою будущую работу с контейнерами Docker и *FindFace CIBR*. Как только вы будете в состоянии управлять Docker как пользователь без полномочий root, вам больше не понадобится применять `sudo` в командах, связанных с Docker.

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

9. Настройте сеть Docker.

```

sudo su
BIP=10.$((RANDOM % 256)).$((RANDOM % 256)).1
cat > /etc/docker/daemon.json <<EOF
{
    "bip": "$BIP/24",
    "fixed-cidr": "$BIP/24"
}
EOF

```

GPU: Установка NVIDIA Container Runtime

Для развертывания контейнеризованного FindFace CIBR с ускорением на GPU, потребуется установить NVIDIA Container Runtime. Мы рекомендуем установить NVIDIA Container Toolkit, который включает данную среду выполнения. Выполните следующие действия:

1. Укажите репозиторий и установите из него NVIDIA Container Toolkit, выполнив следующие команды.

```

distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
    && curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg -
↪-dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
    && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/
↪libnvidia-container.list | \
        sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-
↪toolkit-keyring.gpg] https://#g' | \
        sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
sudo apt-get update
sudo apt-get install -y nvidia-container-toolkit
sudo nvidia-ctk runtime configure --runtime=docker
sudo systemctl restart docker

```

2. Настройте сеть Docker. Выполните настройку использования среды NVIDIA Container Runtime, установленной вместе с NVIDIA Container Toolkit.

```

sudo su
BIP=10.$((RANDOM % 256)).$((RANDOM % 256))
cat > /etc/docker/daemon.json <<EOF
{
    "default-address-pools":
        [
            {"base": "$BIP.0/16", "size": 24}
        ],
    "bip": "$BIP.1/24",
    "fixed-cidr": "$BIP.0/24",
    "runtimes": {
        "nvidia": {
            "path": "nvidia-container-runtime",
            "runtimeArgs": []
        }
    },
    "default-runtime": "nvidia"
}
EOF

```

3. Перезапустите Docker.

```
systemctl restart docker
```

Теперь все готово для установки FindFace CIBR. См. следующие разделы:

- *Автоматическое односерверное развертывание FindFace CIBR*
- *Полностью настраиваемая установка*

2.3.2 Подготовка сервера CentOS 7

Для того чтобы подготовить сервер на CentOS 7 для развертывания FindFace CIBR, следуйте приведенным ниже инструкциям, соблюдая последовательность.

Примечание: Для других платформ обратитесь к следующим ресурсам:

- [Драйверы NVIDIA](#)
 - [Docker Engine](#)
 - [Docker Compose](#)
 - [NVIDIA Container Toolkit](#)
-

В этом разделе:

- *Установка обновлений*
- *GPU: Установка драйверов NVIDIA*
- *Установка продуктов Docker*
- *GPU: Установка NVIDIA Container Runtime*

Установка обновлений

1. Запустите обновление системы и перезагрузите сервер.

```
sudo yum update  
sudo reboot
```

2. Установите fuse с помощью следующей команды.

```
sudo yum -y install fuse
```

GPU: Установка драйверов NVIDIA

Первым шагом подготовки сервера является установка драйвера NVIDIA. Данный шаг актуален только для конфигурации GPU. Если ваша конфигурация использует CPU-ускорение, можно переходить к *установке Docker*.

Для работы FindFace CIBR с ускорением на GPU потребуется драйвер NVIDIA 530 или более поздней версии. Загрузите соответствующий установщик `.run` с сайта [NVIDIA Driver Downloads](#).

Так как установка производится через установщик `.run`, необходимо установить следующие зависимости:

```
sudo yum install kernel-devel gcc kernel-headers
```

Установка продуктов Docker

Продукты Docker должны быть установлены как на CPU-, так и на GPU-серверах. Выполните следующие действия:

1. Установите пакет `yum-utils` (который предоставляет утилиту `yum-config-manager`) и настройте репозиторий.

```
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-
↪ce.repo
```

2. Установите последние версии продуктов Docker.

```
sudo yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-
↪compose-plugin
```

Примечание: Последними протестированными версиями являются версии 26.*. Использование более поздних версий пакетов Docker, если таковые имеются, остается на ваше усмотрение.

3. Перезапустите Docker.

```
sudo systemctl start docker
sudo systemctl enable docker
```

4. Проверьте успешность установки Docker. Следующая команда загружает тестовый образ и запускает его в контейнере. Когда контейнер запускается, он печатает подтверждающее сообщение и завершает работу.

```
sudo docker run hello-world

Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
```

(continues on next page)

(продолжение с предыдущей страницы)

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
(amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

```
$ docker run -it ubuntu bash
```

Share images, automate workflows, and more with a free Docker ID:

<https://hub.docker.com/>

For more examples and ideas, visit:

<https://docs.docker.com/get-started/>

5. Установите docker-compose.

```
sudo curl -SL https://github.com/docker/compose/releases/download/v2.15.1/docker-  
compose-linux-x86_64 -o /usr/local/bin/docker-compose  
sudo chmod +x /usr/local/bin/docker-compose  
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

6. Выполните процедуры после установки Docker Engine, чтобы упростить свою будущую работу с контейнерами Docker и *FindFace CIBR*. Как только вы будете в состоянии управлять Docker как пользователь без полномочий root, вам больше не понадобится применять `sudo` в командах, связанных с Docker.

```
sudo groupadd docker  
sudo usermod -aG docker $USER  
newgrp docker
```

7. Настройте сеть Docker и использование devicemapper.

```
sudo su  
BIP=10.$((RANDOM % 256)).$((RANDOM % 256)).1  
cat > /etc/docker/daemon.json <<EOF  
{  
  "bip": "$BIP/24",  
  "fixed-cidr": "$BIP/24",  
  "storage-driver": "devicemapper"  
}  
EOF
```

GPU: Установка NVIDIA Container Runtime

Для развертывания контейнеризованного FindFace CIBR с ускорением на GPU, потребуется установить NVIDIA Container Runtime. Мы рекомендуем установить NVIDIA Container Toolkit, который включает данную среду выполнения. Выполните следующие действия:

1. Укажите репозиторий и установите из него NVIDIA Container Toolkit, выполнив следующие команды.

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
  && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/
→libnvidia-container.repo | sudo tee /etc/yum.repos.d/nvidia-container-toolkit.repo
sudo yum clean expire-cache
sudo yum install -y nvidia-container-toolkit
sudo nvidia-ctk runtime configure --runtime=docker
sudo systemctl restart docker
```

2. Настройте сеть Docker. Выполните настройку использования `devicemapper` и среды NVIDIA Container Runtime, установленной вместе с NVIDIA Container Toolkit.

```
sudo su
BIP=10.$((RANDOM % 256)).$((RANDOM % 256))
cat > /etc/docker/daemon.json <<EOF
{
  "default-address-pools":
  [
    {"base": "$BIP.0/16", "size": 24}
  ],
  "bip": "$BIP.1/24",
  "fixed-cidr": "$BIP.0/24",
  "runtimes": {
    "nvidia": {
      "path": "nvidia-container-runtime",
      "runtimeArgs": []
    }
  },
  "default-runtime": "nvidia",
  "storage-driver": "devicemapper"
}
EOF
```

3. Перезапустите Docker.

```
systemctl restart docker
```

Теперь все готово для установки FindFace CIBR. См. следующие разделы:

- *Автоматическое односерверное развертывание FindFace CIBR*
- *Полностью настраиваемая установка*

2.3.3 Подготовка сервера Debian 11

Для того чтобы подготовить сервер на Debian 11 для развертывания FindFace CIBR, следуйте приведенным ниже инструкциям, соблюдая последовательность.

Примечание: Для других платформ обратитесь к следующим ресурсам:

- [Драйверы NVIDIA](#)
 - [Docker Engine](#)
 - [Docker Compose](#)
 - [NVIDIA Container Toolkit](#)
-

В этом разделе:

- *Установка FUSE*
- *GPU: Установка драйверов NVIDIA*
- *Установка продуктов Docker*
- *GPU: Установка NVIDIA Container Runtime*

Установка FUSE

1. Установите файловую систему в пользовательском окружении (FUSE) с помощью следующей команды.

```
sudo apt install fuse -y
```

GPU: Установка драйверов NVIDIA

Первым шагом подготовки сервера является установка драйвера NVIDIA. Данный шаг актуален только для конфигурации GPU. Если ваша конфигурация использует CPU-ускорение, можно переходить к *установке Docker*.

Для работы FindFace CIBR с ускорением на GPU потребуется драйвер NVIDIA 530 или более поздней версии. Загрузите соответствующий установщик `.run` с сайта [NVIDIA Driver Downloads](#).

Так как установка производится через установщик `.run`, необходимо установить следующие зависимости:

```
sudo apt install linux-headers-$(uname -r)
sudo apt install build-essential
```


Установка продуктов Docker

Продукты Docker должны быть установлены как на CPU-, так и на GPU-серверах. Выполните следующие действия:

1. Обновите индекс пакетов `apt` и установите пакеты, чтобы разрешить `apt` использовать репозиторий через HTTPS.

```
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
```

2. Добавьте на сервер официальный ключ GPG для Docker (ключ GNU Privacy Guard).

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o /
↳etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

3. Настройте репозиторий Docker.

```
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]
↳https://download.docker.com/linux/debian \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

4. Повторно обновите индекс пакетов `apt`.

```
sudo apt-get update
```

5. Установите последние версии продуктов Docker.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
↳docker-compose-plugin
```

Примечание: Последними протестированными версиями являются версии 26.*. Использование более поздних версий пакетов Docker, если таковые имеются, остается на ваше усмотрение.

6. Проверьте успешность установки Docker.

```
sudo docker run hello-world
```

Эта команда загружает тестовый образ и запускает его в контейнере. Когда контейнер запускается, он печатает подтверждающее сообщение и завершает работу.

7. Установите `docker-compose`.

```
sudo curl -SL https://github.com/docker/compose/releases/download/v2.15.1/docker-
↳compose-linux-x86_64 -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

8. Выполните [процедуры после установки Docker Engine](#), чтобы упростить свою будущую работу с контейнерами Docker и *FindFace CIBR*. Как только вы будете в состоянии управлять Docker

как пользователь без полномочий root, вам больше не понадобится применять `sudo` в командах, связанных с Docker.

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

9. Настройте сеть Docker.

```
sudo su
BIP=10.$((RANDOM % 256)).$((RANDOM % 256)).1
cat > /etc/docker/daemon.json <<EOF
{
    "bip": "$BIP/24",
    "fixed-cidr": "$BIP/24"
}
EOF
```

GPU: Установка NVIDIA Container Runtime

Для развертывания контейнеризованного FindFace CIBR с ускорением на GPU, потребуется установить NVIDIA Container Runtime. Мы рекомендуем установить NVIDIA Container Toolkit, который включает данную среду выполнения. Выполните следующие действия:

1. Укажите репозиторий и установите из него NVIDIA Container Toolkit, выполнив следующие команды.

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
    && curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg -
↳-dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
    && curl -s -L https://nvidia.github.io/libnvidia-container/$distribution/
↳libnvidia-container.list | \
        sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-
↳toolkit-keyring.gpg] https://#g' | \
        sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
sudo apt-get update
sudo apt-get install -y nvidia-container-toolkit
sudo nvidia-ctk runtime configure --runtime=docker
sudo systemctl restart docker
```

2. Переключитесь на учетную запись суперпользователя.

```
sudo su
```

3. Настройте сеть Docker. Выполните настройку использования среды NVIDIA Container Runtime, установленной вместе с NVIDIA Container Toolkit.

```
BIP=10.$((RANDOM % 256)).$((RANDOM % 256))
cat > /etc/docker/daemon.json <<EOF
{
    "default-address-pools":
        [
            {"base": "$BIP.0/16", "size": 24}
        ]
}
```

(continues on next page)

(продолжение с предыдущей страницы)

```

    ],
    "bip": "$BIP.1/24",
    "fixed-cidr": "$BIP.0/24",
    "runtimes": {
        "nvidia": {
            "path": "nvidia-container-runtime",
            "runtimeArgs": []
        }
    },
    "default-runtime": "nvidia"
}
EOF

```

4. Перезапустите Docker.

```
systemctl restart docker
```

Теперь все готово для установки FindFace CIBR. См. следующие разделы:

- [Автоматическое односерверное развертывание FindFace CIBR](#)
- [Полностью настраиваемая установка](#)

2.3.4 Автоматическое односерверное развертывание FindFace CIBR

Для автоматического односерверного развертывания FindFace CIBR следуйте приведенным ниже инструкциям. Будучи самым простым, данный тип установки отлично подходит для начала работы с FindFace CIBR. Перед процедурой обязательно учтите [системные требования](#) и, в зависимости от вашей ОС, сначала подготовьте сервер:

См.:

- [Подготовка сервера Ubuntu](#)
- [Подготовка сервера CentOS 7](#)
- [Подготовка сервера Debian 11](#)

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](#).

Выполните следующие действия:

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

Примечание: Вместо `findface-*` укажите актуальное имя файла.

```
chmod +x findface-*.run
```

4. Запустите файл `.run`.

```
sudo ./findface-*.run
```

Инсталлятор задаст вам несколько вопросов, после чего проверит, соответствует ли сервер системным требованиям. Ответьте на вопросы надлежащим образом, после ответа каждый раз нажимая `Enter`. Вопросы и ответы следующие:

1. B: Which product should be installed? (Какой продукт следует установить?)

O: 1

```
- 1 [cibr      ] FindFace CIBR
- 2 [video-worker] FindFace Video Worker

(default: cibr)
product> 1
```

2. B: Please choose installation type: (Пожалуйста, выберите тип установки:)

O: 1

```
- 1 [stand-alone ] Single Server
- 2 [multi-worker] Single Server, Multiple video workers
- 3 [images      ] Don't configure or start anything, just load the images
  and copy the models
- 4 [custom      ] Fully customized installation

(default: stand-alone)
type> 1
```

3. B: Directory to install into: (Директория установки:)

O: Укажите директорию установки FindFace CIBR. По умолчанию установка выполняется в `/opt/findface-cibr`. Нажмите `Enter` для подтверждения. В противном случае укажите нужную директорию и нажмите `Enter`.

```
Directory to install into:
(default: /opt/findface-cibr)
dest_dir>
```

4. B: Found X interface(s). Which one should we announce as our external address? (Было найдено X сетевых интерфейсов. Какой интерфейс следует использовать в качестве внешнего IP-адреса?)

O: Выберите интерфейс, который вы собираетесь использовать в качестве IP-адреса FindFace CIBR.

```
Found 2 interface(s). Which one should we announce as our external address?

- 1 [lo      ] 127.0.0.1
- 2 [ens3    ] 192.168.112.254

(default: 192.168.112.254)
ext_ip.advertised> 2
```

5. B: Which variant of Video Worker should be installed? (Какой вариант Video Worker следует установить?)

O: Укажите тип пакета findface-video-worker, CPU или GPU.

```
Which variant of Video Worker should be installed?

- 1 [cpu] CPU-based implementation, slower but doesn't require GPU
- 2 [gpu] CUDA-based implementation of video detector, requires NVIDIA GPU

(default: cpu)
findface-video-worker.variant> 1
```

6. B: Which variant of Extraction API should be installed? (Какой вариант Extraction API следует установить?)

O: Укажите тип пакета findface-extraction-api, CPU или GPU.

```
Which variant of Extraction API should be installed?

- 1 [cpu] CPU-only implementation, slower but doesn't require GPU
- 2 [gpu] CUDA-based implementation, faster, requires NVIDIA GPU (supports
↳ both CPU and GPU models)

(default: cpu)
findface-extraction-api.variant> 1
```

7. B: Do you want to configure face features right now?(y/n) (Хотите ли вы настроить распознавание атрибутов лица сейчас?)

O: Мы рекомендуем настроить функциональность распознавания атрибутов лица на этапе установки. Ответьте у, чтобы начать процесс. Вы можете пропустить данный шаг, ответив n и выполнить необходимые действия позже, следуя инструкции в разделе *Распознавание атрибутов лица*.

```
Do you want to configure face features right now?(y/n)
configure> y
```

8. B: Please select face features to install: (Пожалуйста, выберите атрибуты лиц для установки:)

О: Данный вопрос появляется после запроса на настройку распознавания атрибутов лица. По умолчанию установке подлежат все атрибуты лица. Ответьте **done** для подтверждения. Если определенный атрибут не нужен, вы можете ввести ключевое слово (порядковый номер), связанное с ним. Например, введите 3, чтобы исключить распознавание эмоций. Затем введите **done**.

```
Please select face features to install:
```

```
- 1 [v] Age
- 2 [v] Gender
- 3 [v] Emotions
- 4 [v] Beard
- 5 [v] Glasses
- 6 [v] Medicine masks
```

```
Enter keyword to select matching choices or -keyword to clear selection.
Enter "done" to save your selection and proceed to another step.
face_features> done
```

9. B: Please set findface-cibr admin password (Пожалуйста, установите пароль для администратора FindFace CIBR)

О: Задайте пароль для Супер Администратора (**superuser**).

```
Please set findface-cibr admin password
findface-multi-admin-password> admin
```

После ответа на последний вопрос вы увидите путь к файлу с параметрами установки:

```
Your answers were saved to /tmp/findface-installer-t8qk_isw.json
```

Инсталлятор извлечет образы FindFace CIBR из реестра Ntechlab и запустит следующие сервисы в контейнерах Docker:

Сервис	Контейнер	Конфигурация
pause	findface-cibr-pause-1	Запущен Started
nats-jetstream	findface-cibr-nats-jetstream-1	Запущен Started
mongodb	findface-cibr-mongodb-1	Запущен Started
findface-ntls	findface-cibr-findface-ntls-1	Запущен Started
nats	findface-cibr-nats-1	Запущен Started
postgresql	findface-cibr-postgresql-1	Запущен Started
memcached	findface-cibr-memcached-1	Запущен Started
findface-upload	findface-cibr-findface-upload-1	Запущен Started
etcd	findface-cibr-etcd-1	Запущен Started
findface-sf-api	findface-cibr-findface-sf-api-1	Запущен Started
findface-extraction-api	findface-cibr-findface-extraction-api-1	Запущен Started (CPU/GPU-ускорение).
findface-tarantool-server-shard-*	findface-cibr-findface-tarantool-server-shard-* 1	Запущен Started . Количество экземпляров (шардов) рассчитывается по формуле: $N = \min(\max(\min(\text{mem_mb} // 2000, \text{cpu_cores}), 1), 16 * \text{cpu_cores})$. Т. е. количество равно размеру оперативной памяти в Мб, разделенному на 2000, или количеству физических ядер процессора (но не менее 1 шарда), или же количеству физических ядер процессора, умноженному на 16, если первое полученное значение больше.
findface-video-manager	findface-cibr-findface-video-manager-1	Запущен Started

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

Совет: Обязательно сохраните эти данные: они вам понадобятся.

```
#####
#                               Installation is complete                               #
#####
- all configuration and data is stored in /opt/findface-cibr
- upload your license to http://192.168.112.254/#/license/
- user interface: http://192.168.112.254/
superuser:      admin
documentation:  http://192.168.112.254/doc/
2023/10/17 09:24:25 Installer finished
```

5. Выполните *процедуры после развертывания*.

Совет: Для установки той же конфигурации FindFace CIBR на другом сервере используйте автоматическое развертывание из файла с параметрами установки. В этом случае вам не придется снова отвечать на вопросы инсталлятора. Точный путь к файлу с параметрами установки отображается сразу после последнего вопроса инсталлятора, до начала активной фазы установки:

```
[I 2023-10-17 09:20:57,264 main:142] Your answers were saved to /tmp/findface-installer-
↳ t8qk_isw.json
```

Обязательно удалите поля `*.config`, `exp_ip` и `int_ip` перед установкой FindFace CIBR на сервере с другим IP-адресом.

Для запуска автоматического развертывания из файла `/tmp/<findface-installer-*>.json`, выполните следующую команду:

```
sudo ./<findface-*>.run -f /tmp/<findface-installer-*>.json
```

2.3.5 Полностью настраиваемая установка

Консольный инсталлятор FindFace CIBR предоставляет несколько вариантов установки, в том числе полностью настраиваемый вариант (установку отдельно выбранных пакетов). Данный вариант в основном используется при развертывании FindFace CIBR в сильно распределенной среде и требует определенного уровня знаний и опыта.

Для запуска полностью настраиваемой установки выполните следующие действия:

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

Примечание: Вместо `findface-*` укажите актуальное имя файла.

```
chmod +x findface-*.run
```

4. Запустите файл `.run`.

```
sudo ./findface-*.run
```

Инсталлятор задаст вам несколько вопросов, после чего проверит, соответствует ли сервер системным требованиям. Ответьте на вопросы надлежащим образом, после ответа каждый раз нажимая `Enter`. Вопросы и ответы следующие:

1. B: Which product should be installed? (Какой продукт следует установить?)

O: 1

```
Which product should be installed?

- 1 [cibr      ] FindFace CIBR
- 2 [video-worker] FindFace Video Worker

(default: cibr)
product> 1
```

2. B: Please choose installation type: (Пожалуйста, выберите тип установки:)

O: 4

```
Please choose installation type:

- 1 [stand-alone ] Single Server
- 2 [multi-worker] Single Server, Multiple video workers
- 3 [images      ] Don't configure or start anything, just load the images
  ↪and copy the models
- 4 [custom      ] Fully customized installation

(default: stand-alone)
type> 4
```

3. B: Directory to install into: (Директория установки:)

O: Укажите директорию установки FindFace CIBR. По умолчанию установка выполняется в `/opt/findface-cibr`. Нажмите `Enter` для подтверждения. В противном случае укажите нужную директорию и нажмите `Enter`.

```
Directory to install into:
(default: /opt/findface-cibr)
dest_dir>
```

4. B: Please enter path to docker-compose binary: (Пожалуйста, введите путь к бинарному файлу `docker-compose`:)

O: Укажите фактический путь к бинарному файлу `docker-compose`. По умолчанию предлагается `/usr/local/bin/docker-compose`, и это тот же путь, что и при установке `docker-compose` согласно инструкциям по подготовке сервера *Ubuntu / CentOS / Debian*. Нажмите `Enter` для подтверждения. В противном случае укажите другой путь и нажмите `Enter`.

```
Please enter path to docker-compose binary
(default: /usr/local/bin/docker-compose)
docker_compose>
```

5. B: Found X interface(s). Which one should we announce as our external address? (Было найдено X сетевых интерфейсов. Какой интерфейс следует использовать в качестве внешнего IP-адреса?)

O: Выберите интерфейс, который вы собираетесь использовать в качестве IP-адреса FindFace.

```
Found 2 interface(s). Which one should we announce as our external address?

- 1 [lo      ] 127.0.0.1
- 2 [ens3    ] 192.168.112.254

(default: 192.168.112.254)
ext_ip.advertised> 2
```

6. B: Found X interface(s). Which one should we announce as our inter-service communication address? (Было найдено X сетевых интерфейсов. Какой интерфейс следует использовать в качестве адреса для межсервисной коммуникации?)

O: Выберите интерфейс для межсервисного взаимодействия.

```
Found 2 interface(s). Which one should we announce as our inter-service
↪communication address?

- 1 [lo      ] 127.0.0.1
- 2 [ens3    ] 192.168.112.254

(default: 192.168.112.254)
inter_ip.advertised> 2
```

7. B: Please select FindFace CIBR components to install: (Пожалуйста, выберите компоненты FindFace CIBR для установки:)

O: Выберите компоненты FindFace CIBR для установки. По умолчанию установке подлежат все компоненты. Вы можете оставить все как есть, введя **done**, или выбрать определенные компоненты. 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 13** и т. д. Введите **done** для сохранения выбора и перехода к следующему шагу.

Предупреждение: В компоненте **pause** указан сетевой идентификатор (namespace) каждого компонента. Не рекомендуется исключать компонент **pause** из установки, так как оставшиеся компоненты останутся без сетевого идентификатора (namespace).

Если вы исключаете компонент **pause** намеренно, вам потребуется отредактировать файл `/opt/findface-cibr/docker-compose.yaml` и указать имя хоста для каждого сервиса в параметре `network_mode`.

```

Please select FindFace CIBR components to install:

- 1 [v]  findface-data          - Recognition models
...
...

Enter keyword to select matching choices or -keyword to clear selection.
Enter "done" to save your selection and proceed to another step.
components> done

```

8. Конкретные вопросы, связанные с выбранными компонентами: тип ускорения, необходимое количество экземпляров компонента, модели нейронных сетей и т. д. Если вы затрудняетесь с ответом, попробуйте найти его в этой документации или отправьте свой вопрос по адресу support@ntechlab.com.
9. В: Please set findface-cibr admin password (Пожалуйста, установите пароль для администратора FindFace CIBR)
 О: Задайте пароль для Супер Администратора (superuser).

```

Please set findface-cibr admin password
findface-multi-admin-password> admin

```

Инсталлятор извлечет образы FindFace CIBR из реестра Ntechlab и запустит ассоциированные сервисы в контейнерах Docker.

5. Выполните *процедуры после развертывания*.

Совет: Для установки той же конфигурации FindFace CIBR на другом сервере используйте автоматическое развертывание из файла с параметрами установки. В этом случае вам не придется снова отвечать на вопросы инсталлятора. Точный путь к файлу с параметрами установки отображается сразу после последнего вопроса инсталлятора, до начала активной фазы установки:

```

[I 2023-10-29 14:19:27,189 main:142] Your answers were saved to /tmp/findface-installer-
↳8upq3abq.json

```

Обязательно удалите поля *.config, exp_ip и int_ip перед установкой FindFace CIBR на сервере с другим IP-адресом.

Для запуска автоматического развертывания из файла /tmp/<findface-installer-*>.json, выполните следующую команду:

```

sudo ./<findface-*>.run -f /tmp/<findface-installer-*>.json

```

2.3.6 Установка моделей нейронных сетей

Для обнаружения и распознавания лиц и их атрибутов `findface-extraction-api` использует нейронные сети.

Если необходим ручной запуск установки моделей, используйте консольный инсталлятор следующим образом:

1. Запустите файл `findface-*.run`.

Примечание: Вместо `findface-*` укажите актуальное имя файла.

```
sudo ./findface-*.run
```

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

Вы можете найти установленные модели для распознавания лиц и их атрибутов в директории `/opt/findface-cibr/models/`. См. [Обзор моделей нейронных сетей](#).

2.3.7 Процедуры и первичные инструкции после развертывания

После завершения развертывания FindFace CIBR выполните описанные ниже процедуры.

В этом разделе:

- [Задание временной зоны](#)
- [Лицензирование экземпляра продукта](#)
- [Настройка логирования](#)
- [Полезные команды Docker](#)

Задание временной зоны

Временная зона определяет время в отчетах, логах, именах артефактов FindFace CIBR, таких как полные кадры, миниатюры событий и т. п.

Временная зона указывается в файле конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` в формате Регион/Страна/Город или Etc/GMT+Н. Лучший способ сделать это — скопировать и вставить часовой пояс из [таблицы](#) в Википедии.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# time zone
TIME_ZONE = 'America/Argentina/Buenos_Aires'
```

Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

Лицензирование экземпляра продукта

FindFace CIBR предоставляет несколько вариантов лицензирования. Какой бы вариант вы ни выбрали, вам потребуется загрузить файл лицензии FindFace CIBR через основной веб-интерфейс `http://<Host_IP_address>/#/license`. Для доступа к веб-интерфейсу используйте учетные данные `superuser`.

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.

Обратитесь к разделу *Лицензирование*, чтобы узнать о доступных вариантах лицензирования.

Настройка логирования

По умолчанию процессы FindFace CIBR регистрируются в *логах контейнеров Docker*, доступ к которым можно получить с помощью команд `docker logs` и `docker service logs`. Кроме этого, Docker использует драйвер перевода логов в *json-файлы*, который кэширует логи контейнеров в формате JSON. Вы можете настроить Docker на использование другого драйвера логов, выбрав из множества доступных механизмов логирования. См. *Логирование*, чтобы узнать, как это сделать.

Полезные команды Docker

Для быстрого и эффективного администрирования FindFace CIBR вы должны обладать обширными знаниями и навыками работы с Docker. Если вы новичок в Docker, начните с приведенных ниже команд. Затем изучите документацию Docker для получения дополнительных навыков.

- Просмотр всех контейнеров Docker, включая остановленные:

```
docker ps -a
```

Чтобы получить более компактный и понятный вывод, используйте следующую команду:

```
docker ps -a --format "table {{.ID}}\t{{.Names}}\t{{.Status}}\t{{.State}}"
```

Чтобы расширить предыдущий вывод, пригодится следующая команда:

```
docker ps --format='{{json .}}' | jq
```

- Перезапуск Docker:

```
sudo systemctl restart docker
```

- Просмотр логов контейнера, если *включен* драйвер перевода логов в journald:

```
journalctl CONTAINER_NAME=findface-cibr-findface-multi-legacy-1 -f
```

- Остановка контейнера Docker:

```
sudo docker container stop <container_name>/<container_id>
```

Остановка всех контейнеров Docker:

```
sudo docker container stop $(sudo docker ps -a -q)
```

- Запуск контейнера Docker:

```
sudo docker container start <container_name>/<container_id>
```

Запуск всех контейнеров Docker:

```
sudo docker container start $(sudo docker ps -a -q)
```

- Просмотр файла конфигурации docker-compose.yaml для FindFace CIBR:

```
cat /opt/findface-cibr/docker-compose.yaml
```

- Файлы конфигурации FindFace CIBR можно найти здесь:

```
cd /opt/findface-cibr/configs/
```

После внесения изменений в файл конфигурации перезапустите соответствующий контейнер, выполнив команду:

```
sudo docker container restart <container_name>/<container_id>
```

- Вход в работающий контейнер Docker для выполнения в нем команды:

```
sudo docker container exec -it <container_name> /bin/bash
```

- Остановка и удаление всех контейнеров FindFace CIBR:

```
cd /opt/findface-cibr/  
sudo docker-compose down
```

- Сборка, воссоздание и запуск контейнеров FindFace CIBR:

```
cd /opt/findface-cibr/  
sudo docker-compose up -d
```

2.3.8 Удаление экземпляра продукта

Если вы выбрали *автоматическое односерверное развертывание* FindFace CIBR, то для удаления экземпляра продукта воспользуйтесь скриптом `uninstall.sh`, встроенным в файл инсталлятора.

Важно: Обязательно создайте *резервную копию* вашего экземпляра перед его удалением, если вы планируете позже *восстановить* FindFace CIBR и его данные.

Запустите скрипт `uninstall.sh` из папки `/opt/findface-cibr/`:

```
cd /opt/findface-cibr/
sudo /opt/findface-cibr/uninstall.sh /opt/findface-cibr/
```

Вам будет задан вопрос:

В: This script will remove all docker containers, images, volumes, config files and directory `"/opt/findface-cibr/"`. Do you want to proceed(y/n)?: (Этот скрипт удалит все контейнеры Docker, образы, тома, файлы конфигурации и директорию `/opt/findface-cibr/`. Хотите продолжить (да/нет)?)

О: Ответьте у, чтобы удалить экземпляр продукта.

В случае если развертывание FindFace CIBR было нестандартным, и вы не можете воспользоваться скриптом для удаления экземпляра продукта, удалите все компоненты вручную. Следующая команда повторяет поведение скрипта `uninstall.sh`:

```
sudo docker-compose -f /opt/findface-cibr/docker-compose.yaml down -v --rmi all
sudo rm -rf /opt/findface-cibr/
```

2.4 Администрирование и конфигурирование

2.4.1 Лицензирование

В этой главе:

- Принцип лицензирования
- Просмотр и обновление лицензии
- Лицензирование через USB-ключ аппаратной защиты
- Лицензирование по слепку с оборудования

Принцип лицензирования

1. Общее количество извлеченных векторов признаков.

Примечание: Векторы признаков извлекаются из обнаруженных на видео объектов, из фотографий в карточках и фотографий пользователей, при построении т. н. центроидов в процессе формирования кластеров.

Схема лицензирования выглядит следующим образом:

- События: 1 событие распознавания объекта на видео = 1 объект в лицензии.
 - Карточка: 1 фотография в карточке = 1 объект в лицензии.
 - Кластеры: 1 человек = 1 объект в лицензии.
 - Пользователи: 1 фотография пользователя = 1 объект в лицензии.
2. Количество источников видеоизображения, используемых в данный момент времени (т. е. количество активных на данный момент задач на обработку видео).
 3. Количество экземпляров моделей, используемых **findface-extraction-api**.
 4. Распознавание атрибутов лица: пол/возраст/эмоции/очки/борода/медицинская маска/др.

Вы можете выбрать между следующими методами лицензирования:

- Онлайн-лицензирование выполняется через Глобальный менеджер лицензий NtechLab **license.ntechlab.com** и требует наличия стабильного интернет-соединения, DNS и открытого порта 443 TCP. После отключения от интернета система продолжит работать в автономном режиме в течение четырех часов.

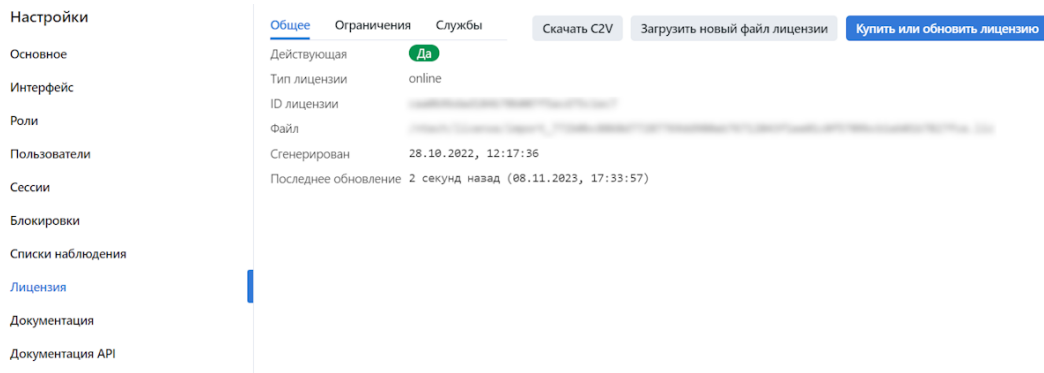
Примечание: Уведомите своего менеджера, если требуется продлить данный период (до 2-х дней).

- Для лицензирования в закрытой сети через USB-ключ необходимо наличие USB-порта на физическом сервере с компонентом **findface-ntls** (сервер лицензирования в составе *ядра FindFace*).
- Для лицензирования в закрытой сети по слепку с оборудования необходимо наличие установленных драйверов Sentinel на физическом сервере с компонентом **findface-ntls**.

Важно: Для обеспечения функционирования системы достаточно одного экземпляра **findface-ntls**. Если ваша система нуждается в большем количестве серверов лицензирования, заблаговременно сообщите об этом своему менеджеру NtechLab, чтобы предотвратить блокировку системы.

Просмотр и обновление лицензии

После установки FindFace CIBR загрузите в систему полученный от менеджера файл лицензии. Для этого перейдите в *Настройки -> Лицензия*.



Используйте ту же вкладку для просмотра текущей информации по лицензии и обновления лицензии.

Лицензирование через USB-ключ аппаратной защиты

Для того чтобы реализовать лицензирование через USB-ключ, выполните следующие действия:

1. Сообщите своему менеджеру о том, что намерены использовать данный метод лицензирования, и запросите свой USB-ключ и файл лицензии.
2. Откройте файл конфигурации `/opt/findface-cibr/docker-compose.yaml`.

```
sudo vi /opt/findface-cibr/docker-compose.yaml
```

3. Добавьте строку `privileged: true`. Смонтируйте каталог `/dev` в контейнер `findface-cibr-findface-ntls-1`, указав его в томах (volumes) секции `findface-ntls`. В результате секция будет выглядеть следующим образом:

```
findface-ntls:
  command: [--config=/etc/findface-ntls.cfg]
  image: docker.int.ntl/ntech/universe/ntls:ffserver-9.230407.1
  network_mode: service:pause
  privileged: true
  restart: always
  user: root
  volumes: ['./configs/findface-ntls/findface-ntls.yaml:/etc/findface-ntls.cfg:ro',
    './data/findface-ntls/ntech/license', '/dev:/dev']
```

4. Создайте новое правило `udev`.
 1. Загрузите файл `95-grdnt.rules` в любой каталог, например, в `/home/username/tmp/`.
 2. Скопируйте файл `95-grdnt.rules` в каталог `/etc/udev/rules.d/`.

```
sudo cp /home/username/tmp/95-grdnt.rules /etc/udev/rules.d/
```

5. Выполните пересборку контейнеров FindFace CIBR.

```
cd /opt/findface-cibr

sudo docker-compose down
```

(continues on next page)

(продолжение с предыдущей страницы)

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

6. Вставьте USB-ключ в USB-порт.
7. Загрузите файл лицензии на вкладке *Лицензия*.

Лицензирование по слепку с оборудования

Примечание: Sentinel — вид офлайн лицензий, которые, в отличие от guardant лицензий, не требуют физического носителя для своей работы.

Словарь:

- Sentinel — система защиты ПО и лицензирования от [Thales](#). Она позволяет реализовать офлайн-новое лицензирование без доступа к глобальному серверу.
- Файл C2V — это файл со слепком с оборудования клиентской машины, который служит для привязки лицензии только в этой машине. Этот файл генерируется библиотекой Sentinel у клиента на той машине, где впоследствии будет устанавливаться лицензионный ключ.

Для того чтобы реализовать лицензирование по слепку, выполните следующие действия:

1. Сообщите своему менеджеру о том, что намерены использовать данный метод лицензирования, и запросите свой уникальный идентификатор лицензии. Менеджер также предоставит вам пакет *findface-sentinel-lib_*.deb*, необходимый для интеграции FindFace CIBR с Sentinel.
2. Установите драйверы Sentinel на физическом сервере с компонентом *findface-ntls*.

Выполните следующие действия:

1. Загрузите [драйверы Sentinel](#) с официального сайта.
2. Разархивируйте скачанный архив и перейдите в него.

```
tar -xvzf Sentinel_LDK_Linux_Runtime_Installer_script.tar.gz
cd Sentinel_LDK_Linux_Runtime_Installer_script/
```

3. Внутри архива есть еще архив *aksusbd-*.tar.gz*. Разархивируйте его и перейдите в полученную директорию.

```
tar -xvzf aksusbd-*.tar.gz
cd aksusbd-*/
```

4. Выполните команду установки.

```
sudo ./dinst
```

5. Запустите и проверьте статусы сервисов Sentinel.

```
sudo systemctl start aksusbd.service hasplmd.service
sudo systemctl status aksusbd.service hasplmd.service
```

3. Смонтируйте каталоги */var/hasplm* и */etc/hasplm* в контейнер *findface-cibr-findface-ntls-1*. Для этого откройте файл конфигурации */opt/findface-cibr/docker-compose.yaml* и перечислите их в томах (volumes) секции *findface-ntls*.

```
sudo vi /opt/findface-cibr/docker-compose.yaml

findface-ntls:
  ...
  volumes: ['./configs/findface-ntls/findface-ntls.yaml:/etc/findface-ntls.cfg:ro',
    ↪ './data/findface-ntls:/ntech/license', '/var/hasplm:/var/hasplm', '/etc/hasplm:/
    ↪ etc/hasplm']
```

4. Выполните пересборку контейнеров FindFace CIBR.

```
cd /opt/findface-cibr

sudo docker-compose down

sudo docker-compose up -d
```

5. Поместите пакет `findface-sentinel-lib_*.deb`, полученный от своего менеджера, в любой каталог на этом же сервере. Установите пакет.

```
sudo dpkg -i /path/to/findface-sentinel-lib_*.deb
```

6. В веб-интерфейсе FindFace CIBR перейдите на вкладку *Настройки* -> *Лицензия*. Снимите слепок с оборудования (файл C2V), нажав на кнопку *Скачать C2V для активации*.

Совет: Если вы предпочитаете работать с консолью, можно вместо этого отправить следующий API-запрос в `findface-ntls`:

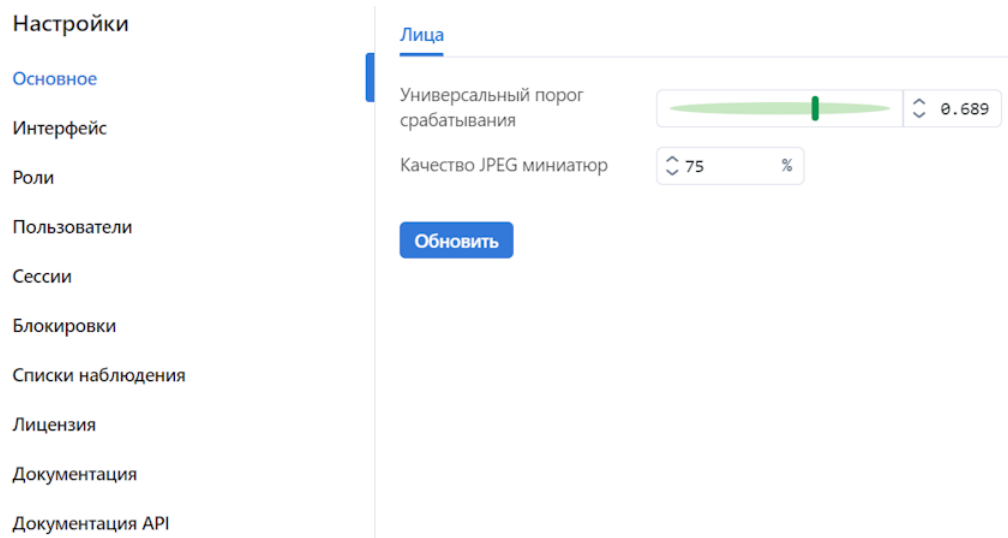
```
curl <findface-ntls-server-ip>:3185/c2v >my_pc.c2v
```

7. Отправьте идентификатор лицензии и файл C2V своему менеджеру и получите в ответном письме файл лицензии.
8. Загрузите файл лицензии на вкладке *Лицензия*.

2.4.2 Основные настройки

Основные настройки FindFace CIBR, такие как универсальный порог срабатывания для распознавания лиц и качество JPEG миниатюр, определяют работу системы и потребление ресурсов.

Для изменения основных настроек перейдите *Настройки* → *Основное*. После внесения изменений нажмите *Обновить*. Подробное описание каждой настройки приведено ниже.



В этом разделе:

- *Универсальный порог срабатывания*
- *Качество JPEG миниатюр*

Универсальный порог срабатывания

FindFace CIBR принимает решение о совпадении лиц на основании предустановленного порога срабатывания. По умолчанию установлено оптимальное пороговое значение. При необходимости вы можете изменить его.

Примечание: Чем выше порог, 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 каждого *списка наблюдения*.

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

Качество JPEG миниатюр

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

2.4.3 Управление пользователями и безопасность системы

Важно: Хотя FindFace CIBR предоставляет инструменты для защиты от несанкционированного доступа, они не заменяют правильно настроенный межсетевой экран. Обязательно используйте брандмауэр, чтобы усилить защиту сети FindFace CIBR.

Управление ролями и пользователями

В этой главе:

- *Предустановленные роли*
- *Создание новой роли в веб-интерфейсе*
- *Основная и дополнительная роль пользователя*
- *Создание пользовательского аккаунта вручную*
- *Работа с ролями и пользователями в консоли*
- *Деактивация или удаление пользователей*

Предустановленные роли

Для работы с CIBR предусмотрены следующие предустановленные роли:

- Администратор: предоставляется полный доступ к функциональности FindFace CIBR, интеграционным и административным инструментам.

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

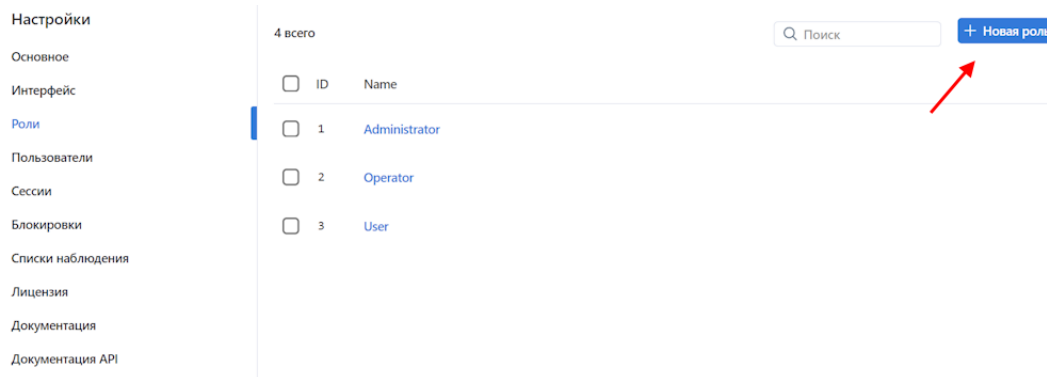
- Оператор: предоставляется полный доступ к функциональности FindFace CIBR.
- Пользователь: обладает правами на управление происшествиями, а также на изменение данных в своем профиле. Остальные функции доступны в режиме чтения.

Вы можете изменить привилегии ролей оператора и пользователя, а также создать новые роли, но не можете изменить привилегии ролей для администратора.

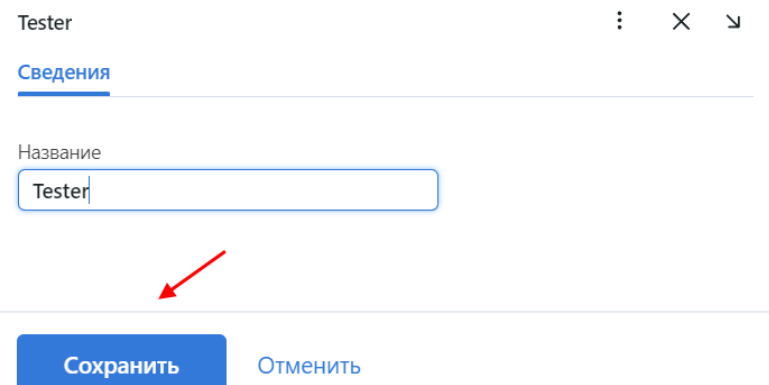
Создание новой роли в веб-интерфейсе

Для создания новой роли в веб-интерфейсе выполните следующие действия:

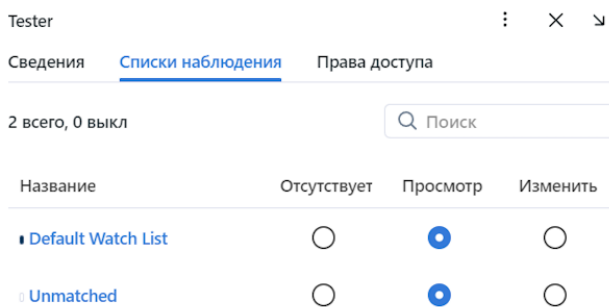
1. Перейдите *Настройки* -> *Роли*.
2. Нажмите *+ Новая роль*.



3. На вкладке *Сведения* задайте название роли. Сохраните роль.



4. После сохранения роли вы увидите, что рядом с вкладкой *Сведения* появились следующие вкладки:



- *Списки наблюдения*: права роли на определенные списки наблюдения
- *Права доступа*: привилегии роли на работу с системными функциями и сущностями

Задайте права роли в соответствии со своими потребностями. Обратите внимание на разницу между правами на определенный список наблюдения и правами на системную сущность `watchlist`. Например, если вы установите `Отсутствует` для определенного списка наблюдения на вкладке *Списки наблюдения*, пользователи с соответствующей ролью не смогут работать с данным списком наблюдения. Снятие всех флажков для сущности `watchlist` на вкладке *Права доступа* не позволит пользователям просматривать и работать со всеми списками наблюдения.

Полный список сущностей FindFace CIBR, которые используются в текущей версии:

- `all_own_sessions`: все *сессии* текущего пользователя на разных устройствах

Примечание: Если для данной сущности установлены соответствующие разрешения, пользователи смогут просматривать (`Просмотр`) и закрывать (`Удалить`) все свои сессии на разных устройствах. В противном случае пользователям будет разрешено только просматривать и закрывать свою сессию на текущем устройстве. Работа с сессиями производится на вкладке *Сессии (Настройки)*.

- `case`: происшествие
- `dailysearchevent`: ежедневный поиск
- `deviceblacklistrecord`: *список блокировок*
- `faceobject`: *фотография лица в карточке*
- `group`: *роли*
- `humancard`: *карточка человека*
- `remotemonitoringrecord`: *удаленный мониторинг*
- `report`: *отчеты*
- `searchrequest`: удаленный поиск
- `upload`: элемент (фото) в пакетной загрузке
- `uploadlist`: список фотографий в пакетной загрузке
- `user`: *пользователь*
- `videoarchive`: идентификация объектов в видеофайлах
- `watchlist`: *список наблюдения*

Вы также можете включать и отключать права на следующую функциональность:

- `batchupload_cards`: *пакетная загрузка карточек*
- `change_runtime_setting`: изменение *основных настроек* FindFace CIBR
- `configure_ntls`: конфигурация *сервера лицензий findface-ntls*
- `view_auditlog`: просмотр и работа с *аудит логами*

5. Сохраните изменения.

Основная и дополнительная роль пользователя

Вы можете назначить пользователю привилегии, используя следующие роли:

- *Основная роль*: основная роль пользователя, обязательная для назначения. Пользователю можно назначить только одну основную роль.
- *Дополнительная пользовательская роль*, необязательная для назначения. Одному пользователю можно назначить несколько ролей. Связанные с ними права будут добавлены к правам, предоставляемым основной ролью.

Все пользователи, принадлежащие к определенной основной роли, автоматически получают доступ к видео архивам внутри группы и спискам наблюдения (и карточкам в этих списках), созданным пользователем с той же основной ролью, с учетом привилегий, определенных их дополнительными ролями.

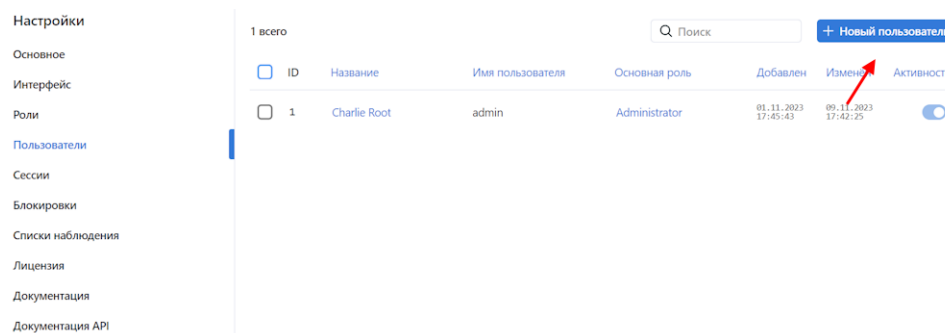
См.также:

Создание пользовательского аккаунта вручную

Создание пользовательского аккаунта вручную

Для создания пользовательского аккаунта вручную выполните следующие действия:

1. Перейдите *Настройки* -> *Пользователи*.
2. Нажмите *+ Новый пользователь*.



3. На вкладке *Сведения* введите данные пользователя, такие как название, имя пользователя и пароль. При необходимости добавьте комментарий.

Примечание: При установке пароля учитывайте требования к паролю:

- содержит не менее 8 символов
- содержит не только цифры
- не входит в список 20000 распространенных паролей
- не похож на другие атрибуты пользователя
- может содержать только буквы латинского алфавита, цифры и специальные символы

4. Из раскрывающегося списка *Роли* выберите одну или несколько пользовательских ролей. Назначьте одну из них основной.

5. На вкладке *Фотографии* прикрепите фотографию пользователя.
6. Сохраните аккаунт пользователя.

Marsha Johnes
Marsha Johnes

Изменено X
:
X
↵

Сведения
Фотографии

Название

Marsha Johnes

Имя пользователя

Marsha Johnes

Сменить пароль

Комментарий

Роли	Основная роль
User	<input type="radio"/> <div></div>
Tester	<input checked="" type="radio"/> <div></div>
<div>Добавить роль</div>	

ID 2

Создано 09.11.2023 17:56:31

Сохранить
Отменить

Работа с ролями и пользователями в консоли

В случае если *предустановленные роли* были удалены из системы, используйте следующую команду, для того чтобы создать их:

```
sudo docker exec -it findface-cibr-findface-multi-identity-provider-1 /opt/findface-
security/bin/python3 /tigre_prototype/manage.py create_groups
```

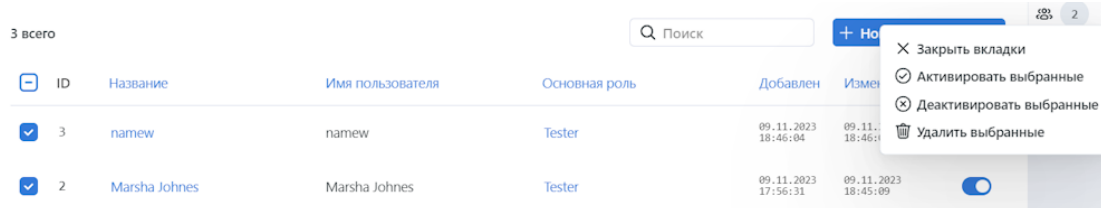
Для создания пользователя с правами Супер Администратора (*superuser*) выполните следующую команду (обратите внимание, что *password* является обязательным аргументом):

```
sudo docker exec -it findface-cibr-findface-multi-identity-provider-1 /opt/findface-
security/bin/python3 /tigre_prototype/manage.py create_default_user --password
<password>
```

Деактивация или удаление пользователей

Для того чтобы деактивировать пользователя, переведите ползунок *Активность* в неактивное положение в списке пользователей (*Настройки -> Пользователи*).

Если вы собираетесь деактивировать нескольких пользователей, выберите их в списке и нажмите *Деактивировать выбранные*.



Для того чтобы удалить пользователей из FindFace CIBR, выберите их в списке и нажмите *Удалить выбранные*.

Настройка шифрования данных

Для обеспечения безопасности данных включите SSL-шифрование. Выполните следующие действия:

1. На хосте создайте директорию с конфигурацией nginx, включающую каталог для хранения информации о SSL-шифровании:

```
sudo mkdir -p /etc/nginx/ssl/
```

2. Создайте ключ и сертификат SSL. Если будет использоваться самоподписанный сертификат, выполните следующую команду:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/nginx/ssl/my-  
example-domain.key -out /etc/nginx/ssl/my-example-domain.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.key` и сертификата `my-example-domain.crt` будут сохранены в каталоге `/etc/nginx/ssl/`.

3. Если используется подтвержденный сертификат, то нужно добавить путь сертификата в **volumes** для сервиса `findface-video-worker`, добавить установку `ca-certificates` и обновить корневое хранилище сертификатов в контейнере сервиса.

1. Откройте файл `/opt/findface-cibr/docker-compose.yaml`:

```
sudo vi /opt/findface-cibr/docker-compose.yaml
```

2. Найдите раздел `findface-video-worker` и отредактируйте его, чтобы он выглядел так, как показано ниже.

Для CPU:

```
findface-video-worker:  
  entrypoint: ["sh", "-c", "apt-get update && DEBIAN_FRONTEND=noninteractive  
  apt-get install --no-install-recommends --yes ca-certificates && update-ca-  
  certificates && exec /tini -- /findface-video-worker-cpu --config=/etc/
```

(continues on next page)

(продолжение с предыдущей страницы)

```

↪findface-video-worker.yaml"]
  depends_on: [findface-video-manager, findface-ntls, mongoddb]
  image: docker.int.ntl/ntech/universe/video-worker-cpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  volumes: ['./configs/findface-video-worker/findface-video-worker.yaml:/etc/
↪findface-video-worker.yaml:ro',
    './models:/usr/share/findface-data/models:ro', './cache/findface-video-
↪worker/models:/var/cache/findface/models_cache',
    './cache/findface-video-worker/recorder:/var/cache/findface/video-worker-
↪recorder',
    '/etc/nginx/ssl/my-example-domain.crt:/usr/local/share/ca-certificates/my-
↪example-domain.crt:ro']

```

Для GPU достаточно будет добавить путь до сертификата и обновить корневое хранилище:

```

findface-video-worker:
  entrypoint: ["sh", "-c", "update-ca-certificates && exec /tini -- /findface-
↪video-worker-gpu --config=/etc/findface-video-worker.yaml"]
  depends_on: [findface-video-manager, findface-ntls, mongoddb]
  environment: [CUDA_VISIBLE_DEVICES=0]
  image: docker.int.ntl/ntech/universe/video-worker-gpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  runtime: nvidia
  volumes: ['./configs/findface-video-worker/findface-video-worker.yaml:/etc/
↪findface-video-worker.yaml:ro',
    './models:/usr/share/findface-data/models:ro', './cache/findface-video-
↪worker/models:/var/cache/findface/models_cache',
    './cache/findface-video-worker/recorder:/var/cache/findface/video-worker-
↪recorder',
    '/etc/nginx/ssl/my-example-domain.crt:/usr/local/share/ca-certificates/my-
↪example-domain.crt:ro']

```

Важно: Для CPU версии настройка требует доступ к интернету. Если он отсутствует, обратитесь к нашим техническим специалистам (support@ntechlab.com).

Предупреждение: Для CPU версии вырастет время запуска контейнера findface-video-worker на ~15 секунд.

3. Выполните пересборку контейнеров FindFace CIBR.

```

cd /opt/findface-cibr/
docker-compose down
docker-compose up -d

```

4. Настройте nginx для использования SSL. Откройте файл конфигурации nginx /opt/findface-cibr/configs/findface-multi-ui/nginx-site.conf. Внесите в файл следующие из-

менения:

1. Добавьте новый раздел `server {...}`, содержащий правило замены URL. В строке `rewrite ^(.*) https://...` замените `ip_address_server_ffcibr` IP-адресом сервера, на котором установлен экземпляр FindFace CIBR.

```
server {  
    listen 80;  
    server_name my-example-domain.com www.my-example-domain.com;  
    rewrite ^(.*) https://ip_address_server_ffcibr$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/my-example-domain.com.crt;  
ssl_certificate_key  /etc/nginx/ssl/my-example-domain.com.key;
```

Пример файла конфигурации `/opt/findface-cibr/configs/findface-multi-ui/nginx-site.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;  
}  
  
upstream ffsecurity-django {  
    server 127.0.0.1:8004;  
}  
  
upstream audit {  
    server 127.0.0.1:8012;  
}  
  
upstream identity-provider {  
    server 127.0.0.1:8022;  
}  
  
map $http_upgrade $ffsec_upstream {  
    default "http://ffsecurity-ws";  
    "" "http://ffsecurity";  
}
```

(continues on next page)

(продолжение с предыдущей страницы)

```

server {
    listen 80;
    server_name my-example-domain.com www.my-example-domain.com;
    rewrite ^(.*) https://my-example-domain.com$1 permanent;
    access_log off;
}

server {
    # listen 80 default_server;
    # listen [::]:80 default_server;

    listen 443 ssl;
    ssl_certificate /etc/nginx/ssl/my-example-domain.com.crt;
    ssl_certificate_key /etc/nginx/ssl/my-example-domain.com.key;

    root /var/lib/findface-security;

    autoindex off;

    server_name _;

    location = / {
        alias /usr/share/findface-security-ui/;
        try_files /index.html =404;
    }
    location /static/ {
    }
    location /uploads/ {
        # internal; # uncomment if you intend to enable OVERPROTECT_
        ↪MEDIA
        add_header 'Access-Control-Allow-Origin' '*';
        add_header 'Access-Control-Allow-Methods' '*';
        add_header 'Access-Control-Allow-Headers' '*';
        ↪Content-Range';
        add_header 'Access-Control-Expose-Headers' 'Content-Length,
        ↪Content-Range';
        add_header 'Access-Control-Max-Age' 2592000;

        location ~ /card/(?<card_type>[a-zA-Z]+)/(?<card_id>[0-9]+)/
        ↪attachments/(.*)$ {
            add_header 'Access-Control-Allow-Origin' '*';
            add_header 'Access-Control-Allow-Methods' '*';
            add_header 'Access-Control-Allow-Headers' '*';
            ↪Length,Content-Range';
            add_header 'Access-Control-Expose-Headers' 'Content-
            ↪Length,Content-Range';
            add_header 'Access-Control-Max-Age' 2592000;
            add_header 'Content-Disposition' 'attachment';
            add_header 'Content-Security-Policy' 'sandbox';
        }
    }
    location /ui-static/ {
        alias /usr/share/findface-security-ui/ui-static/;
    }
}

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    }
    location /doc/ {
        alias /opt/findface-security/doc/;
    }
    location /api-docs {
        alias /opt/findface-security/rapidoc;
        index index.html;
    }
    location /api-docs/ {
        alias /opt/findface-security/rapidoc/;
        try_files $uri index.html =404;
    }
    location ~ /videos/(?<video_id>[0-9]+)/upload/(.*)$ {
        client_max_body_size 15g;

        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 @django {
        internal;
        client_max_body_size 1g;
        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_read_timeout 5m;
        proxy_pass http://ffsecurity-django;
    }

# location /v1/video-liveness {
#     add_header Access-Control-Allow-Headers "*" always;
#     add_header Access-Control-Allow-Methods "*" always;
#     add_header Access-Control-Allow-Origin "*" always;
#
#     if ($request_method = 'OPTIONS') {
#         return 204;
#     }
#
#     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_pass http://127.0.0.1:18301;
#     proxy_read_timeout 5m;
# }

    location / {
        client_max_body_size 1g;
        proxy_set_header Host $http_host;
        proxy_set_header X-Forwarded-For $remote_addr;

```

(continues on next page)

(продолжение с предыдущей страницы)

```

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|vms|external-vms).*/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 $1$is_args$args;
}

location /audit-logs {
    proxy_pass http://audit;
}

location ~ ^/(auth|ad_
↪groups|cproauth|groups|permissions|sessions|users|user-face|device-blacklist-
↪records) {
    proxy_pass http://identity-provider;
}

# location /users/me/ad {
#
#     proxy_pass <FFmulti_address>/auth/ad_login/; e.g http://127.0.
↪0.1/auth/ad_login/;
#     proxy_method POST;
#
#     proxy_set_header    X-Real-IP $remote_addr;
#     proxy_set_header    Host $http_host;
#     proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
#     proxy_set_header    Authorization $http_authorization;
#     proxy_pass_header    Authorization;
#     proxy_no_cache 1;
#     proxy_cache_bypass 1;
#
#     auth_gss on;
#     auth_gss_realm <REALM>; # e.g. TESTNTL.LOCAL;
#     auth_gss_keytab <path/to/file.keytab>; # e.g. /var/lib/web.
↪keytab
#     auth_gss_service_name <service_name>; # e.g. HTTP/web.testntl.
↪local;
#     auth_gss_allow_basic_fallback on;
#     }

```

(continues on next page)

(продолжение с предыдущей страницы)

}

4. Скопируйте общий файл конфигурации nginx `nginx.conf` из контейнера `findface-cibr-findface-multi-ui-1` в директорию `/etc/nginx/`:

```
sudo docker cp findface-cibr-findface-multi-ui-1:/etc/nginx/nginx.conf /etc/
↪ nginx/nginx.conf
```

5. 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;
```

5. В файле `/opt/findface-cibr/docker-compose.yaml` добавьте (смонтируйте) каталог, хранящий данные SSL-шифрования `/etc/nginx/ssl/` и файл конфигурации `/etc/nginx/nginx.conf` системы-хоста в контейнер `findface-cibr-findface-multi-ui-1`:

1. Откройте файл `/opt/findface-cibr/docker-compose.yaml`:

```
sudo vi /opt/findface-cibr/docker-compose.yaml
```

2. Найдите раздел `findface-multi-ui` и отредактируйте его, чтобы он выглядел так, как показано в примере ниже:

```
findface-multi-ui:
  depends_on: [findface-multi-legacy]
  image: docker.int.ntl/ntech/multi/multi/ui-cibr:ffcibr-2.1.2
  network_mode: service:pause
  restart: always
  volumes: [ './configs/findface-multi-ui/nginx-site.conf:/etc/nginx/conf.d/
↪ default.conf:ro',
            './data/findface-multi-legacy/uploads:/var/lib/findface-security/uploads',
            '/etc/nginx/ssl:/etc/nginx/ssl',
            '/etc/nginx/nginx.conf:/etc/nginx/nginx.conf:ro' ]
```

6. Отредактируйте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`.

1. В параметрах `ROUTER_URL` и `IMAGE_CROP_URL` замените префикс `http://` на `https://`.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.
↪ py

...
'ROUTER_URL': 'https://127.0.0.1',
'IMAGE_CROP_URL': 'https://127.0.0.1',
...
```

2. При использовании подтвержденного сертификата необходимо в `ROUTER_URL` указать домен, по которому был создан сертификат:

```
'ROUTER_URL': 'https://my-example-domain.com'
```

3. Добавьте адрес `https://my-example-domain.com` в параметр `EXTERNAL_ADDRESS`:


```
...
EXTERNAL_ADDRESS = 'https://my-example-domain.com'
...
```

7. На сервере с установленным FindFace CIBR откройте файл `/etc/hosts` и добавьте следующую строку:

```
sudo vi /etc/hosts

...
127.0.0.1 my-example-domain.com
```

8. В системе, где вы используете браузер для взаимодействия с FindFace CIBR, откройте файл `hosts`. Добавьте IP-адрес сервера, на котором установлен FindFace CIBR, вместо `ip_address_server_ffcibr`. Замените `my-example-domain.com` адресом вашего домена — так же, как вы делали в предыдущих шагах.

1. Для операционной системы Linux выполните следующее:

```
sudo vi /etc/hosts

...
*ip_address_server_ffcibr* my-example-domain.com
```

2. Если вы используете операционную систему Windows, откройте файл `C:\Windows\System32\drivers\etc\hosts` от имени администратора. Добавьте следующую строку в файл `hosts`:

```
*ip_address_server_ffcibr* my-example-domain.com
```

9. Перезапустите контейнеры:

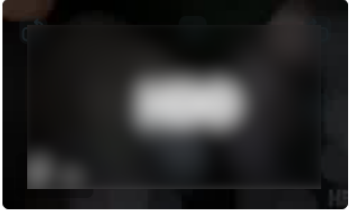
```
cd /opt/findface-cibr/
sudo docker-compose down
sudo docker-compose up -d
```

10. При использовании самоподписанного сертификата для загруженных видео нужно отключать проверку SSL-сертификата:

1. Перейдите на вкладку *Видео* внутри происшествия.
2. Нажмите на загруженное видео.
3. На вкладке *Дополнительные* снимите флажок *Проверить SSL-сертификат*.


Сведения Основные Дополнительные Зоны Лица

Трансформация



☐ Отразить по вертикали
☐ Отразить по горизонтали
☐ Повернуть по часовой стрелке
☐ Повернуть против часовой стрелки

Отправка лиц

Таймаут  15000 Мс

☒ Проверить SSL-сертификат

Список сессий пользователей. Блокировки

В этой главе:

- *Предоставление разрешений для работы с сессиями*
- *Просмотр сессий пользователей*
- *Блокировка устройства*

FindFace CIBR позволяет увидеть текущие пользовательские сессии и узнать связанные с ними данные, такие как UUID подключенного устройства, тип пользовательского интерфейса, IP-адрес, время последнего соединения и т. д.

При необходимости вы можете добавить устройство в список заблокированных, при этом учетная запись пользователя останется активной. Блокировка устройства может пригодиться в различных ситуациях, например, если вы хотите, чтобы пользователи работали с системой только со своих рабочих мест. Данные меры помогут вам поднять безопасность системы на новый уровень.

Предоставление разрешений для работы с сессиями

Доступ пользователей к списку сессий зависит от предоставленных *прав доступа*:

- Администратор: может просматривать и закрывать сессии всех пользователей
- Пользователь с правами доступа `all_own_sessions`: может просматривать/закрывать все сессии со своего логина
- Пользователь без прав доступа `all_own_sessions`: может просмотреть/закрыть только свою текущую сессию

Просмотр сессий пользователей

Для просмотра списка пользователей перейдите *Настройки -> Сессии*.

Настройки

- Основное
- Интерфейс
- Роли
- Пользователи
- Сессии**
- Блокировки
- Списки наблюдения
- Лицензия
- Документация
- Документация API

Сессии | Блокировки

Пользователь | Устройство | Статус

UUID | Пользователь | Устройство | IP-адрес | Статус | Последнее соединение

Толчок: { "user_agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.0.0 Safari/537.36" }

UUID	Пользователь	Устройство	IP-адрес	Статус	Последнее соединение
12345678-9012-3456-7890-123456789012	admin	📱	172.26.76.6	Онлайн	13.11.2023 11:28:36
23456789-0123-4567-8901-234567890123	admin	📱	172.26.127.160	Офлайн	11.11.2023 13:27:58
34567890-1234-5678-9012-345678901234	admin	📱	172.26.127.160	Офлайн	10.11.2023 17:36:22
45678901-2345-6789-0123-456789012345	admin	📱	172.26.126.86	Офлайн	10.11.2023 10:39:34

Каждая запись сессии содержит следующие данные:

- UUID устройства
- логин
- тип пользовательского интерфейса (мобильный/веб)
- информация об устройстве
- IP-адрес
- статус (онлайн, офлайн, заблокировано)
- время последнего соединения

Используйте панель фильтров сверху, чтобы задать условия поиска в списке сессий.

Для закрытия сессии выберите ее в списке и нажмите *Завершить*.

Настройки

- Основное
- Интерфейс
- Роли
- Пользователи
- Сессии**
- Блокировки
- Списки наблюдения
- Лицензия
- Документация
- Документация API

Сессии | Блокировки

Пользователь | Устройство | Статус

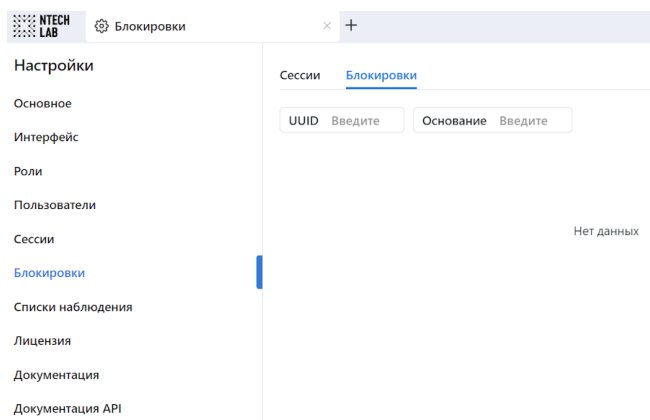
☑ Блокировать | ✕ **Завершить**

☑ UUID | Пользователь | Устройство | IP-адрес | Статус | Последнее соединение

☑ UUID	Пользователь	Устройство	IP-адрес	Статус	Последнее соединение
12345678-9012-3456-7890-123456789012	admin	📱	172.26.76.6	Онлайн	13.11.2023 11:28:36
23456789-0123-4567-8901-234567890123	admin	📱	172.26.127.160	Офлайн	11.11.2023 13:27:58
34567890-1234-5678-9012-345678901234	admin	📱	172.26.127.160	Офлайн	10.11.2023 17:36:22
45678901-2345-6789-0123-456789012345	admin	📱	172.26.126.86	Офлайн	10.11.2023 10:39:34
56789012-3456-7890-1234-567890123456	admin	📱	172.26.25.127	Офлайн	09.11.2023 21:00:33

Блокировка устройства

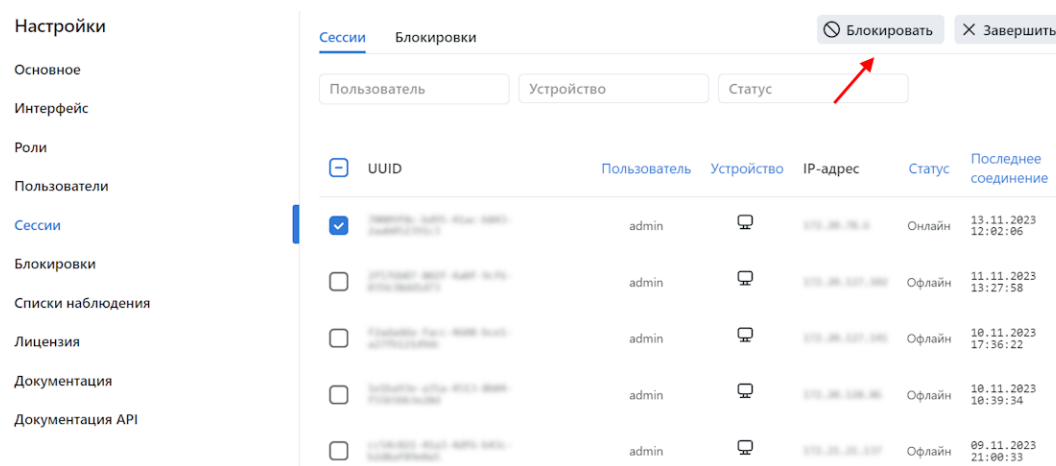
Список заблокированных устройств доступен на вкладке *Блокировки*.



Вы можете заблокировать устройство на вкладке *Сессии*. Блокировка устройства приводит к автоматическому выводу пользователя из системы.

Для блокировки устройства выполните следующие действия:

1. Выберите соответствующие записи сессий.
2. Нажмите *Блокировать*.



3. Укажите основание для блокировки устройства (обязательно) и дату окончания срока действия блокировки (опционально). Если дата не указана, блокировка будет постоянной.
4. Нажмите *Сохранить*.

Создание блокировки

UUID

10000000-0000-414e-8000-000000000000

Основание

Основание

Истекает

15.06.2036 12:00

Отмена

Сохранить

Разрешенные расширения файлов в карточках

По умолчанию вы можете прикрепить к *карточке* файл с любым расширением. Существует возможность повысить безопасность системы, создав список разрешенных расширений файлов. Это предотвратит загрузку пользователями файлов нежелательных форматов, в том числе тех, которые могут включать скрытый вредоносный код, например, `.js`, `.swf` и других.

Для того чтобы создать список разрешенных расширений файлов, выполните следующие действия:

1. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py
```

2. В разделе `FFSECURITY` найдите параметр `CARD_ATTACHMENTS_FILENAME_REGEX`. Задайте выражение, описывающее разрешенные файловые расширения. Подойдет любое действительное регулярное выражение Python.

Примеры:

- `r'.*\.png'`: разрешает только файлы с расширением `.png`
- `r'.*\.(png|jpg)'`: разрешает только расширения `.png` и `.jpg`
- `r'.*'`: разрешает все расширения
- `None`: разрешает все расширения
- `'XXXXXX'`: загрузка файлов с любым расширением запрещена

```
FFSECURITY = {
    ...
    'CARD_ATTACHMENTS_FILENAME_REGEX': r'.*\.txt',
    ...
}
```

Совет: Если закомментировать параметр `CARD_ATTACHMENTS_FILENAME_REGEX`, это также разрешит все расширения.

3. Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

2.4.4 Списки наблюдения

Появление определенных лиц на видеоизображениях отслеживается с помощью набора стандартных и пользовательских списков наблюдения.

Карточки лиц добавляются в списки наблюдения. После активации списка наблюдения система будет искать каждое лицо из него при обработке видеоматериалов и *удаленном мониторинге*.

Вы можете создать столько пользовательских списков наблюдения, сколько необходимо: розыск, подозреваемые и т. д. — в зависимости от потребностей.

В этом разделе:

- *Мониторинг незарегистрированных лиц*
- *Создание списка наблюдения*
- *Удаление списка наблюдения*

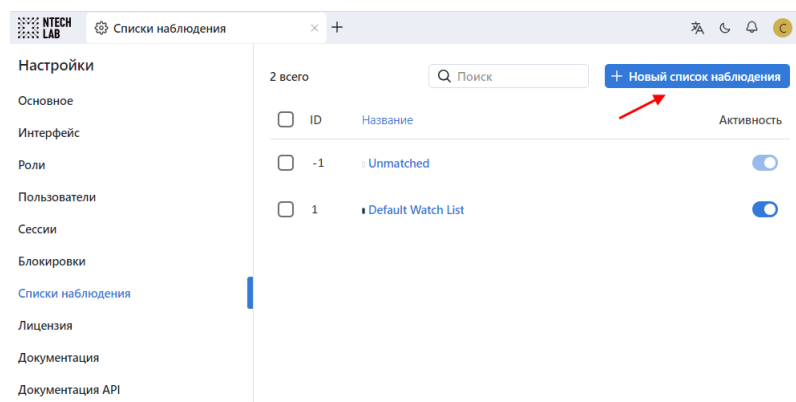
Мониторинг незарегистрированных лиц

Базовая конфигурация FindFace CIBR уже содержит предустановленный список наблюдения, предназначенный для мониторинга незарегистрированных в системе лиц, т. е. лиц, для которых отсутствуют совпадения с картотекой. Данный список наблюдения не может быть удален из системы. Для редактирования настроек списка, перейдите на вкладку *Настройки*. Выберите *Списки наблюдения* и щелкните *Unmatched*.

Создание списка наблюдения

Для создания пользовательского списка наблюдения выполните следующие действия:

1. Перейдите *Настройки* → *Списки наблюдения*.
2. Нажмите + *Новый список наблюдения*.

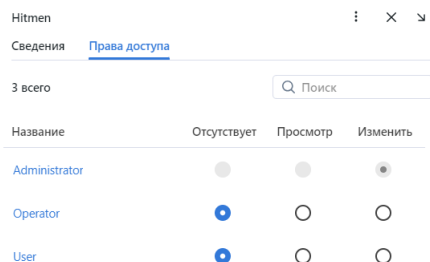


3. На вкладке *Сведения* задайте название списка наблюдения.
4. Из палитры *Цвет* выберите цвет уведомлений для данного списка.

5. При необходимости опишите назначение списка наблюдения в комментарии.
6. По умолчанию ко всем спискам наблюдения в системе применяется *универсальный порог срабатывания*. Для того чтобы задать индивидуальный порог для списка наблюдения, включите *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 предварительно проконсультируйтесь с нашими техническими специалистами (support@ntechlab.com).

7. (Опционально) Если вы включили *ежедневный поиск*, установите флажок *Сбор данных о местоположении* для того, чтобы все привязанные карточки к данному списку наблюдения получали события ежедневного поиска.
8. На вкладке *Права доступа* назначьте права на список наблюдения, указав роли пользователей, которые смогут изменять/просматривать его настройки.

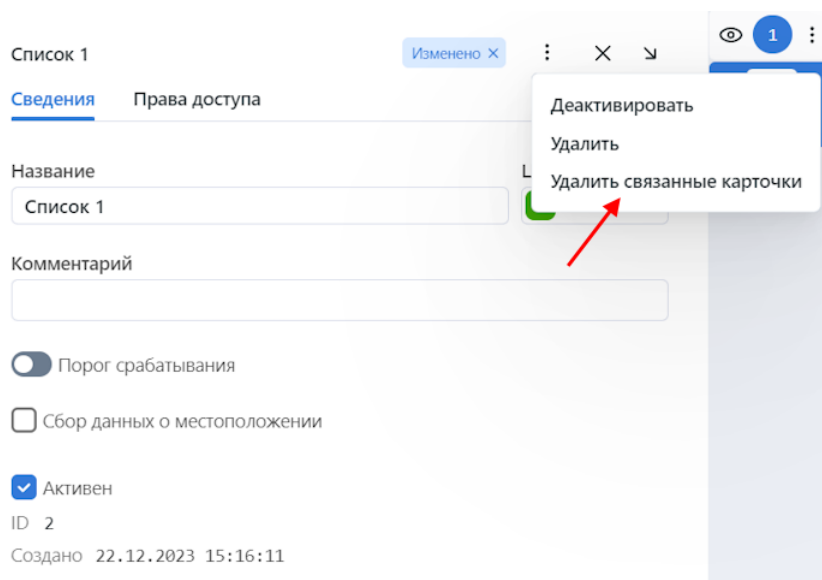


Название	Отсутствует	Просмотр	Изменить
Administrator			
Operator			
User			

9. Активируйте и сохраните список наблюдения.

Удаление списка наблюдения

Чтобы удалить пользовательский список наблюдения, сначала удалите связанные с ним карточки. В противном случае система выдаст ошибку и не удалит список наблюдения.



Списки наблюдения *Default Watch List* и *Unmatched* не могут быть удалены из системы.

2.4.5 Пользовательские вкладки, поля и фильтры в карточке

См.также:

Для создания пользовательских полей в базе данных векторов признаков см. [Пользовательские метаданные в Tarantool](#).

Для добавления пользовательских вкладок и полей в карточки людей выполните следующие действия:

1. Подготовьте список пользовательских вкладок и полей для добавления в карточки.
2. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`.


```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py
```

3. Настройте карточки людей. Для этого раскомментируйте секцию FFSECURITY -> CUSTOM_FIELDS -> `human_card` и измените ее примерное содержимое с учетом следующих правил:

- `'items'`: список полей в карточке. Опишите каждое поле следующими параметрами:
 - `'name'`: внутреннее имя поля, `string`.
 - `default`: значение по умолчанию. Если значение по умолчанию больше `1e14 - 1`, то его следует записывать в виде строки, т. е. `"123123..."` вместо `123123...`
 - `'label'`: название поля в карточке, `string`.
 - `'tab'`: вкладка, на которой располагается поле.
 - `'display'`: формат отображения (`form` или `list`), `string` или `array`.
 - `'description'`: описание поля, `string`.
 - `'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 полей в карточке.

```

FFSECURITY = {

...

# -- Custom model fields --
# Edit CUSTOM_FIELDS -> `human_card` section to customize human card fields.
...
'CUSTOM_FIELDS': {
    'human_card': {
        ...
        'items': [
            {
                'name': 'personid',
                'default': '',
                'label': 'PersonID',
                'display': ['list', 'form'],
                'description': 'Sigur person ID',
                'editable': False
            },
            {
                'name': 'firstname',
                'default': '',
                'label': 'First Name',
                'display': ['list', 'form'],
                'description': 'Sigur first name',
                'editable': False
            },
            {
                'name': 'lastname',
                'default': '',
                'label': 'Last Name',
                'display': ['list', 'form'],
                'description': 'Sigur last name',
                'editable': False
            },
            {
                'name': 'version',
                'default': '',
                'label': 'Version',
                'display': ['list', 'form'],
                'description': 'Sigur photo version',
                'editable': False
            }
        ],
        'filters': [
            {
                'name': 'personid',
                'label': 'Sigur person ID filter',
                'field': 'personid'
            }
        ]
    },

```

(continues on next page)

(продолжение с предыдущей страницы)

```

    },
}

```

4. Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

Вы увидите, что в карточках появилось новое содержимое.

2.4.6 Пользовательские метаданные в Tarantool

Часто бывает необходимо присвоить дополнительные метаданные лицам, извлеченным из загруженных в картотеку изображений и теперь хранящимся в базе данных векторов признаков.

Настройка метаполей для лиц

Для присвоения пользовательских метаданных лицам выполните следующие действия:

1. Подготовьте список пользовательских метаполей, которые будут присвоены лицам.
2. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py
```

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

- **field_name**: имя поля;
- **type**: тип данных (`uint`, `string` или `bool`);
- **default**: значение по умолчанию. Если значение по умолчанию больше `1e14 - 1`, то его следует записывать в виде строки, т. е. `"123123..."` вместо `123123...`

```
FFSECURITY = {
...

    # -- Custom model fields --
    ...
    # Edit CUSTOM_FIELDS -> `face_object` section to customize face object fields.
    ...
    # 'CUSTOM_FIELDS': {
        ...
        'face_object': {
            'items': [
                {
                    "field_name": "tag_name_1",
                    "type": "string",
                    "default": "change_me"
                },
            ]
        }
    }
}
```

(continues on next page)

(продолжение с предыдущей страницы)

```

        {
            "field_name": "tag_name_2",
            "type": "uint",
            "default": 123
        },
        {
            "field_name": "tag_name_3",
            "type": "bool",
            "default": True
        },
    ]
},
}

```

4. Добавьте новые метаполя в структуру базы данных векторов признаков.
5. Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

Вы можете работать с новыми метаполями через *HTTP API*, используя методы `objects/faces/`.

См. также:

Для создания пользовательских вкладок, полей и фильтров в карточках см. *Пользовательские вкладки, поля и фильтры в карточке*.

2.4.7 Распознавание атрибутов лица

Атрибуты лица, такие как возраст, пол, эмоции и т. д., присутствуют в наборе фильтров для анализа обнаруженных лиц при *расследовании происшествий*.

Распознавание атрибутов лица можно автоматически включить и настроить в ходе *установки FindFace CIBR*. При пропуске данного шага можно впоследствии сделать это вручную. Распознавание атрибутов лица работает как на GPU-, так и на CPU-ускорении.

Выполните следующие действия:

1. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml`.

```
sudo vi /opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.
↵yaml
```

2. В секции `extractors` укажите соответствующие модели, как показано в примере ниже. 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.

GPU

```
extractors:
    ...
    models:
        face_age: faceattr/age.v2.gpu.fnk
        face_beard: faceattr/beard.v0.gpu.fnk
        face_beard4: ''
        face_countries47: ''
        face_emben: face/mango_320.gpu.fnk
        face_emotions: faceattr/emotions.v1.gpu.fnk
        face_eyes_attrs: ''
        face_eyes_openness: ''
        face_gender: faceattr/gender.v2.gpu.fnk
        face_glasses3: faceattr/glasses3.v0.gpu.fnk
        face_glasses4: ''
        face_hair: ''
        face_headpose: ''
        face_headwear: ''
        face_highlight: ''
        face_liveness: faceattr/liveness.web.v0.gpu.fnk
        face_luminance_overexposure: ''
        face_luminance_underexposure: ''
        face_luminance_uniformity: ''
        face_medmask3: faceattr/medmask3.v2.gpu.fnk
        face_medmask4: ''
        face_mouth_attrs: ''
        face_quality: faceattr/quality_fast.v1.gpu.fnk
        face_scar: ''
        face_sharpness: ''
        face_tattoo: ''
        face_validity: ''
```

CPU

```
extractors:
    ...
    models:
        face_age: faceattr/age.v2.cpu.fnk
        face_beard: faceattr/beard.v0.cpu.fnk
        face_beard4: ''
        face_countries47: ''
        face_emben: face/mango_320.cpu.fnk
        face_emotions: faceattr/emotions.v1.cpu.fnk
        face_eyes_attrs: ''
        face_eyes_openness: ''
        face_gender: faceattr/gender.v2.cpu.fnk
        face_glasses3: faceattr/glasses3.v0.cpu.fnk
        face_glasses4: ''
        face_hair: ''
        face_headpose: ''
```

(continues on next page)

(продолжение с предыдущей страницы)

```

face_headwear: ''
face_highlight: ''
face_liveness: faceattr/liveness.web.v0.cpu.fnk
face_luminance_overexposure: ''
face_luminance_underexposure: ''
face_luminance_uniformity: ''
face_medmask3: faceattr/medmask3.v2.cpu.fnk
face_medmask4: ''
face_mouth_attrs: ''
face_quality: faceattr/quality_fast.v1.cpu.fnk
face_scar: ''
face_sharpness: ''
face_tattoo: ''
face_validity: ''

```

Доступны следующие модели экстракторов:

Экстрактор	Ускоре- ние	Настройка
возраст	CPU	face_age: faceattr/age.v2.cpu.fnk
	GPU	face_age: faceattr/age.v2.gpu.fnk
борода	CPU	face_beard: faceattr/beard.v0.cpu.fnk
	GPU	face_beard: faceattr/beard.v0.gpu.fnk
вектор признаков объекта лица	CPU	face_emben: face/mango_320.cpu.fnk
	GPU	face_emben: face/mango_320.gpu.fnk
пол	CPU	face_gender: faceattr/gender.v2.cpu.fnk
	GPU	face_gender: faceattr/gender.v2.gpu.fnk
эмоции	CPU	face_emotions: faceattr/emotions.v1.cpu.fnk
	GPU	face_emotions: faceattr/emotions.v1.gpu.fnk
очки	CPU	face_glasses3: faceattr/glasses3.v0.cpu.fnk
	GPU	face_glasses3: faceattr/glasses3.v0.gpu.fnk
витальность	CPU	face_liveness: faceattr/liveness.web.v0.cpu.fnk
	GPU	face_liveness: faceattr/liveness.web.v0.gpu.fnk
медицинская маска	CPU	face_medmask3: faceattr/medmask3.v2.cpu.fnk
	GPU	face_medmask3: faceattr/medmask3.v2.gpu.fnk
качество изображения лица	CPU	face_quality: faceattr/quality_fast.v1.cpu.fnk
	GPU	face_quality: faceattr/quality_fast.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. Например, чтобы отключить распознавание эмоций и маски на лице, передайте пустое значение в соответствующие модели экстракторов:

```

extractors:
...
models:

```

(continues on next page)

(продолжение с предыдущей страницы)

```
...
face_emotions: ''
face_medmask3: ''
```

Примечание: Вы можете найти модели для распознавания атрибутов лица в каталоге `/opt/findface-cibr/models/faceattr/`.

```
ls /opt/findface-cibr/models/faceattr/

age.v2.cpu.fnk      emotions.v1.cpu.fnk  glasses3.v0.cpu.fnk  medmask3.v2.cpu.fnk
age.v2.gpu.fnk      emotions.v1.gpu.fnk  glasses3.v0.gpu.fnk  medmask3.v2.gpu.fnk
beard.v0.cpu.fnk    gender.v2.cpu.fnk    liveness.web.v0.cpu.fnk  quality_fast.v1.cpu.
↪fnk
beard.v0.gpu.fnk    gender.v2.gpu.fnk    liveness.web.v0.gpu.fnk  quality_fast.v1.gpu.
↪fnk
```

3. Перезапустите контейнер `findface-cibr-findface-extraction-api-1`.

```
sudo docker container restart findface-cibr-findface-extraction-api-1
```

4. Включите распознавание атрибутов лица в файле конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`. В разделе `FFSECURITY` укажите атрибуты лица, которые требуется отображать в событиях распознавания.

```
# make sure that corresponding extractors are licensed
# and enabled at findface-extraction-api config file
# available features: age, beard, emotions, gender, glasses, medmask
'FACE_EVENTS_FEATURES': ['emotions', 'beard', 'gender', 'age', 'medmask', 'glasses
↪'],
```

5. Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

2.4.8 Интеграция с удаленными системами распознавания лиц

Вы можете интегрировать свой экземпляр FindFace CIBR с удаленными системами распознавания лиц. В этом случае сервер, известный как «кукловод», будет отправлять биометрические данные из поставленных на мониторинг карточек на удаленные серверы-«марионетки». В ответ он будет получать события распознавания с совпадениями по данным карточкам. Вы можете настроить *ежедневный поиск*, который позволяет серверу-кукловоду получать события по расписанию.

Спектр возможных применений данной функциональности широк. Одним из направлений является отслеживание местонахождения и маршрутов преступников и выявление предполагаемых сообщников. Другое направление — поиск пропавших людей. Результаты особенно хороши, если данная функциональность применяется к системам общественной и транспортной безопасности с тысячами камер.

В текущей версии поддерживается только интеграция с системами распознавания лиц из семейства FindFace.

В этом разделе:

- *Расписание синхронизации*
- *Настройка сервера-кукловода*
- *Настройка сервера-марионетки*

Расписание синхронизации

Данные между сервером-кукловодом и сервером-марионеткой синхронизируются следующим образом:

- Сервер-кукловод доставляет предназначенные для мониторинга карточки серверу-марионетке с интервалом, указанным в параметре `REMOTE_MONITORING_SYNC_INTERVAL` (см. настройку ниже).
- Сервер-марионетка отправляет совпадающие события распознавания серверу-кукловоду, как только они появляются.

Настройка сервера-кукловода

Для настройки экземпляра FindFace CIBR в качестве сервера-кукловода выполните следующие действия:

1. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`. Убедитесь, что параметр `EXTERNAL_ADDRESS` заполнен.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py

EXTERNAL_ADDRESS = 'http://192.168.0.4'
```

Примечание: Значение `EXTERNAL_ADDRESS` определяется во время развертывания.

2. Найдите раздел `Puppeteer`.

```
# ===== Puppeteer =====
# INSTALLED_APPS.append('ffsecurity_puppeteer')

# PUPPETEER_CONFIG = {
#     'UNSAVED_RESULTS_DELETION_TIMEOUT': 3600,          # maximum lifetime of search
#     ↳ results not saved involuntarily
#     'REMOTE_MONITORING_SYNC_INTERVAL': 600,          # monitoring data
#     ↳ synchronization interval, seconds
#     'REMOTE_MONITORING_EVENTS_MAX_AGE': 30*24*60*60, # monitoring events older
#     ↳ than this number of seconds will be
#     #                                                  # automatically deleted
#     ↳ (every night at 1:17 am by default)
#     'ENABLE_DAILY_SEARCH': False,                   # daily search activation
#     ↳ (default False)
#     'DAILY_SEARCH_PUSH_HOUR': 2,                    # daily search cards
#     ↳ synchronization hour
#     'DAILY_SEARCH_PULL_HOUR': 6,                    # hour in which results of
```

(continues on next page)

(продолжение с предыдущей страницы)

```

→ daily search will be obtained
#   'DAILY_SEARCH_EVENTS_MAX_AGE': 0,           # daily search events older
→ than this number of seconds will be         # automatically deleted
#
→ (every night at 1:17 am by default)
#   'DAILY_SEARCH_EVENTS_FULLFRAME_MAX_AGE': 30*24*60*60, # daily search events
→ fullframes                                  # older than this number of
#                                              # automatically deleted
→ (every night at 1:17 am by default)
#   'puppets': [
#       {
#           'id': 'first_puppet',                # puppet ID
#           'url': 'http://1.1.1.1:8010/',        # puppet URL
#           'token': 'first_puppet_token',        # use pwgen -s 64 1 (should
→ match the token in puppet)
#           'facen_model': 'mango_320'           # face model in puppet
#       },
#       {
#           'id': 'second_puppet',
#           'url': 'http://1.1.1.1:8010/',
#           'token': 'second_puppet_token',
#
#           # if remote installation has a different face model than the one
→ used in FFSecurity -
#           # you need to specify its name and ExtractionAPI URL where the
→ corresponding face model is specified
#           # other settings like detector, normalizer and other should be set
→ as puppeteer's ExtractionAPI
#           # contact support for details
#           'facen_model': 'grapefruit_480',
#           'extractor': 'http://127.0.0.1:18667',
#       },
#   ]
# }
#

```

3. Раскомментируйте раздел, как показано в примере ниже, и укажите следующие параметры:

- **REMOTE_MONITORING_SYNC_INTERVAL**: интервал в секундах, с которым сервер-кукловод отправляет предназначенные для мониторинга карточки на сервер-марионетку.
- **REMOTE_MONITORING_EVENTS_MAX_AGE**: события мониторинга старше указанного количества дней будут автоматически удаляться (по умолчанию каждую ночь в 1:17).
- **puppets** → **id**: идентификатор сервера-марионетки.
- **puppets** → **url**: IP-адрес и порт сервера-марионетки.

Укажите порт следующим образом:

- Оставьте без изменений, если сервер-марионетка является общественной или транспортной системой безопасности (т. е. в его основе лежит сервис **public-security**).
- Измените номер на 80 или вообще не указывайте его, если сервер-марионетка построен

на сервисе `findface-security` (т. е. на нем установлен продукт FindFace Security или FindFace Multi).

- `puppets` → `token`: токен, который будут использовать сервер-кукловод и сервер-марионетка для взаимной аутентификации.

Совет: Для создания случайного токена используйте следующую команду.

```
pwgen -s 64 1
```

- `puppets` → `facen_model`: модель нейронной сети, используемая на сервере-марионетке для распознавания лиц.
- `puppets` → `extractor`: IP-адрес и порт сервиса, используемого для извлечения биометрических данных на сервере-марионетке, если модель нейросети для распознавания лиц на нем отличается от модели на сервере-кукловоде.

Оставьте закомментированными остальные параметры в разделе.

```
# ===== Puppeteer =====
INSTALLED_APPS.append('ffsecurity_puppeteer')

PUPPETEER_CONFIG = {
#     'UNSAVED_RESULTS_DELETION_TIMEOUT': 3600,           # maximum lifetime of search
↳ results not saved involuntarily
    'REMOTE_MONITORING_SYNC_INTERVAL': 600,              # monitoring data
↳ synchronization interval, seconds
    'REMOTE_MONITORING_EVENTS_MAX_AGE': 30*24*60*60,      # monitoring events older than
↳ this number of seconds will be
#                                                         # automatically deleted
↳ (every night at 1:17 am by default)
#     'ENABLE_DAILY_SEARCH': False,                      # daily search activation
↳ (default False)
#     'DAILY_SEARCH_PUSH_HOUR': 2,                       # daily search cards
↳ synchronization hour
#     'DAILY_SEARCH_PULL_HOUR': 6,                       # hour in which results of
↳ daily search will be obtained
#     'DAILY_SEARCH_EVENTS_MAX_AGE': 0,                  # daily search events older
↳ than this number of seconds will be
#                                                         # automatically deleted
↳ (every night at 1:17 am by default)
#     'DAILY_SEARCH_EVENTS_FULLFRAME_MAX_AGE': 30*24*60*60, # daily search events
↳ fullframes
#                                                         # older than this number of
↳ seconds will be
#                                                         # automatically deleted
↳ (every night at 1:17 am by default)
    'puppets': [
        {
            'id': 'first_puppet',                        # puppet ID
            'url': 'http://1.1.1.1:8010/',               # puppet URL
            'token': 'first_puppet_token',               # use pwgen -s 64 1 (should
↳ match the token in puppet)
```

(continues on next page)

(продолжение с предыдущей страницы)

```

        'facen_model': 'mango_320'           # face model in puppet
    },
    {
        'id': 'second_puppet',
        'url': 'http://1.1.1.1:8010/',
        'token': 'second_puppet_token',

#           # if remote installation has a different face model than the one
→used in FFSecurity -
#           # you need to specify its name and ExtractionAPI URL where the
→corresponding face model is specified
#           # other settings like detector, normalizer and other should be set
→as puppeteer's ExtractionAPI
#           # contact support for details
        'facen_model': 'grapefruit_480',
        'extractor': 'http://127.0.0.1:18667',
    },
]
}

```

4. При необходимости включите ежедневный поиск. Для этого снимите комментарий и установите значение 'ENABLE_DAILY_SEARCH': True. Установите расписание отправки карточек на сервер-марионетку и получения событий, имеющих совпадение с отправленными карточками.

```

PUPPETEER_CONFIG = {
    ...
    'ENABLE_DAILY_SEARCH': True,           # daily search activation
→(default False)
    'DAILY_SEARCH_PUSH_HOUR': 2,           # daily search cards
→synchronization hour
    'DAILY_SEARCH_PULL_HOUR': 6,           # hour in which results of
→daily search will be obtained
    ...

```

Перейдите во вкладку *Списки наблюдения* и установите флажок **Сбор данных о местоположении** для интересующего вас списка наблюдения.

5. При необходимости включите очистку ежедневного поиска, раскомментировав следующие параметры:

```

PUPPETEER_CONFIG = {
    ...
    'DAILY_SEARCH_EVENTS_MAX_AGE': 0,      # daily search events older
→than this number of seconds will be
                                           # automatically deleted (every
→night at 1:17 am by default)
    'DAILY_SEARCH_EVENTS_FULLFRAME_MAX_AGE': 30*24*60*60, # daily search events
→fullframes
                                           # older than this number of
→seconds will be
                                           # automatically deleted (every
→night at 1:17 am by default)
    ...

```

- 'DAILY_SEARCH_EVENTS_MAX_AGE': 0: продолжительность (в секундах) хранения данных о местоположении объекта интереса. По умолчанию: 0, постоянное хранение.
- 'DAILY_SEARCH_EVENTS_FULLFRAME_MAX_AGE': 30*24*60*60: длительность хранения (в секундах) полных кадров событий ежедневного поиска, полученных от сервера-марионетки. По умолчанию: 30*24*60*60 секунд (30 дней). Измените 30 на нужное количество дней при необходимости.

Примечание: Вы можете вручную удалить данные о местоположении объекта и полные кадры событий ежедневного поиска старше заданного количества секунд, выполнив команды, приведенные ниже.

Чтобы задать количество секунд для удаления данных о местоположении объекта, используйте аргумент `--daily-search-events-max-age`.

```
sudo docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/
python3 /tigre_prototype/manage.py cleanup --daily-search-events-max-age 1
```

Для удаления полных кадров событий ежедневного поиска, полученных от сервера-марионетки, старше 5 дней ($5*24*60*60 = 432000$ секунд), используйте аргумент `--daily-search-events-fullframe-max-age`.

```
sudo docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/
python3 /tigre_prototype/manage.py cleanup --daily-search-events-fullframe-max-
age 432000
```

6. Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

Настройка сервера-марионетки

Для настройки удаленного экземпляра **FindFace Multi 2.0+** в качестве сервера-марионетки выполните следующие действия:

1. Откройте файл конфигурации `/opt/findface-multi/configs/findface-multi-legacy/findface-multi-legacy.py`. Убедитесь, что параметр `EXTERNAL_ADDRESS` заполнен.

```
sudo vi /opt/findface-multi/configs/findface-multi-legacy/findface-multi-legacy.py

EXTERNAL_ADDRESS = 'http://192.168.0.5'
```

2. Найдите раздел `Vns`.

```
# ===== Vns =====
# A plugin for using FindFace Security as a puppeteer server
# INSTALLED_APPS.append('ffsecurity_vns')

# VNS_CONFIG = {
#     'USERS': {
#         'user1': 'token1',
#         'user2': 'token2'
#     },
# }
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# 'MONITORING_THRESHOLD': 0.75,
# 'DAILY': {
#     'ENABLED': False,
#     'THRESHOLD': 0.75,
#     'START_TIME': "00:00:00"
# }
# }
```

3. Раскомментируйте раздел, как показано в примере ниже, и укажите следующие параметры:

- **token**: токен, который будут использовать сервер-марионетка и сервер-кукловод для взаимной аутентификации. Вы можете указать несколько пользователей и токенов, если марионетка взаимодействует с несколькими кукловодами. Имена пользователей (**user**) можно оставить без изменений.
- **MONITORING_THRESHOLD**: порог достоверности в событиях распознавания лиц, отправляемых на сервер-кукловод.

Оставьте закомментированными остальные параметры в разделе.

```
# ===== Vns =====
# A plugin for using FindFace Security as a puppeteer server
INSTALLED_APPS.append('ffsecurity_vns')

VNS_CONFIG = {
    'USERS': {
        'user1': '1234567890'
    },
    'MONITORING_THRESHOLD': 0.75,
#     'DAILY': {
#         'ENABLED': False,
#         'THRESHOLD': 0.75,
#         'START_TIME': "00:00:00"
#     }
}
```

4. Если вы включили ежедневный поиск на сервере-кукловоде (см. шаг №4 выше), раскомментируйте и включите параметры 'DAILY'. Укажите время для сопоставления событий с карточками, которые были получены от сервера-кукловода.

```
===== Vns =====
A plugin for using FindFace Security as a puppeteer server
INSTALLED_APPS.append('ffsecurity_vns')

VNS_CONFIG = {
    ...
    'DAILY': {
        'ENABLED': True,
        'THRESHOLD': 0.75,
        'START_TIME': "00:00:00"
    }
}
```

5. Перезапустите контейнер `findface-multi-findface-multi-legacy-1`.

```
sudo docker container restart findface-multi-findface-multi-legacy-1
```

Для настройки удаленного экземпляра **FindFace Multi** версии 1.2 и более ранней в качестве сервера-марионетки выполните следующие действия:

1. Откройте файл конфигурации `/etc/findface-security/config.py`. Убедитесь, что параметр `EXTERNAL_ADDRESS` заполнен.

```
sudo vi /etc/findface-security/config.py  
  
EXTERNAL_ADDRESS = 'http://192.168.0.5'
```

2. Найдите *раздел Vns*.
3. Раскомментируйте раздел, как показано в *примере* выше, и укажите параметры `token` и `MONITORING_THRESHOLD`.
4. Если вы включили ежедневный поиск на сервере-кукловоде (см. шаг №4 выше), снимите комментарий и включите параметры `'DAILY'`. Укажите время для сопоставления событий с карточками, которые были получены от сервера-кукловода.
5. Перезапустите сервис `findface-security`.

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

6. Выполните миграцию, чтобы синхронизировать сервер-марионетку с серверами-кукловодами.

```
sudo findface-security migrate
```

См.также:

Удаленный мониторинг и удаленный поиск

2.4.9 Использование нескольких видеокарт

Если в вашей системе на базе GPU на физическом сервере установлено несколько видеокарт, вы можете создать дополнительные экземпляры `findface-extraction-api` или `findface-video-worker` и распределить их по одному экземпляру на карту.

Если вы следовали инструкции по подготовке сервера на *Ubuntu* (*CentOS*, *Debian*), то у вас уже должно быть все готово к работе.

Однако прежде чем приступать к дальнейшим действиям, убедитесь, что в файле `/etc/docker/daemon.json` указана правильная конфигурация. В примере, приведенном для операционной системы Ubuntu, показано как настроить сеть Docker, а также как настроить использование среды NVIDIA Container Runtime. Обратите внимание, что в нашем примере мы подразумеваем, что среда NVIDIA Container Runtime уже *установлена*.

```
sudo su  
BIP=10.$((RANDOM % 256)).$((RANDOM % 256))  
cat > /etc/docker/daemon.json <<EOF  
{  
  "default-address-pools":  
    [  
      {"base": "$BIP.0/16", "size": 24}  
    ],  
}
```

(continues on next page)

(продолжение с предыдущей страницы)

```

"bip": "$BIP.1/24",
"fixed-cidr": "$BIP.0/24",
"runtimes": {
  "nvidia": {
    "path": "nvidia-container-runtime",
    "runtimeArgs": []
  }
},
"default-runtime": "nvidia"
}
EOF

```

Обратитесь к соответствующим разделам документации для проверки конфигурации файла `/etc/docker/daemon.json` для *CentOS* и *Debian*.

В этом разделе:

- *Распределение экземпляров `findface-extraction-api` между несколькими видеокартами*
- *Привязка `findface-video-worker` к дополнительной видеокарте*

Распределение экземпляров `findface-extraction-api` между несколькими видеокартами

Для распределения экземпляров `findface-extraction-api` по нескольким видеокартам создайте несколько экземпляров сервиса `findface-extraction-api` в конфигурационном файле `docker-compose.yaml`. Затем, чтобы экземпляры `findface-extraction-api` работали в одной системе, объедините их через балансировщик нагрузки, например, Nginx.

Сделайте следующее:

1. Внесите изменения в конфигурационный файл `docker-compose.yaml`.
 1. Откройте файл `/opt/findface-cibr/docker-compose.yaml` и создайте несколько экземпляров конфигурации `findface-extraction-api` в секции `findface-extraction-api`. Каждый новый экземпляр конфигурации `findface-extraction-api` должен быть настроен аналогично другому, чтобы не возникало проблем при обработке запросов. Сделайте следующее:
 - Переименуйте используемое по умолчанию название сервиса `findface-extraction-api`.
 - Создайте копии конфигурации для новых экземпляров сервиса.
 - Отредактируйте новые экземпляры конфигурации.

Ниже приведен пример конфигурации для двух сервисов:

```

sudo vi /opt/findface-cibr/docker-compose.yaml

findface-extraction-api-0:
command: [--config=/etc/findface-extraction-api.ini]
depends_on: [findface-ntls]
environment:
  - CUDA_VISIBLE_DEVICES=0
  - CFG_LISTEN=127.0.0.1:18660

```

(continues on next page)

(продолжение с предыдущей страницы)

```

image: docker.int.ntl/ntech/universe/extraction-api-gpu:ffserver-9.230407.1
logging: {driver: journald}
network_mode: service:pause
restart: always
runtime: nvidia
volumes: ['./configs/findface-extraction-api/findface-extraction-api.yaml:/
↪etc/findface-extraction-api.ini:ro',
          './models:/usr/share/findface-data/models:ro', './cache/findface-extraction-
↪api/models:/var/cache/findface/models_cache']
findface-extraction-api-1:
  command: [--config=/etc/findface-extraction-api.ini]
  depends_on: [findface-ntls]
  environment:
    - CUDA_VISIBLE_DEVICES=1
    - CFG_LISTEN=127.0.0.1:18661
  image: docker.int.ntl/ntech/universe/extraction-api-gpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  runtime: nvidia
  volumes: ['./configs/findface-extraction-api/findface-extraction-api.yaml:/
↪etc/findface-extraction-api.ini:ro',
            './models:/usr/share/findface-data/models:ro', './cache/findface-extraction-
↪api/models:/var/cache/findface/models_cache']

```

- `findface-extraction-api-0` — читаемое название экземпляра. Должно быть уникальным для каждого экземпляра;
- `CUDA_VISIBLE_DEVICES=0` — идентификатор GPU для запуска экземпляра сервиса. Должен быть уникальным для каждого экземпляра;
- `CFG_LISTEN=127.0.0.1:18660` — слушающий IP:Порт. Должен быть уникальным для каждого экземпляра;
- `./cache/findface-extraction-api/models:/var/cache/findface/models_cache` — том (volume) для хранения кэша моделей нейронных сетей. Он может быть одинаковым, если у вас одинаковые модели GPU. В других случаях кэши хранятся отдельно.

Остальные параметры конфигурации могут быть одинаковыми для каждого экземпляра сервиса `findface-extraction-api`.

2. Внесите изменения в файл `/opt/findface-cibr/docker-compose.yaml`, чтобы настроить балансировку нагрузки. Для этого добавьте новый сервис (`findface-extraction-api-lb` в нашем примере) в файл `docker-compose.yaml`, например, в конец файла.

```

findface-extraction-api-lb:
  depends_on: [findface-ntls]
  image: docker.int.ntl/ntech/multi/multi/ui-cibr:ffcibr-2.1.2
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  volumes: ['./configs/findface-extraction-api/loadbalancer.conf:/etc/nginx/
↪conf.d/default.conf:ro']

```

В томах (volumes) секции `findface-extraction-api-lb` мы указали конфигурационный

файл `loadbalancer.conf`. Теперь нам нужно создать его.

2. Создайте файл конфигурации балансировщика нагрузки со следующим содержимым:

```
sudo vi /opt/findface-cibr/configs/findface-extraction-api/loadbalancer.conf

upstream findface-extraction-api-lb {
    least_conn;
    server 127.0.0.1:18660 max_fails=3 fail_timeout=60s;
    server 127.0.0.1:18661 max_fails=3 fail_timeout=60s;
}

server {
    listen 18666 default_server;
    server_name _;
    location / {
        proxy_pass http://findface-extraction-api-lb;
    }
}
```

В секции `upstream` настройте поведение балансировки и перечислите экземпляры `findface-extraction-api`, между которыми будет распределяться нагрузка.

Вы можете выбрать один из приведенных ниже вариантов поведения отправки, но мы рекомендуем использовать вариант `least_conn`.

- `round-robin` — запросы равномерно распределяются по серверам циклически;
- `least_conn` — запросы отправляются на сервер с наименьшим количеством активных соединений. Этот алгоритм удобен, когда у вас есть внутренние серверы с разной пропускной способностью;
- `weighted` — внутренним серверам присваиваются различные веса, и запросы распределяются в зависимости от этих весов. Таким образом, у определенных серверов приоритет будет выше, чем у других.

Строка `server 127.0.0.1:18666 max_fails=3 fail_timeout=60s` состоит из следующих значимых частей:

- `server` — определяет вышестоящий сервер в Nginx;
- `127.0.0.1:18660` — указывает IP-адрес (`127.0.0.1`) и порт (`18660`) экземпляра `findface-extraction-api`;
- `max_fails=3` — параметр задает максимальное количество неудачных попыток, которое Nginx допустит при подключении к серверу, прежде чем посчитает его недоступным;
- `fail_timeout=60s` — параметр задает 1) время, в течение которого должно произойти заданное в параметре `max_fails` число неудачных попыток работы с сервером для того, чтобы сервер считался недоступным; 2) время, в течение которого сервер будет считаться недоступным (в данном случае 60 секунд);
- `weight=2` (в текущем примере не используется) — параметр позволяет назначить относительный вес или приоритет каждому серверу в группе `upstream`. Это означает, что на каждый запрос сервер с `weight=2` получит в два раза больше запросов. Выставление параметра может понадобиться в сценариях, где используются графические процессоры разной производительности.

3. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml`. Адрес и порт балансировщика нагрузки в секции `extraction-api`

должны иметь вид 127.0.0.1:18666. Если адрес и порт балансировщика нагрузки в вашем примере отличаются от вышеуказанных значений, приведите их к виду 127.0.0.1:18666.

```
sudo vi /opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml

extraction-api:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  max-idle-conns-per-host: 20
  keepalive: 24h0m0s
  trace: false
  url: http://127.0.0.1:18666
```

4. Запустите пересборку контейнеров FindFace CIBR, выполнив при этом очистку старых контейнеров для сервисов, которые больше не включены в файл `docker-compose.yaml`.

```
cd /opt/findface-cibr/

docker-compose down
docker-compose up -d --remove-orphans
```

Чтобы убедиться в правильной работе сервисов, проверьте их логи.

```
docker compose logs --tail 10 -f findface-extraction-api-0
docker compose logs --tail 10 -f findface-extraction-api-1
docker compose logs --tail 10 -f findface-extraction-api-lb
```

Привязка `findface-video-worker` к дополнительной видеокарте

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

1. В файле `/opt/findface-cibr/docker-compose.yaml` укажите конфигурацию `findface-video-worker` для каждого запущенного экземпляра `findface-video-worker`. Скопируйте текущую конфигурацию `findface-video-worker`.

```
sudo vi /opt/findface-cibr/docker-compose.yaml

findface-video-worker:
  command: [--config=/etc/findface-video-worker.yaml]
  depends_on: [findface-video-manager, findface-ntls, mongoddb]
  environment: [CUDA_VISIBLE_DEVICES=0]
  image: docker.int.ntl/ntech/universe/video-worker-gpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  runtime: nvidia
  volumes: ['./configs/findface-video-worker/findface-video-worker.yaml:/etc/
  ↪findface-video-worker.yaml:ro',
    './models:/usr/share/findface-data/models:ro', './cache/findface-video-worker/
```

(continues on next page)

(продолжение с предыдущей страницы)

```
↪models:/var/cache/findface/models_cache',
  './cache/findface-video-worker/recorder:/var/cache/findface/video-worker-
↪recorder']
```

Затем отредактируйте каждую конфигурацию соответствующим образом.

```
findface-video-worker-0:
  command: [--config=/etc/findface-video-worker.yaml]
  depends_on: [findface-video-manager, findface-ntls, mongodb]
  environment:
    - CUDA_VISIBLE_DEVICES=0
    - CFG_STREAMER_PORT=18990
    - CFG_STREAMER_URL=127.0.0.1:18990
  image: docker.int.ntl/ntech/universe/video-worker-gpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  runtime: nvidia
  volumes: ['./configs/findface-video-worker/findface-video-worker.yaml:/etc/
↪findface-video-worker.yaml:ro',
    './models:/usr/share/findface-data/models:ro', './cache/findface-video-worker/
↪models:/var/cache/findface/models_cache',
    './cache/findface-video-worker/recorder:/var/cache/findface/video-worker-
↪recorder']
findface-video-worker-1:
  command: [--config=/etc/findface-video-worker.yaml]
  depends_on: [findface-video-manager, findface-ntls, mongodb]
  environment:
    - CUDA_VISIBLE_DEVICES=1
    - CFG_STREAMER_PORT=18991
    - CFG_STREAMER_URL=127.0.0.1:18991
  image: docker.int.ntl/ntech/universe/video-worker-gpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  runtime: nvidia
  volumes: ['./configs/findface-video-worker/findface-video-worker.yaml:/etc/
↪findface-video-worker.yaml:ro',
    './models:/usr/share/findface-data/models:ro', './cache/findface-video-worker/
↪models:/var/cache/findface/models_cache',
    './cache/findface-video-worker/recorder:/var/cache/findface/video-worker-
↪recorder']
```

Здесь есть следующие основные параметры:

- `findface-video-worker-0` — новое название экземпляра `findface-video-worker` (в данном примере экземпляру `findface-video-worker` присвоен номер, равный ID GPU на устройстве);
- `CUDA_VISIBLE_DEVICES=0` — идентификатор устройства CUDA для запуска нового экземпляра;
- `CFG_STREAMER_PORT=18991` — порт стримера (должен быть уникальным);
- `CFG_STREAMER_URL=127.0.0.1:18991` — URL для подключения и получения потока от экзем-

пляра `findface-video-worker` (должен быть уникальным).

Все остальные параметры остаются неизменными.

2. Запустите пересборку контейнеров FindFace CIBR, выполнив при этом очистку старых контейнеров для сервисов, которые больше не включены в файл `docker-compose.yaml`.

```
cd /opt/findface-cibr/

docker-compose down
docker-compose up -d --remove-orphans
```

Чтобы убедиться в правильной работе экземпляров `findface-video-worker`, проверьте их логи.

```
docker compose logs --tail 10 -f findface-video-worker-0
docker compose logs --tail 10 -f findface-video-worker-1
```

2.4.10 Настройка сохранения изображений в отчетах

При создании *отчетов* предоставляется возможность выбрать способ представления в нем изображений: в виде ссылок, миниатюр или полных кадров. Вы можете настроить параметры изображений. Для этого откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` и измените заданные по умолчанию качество JPEG и максимальную высоту миниатюр и полных кадров в зависимости от свободного дискового пространства.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py

# reports image saving options
'REPORT_THUMBNAIL_JPEG_QUALITY': 75,
'REPORT_THUMBNAIL_MAX_HEIGHT': 100,
'REPORT_FULLFRAME_JPEG_QUALITY': 75,
'REPORT_FULLFRAME_MAX_HEIGHT': 250,
```

Перезапустите контейнер `findface-cibr-findface-multi-legacy-1`.

```
sudo docker container restart findface-cibr-findface-multi-legacy-1
```

2.4.11 Пакетная загрузка карточек через консоль

Помимо *веб-интерфейса* для пакетной загрузки фотографий в картотеку можно использовать консольную утилиту `uploader.py`. Рекомендуется сделать выбор в пользу данной утилиты, если количество загружаемых фотографий превышает 10 000.

Совет: Для вызова справки `uploader.py` выполните команду:

```
docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/python3 /
↳ tigre-prototype/ffsecurity/uploader.py --help

Usage: uploader.py [OPTIONS] COMMAND [ARGS]...

Options:
--job FILE          Job file (default: enroll-job.db)
```

(continues on next page)

(продолжение с предыдущей страницы)

```
--log-level TEXT  Log level
--fsync BOOLEAN   Call fsync() to prevent data loss on power failure
--help           Show this message and exit.
```

Commands:

```
add    Add items from CSV or TSV file to job
print  Print contents of job file as JSON
run    Run upload job
```

```
docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/python3 /
↳tigre_prototype/ffsecurity/uploader.py add --help
```

Usage: uploader.py add [OPTIONS] FILES...

Options:

```
--format [csv|tsv]  Input file format - CSV or TSV
--delimiter TEXT    Field delimiter - by default it's "\t" for TSV and ","
                    for CSV
--help             Show this message and exit.
```

```
docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/python3 /
↳tigre_prototype/ffsecurity/uploader.py print --help
```

Usage: uploader.py print [OPTIONS]

Print contents of job file as JSON

Options:

```
--failed  Show only failed images
--noface  Show only images without detection
--help    Show this message and exit.
```

```
docker exec findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/python3 /
↳tigre_prototype/ffsecurity/uploader.py run --help
```

Usage: uploader.py run [OPTIONS]

Run upload job

Options:

```
--parallel INTEGER  Number of enroll threads (default: 10)
--api TEXT          API url (default: http://127.0.0.1:80/)
                    [required]
--user TEXT         API username [required]
--password TEXT     API password [required]
--watch-lists TEXT  Comma-separated list of card list ids [required]
--inactive          Mark new cards as inactive
--failed            Include failed images
--noface            Include images without detection
```

(continues on next page)

(продолжение с предыдущей страницы)

```
--all-faces           Enroll all found faces on each image
--detect-timeout INTEGER Request timeout for detect photos
--logging-delta INTEGER Logging period delta
--help               Show this message and exit.
```

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, либо сгенерировать его с помощью python или другой утилиты.

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 домашней директории (`/home/ubuntu` в примере) выполните следующую команду:

```
sudo docker run -it --rm --network host --volume ${PWD}:/home/ubuntu/create_cards --
↪ volume /home/ubuntu/photos:/home/ubuntu/photos docker.int.ntl/ntech/multi/multi/
↪ legacy:ffcibr-2.1.2 sh -c "cd /home/ubuntu/create_cards && /opt/findface-security/
↪ bin/python3 tsv_builder.py /home/ubuntu/photos"
```

где `/home/ubuntu/photos` путь с вашим фотографиями.

TSV файл может быть сгенерирован с помощью следующей команды:

```
python3 tsv_builder.py /home/ubuntu/photos/
```

Вы можете использовать файл `images.tsv` в качестве примера.

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

```
sudo docker run -it --rm --network host --volume ${PWD}:/home/ubuntu/create_cards
↪ docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 sh -c "cd /home/ubuntu/
↪ create_cards && /opt/findface-security/bin/python3 /tigre_prototype/ffsecurity/
↪ uploader.py add /home/ubuntu/create_cards/images.tsv"
```

Опции команды `add` утилиты:

- `--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, указав путь к фотографиям вместо (например, /home/ubuntu/photos) и передав необходимые аргументы:

```
sudo docker run -it --rm --network host --volume ${PWD}:/home/ubuntu/create_cards --
↪volume /home/ubuntu/photos:/home/ubuntu/photos docker.int.ntl/ntech/multi/multi/
↪legacy:ffcibr-2.1.2 sh -c "cd /home/ubuntu/create_cards && /opt/findface-security/
↪bin/python3 /tigre_prototype/ffsecurity/uploader.py run --user admin --password
↪password --watch-lists 1"
```

Опции команды run утилиты:

- **--parallel**: количество потоков загрузки фотографий, по умолчанию 10. Чем больше потоков, тем быстрее будет завершена загрузка, однако также потребуется и большее количество ресурсов.
 - **--api**: API URL, компонента findface-security, по умолчанию http://127.0.0.1:80/. Обязательная опция.
 - **--user**: имя пользователя. Обязательная опция.
 - **--password**: пароль. Обязательная опция.
 - **--watch-lists**: перечень разделенных запятой id списков наблюдения, в которые нужно добавить фотографии. Обязательная опция.
 - **--inactive**: отметить новые карточки как неактивные.
 - **--failed**: в случае неудачи при обработке job-файла исправьте ошибку и повторите попытку с данной опцией.
 - **--noface**: по умолчанию изображения, классифицированные как не имеющие лиц, получают статус NOFACE и автоматически исключаются из загрузки. Чтобы попытаться повторно обнаружить лица на таких изображениях, перезапустите job-задание с данной опцией. Если повторное обнаружение снова даст отрицательный результат, изображение будет пропущено, а в журнале загрузки появится соответствующая запись.
 - **--all-faces**: загрузить все лица, если на фотографии их несколько.
 - **--detect-timeout**: тайм-аут запросов на обнаружение фотографий.
 - **--logging-delta**: частота логирования статуса загрузки.
4. (Опционально) Выведите в формате JSON результаты обработки задания job. При необходимости можно вывести только изображения, которые не удалось загрузить, и изображения без обнаруженных лиц. Используйте следующую команду:

```
sudo docker run -it --rm --network host --volume ${PWD}:/home/ubuntu/create_cards --
↪volume /home/ubuntu/NTL_impersonated:/home/ubuntu/NTL_impersonated docker.int.ntl/
↪ntech/multi/multi/legacy:ffcibr-2.1.2 sh -c "cd /home/ubuntu/create_cards && /opt/
↪findface-security/bin/python3 /tigre_prototype/ffsecurity/uploader.py print --
↪noface --failed"
```

Опции команды print утилиты:

- **--failed**: вывести только изображения, которые не удалось загрузить.
- **--noface**: вывести только изображения без обнаруженных лиц.

2.5 Обслуживание и устранение неисправностей

2.5.1 Обновление до FindFace CIBR 2.1.2

Обновление FindFace CIBR 1.3 до FindFace CIBR 2.1.2

Для обновления FindFace CIBR 1.3 до версии FindFace CIBR 2.1.2 выполните следующие действия:

1. Создайте резервную копию старой схемы базы данных Tarantool:

```
sudo cp /etc/findface-security/tnt_schema.lua /etc/findface-security/old_tnt_schema.  
↪lua
```

Начиная с версии 2.1.1, одним из самых существенных отличий FindFace CIBR от предыдущих версий продукта является структура биометрической базы данных Tarantool (так называемая «метасхема»). В новой структуре выделено несколько пространств (spaces), тогда как в прежних версиях было только одно пространство (space) по умолчанию.

2. Откройте файл конфигурации `/etc/findface-security/config.py`. Сохраните значения параметров `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN`, `ROUTER_URL`, `CUSTOM_FIELDS` для последующего использования.

```
sudo vi /etc/findface-security/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',
  ...
  # -- Custom model fields --
  # Edit CUSTOM_FIELDS -> `human_card` section to customize human card fields.
  # Edit CUSTOM_FIELDS -> `face_object` section to customize face object fields.
  # Below is an example with every field type possible.
  # 'CUSTOM_FIELDS': {
  #     'human_card': {
  #         'items': [
  #             {
  #                 'name': 'personid',
  #                 'default': '',
  #                 'label': 'PersonID',
  #                 'display': ['list', 'form'],
  #                 'description': 'Sigur person ID',
  #                 'editable': False
  #             },
  #             {
  #                 'name': 'firstname',
  #                 'default': '',
  #                 'label': 'First Name',
  #                 'display': ['list', 'form'],
```

(continues on next page)

(продолжение с предыдущей страницы)

```

#         'description': 'Sigur first name',
#         'editable': False
#     },
#     {
#         'name': 'lastname',
#         'default': '',
#         'label': 'Last Name',
#         'display': ['list', 'form'],
#         'description': 'Sigur last name',
#         'editable': False
#     },
#     {
#         'name': 'version',
#         'default': '',
#         'label': 'Version',
#         'display': ['list', 'form'],
#         'description': 'Sigur photo version',
#         'editable': False
#     }
# ],
# 'filters': [
#     {
#         'name': 'personid',
#         'label': 'Sigur person ID filter',
#         'field': 'personid'
#     }
# ]
# },
# 'face_object': {
#     'items': [
#         {
#             "field_name": "tag_name_1",
#             "type": "string",
#             "default": "change_me"
#         },
#         {
#             "field_name": "tag_name_2",
#             "type": "uint",
#             "default": 123
#         },
#         {
#             "field_name": "tag_name_3",
#             "type": "bool",
#             "default": True
#         }
#     ]
# }
# },
# }

```

3. Остановите сервис `findface-security`.

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

4. Создайте резервную копию базы данных векторов признаков на основе Tarantool в любой выбранной директории, например, /etc/findface_dump.

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

5. Перед установкой новой версии остановите и отключите все сервисы во избежание конфликта портов.

Примечание: В примере ниже восемь шардов. Если количество шардов в вашей системе отличается от количества шардов в примере, измените приведенную ниже команду соответствующим образом. Например, для системы с шестнадцатью шардами замените `tarantool@shard-00{1..8}.service` на `tarantool@shard-00{01..16}.service`. Список активных шардов можно посмотреть с помощью команды `ls /etc/tarantool/instances.enabled/`.

Можно остановить и отключить сервисы один за другим:

```
sudo systemctl stop postgresql.service
sudo systemctl stop postgresql@10-main
sudo systemctl stop findface-*.service
sudo systemctl stop pgbouncer.service
sudo systemctl stop tarantool@shard-00{1..8}.service
sudo systemctl stop nats-server.service
sudo systemctl stop etcd.service
sudo systemctl stop mongod.service
sudo systemctl stop mongod.service
sudo systemctl stop memcached.service
sudo systemctl stop nginx.service

sudo systemctl disable postgresql.service
sudo systemctl disable postgresql@10-main
sudo systemctl disable pgbouncer.service
sudo systemctl disable findface-extraction-api.service
sudo systemctl disable findface-security.service
sudo systemctl disable findface-security-onvif.service
sudo systemctl disable findface-sf-api.service
sudo systemctl disable findface-ntls.service
sudo systemctl disable findface-video-manager.service
sudo systemctl disable findface-video-worker-cpu.service
sudo systemctl disable findface-video-worker-gpu.service
sudo systemctl disable findface-counter.service
sudo systemctl disable findface-liveness-api.service
sudo systemctl disable findface-video-streamer-cpu.service
sudo systemctl disable findface-video-streamer-gpu.service
sudo systemctl disable findface-video-storage.service
sudo systemctl disable tarantool@shard-00{1..8}.service
sudo systemctl disable nats-server.service
sudo systemctl disable etcd.service
sudo systemctl disable mongod.service
```

(continues on next page)

(продолжение с предыдущей страницы)

```
sudo systemctl disable mongodb.service
sudo systemctl disable memcached.service
sudo systemctl disable nginx.service
```

Или использовать более компактные команды:

```
sudo systemctl stop postgresql.service postgresql@10-main findface-*.service
↳ pgbouncer.service tarantool@shard-00{1..8}.service nats-server.service etcd.
↳ service mongod.service mongodb.service memcached.service nginx.service

sudo systemctl disable postgresql.service postgresql@10-main pgbouncer.service
↳ findface-extraction-api.service findface-security.service findface-security-onvif.
↳ service findface-sf-api.service findface-ntls.service findface-video-manager.
↳ service findface-video-worker-cpu.service findface-video-worker-gpu.service
↳ findface-counter.service findface-liveness-api.service findface-video-streamer-
↳ cpu.service findface-video-streamer-gpu.service findface-video-storage.service
↳ tarantool@shard-00{1..8}.service nats-server.service etcd.service mongod.service
↳ mongodb.service memcached.service nginx.service
```

6. *Установите* экземпляр FindFace CIBR 2.1.2. Не забудьте предварительно подготовить сервер:

См.:

- *Подготовка сервера Ubuntu*
- *Подготовка сервера CentOS 7*
- *Подготовка сервера Debian 11*

7. После установки продукта остановите все контейнеры FindFace CIBR из директории `/opt/findface-cibr/`.

```
cd /opt/findface-cibr/

sudo docker-compose stop
```

8. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` и вставьте сохраненные на шаге №2 значения для параметров `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN`, `ROUTER_URL`, и `CUSTOM_FIELDS`.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py
...
# 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)
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# or findface-facerouter url (in some specific cases)
'ROUTER_URL': 'http://127.0.0.1:80',
```

9. Откройте старую версию файла конфигурации `findface-ntls`, доступную по адресу `/etc/findface-ntls.cfg`, и сравните ее с новой версией `/opt/findface-cibr/configs/findface-ntls/findface-ntls.yaml`. Перенесите все пользовательские параметры из старой версии в новую. Сделайте то же самое для других компонентов, например, для `findface-extract-api` проверьте `/etc/findface-extract-api.ini` на соответствие `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml`, для `findface-sf-api` проверьте `/etc/findface-sf-api.ini` на соответствие `/opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml` и т. д.

```
sudo vi /etc/findface-ntls.cfg
sudo vi /opt/findface-cibr/configs/findface-ntls/findface-ntls.yaml
sudo vi /etc/findface-extraction-api.ini
sudo vi /opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.
↪yaml
sudo vi /etc/findface-sf-api.ini
sudo vi /opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml
```

Важно: Правила переноса параметров из старого файла конфигурации в новый:

- Если параметр в старом файле конфигурации имел пустое значение, а в новом файле конфигурации он заполнен – сотрите это значение.
- Оставьте как есть параметры, не включенные в старый файл конфигурации, но присутствующие в новом файле конфигурации.

10. Измените структуру базы данных Tarantool, применив схему `tnt_schema.lua` из версии Findface CIBR 2.1.2.

```
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/findface-
↪multi-legacy/findface-multi-legacy.py:/etc/findface-security/config.py:ro' docker.
↪int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 make-tnt-schema | sudo tee /etc/
↪findface-security/tnt_schema.lua
```

11. Удалите данные из каталогов, соответствующих активным шардам.

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

12. Скопируйте метасхему пространства (space) `face_clusters_space` из старого файла конфигурации `old_tnt_schema.lua` в новый файл `tnt_schema.lua`. Это можно сделать следующим способом:

- 12.1. В новом файле конфигурации `/etc/findface-security/tnt_schema.lua` замените строку в начале файла:

```
cfg_spaces = {
```

на следующую:

```
spaces = {
```

12.2. Скопируйте и добавьте из старого файла конфигурации `old_tnt_schema.lua` пространство (space) `face_clusters_space`:

```
face_clusters_space = {
  meta_scheme = {
    -- <class 'ffsecurity.entities.cluster.face.models.FaceCluster'>.
    ↪normalized_id:
    {
      default = '',
      field_type = 'string',
      id = 1,
      name = 'normalized_id',
    },
    -- <class 'ffsecurity.entities.cluster.face.models.FaceCluster'>.feat:
    {
      default = '',
      field_type = 'string',
      id = 2,
      name = 'feat',
    },
    -- <class 'ffsecurity.entities.cluster.face.models.FaceCluster'>.
    ↪m:nonnormalized_emben:
    {
      default = '',
      field_type = 'string',
      id = 3,
      name = 'm:nonnormalized_emben',
    },
    -- <class 'ffsecurity.entities.cluster.face.models.FaceCluster'>.
    ↪m:case:
    {
      default = '0',
      field_type = 'unsigned',
      id = 4,
      name = 'm:case',
    },
  },
  meta_indexes = {'m:case'}
},
```

13. Удалите генерируемый пакетом `findface-tarantool-server` конфигурационный файл по умолчанию `FindFace.lua` (он мешает сделать перезагрузку).

```
sudo rm -rf /etc/tarantool/instances*/FindFace.lua
```

14. 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
```

После выполненных операций в шардах по-прежнему останутся старые галереи, созданные в пространстве `default`. При этом станут доступны новые пространства (например, `ffsec_face_case_events`, `ffsec_face_case_clusters` и т. д.).

15. Восстановите в базе старые данные из резервной копии. Данные будут восстановлены в том виде,

в котором они существовали в предыдущей версии: все галереи останутся в пространстве `default`.

```
sudo systemctl start findface-ntls.service
cd /etc/findface_dump
for x in *.json; do sudo findface-storage-api-restore -config /etc/findface-sf-api.
↪ini < "$x"; done
```

16. Включите старые сервисы `findface-sf-api.service`, `nginx.service`, они понадобятся при миграции `postgresql`.

```
sudo systemctl start findface-sf-api.service
sudo systemctl start nginx.service
```

17. Выполните миграции базы данных PostgreSQL для совместимости с FindFace CIBR 2.1.2. Сделайте следующее:

17.1. Перейдите в файл конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`. В секции `DATABASES` -> `default` временно поменяйте пароль `PASSWORD` на старый, используемый в конфигурационном файле `/etc/findface-security/config.py`.

Важно: Обязательно сохраните пароль из файла конфигурации `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py`. Позже он понадобится.

```
sudo vi /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-
↪legacy.py

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'DISABLE_SERVER_SIDE_CURSORS': True,
        'NAME': 'ffsecurity', 'HOST': '127.0.0.1', 'PORT': 5439, 'USER':
↪'ntech', 'PASSWORD': 'XXXXXXXXXXXXXXXXXX'
    }
}
```

- 17.2. Выполните миграции базы данных на хосте:

```
sudo systemctl start postgresql.service
sudo systemctl start pgbouncer.service
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/
↪findface-multi-legacy/findface-multi-legacy.py:/etc/findface-security/
↪config.py:ro' docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 /opt/
↪findface-security/bin/python3 /tigre_prototype/manage.py migrate
```

- 17.3. Сделайте резервное копирование существующей базы данных с PostgreSQL, установленной на хост-системе.

```
cd /opt/findface-cibr/
sudo -u postgres pg_dump --verbose --disable-triggers ffsecurity | sudo tee
↪dump_ffsecurity.sql
sudo -u postgres pg_dump -t auth_group -t ffsecurity_adgroupguid -t
↪ffsecurity_deviceblacklistrecord -t ffsecurity_ffsecauthsession -t
```

(continues on next page)

(продолжение с предыдущей страницы)

```

↪ffsecurity_grouppermission -t ffsecurity_runtimesetting -t ffsecurity_
↪user -t ffsecurity_user_groups -t ffsecurity_user_user_permissions -t_
↪ffsecurity_userkeyvalue -t Knox_authtoken -t ffsecurity_
↪watchlistpermission -t ffsecurity_cameragrouppermission -t ffsecurity_
↪casepermission --data-only --verbose --no-acl --no-owner --disable-
↪triggers ffsecurity | sudo tee dump_identity_provider.sql

```

17.4. Сделайте резервное копирование прав доступа ролей.

```

sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/
↪findface-multi-legacy/findface-multi-legacy.py:/etc/findface-security/
↪config.py:ro' docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 /opt/
↪findface-security/bin/python3 /tigre_prototype/manage.py dump_permissions_
↪| sudo tee permissions.csv

```

17.5. Поменяйте обратно пароль, измененный на шаге №17.1

17.6. Остановите все сервисы.

```

sudo systemctl stop findface-sf-api.service nginx.service tarantool@shard-00
↪{1..8}.service postgresql.service pgbouncer.service

```

17.7. Откройте файл /opt/findface-cibr/docker-compose.yaml и сохраните значение POSTGRES_PASSWORD для использования в последующих командах.

17.8. Пересоздайте базу данных ffsecurity, чтобы очистить её от данных по умолчанию. Вставьте значение {POSTGRES_PASSWORD}, которое вы ранее скопировали в шаге №17.7, в команду ниже:

```

sudo docker-compose up -d postgresql
sudo docker exec -it -u postgres findface-cibr-postgresql-1 /bin/bash -c
↪"PGPASSWORD={POSTGRES_PASSWORD} dropdb ffsecurity"
sudo docker exec -it -u postgres findface-cibr-postgresql-1 /bin/bash -c
↪"PGPASSWORD={POSTGRES_PASSWORD} createdb -O ntech --encoding=UTF-8 --lc-
↪collate=C.UTF-8 --lc-ctype=C.UTF-8 --template=template0 ffsecurity"

```

17.9. Восстановите данные в только что созданную базу ffsecurity. Вставьте значение {POSTGRES_PASSWORD}, которое вы ранее скопировали в шаге №17.7, в команду ниже:

```

sudo docker exec -i findface-cibr-postgresql-1 /bin/bash -c "PGPASSWORD=
↪{POSTGRES_PASSWORD} psql --username postgres ffsecurity" < dump_
↪ffsecurity.sql

```

17.10. Пересоздайте базу данных ffsecurity_identity_provider, чтобы очистить её от данных по умолчанию. Вставьте значение {POSTGRES_PASSWORD}, которое вы ранее скопировали в шаге №17.7, в команду ниже:

```

sudo docker exec -it -u postgres findface-cibr-postgresql-1 /bin/bash -c
↪"PGPASSWORD={POSTGRES_PASSWORD} dropdb ffsecurity_identity_provider"
sudo docker exec -it -u postgres findface-cibr-postgresql-1 /bin/bash -c
↪"PGPASSWORD={POSTGRES_PASSWORD} createdb -O ntech --encoding=UTF-8 --lc-
↪collate=C.UTF-8 --lc-ctype=C.UTF-8 --template=template0 ffsecurity_
↪identity_provider"

```

17.11. Запустите миграции.

```
sudo docker-compose up -d pgbouncer
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/
↪findface-multi-identity-provider/findface-multi-identity-provider.py:/etc/
↪findface-security/config.py:ro' docker.int.ntl/ntech/multi/multi/identity-
↪provider:ffcibr-2.1.2 /opt/findface-security/bin/python3 /tigre_prototype/
↪manage.py migrate
```

17.12. Восстановите данные в базу ffsecurity_identity_provider. Вставьте значение {POSTGRES_PASSWORD}, которое вы ранее скопировали в шаге №17.7, в команду ниже:

```
sudo docker exec -i findface-cibr-postgresql-1 /bin/bash -c "PGPASSWORD=
↪{POSTGRES_PASSWORD} psql --username postgres ffsecurity_identity_provider
↪" < dump_identity_provider.sql
```

17.13. Запустите все сервисы.

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

17.14. Запустите команду создания записей в таблице outbox для списков наблюдения, групп камер и происшествий. Вставьте значение {POSTGRES_PASSWORD}, которое вы ранее скопировали в шаге №17.7, в команду ниже:

```
sudo docker exec -i findface-cibr-postgresql-1 /bin/bash -c "PGPASSWORD=
↪{POSTGRES_PASSWORD} pg_dump --username postgres -f permissions.sql -t
↪ffsecurity_cameragrouppermission -t ffsecurity_watchlistpermission -t
↪ffsecurity_casepermission --data-only ffsecurity_identity_provider"
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/
↪findface-multi-legacy/findface-multi-legacy.py:/etc/findface-security/
↪config.py:ro' docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 /opt/
↪findface-security/bin/python3 /tigre_prototype/manage.py export_to_outbox
↪--watchlists --cameragroups --cases
sudo docker exec -i findface-cibr-postgresql-1 /bin/bash -c "PGPASSWORD=
↪{POSTGRES_PASSWORD} psql --username postgres -c 'TRUNCATE ffsecurity_
↪cameragrouppermission, ffsecurity_watchlistpermission, ffsecurity_
↪casepermission RESTART IDENTITY;' ffsecurity_identity_provider"
sudo docker exec -i findface-cibr-postgresql-1 /bin/bash -c "PGPASSWORD=
↪{POSTGRES_PASSWORD} psql --username postgres ffsecurity_identity_provider
↪< permissions.sql"
```

17.15. Прежде чем восстановить права доступа ролей в сервис identity_provider, проверьте файл /opt/findface-cibr/permissions.csv. Обязательно замените *_ffsecAuthToken на *_authToken, если таковой имеется. Чаще всего это применимо к тем случаям, когда установка FindFace CIBR 1.3 представляла собой обновление с более ранних версий продукта.

После этого восстановите права доступа ролей в сервис identity_provider.

```
sudo docker run --rm --network host -v /opt/findface-cibr/configs/findface-
↪multi-identity-provider/findface-multi-identity-provider.py:/etc/findface-
↪security/config.py:ro -v $(pwd)/permissions.csv:/var/permissions.csv:ro
↪docker.int.ntl/ntech/multi/multi/identity-provider:ffcibr-2.1.2 /opt/
↪findface-security/bin/python3 /tigre_prototype/manage.py load_permissions
↪/var/permissions.csv
```

17.16. Скопируйте полные кадры, нормализованные изображения и лицензионный файл. Скопируйте файлы из папки /opt/ntech/license/ в папку /opt/findface-cibr/data/

findface-ntls/, из /var/lib/findface-security/uploads/ в /opt/findface-cibr/data/findface-multi-legacy/uploads/, из /var/lib/ffupload/uploads/ в /opt/findface-cibr/data/findface-upload/uploads/.

```
sudo cp -r /opt/ntech/license/* /opt/findface-cibr/data/findface-ntls/
sudo cp -r /var/lib/findface-security/uploads/* /opt/findface-cibr/data/
↪findface-multi-legacy/uploads/
sudo cp -r /var/lib/ffupload/uploads/* /opt/findface-cibr/data/findface-
↪upload/uploads/
sudo chmod 777 -R /opt/findface-cibr/data/findface-upload/uploads/
sudo chown www-data:www-data -R /opt/findface-cibr/data/findface-upload/
↪uploads/*
```

Также можно напрямую смонтировать указанные выше папки в соответствующие контейнеры через файл docker-compose.yaml, как показано в примере ниже:

```
sudo vi /opt/findface-cibr/docker-compose.yaml

...
findface-upload:
  image: docker.int.ntl/ntech/universe/upload:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  volumes: ['./configs/findface-upload/40-ffupload.sh:/docker-entrypoint.d/
↪40-ffupload.sh:ro',
            '/var/lib/ffupload:/var/lib/ffupload']
...
findface-multi-identity-provider:
  depends_on: [pgbouncer, nats, findface-sf-api, findface-liveness-api,
↪etcd]
  environment: {ADMIN_PASSWORD: <ADMIN_PASSWORD>}
  image: docker.int.ntl/ntech/multi/multi/identity-provider:ffcibr-2.1.2
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  volumes: ['./configs/findface-multi-identity-provider/findface-multi-
↪identity-provider.py:/etc/findface-security/config.py:ro',
            '/var/lib/findface-security/uploads:/var/lib/findface-security/uploads']
...
findface-multi-legacy:
  depends_on: [pgbouncer, nats, findface-sf-api, findface-counter, findface-
↪liveness-api,
            etcd, findface-multi-identity-provider]
  environment: {ADMIN_PASSWORD: <ADMIN_PASSWORD>}
  image: docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
  volumes: ['./configs/findface-multi-legacy/findface-multi-legacy.py:/etc/
↪findface-security/config.py:ro',
            '/var/lib/findface-security/uploads:/var/lib/findface-security/uploads']
...
findface-multi-ui:
```

(continues on next page)

(продолжение с предыдущей страницы)

```

depends_on: [findface-multi-legacy]
image: docker.int.ntl/ntech/multi/multi/ui-cibr:ffcibr-2.1.2
logging: {driver: journald}
network_mode: service:pause
restart: always
volumes: ['. /configs/findface-multi-ui/nginx-site.conf:/etc/nginx/conf.d/
↪default.conf:ro',
  '/var/lib/findface-security/uploads:/var/lib/findface-security/uploads']
...
cleaner:
  command: [run-cleaner-service]
depends_on: [pgbouncer, findface-sf-api, findface-multi-legacy]
image: docker.int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2
logging: {driver: journald}
network_mode: service:pause
restart: always
volumes: ['. /configs/findface-multi-legacy/findface-multi-legacy.py:/etc/
↪findface-security/config.py:ro',
  '/var/lib/findface-security/uploads:/var/lib/findface-security/uploads']
...

```

17.17. Чтобы перенести данные Tarantool, выполните следующие действия:

Остановите все контейнеры FindFace CIBR:

```
sudo docker-compose down
```

Запустите снова старые шарды и сервис findface-sf-api :

```
sudo systemctl start tarantool@shard-00{1..8}.service findface-sf-api.
↪service
```

Создайте новую резервную копию базы данных векторов признаков:

```
sudo mkdir -p /etc/findface_dump_final
sudo findface-storage-api-dump -output-dir=/etc/findface_dump_final -config
↪/etc/findface-sf-api.ini
```

Остановите остальные сервисы, очистите директорию instances.enabled, снова запустите контейнеры и выполните storage-api-restore:

```
sudo systemctl stop tarantool@shard-00{1..8}.service findface-sf-api.
↪service findface-ntls.service
sudo rm /etc/tarantool/instances.enabled/*
sudo docker-compose up -d
sudo findface-storage-api-restore -config /opt/findface-cibr/configs/
↪findface-sf-api/findface-sf-api.yaml /etc/findface_dump_final/*.json
```

Обновление завершено. В новой версии присутствуют новые *модели нейронных сетей*, но миграция векторов признаков на другую модель нейронной сети не требуется. Можно использовать старые нейронные сети, перенеся их из директории /usr/share/findface-data/models/ в директорию /opt/findface-cibr/models/, указав их в файле /opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml, если они включены в FindFace CIBR 2.1.2.

Обратите внимание, что модели обнаружения лиц хранятся в директории `/usr/share/findface-data/models/facedet/`. В то время как в FindFace CIBR 2.1.2 все модели обнаружения лиц можно найти в директории `/opt/findface-cibr/models/detector/`. При переносе старых моделей нейронных сетей из директории `/usr/share/findface-data/models/` в директорию `/opt/findface-cibr/models/` поместите все модели обнаружения лиц (`facedet`) в директорию `/opt/findface-cibr/models/detector/`.

См. для справки:

```
$ ls -lash /usr/share/findface-data/models
total 52K
4.0K drwxr-xr-x  2 root root 4.0K Jul 15 14:48 face
4.0K drwxr-xr-x  2 root root 4.0K Jul 15 14:48 faceattr
4.0K drwxr-xr-x  2 root root 4.0K Jul 15 14:48 facedet
4.0K drwxr-xr-x  2 root root 4.0K Jul 15 14:48 facenorm

$ ls -lash /opt/findface-cibr/models/
total 44K
4.0K drwxr-xr-x  2 root root 4.0K Jul 17 16:20 detector
4.0K drwxr-xr-x  2 root root 4.0K Jul 17 16:24 face
4.0K drwxr-xr-x  2 root root 4.0K Jul 17 16:24 faceattr
4.0K drwxr-xr-x  2 root root 4.0K Jul 17 13:37 facenorm
```

Важно: Настоятельно рекомендуется отключить автоматическое обновление Ubuntu, чтобы сохранить совместимость FindFace CIBR со средой установки. В этом случае вы сможете обновлять ОС вручную, контролируя процесс обновления отдельных пакетов.

Для отключения автоматического обновления Ubuntu выполните следующие команды:

```
sudo apt-get remove unattended-upgrades
sudo systemctl stop apt-daily.timer
sudo systemctl disable apt-daily.timer
sudo systemctl disable apt-daily.service
sudo systemctl daemon-reload
```

Обновление FindFace CIBR 2.1.1 до FindFace CIBR 2.1.2

Для обновления FindFace CIBR 2.1.1 до версии FindFace CIBR 2.1.2 выполните следующие действия:

1. В домашней директории пользователя создайте папку для резервного копирования данных FindFace CIBR 2.1.1.

```
mkdir ~/backup_data/
```

2. Создайте резервную копию конфигурационных файлов FindFace CIBR и файла `docker-compose.yml`, а затем перейдите в директорию `/opt/findface-cibr/`.

```
sudo tar -cvzf ~/backup_data/configs.tar.gz -C /opt/findface-cibr/ configs
sudo cp /opt/findface-cibr/docker-compose.yml ~/backup_data/

cd /opt/findface-cibr/
```

3. Сделайте резервное копирование данных PostgreSQL.

```
sudo docker-compose cp postgresql:/bitnami/postgresql ~/backup_data/pg_bkp
```

4. Сделайте резервное копирование данных Tarantool.

```
sudo docker exec -it findface-cibr-findface-sf-api-1 bash -c "mkdir ffmulti_dump;
↪cd ffmulti_dump && /storage-api-dump -config /etc/findface-sf-api.ini"
sudo docker cp findface-cibr-findface-sf-api-1:/ffmulti_dump ~/backup_data/
```

5. Для папки `backup_data/`, расположенной в домашней директории пользователя, выполните команду `ls`, чтобы просмотреть содержимое папки. Оно должно быть таким.

```
ls ~/backup_data/

configs.tar.gz  docker-compose.yaml  ffmulti_dump  pg_bkp
```

6. Остановите все контейнеры FindFace CIBR.

```
sudo docker-compose down
```

7. Удалите папки `/opt/findface-cibr/data/findface-tarantool-server/` и `/opt/findface-cibr/data/postgresql/`.

```
sudo rm -r /opt/findface-cibr/data/findface-tarantool-server/

sudo rm -r /opt/findface-cibr/data/postgresql/
```

8. *Установите* экземпляр FindFace CIBR 2.1.2 в директорию `/opt/findface-cibr`.

9. Восстановите данные Tarantool из резервной копии.

1. Скопируйте папку `~/backup_data/ffmulti_dump/` в контейнер `findface-cibr-findface-sf-api-1`.

```
sudo docker cp ~/backup_data/ffmulti_dump/ findface-cibr-findface-sf-api-1:/
```

2. Восстановите данные Tarantool.

```
sudo docker exec -it findface-cibr-findface-sf-api-1 bash -c 'cd ffmulti_
↪dump && for x in *.json; do /storage-api-restore -config /etc/findface-sf-
↪api.ini < "$x"; done;'
```

10. В новых конфигурационных файлах замените пароль для NTECH USER на тот, который использовался в FindFace CIBR 2.1.1. Старый пароль можно получить из файла `configs/postgresql/40-init.sql`, находящегося в архиве резервной копии `configs.tar.gz`. Пароль NTECH USER должен быть заменен в следующих конфигурационных файлах:

- `findface-multi-legacy.py`
- `findface-multi-identity-provider.py`
- `findface-multi-audit.py`
- `pgbouncer -> userlist.txt`
- `postgresql -> 40-init.sql`
- `findface-counter.yaml`

11. В новом конфигурационном файле `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` замените значения `SECRET_KEY` в секции `GENERAL SETTINGS` и `'VIDEO_DETECTOR_TOKEN'` в секции `FINDFACE SECURITY SETTINGS` на те, которые использовались в FindFace CIBR 2.1.1. Старые значения можно получить из файла `configs/findface-multi-legacy/findface-multi-legacy.py`, находящегося в архиве резервной копии `configs.tar.gz`.
12. Сравните конфигурационные файлы версии 2.1.1 с файлами версии 2.1.2 для сервисов `findface-ntls`, `findface-extract-api`, `findface-sf-api` и т.д. Перенесите все пользовательские параметры из старой версии в новую. Возможно, потребуется внести изменения в файл `docker-compose.yaml`, если ранее вы добавляли или удаляли сервисы или вносили какие-либо другие изменения.

Важно: Правила переноса параметров из старого файла конфигурации в новый:

- Если параметр в старом файле конфигурации имел пустое значение, а в новом файле конфигурации он заполнен – сотрите это значение.
 - Оставьте как есть параметры, не включенные в старый файл конфигурации, но присутствующие в новом файле конфигурации.
-

13. Остановите все контейнеры FindFace CIBR из директории `/opt/findface-cibr/`.

```
cd /opt/findface-cibr
sudo docker-compose down
```

14. Скопируйте данные из директории `~/backup_data/pg_bkp/data/` в директорию `/opt/findface-cibr/data/postgresql`.

```
sudo cp -r ~/backup_data/pg_bkp/data/ /opt/findface-cibr/data/postgresql
```

15. Запустите все контейнеры FindFace CIBR.

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

2.5.2 Резервное копирование и восстановление FindFace CIBR и его данных

Вы можете создать резервную копию FindFace CIBR перед его удалением, чтобы позже восстановить продукт и его данные.

В этом разделе:

- *Резервное копирование FindFace CIBR*
- *Восстановление FindFace CIBR и его данных*

Резервное копирование FindFace CIBR

Чтобы создать резервную копию вашего экземпляра FindFace CIBR и его данных, выполните следующие команды:

```
sudo tar -cvzf ~/configs.tar.gz -C /opt/findface-cibr/ configs
sudo tar -cvzf ~/data.tar.gz -C /opt/findface-cibr/ data
sudo cp /opt/findface-cibr/docker-compose.yaml ~/
```

Восстановление FindFace CIBR и его данных

Чтобы восстановить FindFace CIBR и его данные из резервной копии, выполните следующие действия:

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

Примечание: Вместо `findface-*` укажите актуальное имя файла.

```
chmod +x findface-*.run
```

4. Запустите файл `.run`.

```
sudo ./findface-*.run
```

5. Пройдите процесс установки, как описано *здесь*.
6. Чтобы восстановить FindFace CIBR, его данные и конфигурационные файлы из резервной копии, после завершения установки остановите все контейнеры FindFace CIBR.

```
cd /opt/findface-cibr
sudo docker-compose stop
```

7. Удалите новые конфигурационные файлы и данные, сформированные и созданные инсталлятором, и восстановите их из резервной копии.

Важно: Сравните старый файл `~/docker-compose.yaml`, который вы сохранили в ходе резервного копирования, с новым файлом `/opt/findface-cibr/docker-compose.yaml`. Возможно, потребуется внести изменения в старый файл `~/docker-compose.yaml`, если ранее вы добавляли / удаляли сервисы или вносили какие-либо другие изменения. Вы должны полностью осознавать, что делаете, меняя новый файл `docker-compose.yaml` на старый.

```
sudo rm -r /opt/findface-cibr/configs/*
sudo tar -xvf ~/configs.tar.gz -C /opt/findface-cibr/
sudo rm -r /opt/findface-cibr/data/*
sudo tar -xvf ~/data.tar.gz -C /opt/findface-cibr/
sudo cp ~/docker-compose.yaml /opt/findface-cibr/
```

8. Перезапустите контейнеры FindFace CIBR.

```
cd /opt/findface-cibr
sudo docker-compose up -d
```

2.5.3 Миграция векторов признаков на другую модель нейронной сети

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

Важно: В случае если вы выполняете миграцию как часть обновления системы до новой версии, выполните сначала *обновление*. Только после этого приступайте к миграции.

Этот раздел посвящен миграции векторов признаков объектов на другую модель нейронной сети.

Выполните следующие действия:

1. Создайте резервную копию базы данных векторов признаков на основе Tarantool в любой выбранной директории, например, `/etc/ffcibr_dump`.

```
sudo docker exec -it findface-cibr-findface-sf-api-1 bash -c "mkdir ffcibr_dump; cd \
↪ffcibr_dump && /storage-api-dump -config /etc/findface-sf-api.ini"
sudo docker cp findface-cibr-findface-sf-api-1:/ffcibr_dump /etc
```

2. Создайте новые шарды, в которых будут храниться заново сгенерированные векторы признаков.
 1. Перейдите в директорию `/opt/findface-cibr/data/findface-tarantool-server` и подсчитайте количество шардов по количеству директорий.

Примечание: В примере ниже восемь шардов.

```
cd /opt/findface-cibr/data/findface-tarantool-server

ls -l

shard-001
shard-002
shard-003
shard-004
shard-005
shard-006
shard-007
shard-008
```

2. Создайте директории, в которых будут храниться файлы новых шардов.

```
sudo mkdir -p shard-01{1..8}/{index,snapshots,xlogs}
```

3. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml` и замените модель `extraction` на новую в параметре `face_emben`.

```

sudo vi /opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.
↪yaml

extractors:
  ...
  models:
    ...
    face_emben: face/<new_model_face>.cpu<gpu>.fnk
    ...

```

Перезапустите контейнер findface-cibr-findface-extraction-api-1.

```

cd /opt/findface-cibr/
sudo docker-compose restart findface-extraction-api

```

- В файле docker-compose.yaml создайте новые сервисы для каждого нового шарда. Для этого скопируйте существующий сервис и замените имя шарда на новое, порты в CFG_LISTEN_PORT, TT_LISTEN, а также путь к шарду в volumes.

```

sudo vi docker-compose.yaml

services:
  ...
  findface-tarantool-server-shard-**011**:
    depends_on: [findface-ntls]
    environment: {CFG_EXTRA_LUA: loadfile("/tnt_schema.lua")(), CFG_LISTEN_HOST: ↪
↪127.0.0.1,
    CFG_LISTEN_PORT: '8111', CFG_NTLS: '127.0.0.1:3133', TT_CHECKPOINT_COUNT: 3,
    TT_CHECKPOINT_INTERVAL: '14400', TT_FORCE_RECOVERY: 'true', TT_LISTEN: '127.0.
↪0.1:32011',
    TT_MEMTX_DIR: snapshots, TT_MEMTX_MEMORY: '2147483648', TT_WAL_DIR: xlogs, TT_
↪WORK_DIR: /var/lib/tarantool/FindFace}
    image: docker.int.ntl/ntech/universe/tntapi:ffserver-9.230407.1
    logging: {driver: journald}
    network_mode: service:pause
    restart: always
    volumes: ['./data/findface-tarantool-server/shard-**011*/var/lib/tarantool/
↪FindFace',
    './configs/findface-tarantool-server/tnt-schema.lua:/tnt_schema.lua:ro']
  ...

```

- Запустите новые шарды, подняв контейнеры.

```

sudo docker-compose up -d

```

- На основании приведенного ниже примера создайте файл конфигурации с настройками миграции migration.yaml.

```

sudo vi migration.yaml

extraction-api:
  timeouts:
    connect: 5s

```

(continues on next page)

(продолжение с предыдущей страницы)

```

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://127.0.0.1:8101/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: ''
  - 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/
    slave: ''
storage-api-to:
timeouts:
  connect: 5s
  response_header: 30s
  overall: 35s
  idle_connection: 10s
max-idle-conns-per-host: 20
shards:
  - master: http://127.0.0.1:8111/v2/
    slave: ''
  - master: http://127.0.0.1:8112/v2/
    slave: ''
  - master: http://127.0.0.1:8113/v2/
    slave: ''
  - master: http://127.0.0.1:8114/v2/
    slave: ''
  - master: http://127.0.0.1:8115/v2/
    slave: ''
  - master: http://127.0.0.1:8116/v2/
    slave: ''
  - master: http://127.0.0.1:8117/v2/
    slave: ''
  - master: http://127.0.0.1:8118/v2/
    slave: ''

```

(continues on next page)

(продолжение с предыдущей страницы)

```
workers_num: 3
objects_limit: 1000
extraction_batch_size: 8
normalized_storage:
  type: webdav
  enabled: True
  webdav:
    upload-url: http://127.0.0.1:3333/uploads/
  s3:
    endpoint: ''
    bucket-name: ''
    access-key: ''
    secret-access-key: ''
    secure: False
    region: ''
    public-url: ''
    operation-timeout: 30
```

В разделе `storage-api-from` укажите шарды, с которых осуществляется миграция.

```
storage-api-from: # current location of the gallery
...
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: ''
  - 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/
    slave: ''
...
```

В разделе `storage-api-to` укажите новые шарды, в которых будут храниться данные после миграции.

```
storage-api-to:
...
shards:
  - master: http://127.0.0.1:8111/v2/
    slave: ''
  - master: http://127.0.0.1:8112/v2/
    slave: ''
  - master: http://127.0.0.1:8113/v2/
```

(continues on next page)

(продолжение с предыдущей страницы)

```

slave: ''
- master: http://127.0.0.1:8114/v2/
  slave: ''
- master: http://127.0.0.1:8115/v2/
  slave: ''
- master: http://127.0.0.1:8116/v2/
  slave: ''
- master: http://127.0.0.1:8117/v2/
  slave: ''
- master: http://127.0.0.1:8118/v2/
  slave: ''
...

```

7. Скопируйте файл `migration.yaml` внутрь контейнера `findface-cibr-findface-sf-api-1`. Запустите утилиту `sf-api-migrate` с опцией `-config` и укажите файл конфигурации `migration.yaml`.

```

sudo docker cp migration.yaml findface-cibr-findface-sf-api-1:/
sudo docker exec findface-cibr-findface-sf-api-1 ./sf-api-migrate -config migration.
↪yaml

```

Примечание: Если в системе много событий и карточек, процесс миграции может занять значительное количество времени.

8. После завершения миграции удалите сервисы старых шардов из файла `docker-compose.yaml` и остановите используемые ими контейнеры.

```

sudo docker-compose up -d --remove-orphans

```

9. Откройте файл конфигурации `/opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml` и измените порты шардов в зависимости от новых настроек. Перезапустите контейнер `findface-cibr-findface-sf-api-1`.

```

sudo vi /opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml

storage-api:
  shards:
    - master: http://127.0.0.1:8111/v2/
      slave: ''
    - master: http://127.0.0.1:8112/v2/
      slave: ''
    - master: http://127.0.0.1:8113/v2/
      slave: ''
    - master: http://127.0.0.1:8114/v2/
      slave: ''
    - master: http://127.0.0.1:8115/v2/
      slave: ''
    - master: http://127.0.0.1:8116/v2/
      slave: ''
    - master: http://127.0.0.1:8117/v2/
      slave: ''
    - master: http://127.0.0.1:8118/v2/

```

(continues on next page)

(продолжение с предыдущей страницы)

```
slave: ''

sudo docker-compose restart findface-sf-api
```

10. Мигрируйте кластеры. Для этого выполните следующую команду:

```
sudo docker exec -it findface-cibr-findface-multi-legacy-1 /opt/findface-security/
↪bin/python3 /tigre_prototype/manage.py migrate_clusters --face --use-thumbnail
```

В результате система повторно сгенерирует векторы признаков для существующих кластерных событий и автоматически запустит кластеризацию для пересборки кластеров.

2.5.4 Изменение структуры базы данных векторов признаков

В некоторых случаях вам может потребоваться применить новую структурную схему к базе данных векторов признаков 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 CIBR структура базы данных задается через файл `/opt/findface-cibr/configs/findface-tarantool-server/tnt-schema.lua`.

Структура представляет собой набор пространств (**spaces**) и полей, каждое из которых описывается следующими параметрами:

- **id**: id поля;
- **name**: название поля, должно совпадать с названием соответствующего параметра объекта;
- **field_type**: тип данных (`unsigned|string|set[string]|set[unsigned]`);
- **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` доступен по [ссылке](#).

Изменение структуры

Для изменения структуры базы данных выполните следующие действия:

1. Создайте резервную копию базы данных векторов признаков на основе Tarantool в любой выбранной директории, например, /etc/ffcibr_dump.

```
sudo docker exec -it findface-cibr-findface-sf-api-1 bash -c "mkdir ffcibr_dump; cd ↵
↵ ffcibr_dump && /storage-api-dump -config /etc/findface-sf-api.ini"
sudo docker cp findface-cibr-findface-sf-api-1:/ffcibr_dump /etc
```

2. Измените структуру базы данных, добавив новую схему в файл `tnt-schema.lua`.

```
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/findface-
↵ multi-legacy/findface-multi-legacy.py:/etc/findface-security/config.py:ro' docker.
↵ int.ntl/ntech/multi/multi/legacy:ffcibr-2.1.2 make-tnt-schema | sudo tee /opt/
↵ findface-cibr/configs/findface-tarantool-server/tnt-schema.lua
```

3. Откройте файл `docker-compose.yaml` и убедитесь, что в каждом сервисе шарда Tarantool в разделе `environment` присутствует заполненная переменная `CFG_EXTRA_LUA: loadfile("/tnt_schema.lua")`.

```
sudo vi /opt/findface-cibr/docker-compose.yaml

...
findface-tarantool-server-shard-001:
  depends_on: [findface-ntls]
  environment: {CFG_EXTRA_LUA: loadfile("/tnt_schema.lua")()...
  ...
```

4. Удалите данные из каталогов, соответствующих активным шардам.

```
sudo rm /opt/findface-cibr/data/findface-tarantool-server/shard-*/{index,snapshots,
↵ xlogs}/*
```

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

```
docker restart findface-cibr-findface-tarantool-server-shard-001-1 findface-cibr-
↵ findface-tarantool-server-shard-002-1 findface-cibr-findface-tarantool-server-
↵ shard-003-1 findface-cibr-findface-tarantool-server-shard-004-1 findface-cibr-
↵ findface-tarantool-server-shard-005-1 findface-cibr-findface-tarantool-server-
↵ shard-006-1 findface-cibr-findface-tarantool-server-shard-007-1 findface-cibr-
↵ findface-tarantool-server-shard-008-1
```

6. Восстановите базу данных Tarantool из резервной копии.

```
sudo docker exec -it findface-cibr-findface-sf-api-1 bash -c 'cd ffcibr_dump && for ↵
↵ x in *.json; do /storage-api-restore -config /etc/findface-sf-api.ini < "$x"; ↵
↵ done;'
```

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

См.также:

Пользовательские метаданные в Tarantool

2.5.5 Проверка статуса компонентов

Проверьте статус контейнеров, если вы столкнулись с проблемой в системе, используя следующие команды:

```
docker ps
sudo docker container inspect <container_id>/<container_name>
sudo docker container stats <container_id>/<container_name>
```

2.5.6 Логирование

Просмотр логов — это первое, что нужно сделать, чтобы определить причину проблемы. По умолчанию процессы FindFace CIBR регистрируются в [логах контейнеров Docker](#), доступ к которым можно получить, используя команды `docker logs` и `docker service logs`. Помимо этого Docker использует [драйвер логирования в формате JSON](#), который кэширует логи контейнеров в данный формат. Вы можете настроить Docker на использование другого драйвера логирования, выбрав из [множества доступных механизмов логирования](#).

Данный раздел посвящен настройке Docker на использование драйвера логирования `journald`, который отправляет логи контейнеров в журнал `systemd`. В этом случае записи журнала извлекаются с помощью команды `journalctl`, через `journal API` или команды `docker logs`. Следуя приведенным ниже инструкциям, вы также можете настроить журнал `systemd`.

В этом разделе:

- [Настройка Journald](#)
- [Активация драйвера логирования journald](#)
- [Просмотр логов](#)

Настройка Journald

Для настройки сервиса `systemd-journal` выполните следующие действия:

1. Проверьте наличие директории `/var/log/journal`. Если таковая отсутствует, создайте ее следующей командой:

```
sudo mkdir /var/log/journal
sudo chmod 2755 /var/log/journal
```

2. Откройте файл конфигурации `/etc/systemd/journald.conf`. Включите сохранение логов `journald` на жестком диске, раскомментировав параметр `Storage` и изменив его значение на `persistent`. Также отключите фильтрацию `systemd-journal`:

```
sudo vi /etc/systemd/journald.conf

[Journal]
...
Storage=persistent
...
RateLimitInterval=0
RateLimitBurst=0
...
```

При необходимости раскомментируйте и измените значение параметра `SystemMaxUse`. Данный параметр определяет максимальный объем логов на жестком диске. Укажите его значение в байтах или используйте К, М, Г, Т, Р, Е в качестве единиц для указанного размера (соответственно 1024, 1024², ... байт).

```
...
SystemMaxUse=3G
```

3. Перезапустите сервис `journald`.

```
sudo systemctl restart systemd-journald.service
```

Активация драйвера логирования `journald`

Для того чтобы разрешить Docker использовать драйвер логирования `journald`, выполните следующие действия:

1. Добавьте следующую строку в файл конфигурации `/etc/docker/daemon.json`.

Совет: Данный файл может отсутствовать в вашей системе. Проверьте, существует ли он, и если нет, сначала создайте каталог `/etc/docker`, а затем файл.

```
sudo mkdir /etc/docker
sudo touch /etc/docker/daemon.json
```

```
sudo vi /etc/docker/daemon.json

{
  "log-driver": "journald"
}
```

2. Остановите все контейнеры Docker.

```
sudo docker stop $(sudo docker ps -a -q)
```

3. Перезапустите сервис Docker.

```
sudo systemctl restart docker
```

4. Запустите все контейнеры Docker.

```
sudo docker start $(sudo docker ps -a -q)
```

Просмотр логов

Используйте любой удобный способ работы с логами `journald`. Мы рекомендуем начать со следующих команд:

- Отображение всех логов:

```
journalctl -fa
```

- Отображение логов по ID контейнера:

```
journalctl CONTAINER_ID=<container_id> -f
```

- Отображение логов по имени контейнера:

```
journalctl CONTAINER_NAME=<container-name> -f
```

См.также:

Журнал действий пользователей

2.5.7 Устранение неполадок с лицензированием и `findface-ntls`

При устранении неполадок с лицензией и сервером `findface-ntls` (см. *Лицензирование*) первым шагом является получение информации о лицензии и статусе сервера. Это можно сделать, отправив API-запрос в `findface-ntls`. Действия по устранению неполадок предпринимаются с учетом содержания 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 CIBR 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.

- $(120; \infty)$ — не удастся получить ответ от источника лицензирования в течение длительного времени. Необходимо вмешательство.
- "valid" -> "value": 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
    },
    {

```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "type": "boolean",
    "name": "emotions",
    "value": true
  },
  {
    "type": "boolean",
    "name": "fast-index",
    "value": true
  },
  {
    "type": "boolean",
    "name": "sec-genetec",
    "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"
  }
]
}
```

2.5.8 Ручная очистка базы данных от старых данных

Для ручного удаления старых данных из базы данных FindFace CIBR используйте утилиту `cleanup`. Вы можете выборочно удалить следующие данные:

- журнал действий пользователей,
- события удаленного мониторинга (применимо только в том случае, если настроена *Интеграция с удаленными системами распознавания лиц*).

Утилита очистки работает в контейнере `findface-cibr-findface-multi-legacy-1`. Для того чтобы вызвать справку `cleanup`, выполните следующую команду:

```
sudo docker exec -it findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/
python3 /tigre_prototype/manage.py cleanup --help

usage: manage.py cleanup [-h] [--as-configured]
                        [--face-events-max-fullframe-unmatched-age FACE_EVENTS_MAX_
                        FULLFRAME_UNMATCHED_AGE]
                        [--face-events-max-fullframe-matched-age FACE_EVENTS_MAX_
                        FULLFRAME_MATCHED_AGE]
                        [--face-events-max-unmatched-age FACE_EVENTS_MAX_UNMATCHED_AGE]
                        [--face-events-max-matched-age FACE_EVENTS_MAX_MATCHED_AGE]
                        [--body-events-max-fullframe-unmatched-age BODY_EVENTS_MAX_
                        FULLFRAME_UNMATCHED_AGE]
                        [--body-events-max-fullframe-matched-age BODY_EVENTS_MAX_
                        FULLFRAME_MATCHED_AGE]
                        [--body-events-max-unmatched-age BODY_EVENTS_MAX_UNMATCHED_AGE]
                        [--body-events-max-matched-age BODY_EVENTS_MAX_MATCHED_AGE]
                        [--car-events-max-fullframe-unmatched-age CAR_EVENTS_MAX_
                        FULLFRAME_UNMATCHED_AGE]
                        [--car-events-max-fullframe-matched-age CAR_EVENTS_MAX_
                        FULLFRAME_MATCHED_AGE]
                        [--car-events-max-unmatched-age CAR_EVENTS_MAX_UNMATCHED_AGE]
                        [--car-events-max-matched-age CAR_EVENTS_MAX_MATCHED_AGE]
                        [--car-cluster-events-max-age CAR_CLUSTER_EVENTS_MAX_AGE]
                        [--body-cluster-events-max-age BODY_CLUSTER_EVENTS_MAX_AGE]
                        [--face-cluster-events-max-age FACE_CLUSTER_EVENTS_MAX_AGE]
                        [--car-cluster-events-keep-best-max-age CAR_CLUSTER_EVENTS_KEE_
                        BEST_MAX_AGE]
                        [--body-cluster-events-keep-best-max-age BODY_CLUSTER_EVENTS_
                        KEEP_BEST_MAX_AGE]
                        [--face-cluster-events-keep-best-max-age FACE_CLUSTER_EVENTS_
                        KEEP_BEST_MAX_AGE]
                        [--area-activations-max-age AREA_ACTIVATIONS_MAX_AGE]
                        [--audit-logs-max-age AUDIT_LOGS_MAX_AGE]
                        [--counter-records-max-age COUNTER_RECORDS_MAX_AGE]
                        [--external-vms-events-max-age EXTERNAL_VMS_EVENTS_MAX_AGE]
                        [--external-vms-send-events-status-max-age EXTERNAL_VMS_SEN_
                        EVENTS_STATUS_MAX_AGE]
                        [--remote-monitoring-events-max-age REMOTE_MONITORING_EVENTS_
                        MAX_AGE]
                        [--configuration CONFIGURATION] [--version]
                        [-v {0,1,2,3}] [--settings SETTINGS]
                        [--pythonpath PYTHONPATH] [--traceback] [--no-color]
```

(continues on next page)

(продолжение с предыдущей страницы)

```

[--force-color] [--skip-checks]

Delete FFSecurity entities

optional arguments:
  -h, --help            show this help message and exit
  --as-configured        Apply config age options for events, counter records
                        and clusters. Can't be used with other arguments.
  --face-events-max-fullframe-unmatched-age FACE_EVENTS_MAX_FULLFRAME_UNMATCHED_AGE
                        face events max fullframe unmatched age to clean up
                        (in seconds)
  --face-events-max-fullframe-matched-age FACE_EVENTS_MAX_FULLFRAME_MATCHED_AGE
                        face events max fullframe matched age to clean up (in
                        seconds)
  --face-events-max-unmatched-age FACE_EVENTS_MAX_UNMATCHED_AGE
                        face events max unmatched age to clean up (in seconds)
  --face-events-max-matched-age FACE_EVENTS_MAX_MATCHED_AGE
                        face events max matched age to clean up (in seconds)
  --body-events-max-fullframe-unmatched-age BODY_EVENTS_MAX_FULLFRAME_UNMATCHED_AGE
                        body events max fullframe unmatched age to clean up
                        (in seconds)
  --body-events-max-fullframe-matched-age BODY_EVENTS_MAX_FULLFRAME_MATCHED_AGE
                        body events max fullframe matched age to clean up (in
                        seconds)
  --body-events-max-unmatched-age BODY_EVENTS_MAX_UNMATCHED_AGE
                        body events max unmatched age to clean up (in seconds)
  --body-events-max-matched-age BODY_EVENTS_MAX_MATCHED_AGE
                        body events max matched age to clean up (in seconds)
  --car-events-max-fullframe-unmatched-age CAR_EVENTS_MAX_FULLFRAME_UNMATCHED_AGE
                        car events max fullframe unmatched age to clean up (in
                        seconds)
  --car-events-max-fullframe-matched-age CAR_EVENTS_MAX_FULLFRAME_MATCHED_AGE
                        car events max fullframe matched age to clean up (in
                        seconds)
  --car-events-max-unmatched-age CAR_EVENTS_MAX_UNMATCHED_AGE
                        car events max unmatched age to clean up (in seconds)
  --car-events-max-matched-age CAR_EVENTS_MAX_MATCHED_AGE
                        car events max matched age to clean up (in seconds)
  --car-cluster-events-max-age CAR_CLUSTER_EVENTS_MAX_AGE
                        car cluster events max age to clean up (in seconds)
  --body-cluster-events-max-age BODY_CLUSTER_EVENTS_MAX_AGE
                        body cluster events max age to clean up (in seconds)
  --face-cluster-events-max-age FACE_CLUSTER_EVENTS_MAX_AGE
                        face cluster events max age to clean up (in seconds)
  --car-cluster-events-keep-best-max-age CAR_CLUSTER_EVENTS_KEEP_BEST_MAX_AGE
                        car cluster events keep best max age to clean up (in
                        seconds)
  --body-cluster-events-keep-best-max-age BODY_CLUSTER_EVENTS_KEEP_BEST_MAX_AGE
                        body cluster events keep best max age to clean up (in
                        seconds)
  --face-cluster-events-keep-best-max-age FACE_CLUSTER_EVENTS_KEEP_BEST_MAX_AGE
                        face cluster events keep best max age to clean up (in

```

(continues on next page)

(продолжение с предыдущей страницы)

```

seconds)
--area-activations-max-age AREA_ACTIVATIONS_MAX_AGE
    area activations max age to clean up (in seconds)
--audit-logs-max-age AUDIT_LOGS_MAX_AGE
    audit logs max age to clean up (in seconds)
--counter-records-max-age COUNTER_RECORDS_MAX_AGE
    counter records max age to clean up (in seconds)
--external-vms-events-max-age EXTERNAL_VMS_EVENTS_MAX_AGE
    external vms events max age to clean up (in seconds)
--external-vms-send-events-status-max-age EXTERNAL_VMS_SEND_EVENTS_STATUS_MAX_AGE
    external vms send events status max age to clean up
    (in seconds)
--remote-monitoring-events-max-age REMOTE_MONITORING_EVENTS_MAX_AGE
    remote monitoring events max age to clean up (in
    seconds)
--configuration CONFIGURATION
    The name of the configuration class to load, e.g.
    "Development". If this isn't provided, the
    DJANGO_CONFIGURATION environment variable will be
    used.
--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".
--traceback
    Raise on CommandError exceptions
--no-color
    Don't colorize the command output.
--force-color
    Force colorization of the command output.
--skip-checks
    Skip system checks.

```

Для удаления журнала действий пользователя старше определенного количества дней используйте аргумент `--audit-logs-max-age` (значение задается в секундах). Например, для удаления журнала действий пользователя старше 5 дней (432000 секунд) выполните команду:

```

sudo docker exec -it findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/
↳python3 /tigre_prototype/manage.py cleanup --audit-logs-max-age 432000

```

Для удаления событий удаленного мониторинга старше 5 дней (432000 секунд) выполните команду:

```

sudo docker exec -it findface-cibr-findface-multi-legacy-1 /opt/findface-security/bin/
↳python3 /tigre_prototype/manage.py cleanup --remote-monitoring-events-max-age 432000

```

Важно: Должен быть задан хотя бы один аргумент из указанных.

2.5.9 Сброс пароля

Для того чтобы сбросить пользовательский пароль в веб-интерфейсе FindFace CIBR, выполните следующую команду:

```
sudo docker exec -it findface-cibr-findface-multi-identity-provider-1 /opt/findface-
security/bin/python3 /tigre_prototype/manage.py changepassword <username>
```

2.5.10 Перенос данных на другой диск

Высокая нагрузка на диск может приводить к задержкам при поступлении событий. В тяжелых случаях возможна полная неработоспособность FindFace CIBR. Одним из способов снижения нагрузки на диск является перенос хранилищ данных FindFace CIBR на другой диск.

В этом разделе:

- Подготовка диска
- Миграция хранилища фотографий и видео
- Миграция основной базы данных (PostgreSQL)

Подготовка диска

Для того чтобы подготовить диск к миграции данных, выполните следующие действия:

1. Создайте новую точку монтирования (в примере /mnt/ffdata).

```
sudo mkdir /mnt/ffdata
```

2. Создайте раздел.

```
sudo parted /dev/sdb
mklabel gpt
mkpart primary ext4 1MiB 100%
q
sudo mkfs.ext4 /dev/sdb1
```

3. Узнайте UUID раздела (для sdb1 в примере).

```
sudo blkid | grep sdb1
/dev/sdb1: LABEL="data" UUID="0638ebe0-853e-43ea-8f35-bfae305695d1" TYPE="ext4"
PARTUUID="8cebaacc-77d7-4757-b4c6-14147e92646c"
```

4. Настройте автоматическое монтирование раздела при загрузке ОС, добавив запись в **fstab**.

```
sudo vi /etc/fstab
-----
#DATA mount
UUID=0638ebe0-853e-43ea-8f35-bfae305695d1 /mnt/ffdata/ ext4 auto,user,rw
```

(continues on next page)

(продолжение с предыдущей страницы)

```
↪0      2
-----
```

5. Смонтируйте все файловые системы.

```
sudo mount -a
```

Миграция хранилища фотографий и видео

Примечание: Рекомендуем ознакомиться с разделом *Хранилища данных FindFace CIBR* перед началом миграции хранилищ.

Для миграции хранилищ фотографий и видео FindFace CIBR выполните следующие действия:

1. Внутри точки монтирования создайте каталоги (например, `/mnt/ffdata/ffupload` и `/mnt/ffdata/findface-security`) для хранения фотографий и видео. Затем переместите содержимое директорий `/opt/findface-cibr/data/findface-upload/uploads/` и `/opt/findface-cibr/data/findface-multi-legacy/uploads/` во вновь созданные каталоги.

```
sudo mkdir /mnt/ffdata/ffupload
sudo mkdir /mnt/ffdata/findface-security
sudo cp -ax /opt/findface-cibr/data/findface-upload/uploads/ -R /mnt/ffdata/
↪ffupload/
sudo cp -ax /opt/findface-cibr/data/findface-multi-legacy/uploads/ -R /mnt/ffdata/
↪findface-security/
sudo rm -r /opt/findface-cibr/data/findface-multi-legacy/uploads/
sudo rm -r /opt/findface-cibr/data/findface-upload/uploads/
```

2. Смонтируйте созданные каталоги (`/mnt/ffdata/ffupload` и `/mnt/ffdata/findface-security` в примере) в соответствующие контейнеры. Для этого откройте конфигурационный файл `/opt/findface-cibr/docker-compose.yaml` и перечислите созданные каталоги в указанных в примере томах разделов вместо каталогов по умолчанию.

```
sudo vi /opt/findface-cibr/docker-compose.yaml

findface-upload:
  ...
  volumes: ['./configs/findface-upload/40-ffupload.sh:/docker-entrypoint.d/40-
↪ffupload.sh:ro',
            '/mnt/ffdata/ffupload:/var/lib/ffupload']
  ...
findface-multi-identity-provider:
  ...
  volumes: ['./configs/findface-multi-identity-provider/findface-multi-identity-
↪provider.py:/etc/findface-security/config.py:ro',
            '/mnt/ffdata/findface-security/uploads:/var/lib/findface-security/uploads']
  ...
findface-multi-legacy:
  ...
  volumes: ['./configs/findface-multi-legacy/findface-multi-legacy.py:/etc/findface-
```

(continues on next page)

(продолжение с предыдущей страницы)

```

↪security/config.py:ro',
    '/mnt/ffdata/findface-security/uploads:/var/lib/findface-security/uploads']
...
cleaner:
    ...
    volumes: ['./configs/findface-multi-legacy/findface-multi-legacy.py:/etc/findface-
↪security/config.py:ro',
    '/mnt/ffdata/findface-security/uploads:/var/lib/findface-security/uploads']
...
findface-multi-ui:
    ...
    volumes: ['./configs/findface-multi-ui/nginx-site.conf:/etc/nginx/conf.d/default.
↪conf:ro',
    '/mnt/ffdata/findface-security/uploads:/var/lib/findface-security/uploads']

```

3. Выполните пересборку контейнеров FindFace CIBR.

```

cd /opt/findface-cibr

sudo docker-compose down

sudo docker-compose up -d

```

Миграция основной базы данных (PostgreSQL)

Для того чтобы мигрировать базу данных PostgreSQL, выполните следующие действия:

1. Остановите все контейнеры FindFace CIBR:

```

cd /opt/findface-cibr

sudo docker-compose down

```

2. Переместите каталог /opt/findface-cibr/data/postgresql в каталог /mnt/ffdata.

```

sudo mv /opt/findface-cibr/data/postgresql /mnt/ffdata

```

3. Укажите новый путь каталога postgresql (/mnt/ffdata/postgresql в примере) для монтирования в контейнер findface-cibr-postgresql-1. Для этого в файле конфигурации /opt/findface-cibr/docker-compose.yaml укажите его в томах (volumes) секции postgresql вместо каталога по умолчанию /opt/findface-cibr/data/postgresql.

```

sudo vi /opt/findface-cibr/docker-compose.yaml

postgresql:
    ...
    volumes: ['./configs/postgresql/40-init.sql:/docker-entrypoint-initdb.d/40-init.
↪sql:ro',
    '/mnt/ffdata/postgresql:/bitnami/postgresql']

```

4. Запустите все контейнеры FindFace CIBR.


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

2.6 Приложения

2.6.1 Файл с параметрами установки

При установке FindFace CIBR из инсталлятора параметры установки автоматически сохраняются в файл `/tmp/<findface-installer-*>.json`. Вы можете отредактировать данный файл и использовать его при установке FindFace CIBR на других серверах, не отвечая на вопросы инсталлятора повторно.

Совет: Подробная информация об инсталляторе приведена в разделе *Автоматическое односерверное развертывание FindFace CIBR*.

Важно: Обязательно удалите поля `*.config`, `exp_ip` и `int_ip` перед установкой FindFace CIBR на сервере с другим IP-адресом.

Пример файла с параметрами установки приведен [здесь](#).

2.6.2 Обзор моделей нейронных сетей

В этом разделе вы найдете сводную информацию по моделям нейронных сетей, созданным в нашей лаборатории и используемым в FindFace CIBR.

Установленные модели расположены в каталоге `/opt/findface-cibr/models/`.

Важно: При чистой установке биометрической моделью по умолчанию является `mango_320`.

Обнаружение лиц

```
ls /opt/findface-cibr/models/detector/

face.jasmine_fast.003.cpu.fnk  face.jasmine_fast.003.gpu.fnk
```

Нормализация изображений лиц

```
ls /opt/findface-cibr/models/facenorm/

bee.v3.cpu.fnk                crop2x.v2_maxsize400.cpu.fnk  facenorm.multicrop_full_
↪center_size400.cpu.fnk
bee.v3.gpu.fnk                crop2x.v2_maxsize400.gpu.fnk  facenorm.multicrop_full_
↪center_size400.gpu.fnk
bee_fast.cpu.fnk              crop2x.v2_no_maxsize.cpu.fnk  facenorm.multicrop_full_
↪crop2x_size400.cpu.fnk
```

(continues on next page)

(продолжение с предыдущей страницы)

```
bee_fast.gpu.fnk          crop2x.v2_no_maxsize.gpu.fnk  facenorm.multicrop_full_  
↪ crop2x_size400.gpu.fnk  
crop1x.v2_maxsize400.cpu.fnk  cropbbox.v2.cpu.fnk  
crop1x.v2_maxsize400.gpu.fnk  cropbbox.v2.gpu.fnk
```

Распознавание лиц

```
ls /opt/findface-cibr/models/face/  
  
mango_320.cpu.fnk  nectarine_m_160.cpu.fnk  nectarine_xs_320.cpu.fnk  
mango_320.gpu.fnk  nectarine_m_160.gpu.fnk  nectarine_xs_320.gpu.fnk
```

Распознавание атрибутов лиц

```
ls /opt/findface-cibr/models/faceattr/  
  
age.v2.cpu.fnk      emotions.v1.cpu.fnk  glasses3.v0.cpu.fnk  medmask3.v2.cpu.fnk  
age.v2.gpu.fnk      emotions.v1.gpu.fnk  glasses3.v0.gpu.fnk  medmask3.v2.gpu.fnk  
beard.v0.cpu.fnk    gender.v2.cpu.fnk    liveness.web.v0.cpu.fnk  quality_fast.v1.cpu.fnk  
beard.v0.gpu.fnk    gender.v2.gpu.fnk    liveness.web.v0.gpu.fnk  quality_fast.v1.gpu.fnk
```

2.6.3 Хранилища данных FindFace CIBR

В этом разделе:

- *Список хранилищ*
- *Галереи базы данных векторов признаков*

Список хранилищ

FindFace CIBR использует следующие хранилища данных:

- База данных векторов признаков на основе Tarantool, в которой хранятся векторы признаков лиц и события распознавания: `/opt/findface-cibr/data/findface-tarantool-server`.
- Основная база данных системы на основе PostgreSQL, в которой хранятся внутренние данные системы, карточки и учетные записи пользователей: `/opt/findface-cibr/data/postgresql`.
- Каталог `/opt/findface-cibr/data/findface-upload/uploads`, в котором хранятся нормализованные изображения лиц.
- Каталог `/opt/findface-cibr/data/findface-multi-legacy/uploads`, в котором хранятся загруженные в карточки фотографии; фотографии пользователей; видеофайлы и фотографии, загруженные в происшествия; полные кадры и миниатюры лиц кластеров, событий удаленного поиска и мониторинга.

Галереи базы данных векторов признаков

В базе данных векторов признаков на основе Tarantool есть следующие галереи, относящиеся только к FindFace CIBR:

- `ffsec_face_case_events`: векторы признаков, извлеченные из обнаруженных на видео или фотографиях происшествий лиц.
- `ffsec_face_case_clusters`: центроиды кластеров лиц происшествий.

В этой главе описывается, как работать с веб-интерфейсом FindFace CIBR, включая его расширенные возможности. Данные сведения будут интересны администраторам, операторам и другим пользователям.

3.1 Приступая к работе

После успешного *развертывания и конфигурирования* FindFace CIBR 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.

Создание списков наблюдения и картотеки

1. Создайте *новый список наблюдения* или используйте созданный по умолчанию. Список наблюдения — это системная сущность, с помощью которой выполняется классификация лиц по произвольным критериям, например, розыск, подозреваемые и т. д.
2. Загрузите карточки лиц в картотеку и добавьте их в списки наблюдения. См. *Картотека*.

Подключение FindFace CIBR к удаленным системам

Интегрируйте свой экземпляр FindFace CIBR с удаленными системами распознавания лиц. В этом случае сервер, известный как «кукловод», будет отправлять биометрические данные из созданных карточек на удаленные серверы-«марионетки». В ответ он будет получать события распознавания с совпадениями по данным карточкам.

FindFace CIBR в действии

1. *Создавайте происшествия* и обрабатывайте видеоматериалы и фотографии с мест преступлений, чтобы определить участников.
2. *Ищите лица* по всей системе.
3. *Сравнивайте лица*, чтобы проверить их на совпадение.
4. *Стройте* детализированные отчеты по результатам поиска и картотеке.
5. Поддерживайте безопасность системы, *анализируя журнал действий пользователей* FindFace CIBR.
6. Получайте оповещения от удаленных систем распознавания лиц и ищите в удаленных системах нужных людей. См. *Удаленный мониторинг и удаленный поиск*.

Базовое обслуживание системы

1. При необходимости вручную *удалите* старые данные.
2. Регулярно создавайте *резервную копию* базы данных.

Больше возможностей

Задействуйте функции FindFace CIBR через *HTTP API*.

3.2 Основы веб-интерфейса

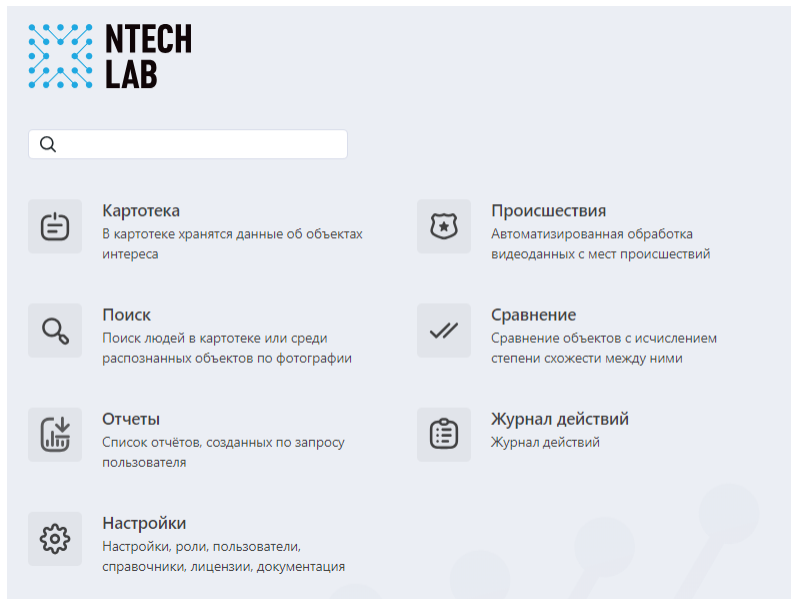
Работа с FindFace CIBR выполняется через веб-интерфейс. Для того чтобы отобразить веб-интерфейс, в адресной строке браузера введите базовый адрес веб-интерфейса и пройдите авторизацию.

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

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

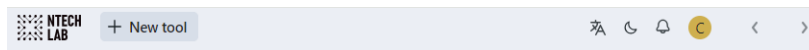
3.2.1 Навигация

В панели навигации разное количество элементов в зависимости от роли пользователя.



3.2.2 Язык веб-интерфейса и тема

Для переключения языка веб-интерфейса и выбора темы нажмите  на верхней панели.



3.3 Картотека

В картотеке хранятся карточки лиц, содержащие биометрические сведения, связанные документы, ссылки на происшествия, в которых принимало участие лицо, и другие важные данные.

Для одновременного создания большого количества карточек используйте функцию *пакетной загрузки через консоль*.

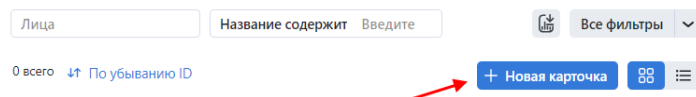
В этом разделе:

- *Создание карточки*
- *Пакетная загрузка карточек*
- *Автозаполнение артефактами расследований*
- *Фильтрация карточек*
- *Очистка картотеки*

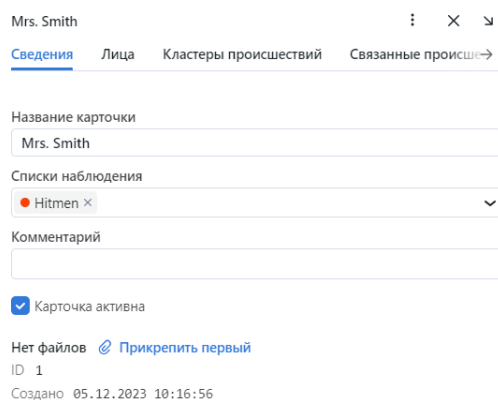
3.3.1 Создание карточки

Для создания карточки вручную выполните следующие действия:

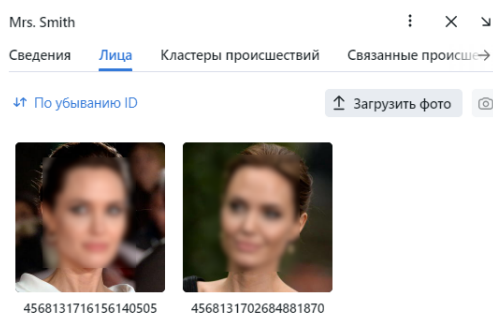
1. Перейдите на вкладку *Картотека*.
2. Нажмите *+ Новая карточка*.



3. Введите название карточки. При необходимости добавьте комментарий.
4. Из раскрывающегося списка *Списки наблюдения* выберите список, в который нужно добавить карточку (или несколько списков, по очереди).
5. Убедитесь, что поставлен флажок *Карточка активна*. Если карточка неактивна, она будет исключена из *анализа происшествий, удаленного мониторинга и поиска*.
6. Сохраните карточку. В результате в ней появятся дополнительные вкладки.
7. На вкладке *Сведения* прикрепите связанные файлы.



8. На вкладке *Лица* загрузите фотографии лица человека. Поддерживаемые форматы: WEBP, JPG, BMP, PNG, HEIC.



9. На вкладках *Кластеры происшествий*, *Связанные происшествия* и *Связанные карточки* данные будут отображены после автоматического совпадения карточки с кластером происшествий или создания ручной связи кластера во время *анализа людей*.

3.3.2 Пакетная загрузка карточек



Для создания большого количества карточек используйте функциональность пакетной загрузки. Выполните следующие действия:

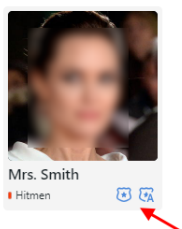
1. Перейдите на вкладку *Картотека*.
2. Нажмите *+ Новая карточка* → Попробуйте *пакетную загрузку карточек*.
3. Выберите файлы или папку и перетащите их для загрузки.

4. Введите префикс и постфикс имени. При необходимости добавьте префикс и постфикс комментария.
5. Выберите список наблюдения (или несколько списков наблюдения, по очереди) из раскрывающегося списка.
6. Нажмите *Начать*.

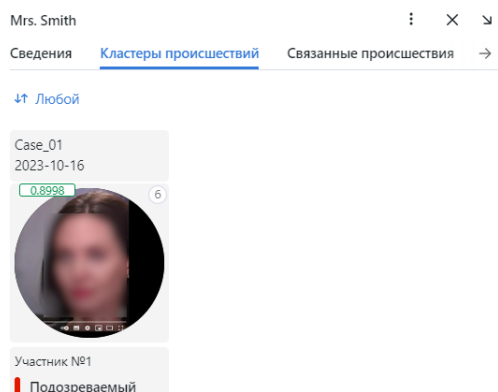
3.3.3 Автозаполнение артефактами расследований

Пока вы выполняете служебные обязанности с помощью FindFace CIBR, расследуете происшествия и анализируете видеозаписи с мест преступления, карточка автоматически заполняется данными происшествия.

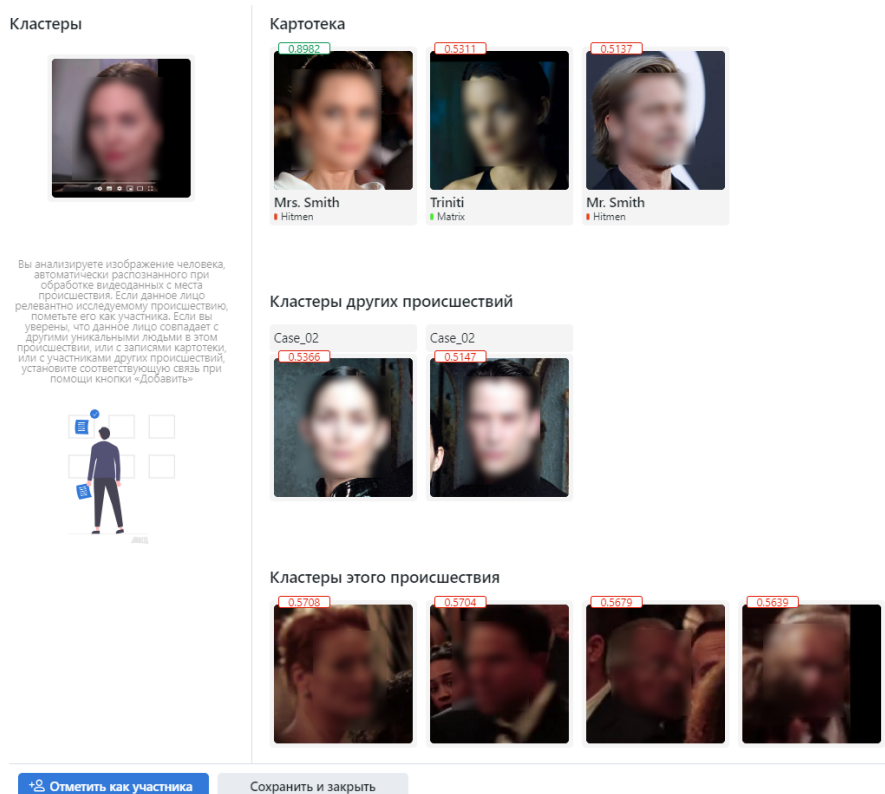
Если карточка имеет автоматическое совпадение с лицом на фотографии или видео, загруженных в происшествие, то на миниатюре карточки будет отображен знак . Если связь установлена вручную, то будет отображен знак .



Нажмите на него, чтобы перейти на вкладку *Кластеры происшествий* или *Связанные происшествия* карточки.

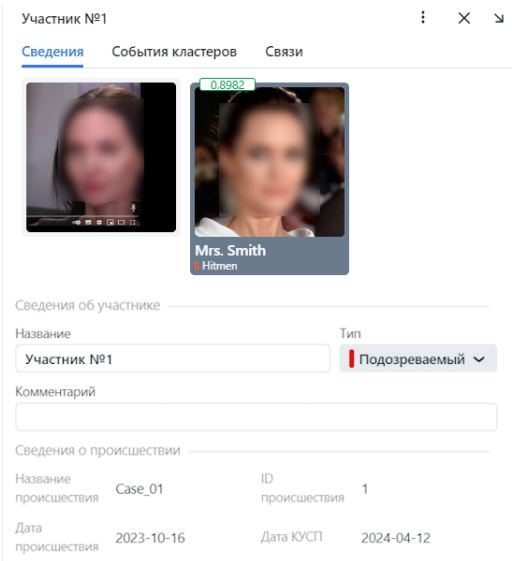


Если выбранный кластер происшествий не является участником происшествия, перейдите в окно мастера связей, нажав на изображение лица. См. *Анализ обнаруженных людей*.

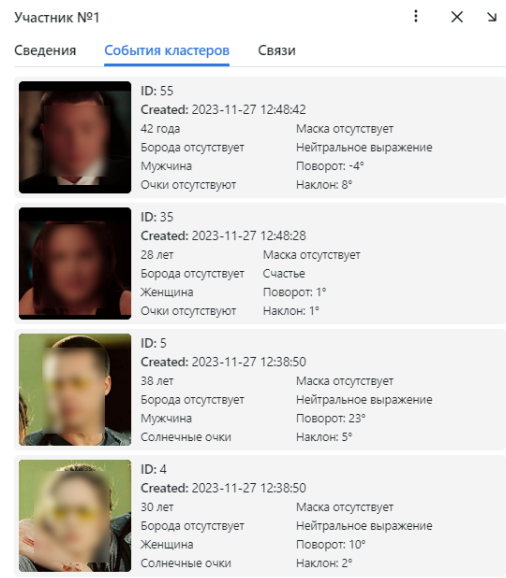


Если выбранный кластер происшествий является участником происшествия, перейдите в *запись участника происшествия*, нажав на изображение лица.

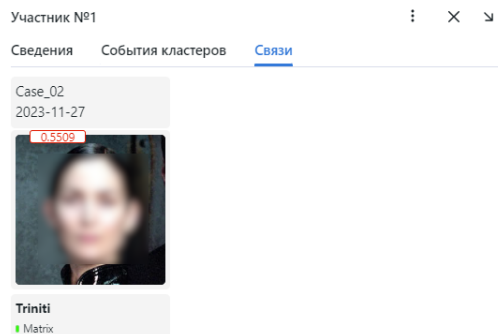
- На вкладке *Сведения* вы можете изменить информацию об участнике при необходимости.



- На вкладке *События кластеров* вы можете посмотреть обнаруженные изображения лица участника и другие изображения лиц кластера происшествий.



- На вкладке *Связи* вы можете просмотреть связи участника с лицами из других кластеров.



3.3.4 Фильтрация карточек

Наиболее часто используемые фильтры для карточек отображаются в верхней части окна.

Для того чтобы отобразить набор доступных фильтров целиком, нажмите на кнопку *Все фильтры*. Набор фильтров следующий:

- *Списки наблюдения*: отобразить карточки, включенные в выбранные списки наблюдения.
- *Лица*: отфильтровать карточки по присутствию изображения лица человека.
- *Заполнение*: отобразить только пустые карточки, только заполненные или любые.
- *Название содержит*: фильтровать карточки по имени.
- *ID*: отобразить карточку с определенным ID.
- *Статус*: фильтровать карточки по статусу (только активные, только неактивные, любые).
- *Тип связи*: фильтровать карточки по типу связи.

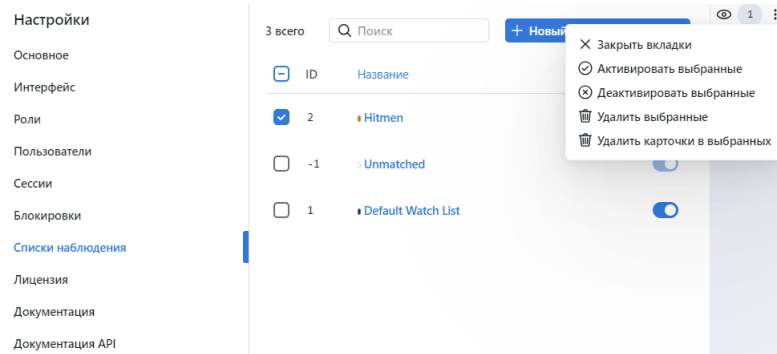


Вы можете отсортировать карточки в списке по параметру *ID*.

3.3.5 Очистка картотеки

Вы можете очистить картотеку целиком или по спискам наблюдения одним щелчком мыши. Выполните следующие действия:

1. Перейдите *Настройки* → *Списки наблюдения*.
2. Выберите один или несколько списков наблюдения.
3. Нажмите *Удалить карточки в выбранных*.



3.4 Происшествия

FindFace CIBR позволяет управлять записями о происшествиях и проводить расследования по связанным фото- и видеоматериалам. Данная функциональность доступна на вкладке *Происшествия*.

В этом разделе:

- *Форматы видеофайлов*
- *Процесс расследования происшествия*
- *Создание происшествия*
- *Права доступа к записи о происшествии и связанные документы*
- *Загрузка и обработка видеофайла*
- *Параметры обработки видео*
- *Загрузка и обработка фотографии*
- *Анализ обнаруженных людей*
- *Записи участников происшествия*
- *Архив происшествий*

3.4.1 Форматы видеофайлов

Видеоматериалы, используемые для расследования происшествий, принимаются в самых разных форматах. Нажмите [здесь](#), чтобы просмотреть список форматов.

3.4.2 Процесс расследования происшествия

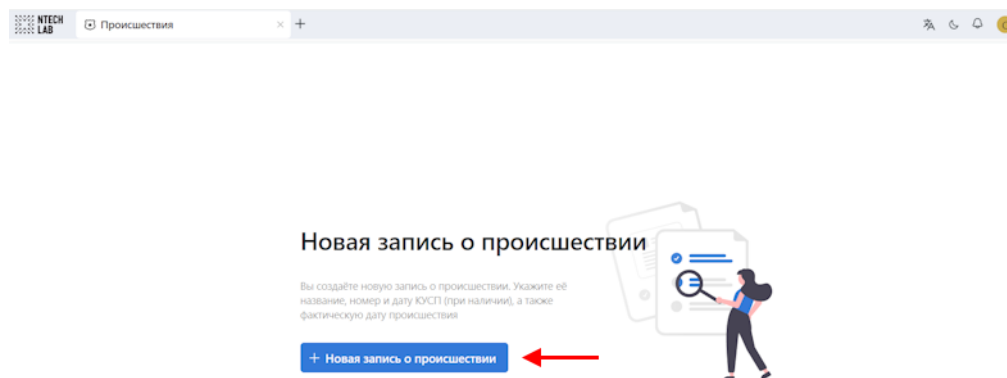
Расследование происшествия производится следующим образом:

1. Создайте новую запись о происшествии. Укажите дату инцидента и название, номер и дату регистрации инцидента в Книге учета сообщений о преступлениях (КУСП).
2. Укажите детали происшествия и настройте для него права доступа.
3. Загрузите фото- и видеоматериалы с места преступления. При необходимости настройте параметры обработки видео. Обработайте видеоматериалы. Система вернет обнаруженные на них человеческие лица.
4. Выполните разбор обнаруженных лиц. Выясните, кто может являться подозреваемым, потерпевшим или свидетелем. Другие лица будут считаться нерелевантными происшествию. Свяжите лица с соответствующими карточками в [картотеке](#) и участниками других происшествий.
5. Заполните записи участников происшествия. Укажите их имена и прикрепите соответствующие документы.
6. Регулярно возвращайтесь к записи о происшествии для дополнения ее новыми материалами по мере продвижения расследования.

3.4.3 Создание происшествия

Для создания происшествия выполните следующие действия:

1. Перейдите на вкладку *Происшествия*.
2. Нажмите *+ Новая запись о происшествии*.



3. Введите описательное название происшествия. Укажите дату инцидента.
4. Добавьте комментарий при необходимости.
5. Укажите номер и дату происшествия в КУСП при необходимости.

6. Нажмите *Добавить* >. В результате запись о происшествии будет добавлена в список происшествий.

Новая запись о происшествии

Вы создаёте новую запись о происшествии. Укажите её название, номер и дату КУСП (при наличии), а также фактическую дату происшествия

Название	Дата происшествия
<input type="text" value="Case_01"/>	<input type="text" value="16.10.2023"/>
Комментарий <input type="text"/>	
Номер КУСП	Дата КУСП
<input type="text" value="1111"/>	<input type="text" value="16.10.2023"/>

Добавить >

3.4.4 Права доступа к записи о происшествии и связанные документы

После создания записи о происшествии укажите его детали и настройте права доступа к записи. Выполните следующие действия:

1. Нажмите на запись в списке происшествий, чтобы открыть ее.

Все 2

Открытые 2

Архивные 0

Неподтвержденные класт...

Все фильтры

2 всего

Поиск

+ Новая запись о происшествии

<input type="checkbox"/>	Название	Создатель	Добавлено	Изменено	Видео	Кластеры	Участники	Статус
<input type="checkbox"/>	Case_01	Charlie Root	2024-01-09 11:07:39	2024-01-09 11:07:39	0	3	0	Открыто
<input type="checkbox"/>	Case_01	Charlie Root	2023-12-29 11:32:56	2023-12-29 11:32:56	2	43	0	Открыто

2. Нажмите *Установить права доступа*, чтобы изменить права доступа, заданные по умолчанию.
3. Прикрепите один или несколько файлов, имеющих отношение к происшествию.
4. Нажмите *Сохранить*.

Case_01 [Сведения](#) → [Видео](#) → [Фотографии](#) → [Кластеры](#) → [Участники](#)

Название	Дата происшествия		
<input type="text" value="Case_01"/>	<input type="text" value="16.10.2023"/>		
Комментарий <input type="text"/>			
Номер КУСП	Дата КУСП		
<input type="text" value="1111"/>	<input type="text" value="16.10.2023"/>		
3 всего <input type="text" value="Поиск"/>			
Наз...	Просмотр	Изменить	Удалить
Administrator	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Operator	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
User	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Нет файлов Прикрепить первый			
<input type="button" value="Сохранить"/>			

3.4.5 Загрузка и обработка видеофайла

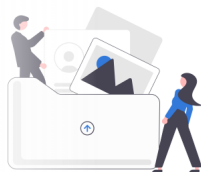
Для того чтобы загрузить и обработать видеозапись с места преступления, выполните следующие действия:

1. Перейдите на вкладку *Происшествия*.
2. Нажмите на запись в списке происшествий, чтобы открыть ее.
3. Перейдите на вкладку *Видео*.
4. Нажмите на кнопку *+ Добавить видео*.

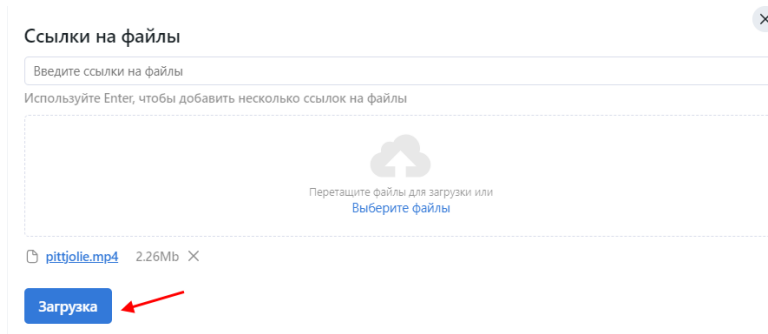
Case_01 [Сведения](#) → [Видео](#) → [Фотографии](#) → [Кластеры](#) → [Участники](#)

Новое видео

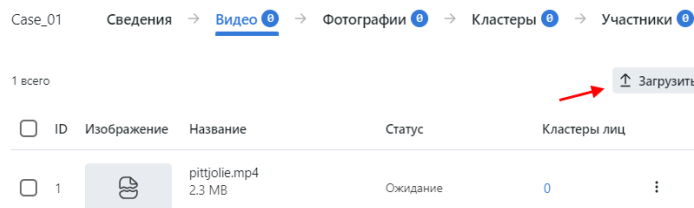
Пожалуйста, загрузите первое видео



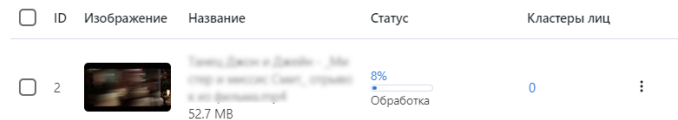
5. Укажите URL или выберите файл. Нажмите *Загрузка*.



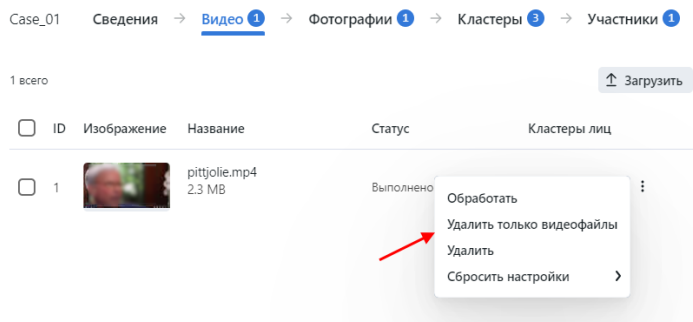
6. Видеофайл будет добавлен в список загруженных видео. Нажмите *Загрузить*, чтобы добавить еще видеофайл.



7. Нажмите на видео в списке, чтобы открыть мастер настройки обработки. Настройте *параметры обработки видео* и запустите идентификацию лиц. С помощью шкалы прогресса в столбце **Статус** вы можете следить за ходом обработки видео.

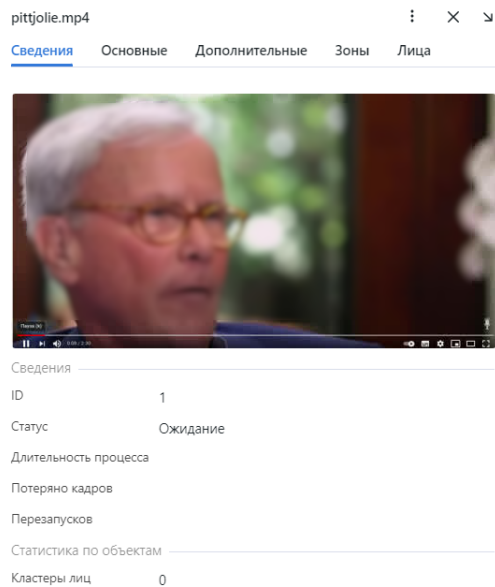


8. Если вам нужно освободить место на диске, но сохранить распознанные лица и кластеры, вы можете *Удалить только видеофайлы*. Чтобы полностью удалить загруженный файл, выберите *Удалить*.

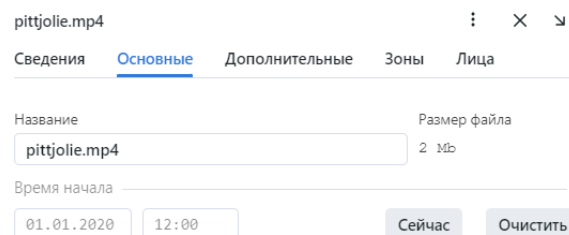


3.4.6 Параметры обработки видео

1. Для каждого видео предоставляется полная статистика: продолжительность текущей сессии обработки, количество объектов, обработанных с ошибками после последнего перезапуска job-задания, количество пропущенных кадров и другие данные. Для просмотра этих данных нажмите на название видео в списке и перейдите на вкладку *Сведения*.



2. На вкладке *Основные* вы можете изменить название видео при необходимости.

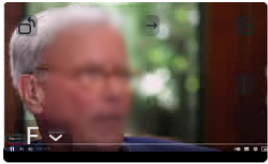


3. На вкладке *Дополнительные* выполните тонкую настройку видео:

pittjolie.mp4

Сведения Основные Дополнительные Зоны Лица

Трансформация



☐ Отразить по вертикали
☐ Отразить по горизонтали
☐ Повернуть по часовой стрелке
☐ Повернуть против часовой стрелки

Отправка лиц

Таймаут Мс

☒ Проверить SSL-сертификат

Временная метка

☐ Получать метки времени из потока
 ☐ Добавить к временным меткам
 Сек

Другие

Параметры FFmpeg

Ограничение скорости воспроизведения
 Формат ввода
 Минимальная интенсивность движения

☒ Считывание кадров из источника без пропусков

URL роутера (router_url)

- При необходимости измените ориентацию видео.

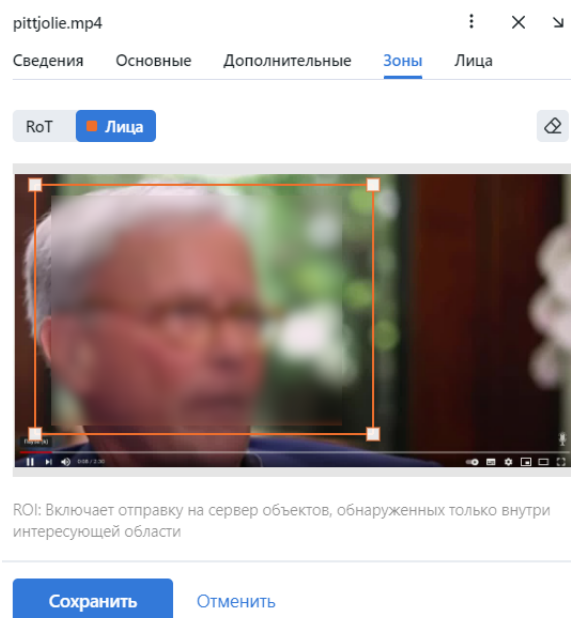
Важно: Вращение потока выполняется на стороне сервера `findface-multi-legacy` средствами постобработки. Это может негативно сказаться на производительности. Рекомендуем по возможности делать вращение потока на стороне камеры.

- *Таймаут:* Укажите таймаут в миллисекундах для отправки обнаруженных объектов.
- *Проверить SSL-сертификат:* Поставьте флажок, чтобы включить проверку подписи SSL-сертификата при взаимодействии по `https` видеодетектора объектов с сервером. Снимите флажок, если вы используете самоподписанный сертификат.
- *Получать метки времени из потока:* Поставьте флажок, чтобы отправлять на сервер временные метки, полученные из потока. Снимите флажок, чтобы отправлять текущие дату и время.
- *Добавить к временным меткам:* Прибавлять указанное количество секунд к временным меткам из потока.
- *Параметры FFmpeg:* Параметры FFmpeg для видеопотока. Задаются массивом строк `ключ=значение`, например, `["rtsp_transpotr=tcp", "ss=00:20:00"]`.
- *Ограничение скорости воспроизведения:* Если меньше нуля, то скорость не ограничивается, в остальных случаях поток читается с заданной скоростью `play_speed`. Не применимо для потоков с камер видеонаблюдения.
- *Формат ввода:* Передать формат FFmpeg (`mxg`, `flv`, и т. д.), если он не может быть автома-

тически определен.

- *Минимальная интенсивность движения*: Минимальная интенсивность движения, которая будет регистрироваться детектором движения.
- *Считывание кадров из источника без пропусков*: Если `findface-video-worker` не обладает достаточными ресурсами для обработки всех кадров с объектами, он будет пропускать некоторые из них. Если данная опция активна (`true`), `findface-video-worker` помещает лишние кадры в очередь, чтобы обработать из впоследствии.
- *URL роутера (router_url)*: IP адрес для отправки обнаруженных объектов во внешние видеообработчики из `findface-video-worker`. По умолчанию `http://127.0.0.1`.

4. Укажите область отслеживания (*Зоны*) в поле зрения камеры и область интереса. Нажмите *Сохранить*.



Область отслеживания позволяет обнаруживать и отслеживать лица только внутри заданного прямоугольника. Используйте данную опцию, чтобы уменьшить нагрузку на видеодетектор объектов. Область интереса позволяет отправлять на сервер только объекты, обнаруженные в ее пределах.

5. На вкладке *Лица* укажите настройки детектора лиц.

pittjolie.mp4 ⋮ × ↵

Сведения Основные Дополнительные Зоны Лица

☒ Включен Сбросить параметры лиц

Фильтр

Размер - Качество

Сжатие

Качество JPEG

☐ Полный кадр PNG

Трекинг

Коэффициент перекрытия границ распознанного объекта

Максимальная длительность трека

Интервал закрытия трека

☐ Отправлять историю трека

ROT

☐ Обрезать полный кадр

Отправка на обработку

☒ Буферный режим

Интервал

☐ Отправлять первый распознанный объект немедленно ☐ Отправлять в каждом интервале

☐ Отправлять первый кадр трека ☐ Отправлять последний кадр трека

- **Размер:** Минимальный и максимальный размер отправляемых объектов, в пикселях.
- **Качество:** Минимальное пороговое значение качества изображения объекта. Допустимый диапазон от 0 до 1. Базовое рекомендуемое значение 0,5 — соответствует удовлетворительному качеству изображения объекта. Не меняйте значение по умолчанию без предварительной консультации с нашими техническими экспертами (support@ntechlab.com).
- **Качество JPEG:** Качество сжатия полного кадра для отправки.
- **Полный кадр PNG:** Отправлять полные кадры в формате PNG, а не в формате JPEG, как установлено по умолчанию. Не включайте этот параметр, не посоветовавшись с нашей командой, так как он может повлиять на работу всей системы.
- **Коэффициент перекрытия границ распознанного объекта:** Процент перекрытия границ между двумя последовательными кадрами, чтобы эти bbox'ы считались одним треком. Диапазон значений: от 0 до 1. Не меняйте значение по умолчанию без предварительной консультации с нашими техническими экспертами (support@ntechlab.com).
- **Максимальная длительность трека:** Максимальное приблизительное количество кадров в треке, по достижении которого трек будет принудительно закрыт. Включите опцию, чтобы автоматически завершать вечные треки, например, треки детектирования объектов с рекламных носителей.
- **Интервал закрытия трека:** Завершить трек, если в нем не было распознано новых изображений объекта в течение заданного времени (в секундах).
- **Отправить историю трека:** Отправлять массив координат границ объекта вместе с событием. Может быть применено для внешних интеграций, чтобы отобразить путь объекта.
- **Обрезать полный кадр:** Поставьте флажок если необходимо обрезать полный кадр по раз-

меру ROT-области перед отправкой на распознавание. Размер полного кадра будет равен размеру ROT-области.

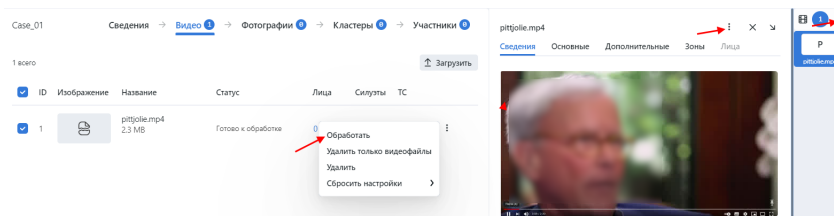
- *Буферный режим*: По умолчанию для обработки видео система использует буферный режим, т.е. отправляет по одному кадру наилучшего качества на трек для экономии места на диске. Отключите данный режим, чтобы получать больше кадров объекта, если это необходимо. Если буферный режим включен, параметры режима реального времени не работают.

Параметры режима реального времени:

Примечание: Эти параметры не работают, если включен буферный режим.

- *Интервал*: Временной интервал в секундах (целое или десятичное число), в течение которого в режиме реального времени выбирается лучший кадр с объектом.
- *Отправлять первый распознанный объект немедленно*: Поставьте флажок, чтобы отправлять первый объект из трека сразу, как только фильтры по качеству/размеру/региону интереса пройдены, не дожидаясь окончания первого `realtime_post_interval` в режиме реального времени. Снимите флажок, чтобы отправлять первый объект из трека только после окончания первого `realtime_post_interval`.
- *Отправлять первый кадр трека*: При завершении трека в дополнение к overall-кадру трека будет дополнительно отправлен первый кадр трека. Может быть применимо для внешних интеграций.
- *Отправлять в каждом интервале*: Поставьте флажок, чтобы отправлять лучший кадр в каждом временном `realtime_post_interval` в режиме реального времени. Снимите флажок, чтобы отправлять лучший кадр, только если его качество улучшилось по сравнению с предыдущим отправленным кадром.
- *Отправлять последний кадр трека*: При завершении трека в дополнение к overall-кадру трека будет дополнительно отправлен последний кадр трека. Может быть применимо для внешних интеграций.

6. В списке загрузок видео нажмите три точки → *Обработать*, чтобы запустить идентификацию лиц.




Результаты обработки будут доступны на вкладке *Кластеры* и на вкладке *События кластеров участника преступления*.

Важно: При обработке фото- и видеофайлов обратите внимание на лицензионный лимит количества изображений лица человека.

В лицензионном лимите количества изображений лица человека учитываются не только изображения, загруженные в *Картотеку*, но и каждое событие кластера, порождённое при обработке видео и фотографии в происшествиях.

Если происшествие деактивируется (помещается в архив), соответствующий лимит высвобождается, и наоборот: при извлечении происшествия из архива (повторном открытии) все связанные

с ним события кластеров учитываются в лимите.

Обработка видеофайла или фотографии в происшествии может оказаться неуспешной. Это также распространяется на случаи исчерпания лимита лицензии. Происшествие, содержащее видеофайлы и фотографии, которые не удалось обработать, помечается значком . В таком происшествии можно подтверждать кластеры, имеющие автоматическое совпадение с лицом карточки, но нельзя сохранять кластеры и конвертировать их в участников. Статус происшествия изменится либо после успешной обработки видеофайлов и фотографий (для этого перезапустите обработку файла), либо после удаления файла.

3.4.7 Загрузка и обработка фотографии


Для того чтобы загрузить и обработать фотографию с места преступления, выполните следующие действия:

1. Перейдите на вкладку *Происшествия*.
2. Нажмите *+ Добавить фото*.

Case_01 Сведения → Видео  → Фотографии  → Кластеры  → Участники 

Новая фотография

Пожалуйста, загрузите первое фото. Поддерживаемые форматы: JPG, PNG, WEBP, HEIC

 **Добавить фото**

3. Выберите фотографии для загрузки.

Выберите фотографии

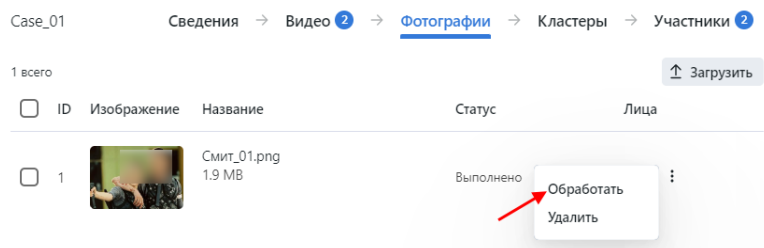
×

Перетащите файлы для загрузки или
[Выберите файлы](#)

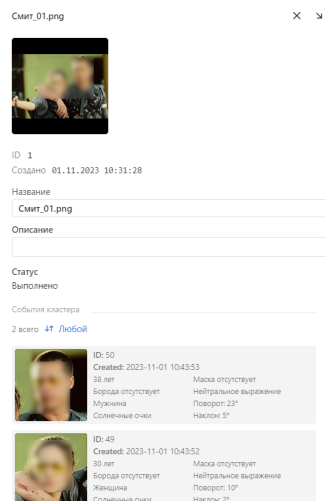
 [Смилг_01.png](#) 1.93Mb ✕

Добавить и обработать

4. Нажмите *Добавить и обработать*. В результате фотография будет добавлена в список загруженных изображений и обработана.
5. Чтобы перезапустить идентификацию объектов на фотографии, нажмите три точки → *Обработать*.



6. Нажмите на название изображения в списке, чтобы посмотреть события кластеров фотографии, изменить название фотографии и добавить комментарий при необходимости.




3.4.8 Анализ обнаруженных людей

Кластеры обнаруженных на видео или фотографии лиц отображаются на вкладке *Кластеры*. Здесь вам нужно проанализировать их в зависимости от той роли, которую человек играет в инциденте, и установить ссылки на соответствующие карточки, записи других происшествий, другие кластеры и других участников данного происшествия.

Если обнаруженный объект имеет соответствие с картотекой, то миниатюра кластера происшествий содержит следующую информацию:

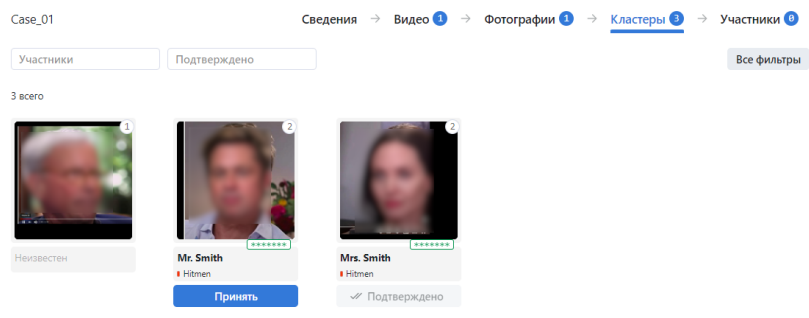
- нормализованное изображение объекта
- название карточки
- степень схожести объектов
- список (списки) наблюдения
- кнопка *Принять*

Нажмите кнопку *Принять*, чтобы подтвердить кластер происшествия.

Примечание: Символ  в списке происшествий указывает на то, что в происшествии есть неподтвержденный кластер происшествий.

Чтобы проанализировать обнаруженных людей, выполните следующие действия:

1. Нажмите на лицо в списке.



Совет: При большом количестве кластеров используйте фильтры. Для их отображения нажмите кнопку *Все фильтры* в правом верхнем углу.

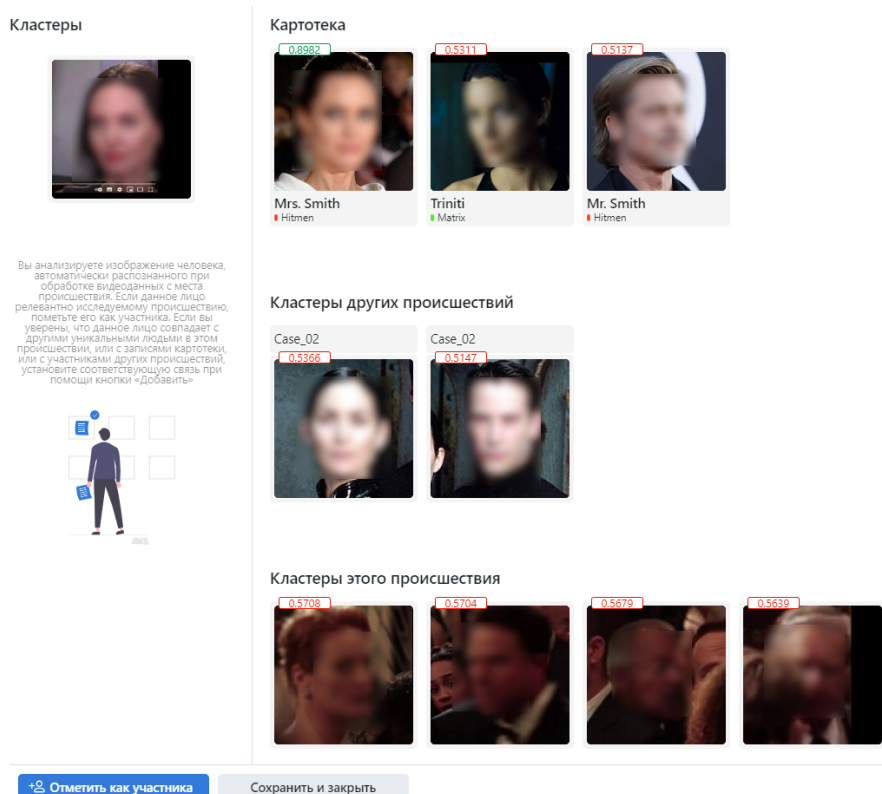
Примечание: Некоторые фильтры из приведенного ниже списка могут быть скрыты, в зависимости от активированной функциональности распознавания.

- *Участники:* отобразить только кластеры лиц/только участников или любой.
- *Подтверждено:* отобразить только неподтвержденные/только подтвержденные кластеры или любой.
- *Совпадения:* отображать только кластеры с совпадением объектов/без совпадения объектов с карточкой или все события.
- *Списки наблюдения:* отобразить кластеры происшествий по определенному списку наблюдения.
- *Дата и время:* отобразить кластеры происшествий, сформированные в определенный период времени.
- *ID:* отобразить кластер происшествий с определенным ID.
- *ID фотографии:* отобразить кластер происшествий с определенной фотографией (по ID фотографии).
- *ID видео:* отобразить кластер происшествий с определенным видео (по ID видео).

Специальные фильтры для кластеров лиц

- *Возраст:* отобразить кластеры происшествий с людьми определенного возраста.
- *Борода:* фильтровать кластеры происшествий по наличию бороды.
- *Эмоции:* отобразить кластеры происшествий с заданными эмоциями.
- *Пол:* отобразить кластеры происшествий с людьми заданного пола.
- *Очки:* фильтровать кластеры происшествий по наличию очков на лице.
- *Медицинская маска:* фильтровать кластеры происшествий по наличию медицинской маски.

- При наличии совпадения свяжите лицо с карточкой из *картотеки* и участниками других происшествий. Для этого используйте кнопку *Добавить*, которая появляется при наведении курсора на фотографию. Вы также можете связать лицо с участниками данного происшествия и другими кластерами.
- Нажмите на лицо в разделе *Кластеры*, чтобы посмотреть обнаруженные лица в полноэкранном режиме и отделить некоторые из них от кластера при необходимости.
- Нажмите *Отметить как участника*.



- Укажите название участника и его тип и при необходимости добавьте комментарий. Это создаст новую запись участника происшествия, с которой вы сможете работать на вкладке *Участники*.

Сохранить участника
✕

Сведения об участнике

Название

Тип

Подозреваемый
▼

Комментарий

Сведения о происшествии

Название происшествия
Case_01

ID происшествия
1

Дата происшествия
2023-10-16

Дата КУСП
2024-04-12

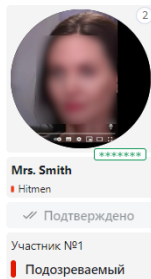
Сохранить и закрыть

Участник может быть одного из следующих типов:

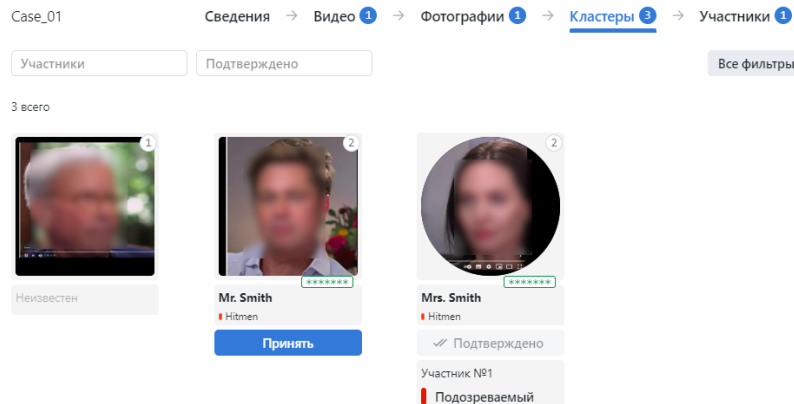
- потерпевший
- свидетель
- подозреваемый

6. Нажмите *Сохранить и закрыть*.

В результате информация будет добавлена к миниатюре кластера происшествий, а изображение лица будет иметь круглую рамку. Количество событий кластера отображается цифрой в правом верхнем углу миниатюры кластера происшествий. Нажмите на цифру, чтобы просмотреть изображение во весь экран в новом окне. При необходимости вы можете отделить лицо из кластера.



7. Вы можете повторно открыть мастер связей, еще раз нажав на изображение участника на вкладке *Кластеры*.

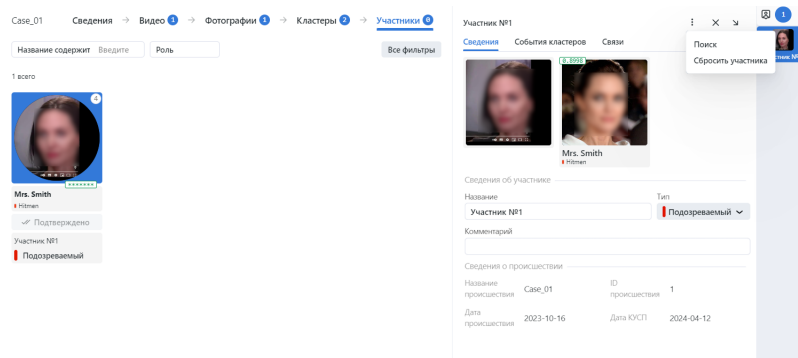


Чтобы получить доступ к записи участника происшествия, перейдите на вкладку *Участники*.

3.4.9 Записи участников происшествия

На вкладке *Участники* доступны все данные участников, собранные на текущий момент.

Чтобы просмотреть запись участника происшествия, нажмите на соответствующее лицо в списке.



- На вкладке *Сведения* вы можете изменить информацию об участнике, при необходимости.
- На вкладке *События кластеров* вы можете посмотреть обнаруженные изображения лица участника и другие изображения лиц кластера происшествий.
- На вкладке *Связи* вы можете просмотреть связи участника с лицами из других кластеров.

FindFace CIBR позволяет искать обнаруженные объекты из записей кластера происшествий. Для перехода на вкладку поиска из записи кластера происшествий нажмите три точки → *Поиск*.

Если во время расследования происшествия необходимо сбросить запись участника, нажмите три точки → *Сбросить участника*.

При большом количестве участников используйте фильтры. Для их отображения нажмите кнопку *Все фильтры* в правом верхнем углу.

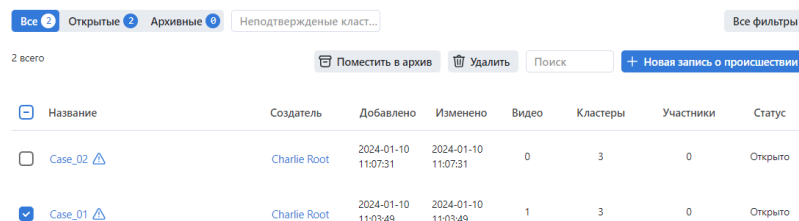
- *Название содержит:* отобразить участников с заданными именем.
- *Роль:* отобразить участников по выбранной роли (Любой/Потерпевший/Свидетель/Подозреваемый).
- *Дата и время:* отобразить участников по происшествиям, сформированным в определенный период времени.

3.4.10 Архив происшествий

Запись о происшествии можно пометить как архивную, чтобы указать, что дело закрыто, или по другой причине.

Для архивации/извлечения из архива записей о происшествиях выполните следующие действия:

1. На вкладке *Происшествия* выберите одну или несколько записей о происшествиях.
2. Нажмите *Поместить в архив*, чтобы поместить записи в архив. Нажмите *Удалить*, чтобы удалить их.



Если происшествие было заархивировано, вы больше не сможете его редактировать (изменять название, добавлять фотографии и видео, изменять настройки обработки и т.д.).

- Нельзя обрабатывать ранее загруженные фото- и видеофайлы.

- Нельзя редактировать его кластеры и конвертировать их в участников.
- Поиск по событиям кластера происшествий больше не может быть выполнен.
- Можно просматривать кластеры, участников и фильтровать их.
- В лицензии события кластеров из заархивированного происшествия не будут учитываться.
- Повторное сопоставление кластеров с картотекой из архивированного происшествия производиться не будет.
- Однако связи, построенные на основании этого происшествия в картотеке, сохраняются.

Если происшествие извлекается из архива (открыто повторно), его снова можно редактировать, а поиск по его событиям кластера снова становится возможным.

Чтобы снова открыть происшествие, выберите его и нажмите кнопку *Извлечь из архива*.

Вы можете отфильтровать список записей о происшествиях по статусу архивирования: *Все*, *Открытые*, *Архивные* или используйте дополнительные фильтры, нажав *Все фильтры*:

- *Неподтвержденные кластеры*: отображать происшествия по статусу подтверждения кластеров.
- *Номер КУСП*: отобразить происшествия с определенным номером КУСП.
- *Дата создания*: отобразить происшествия, сформированные в определенный период времени.
- *ID*: отобразить происшествия с определенным ID.

3.5 Поиск лиц в системе

FindFace CIBR позволяет выполнять поиск лиц по всей системе.

Для того чтобы найти человека, выполните следующие действия:

1. Перейдите на вкладку *Поиск*.
2. Задайте искомый объект одним из следующих способов:
 - задав URL или ID карточки
 - загрузив фотографию

Задайте источник поиска


Введите URL фото в Интернете или ID/URL внутренней сущности

Карточка

Файл

Введите ID или URL

или загрузите медиафайл. Поддерживаемые форматы: JPG, PNG, WEBP, HEIC

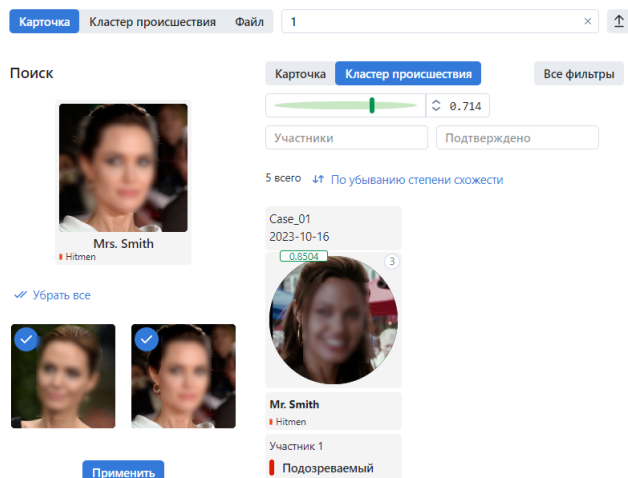


Перетащите файл для загрузки или
выберите файл

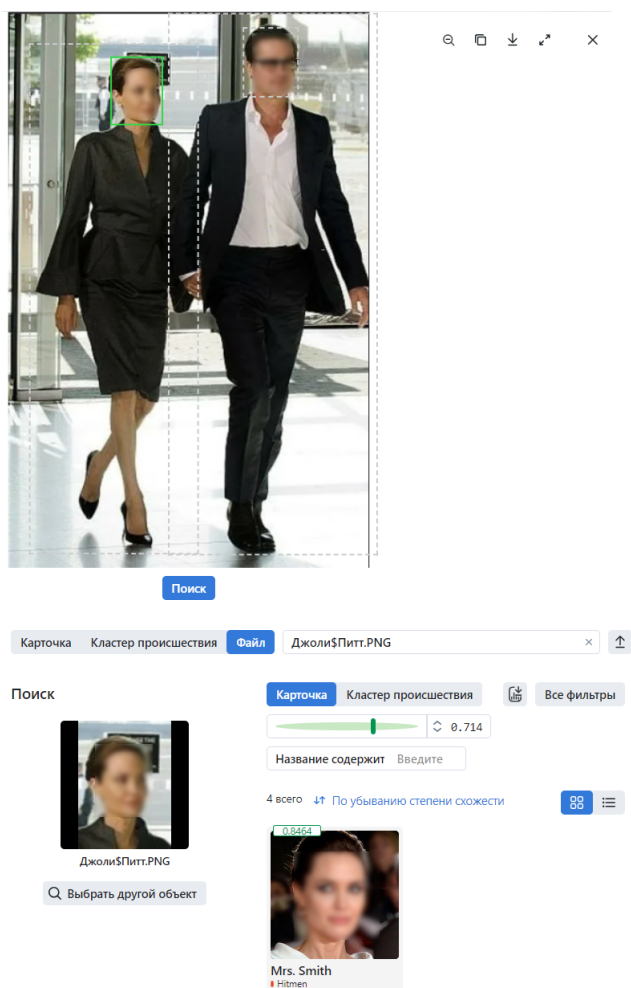
Примечание: Поиск по кластеру происшествия доступен из записей участников происшествия либо по URL или ID кластера.

3. В случае, если вы указали URL или ID карточки, выберите из нее фотографию. Если в карточке несколько фотографий, вы можете выбрать некоторые или все из них. Нажмите на кнопку *Применить*. При выборе нескольких объектов из карточки диапазон поиска сужается. В результаты

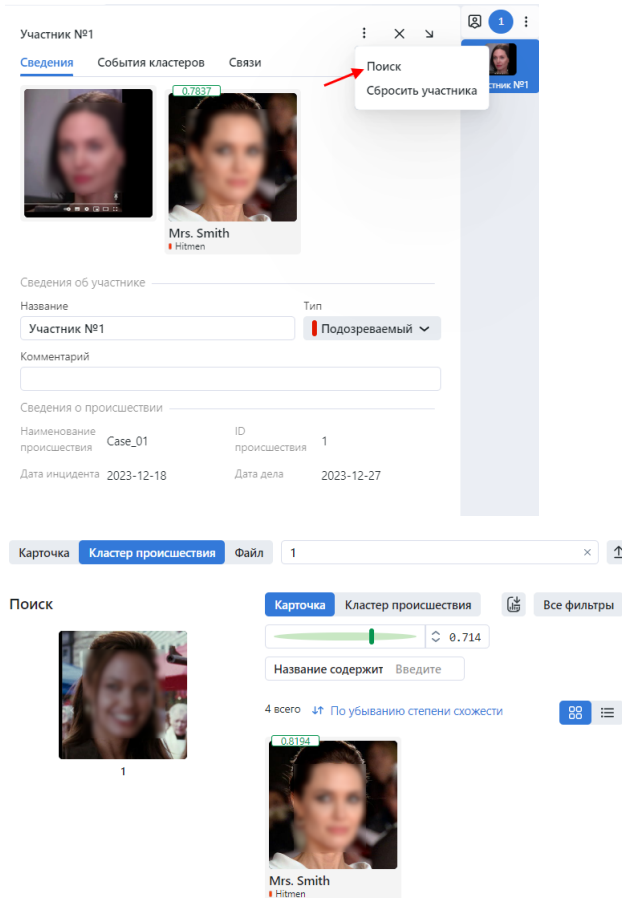
поиска попадут только те объекты, которые похожи на объекты на выбранных фотографиях и находятся в пределах заданного порога поиска.



4. В случае если была загружена фотография, вы увидите ее в новом окне. Если на фотографии присутствует несколько объектов, выберите нужный. Нажмите на кнопку *Поиск*.



5. Чтобы выполнить поиск по кластеру происшествий, связанному с определенным участником происшествия, перейдите к записи участника, нажмите три точки, затем *Поиск*, либо укажите URL или идентификатор кластера происшествия на вкладке *Поиск*. В результаты поиска попадут только те объекты, которые похожи на объекты на выбранных кластерах происшествий и находятся в пределах заданного порога поиска.



6. Определите секцию поиска – *Карточка* или *Кластер происшествий*.
7. Вы увидите результаты поиска. При необходимости можно сузить область поиска, указав фильтры или снизив порог сходства.

Для карточек доступны следующие фильтры:

- *Списки наблюдения*: отобразить карточки, включенные в выбранные списки наблюдения.
- *Заполнение*: отобразить только пустые карточки, только заполненные или любые.
- *Название содержит*: фильтровать карточки по названию.
- *ID*: отобразить карточку с определенным ID.
- *Статус*: фильтровать карточки по статусу (только активные, только неактивные, любые).
- *Тип связи*: фильтровать карточки по типу связи.
- *Лимит*: отобразить карточки в пределах заданного лимита.

Для кластеров происшествий доступны следующие фильтры:

- *Участники*: отобразить только кластеры происшествий/только участников.

- *Подтверждено*: отобразить только подтвержденные/неподтвержденные кластеры происшествий или любые.
- *Совпадения*: отобразить только кластеры с совпадением объектов/без совпадения объектов с карточкой или все события.
- *Списки наблюдения*: отобразить только кластеры с карточкой, включенной в выбранные списки наблюдения.
- *Дата и время*: отобразить только события, случившиеся в определенный период времени.
- *ID*: отобразить кластер происшествий с определенным ID.
- *ID фотографии*: отобразить кластер происшествий с определенной фотографией (по ID фотографии).
- *ID видео*: отобразить кластер происшествий с определенным видео (по ID видео).

Для кластеров происшествий доступны следующие специальные фильтры:

Примечание: Некоторые фильтры из приведенного ниже списка могут быть скрыты, в зависимости от активированной функциональности распознавания.

- *Возраст*: отобразить кластеры происшествий с людьми определенного возраста.
- *Борода*: фильтровать кластеры происшествий по наличию бороды.
- *Эмоции*: отобразить кластеры происшествий с заданными эмоциями.
- *Пол*: отобразить кластеры происшествий с людьми заданного пола.
- *Очки*: фильтровать кластеры происшествий по наличию очков на лице.
- *Медицинская маска*: фильтровать кластеры происшествий по наличию медицинской маски.
- *Лимит*: отобразить кластеры происшествий в пределах заданного лимита.

Вы можете отсортировать результаты поиска по следующим параметрам:

- по убыванию степени схожести
- сначала самые новые
- сначала самые старые

3.6 Сравнение двух лиц

FindFace CIBR позволяет сравнить два лица и убедиться, что они совпадают. Выполните следующие действия:

1. Перейдите на вкладку *Сравнение*.
2. Укажите два объекта для проверки одним из следующих способов:
 - задав URL события
 - загрузив фотографию
3. Если на изображении несколько объектов, выберите тот, который вас интересует.



4. В результате будет отображена вероятность совпадения.

3.7 Отчеты

FindFace CIBR предоставляет возможность построения отчетов по следующим системным объектам:


- *результаты поиска*
- *карточки*
- *журнал действий пользователя*

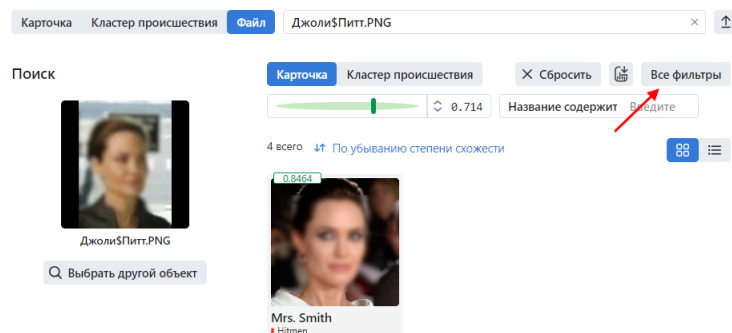
В этом разделе:


- *Построение отчета*
- *Работа с отчетами*

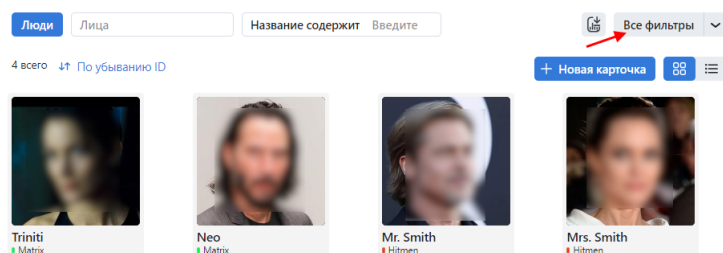
3.7.1 Построение отчета


Для того чтобы построить отчет по системному объекту, выполните следующие действия:

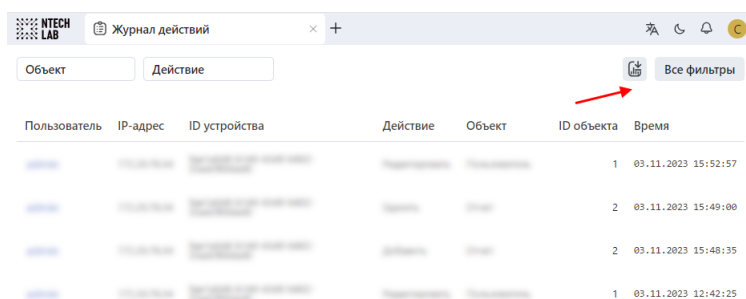
1. Перейдите на нужную вкладку: *Поиск*, *Картотека* или *Журнал действий*.
2. Выполните поиск, если вы на вкладке *Поиск*. См. [Поиск лиц в системе](#). Нажмите на кнопку  для построения отчета или *Все фильтры*, чтобы сначала задать фильтры.



3. Если вы на вкладке *Карточка*, нажмите на кнопку  для построения отчета или *Все фильтры*, чтобы сначала задать фильтры.



4. Если вы на вкладке *Журнал действий*, нажмите на кнопку  для построения отчета или *Все фильтры*, чтобы сначала задать фильтры.



5. Установите фильтры для отчета. Фильтры различны для каждого системного объекта.
6. Нажмите *Применить и создать отчет*.

Дата и время ✕ 01.01.1970 03:00 - 17.04.2024 15:34

IP-адрес ✕ Введите IP-адрес

ID устройства ✕ Введите ID устройства

Объект ✕ Выберите объект

Действие ✕ Выберите действие

ID объекта ✕ Введите ID объекта

Пользователь ✕ Введите имя пользователь

Применить и создать отчет Сбросить фильтры Применить фильтры

7. Укажите название отчета.
8. Поставьте флажки для нужных форматов отчета: XLS, JSON, CSV.
9. Выберите, следует ли сохранять изображения отчета в виде ссылок, миниатюр или полных кадров. (Только для результатов поиска и карточек).

Создать отчет

Название

Отчет по карточкам людей

Формат отчета

☒ XLS
☐ JSON
☐ CSV

Сохранить изображения в отчете как

Ссылка

Добавить

10. Нажмите *Добавить*. Отчет будет доступен для загрузки на вкладке *Отчеты*.

3.7.2 Работа с отчетами

Вы можете получить доступ к отчетам, ранее созданным в системе, на вкладке *Отчеты*. Доступны следующие операции:

- Скачивание выбранных отчетов.
- Обновление выбранных отчетов.
- Удаление выбранных отчетов.

Поиск							Скачать	Обновить	Удалить
<input checked="" type="checkbox"/>	ID	Имя	Тип	Изменено	Записи	Размер	Статус		
<input checked="" type="checkbox"/>	4	Отчет по журналу действий	Журнал действий	19.04.2024 14:13:50	2	6.84 Кбайт	Завершено		

См.также:

Настройка сохранения изображений в отчетах

3.8 Журнал действий пользователей

Комплексный журнал действий с возможностью поиска является отличным дополнительным инструментом для управления пользователями, обеспечивающим подробный аудит их действий и повышающим защиту системы. Вы можете получить доступ к этой функции на вкладке *Журнал действий*.

NTECH LAB		Журнал действий		+		🔍 🌙 🔄 📄	
Объект		Действие		📄		Все фильтры	
Пользователь	IP-адрес	ID устройства	Действие	Объект	ID объекта	Время	Изображение
...	29.12.2023 14:29:19	🖼️
...	29.12.2023 14:26:53	🖼️
...	29.12.2023 14:26:51	🖼️
...	29.12.2023 14:26:49	🖼️
...	29.12.2023 13:30:21	🖼️
...	29.12.2023 13:30:20	🖼️
...	27.12.2023 15:26:13	🖼️

Каждая запись журнала содержит следующие данные:

- логин пользователя, выполнившего действие
- IP-адрес, с которого поступил запрос на выполнение действия
- ID устройства: уникальный идентификатор клиентского устройства пользователя
- тип действия, например, авторизация, поиск, изменение объекта, перезагрузка и т. д.
- тип объекта, к которому было приложено действие, например, карточка или происшествие
- идентификатор объекта
- подробности в зависимости от типа действия
- временная метка
- изображение (если применимо).

Для действий, связанных с использованием фотографии, запись журнала также содержит информацию об этой фотографии (включая миниатюру и полный кадр фотографии). Это относится к таким действиям, как поиск лица по изображению, удаление фотографии из карточки или добавление фотографии в карточку. При удалении карточки с фотографией также регистрируется удаление изображения, прикрепленного к карточке. При нажатии на кнопку 🖼️ отображается эта фотография.

Используйте панель фильтров сверху, чтобы задать условия поиска в журнале.

Вы можете создать отчет, нажав кнопку 📄 или задав фильтры, далее → *Применить и создать отчет*.

Дата и время ⚙️

01.01.1970 03:00

–

03.11.2023 12:42

⌵

×

IP-адрес ⚙️

Введите IP-адрес

ID устройства ⚙️

Введите ID устройства

Объект ⚙️

Выберите объект ⌵

Действие ⚙️

Выберите действие ⌵

ID объекта ⚙️

Введите ID объекта

Пользователь ⚙️

Введите имя пользователя

Применить и создать отчет

Сбросить фильтры

Применить фильтры

Отчет будет доступен для загрузки на вкладке *Отчеты*.

3.9 Удаленный мониторинг и удаленный поиск

FindFace CIBR предоставляет возможность извлечения событий распознавания лиц, совпадающих с карточками на локальном сервере, из удаленных систем распознавания лиц. Данная функциональность имеет широкую сферу возможного применения. Одним из направлений является отслеживание местонахождения и маршрутов преступников и выявление предполагаемых сообщников. Другим — поиск пропавших без вести людей. Функциональность спроектирована с учетом возможного подключения к системам общественной и транспортной безопасности с тысячами камер, но может применяться и в системах меньшего масштаба.

Удаленный мониторинг по умолчанию отключен. Для его включения см. раздел *Интеграция с удаленными системами распознавания лиц*.

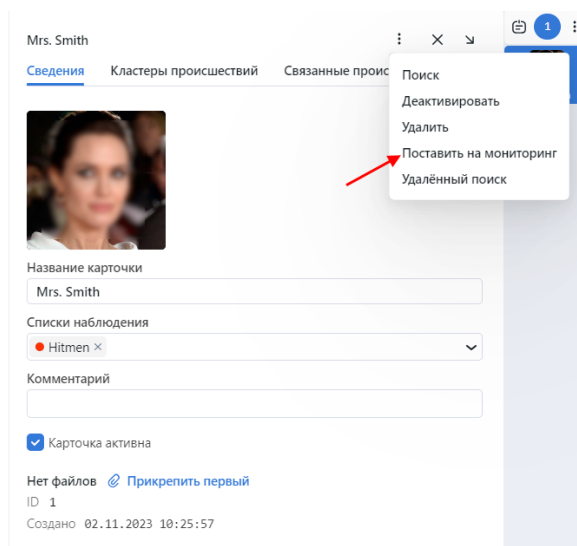
В этом разделе:

- *Постановка и снятие с мониторинга*
- *Просмотр оповещений удаленного мониторинга*
- *Поиск лиц в удаленных системах*
- *Ежедневный поиск*

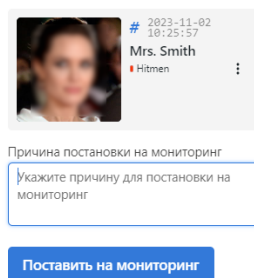
3.9.1 Постановка и снятие с мониторинга

Чтобы включить/отключить удаленные оповещения для определенного лица, выполните следующие действия:

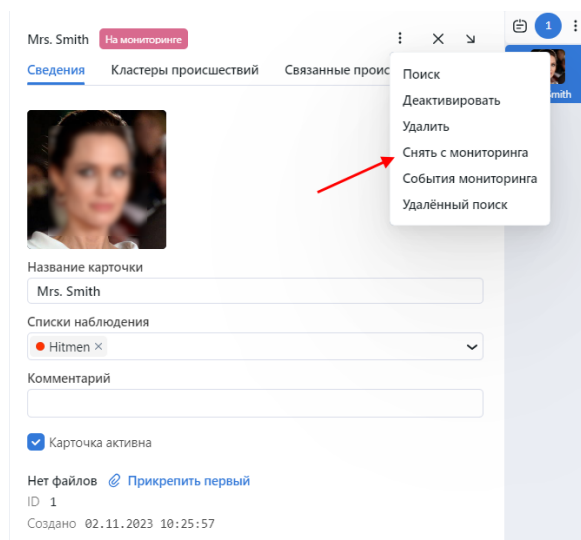
1. Перейдите на вкладку *Картотека*.
2. Откройте карточку лица.
3. Нажмите три точки → *Поставить на мониторинг*, чтобы включить удаленные оповещения.



4. Если вы включаете удаленные оповещения, укажите причину. Нажмите *Поставить на мониторинг*.



5. Нажмите три точки → *Снять с мониторинга*, чтобы отключить удаленные оповещения.

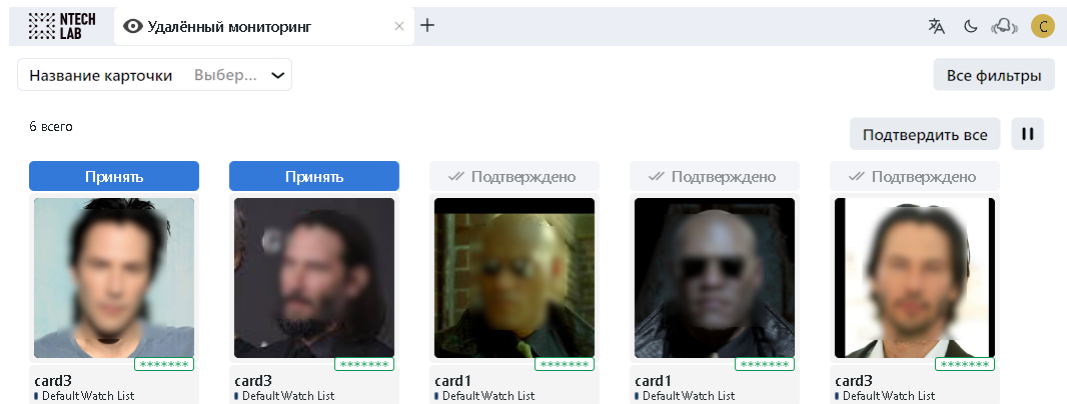


3.9.2 Просмотр оповещений удаленного мониторинга

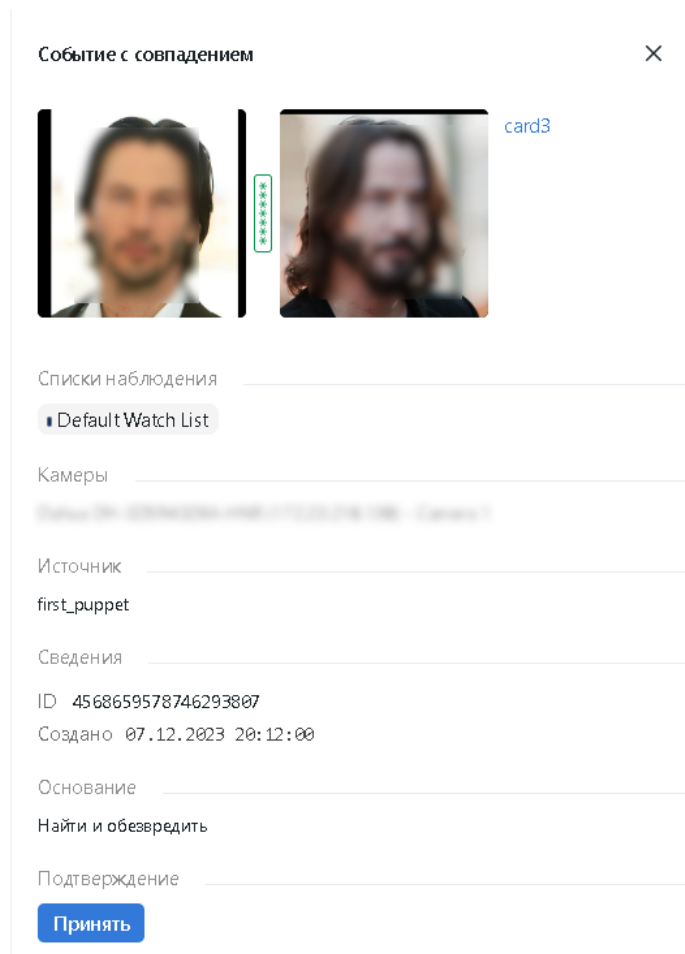
Просмотр оповещений от удаленных систем распознавания лиц производится на вкладке *Удалённый мониторинг* навигационной панели. Когда поступает новое событие, срабатывает звуковое оповещение, и после принятия события, оповещение прекращается. Вы также можете перейти в события удаленного мониторинга, нажав в колокольчике на верхней панели на *Показать события*. Вы можете прекратить воспроизведение звука оповещения, нажав *Выключить звук*.

Выполните следующие действия:

1. Перейдите на вкладку *Удалённый мониторинг*.



2. При необходимости отфильтруйте оповещения по дате и времени, а также по имени карточки, нажав *Все фильтры*.
3. Нажмите на изображение лица, чтобы перейти в боковую панель события с совпадением.



4. Нажмите на текстовую строку миниатюры, чтобы перейти в карточку лица.
5. Нажмите *Принять* для подтверждения события.

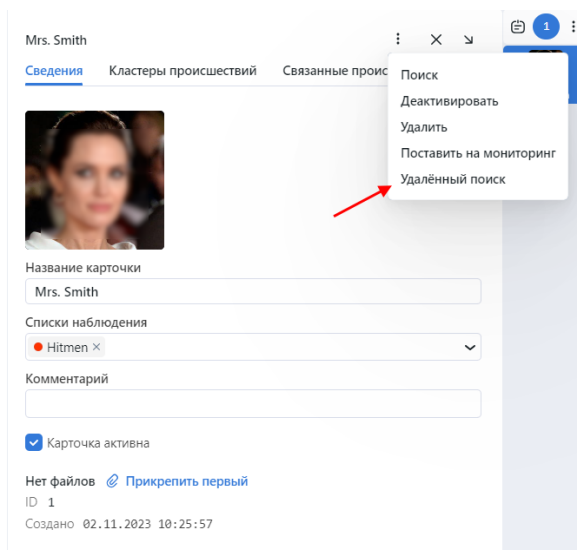
Примечание: Даже если одно событие на сервере-марионетке совпадет с несколькими фотографиями, связанными с одной и той же карточкой, в списке удалённого мониторинга будет не более одного события – того, у которого степень схожести выше (и в списке результатов мониторинга будет отображено соответствующее фото из карточки).

Если одно и то же событие на сервере-марионетке совпадет с несколькими фотографиями, но эти фотографии связаны с разными карточками, то в списке удалённого мониторинга будет столько событий, сколько имеется совпадений с карточками.

3.9.3 Поиск лиц в удаленных системах

Для поиска лица в удаленных системах выполните следующие действия:

1. Перейдите на вкладку *Карточка*.
2. Откройте карточку лица.
3. Нажмите три точки → *Удалённый поиск*.



4. Укажите условия поиска, такие как причина поиска, порог срабатывания, дата и время появления лица, а также максимальное количество результатов поиска. Нажмите *Выполнить*.

5. Результаты поиска будут отображены на вкладке *Удалённый поиск*.

7
всего

01.01.1970 03:00 – 29.12.2023 12:29

Имя пользователя Имя карточки

#7

0 событий

27.12.2023 15:26:13

Добавил(а) Charlie Root

2023-12-22 16:53:04

Дмитрий Кирюшин

Добавил(а) Charlie Root

#6

120 событий

27.12.2023 12:11:04

Добавил(а) Charlie Root

2023-12-21 20:26:48

Генри Димитров

Контроль

Вы можете отфильтровать события по дате и времени, названию карточки и пользователю, инициировавшему поиск.

6. Если вы нажмете на левую часть с информацией о событиях, вы перейдете к событиям сервера-марионетки.

← Поиск ↕ Сначала самые новые

01.01.1970 03:00 – 29.12.2023 12:48

#500

26.12.2023 12:26:09

#4573020196888961697

Office entrance #1

2023-12-21 20:26:48

Генри Димитров

Контроль

Результаты поиска содержат следующую информацию:

- дата события на сервере-марионетке
- название камеры сервера-марионетки
- название сервера-марионетки
- ID события

- миниатюра события лица
- карточка, по которой был удаленный поиск
- вероятность совпадения

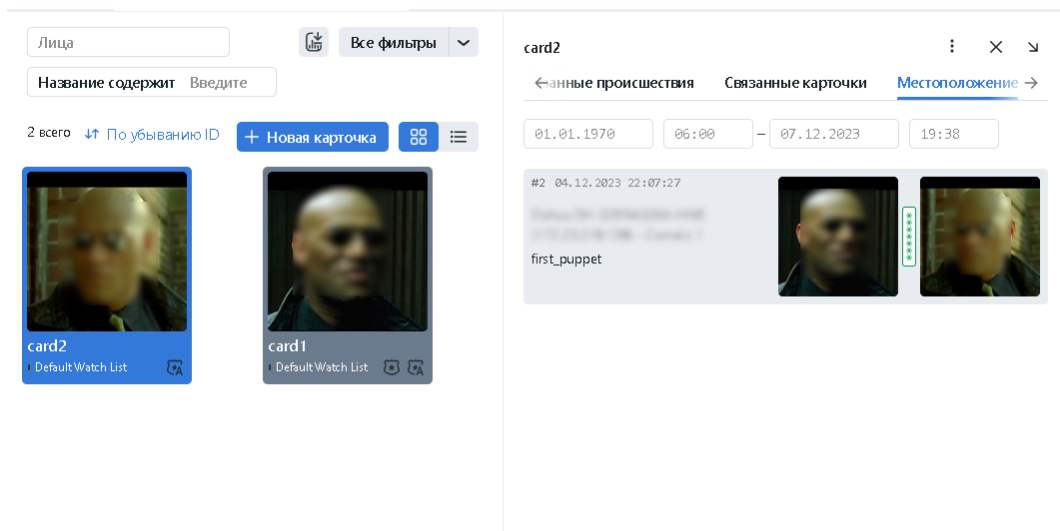
Вы можете отфильтровать события поиска по уровню новизны.

3.9.4 Ежедневный поиск

Ежедневный поиск позволяет серверу-кукловоду получать по расписанию события из сервера-марионетки с совпадениями с карточкой.

Чтобы использовать данную функциональность, см. [интеграцию с удаленными системами распознавания лиц](#) и выполните следующие действия:

1. Включите ежедневный поиск в конфигурационном файле `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` сервера-кукловода.
2. Включите параметры 'DAILY' в конфигурационном файле сервера-марионетки и укажите время для сопоставления событий с карточками, которые были получены от сервера-кукловода.
3. Перейдите во вкладку *Списки наблюдения* и установите флажок **Сбор данных о местоположении** для интересующего вас списка наблюдения. Для всех карточек, включенных в этот список, все подключенные серверы-марионетки возвращают ежедневно по заданному расписанию данные о местонахождении объектов интереса за последние 24 часа, а если флажок снимается, серверы-марионетки прекращают ежедневный мониторинг по карточкам этого списка.
4. Результаты будут показаны на вкладке *Местоположение* карточки человека и по умолчанию отсортированы в порядке убывания новизны.



Руководство по интеграции

Данная глава посвящена возможностям интеграции с FindFace. В текущей версии для интеграции своей системы можно использовать только HTTP API.

4.1 HTTP API

Подробная интерактивная документация HTTP API FindFace доступна после установки по адресу <http://<findface-ip:port>/api-docs>. Изучайте и пробуйте.

Совет: Документацию также можно найти в веб-интерфейсе, перейдя в меню по пунктам *Настройки* -> *Документация API*.
