
FindFace

Release 2.1.2

NtechLab

Apr 24, 2024

CONTENTS

1	What's New in FindFace CIBR 2.1.2	3
2	System Administrator's Guide	5
2.1	Architecture	5
2.2	Requirements	10
2.3	Deploy and Remove FindFace CIBR	12
2.4	Administration and Configuration	36
2.5	Maintenance and Troubleshooting	82
2.6	Appendices	114
3	User's Guide	117
3.1	Getting Started	117
3.2	Web Interface Basics	118
3.3	Record Index	119
3.4	Case Files	124
3.5	Search Faces in System	140
3.6	Compare Two Faces	143
3.7	Reports	143
3.8	Audit Log	146
3.9	Remote Alerting and Remote Search	147
4	Integrations	155
4.1	HTTP API	155

FindFace CIBR (Criminal Investigation Biometric Registry) is a forensic platform powered by a cutting-edge facial recognition technology. It is designed to conduct criminal investigations based on associated video footage and to search individuals across Public and Transport Safety systems.

FindFace CIBR forensic functionality

- **Biometric forensic databases.** Upload the databases to FindFace CIBR by creating a Record Index and assigning records of individuals under observation to relevant watch lists (Wanted, Fugitives, etc.). It can be done in bulk. A record accommodates aggregated data about an individual: a face biometric sample, document scans, criminal record, and other information.
- **Case files.** Upload a video footage of an incident or a photo to FindFace CIBR to detect and identify human faces in them. If there are individuals whose facial biometric data are in the forensic databases, FindFace CIBR will be able to spot them during this stage.

Go ahead and process the facial recognition results, using a built-in tool. Tell apart participants from bystanders and identify suspects and victims.
- **Search.** Search the system for specific individuals.
- **Remote alerting.** Combine FindFace CIBR with Public and Transport Safety systems. Receive real-time alerts on appearance of specific individuals from remote facial recognition systems. This will help track the offender's location and routes, detect alleged accomplices, find missing people.
- **Remote search.** Search specific individuals in remote facial recognition systems.
- **Facial verification.** Verify that two given faces belong to the same individual.
- **Reports.** Detailed reports on search results and records.

Technical features

- AI-based platform.
- Developer-friendly installer and user-friendly interface.
- Single- and multi-host deployment.
- Increased performance and fault-tolerance in high load systems with numerous cameras and clients.
- Network or on-premise licensing.
- CPU- and GPU-based acceleration for your choice.

System security

- Advanced user management.
- Comprehensive, friendly, searchable audit logs.
- Backup and recovery utilities.
- Possibility of monitoring user sessions and blocking devices without deactivating user accounts.

Useful little things

- Quick record index creation.
- Extended set of search filters.

Integration

- Integration via HTTP API.

WHAT'S NEW IN FINDFACE CIBR 2.1.2

FindFace CIBR 2.1.2 rests upon significant backend changes, introduced in the [previous release](#), and brings some exciting features and enhancements.

New Features:

- Features aimed at preventing abuse of authority and data leakage:
 - System media protection is always enabled and cannot be disabled.
 - For actions that involve usage of a photo, a log record now *displays that photo*.
- Other features:
 - Daily search cleanup functionality: helps save storage resources.
See *Integration with Remote Facial Recognition Systems*

Enhanced Algorithms, UI, UX:

- Search improvements: search for an object of interest among all recognized faces, and not only among the participants of a case.
See *Search Faces in System*
- A single event on a puppet results in a single event on a puppeteer, the one with the highest confidence match.
See *Remote Alerting and Remote Search*
- License limits do not apply to the archived cases.
See:
 - *Case File Archive*
 - *License Limits*
- While processing a video within a case, follow the actual progress of video processing in a progress bar.
See *Case Files*

SYSTEM ADMINISTRATOR'S GUIDE

This chapter is all about FindFace CIBR deployment and further updates and maintenance during exploitation.

2.1 Architecture

Though you mostly interact with FindFace CIBR through its web interface, be sure to take a minute to learn the FindFace CIBR architecture. This knowledge is essential for the FindFace CIBR deployment, integration, maintenance, and troubleshooting.

In this chapter:

- *Recognition Process*
- *Docker Platform*
- *Architectural Elements*
 - *Architecture scheme*
 - *FindFace core*
 - *Application Module (FindFace CIBR)*
- *CPU- and GPU-acceleration*

2.1.1 Recognition Process

FindFace CIBR detects a human face in the photo or video and prepares its image through normalization. The normalized image is then used for extracting the face's feature vector (an n-dimensional vector of numerical features that represent the face). Face feature vectors are stored in the database and further used for verification and identification purposes.

2.1.2 Docker Platform

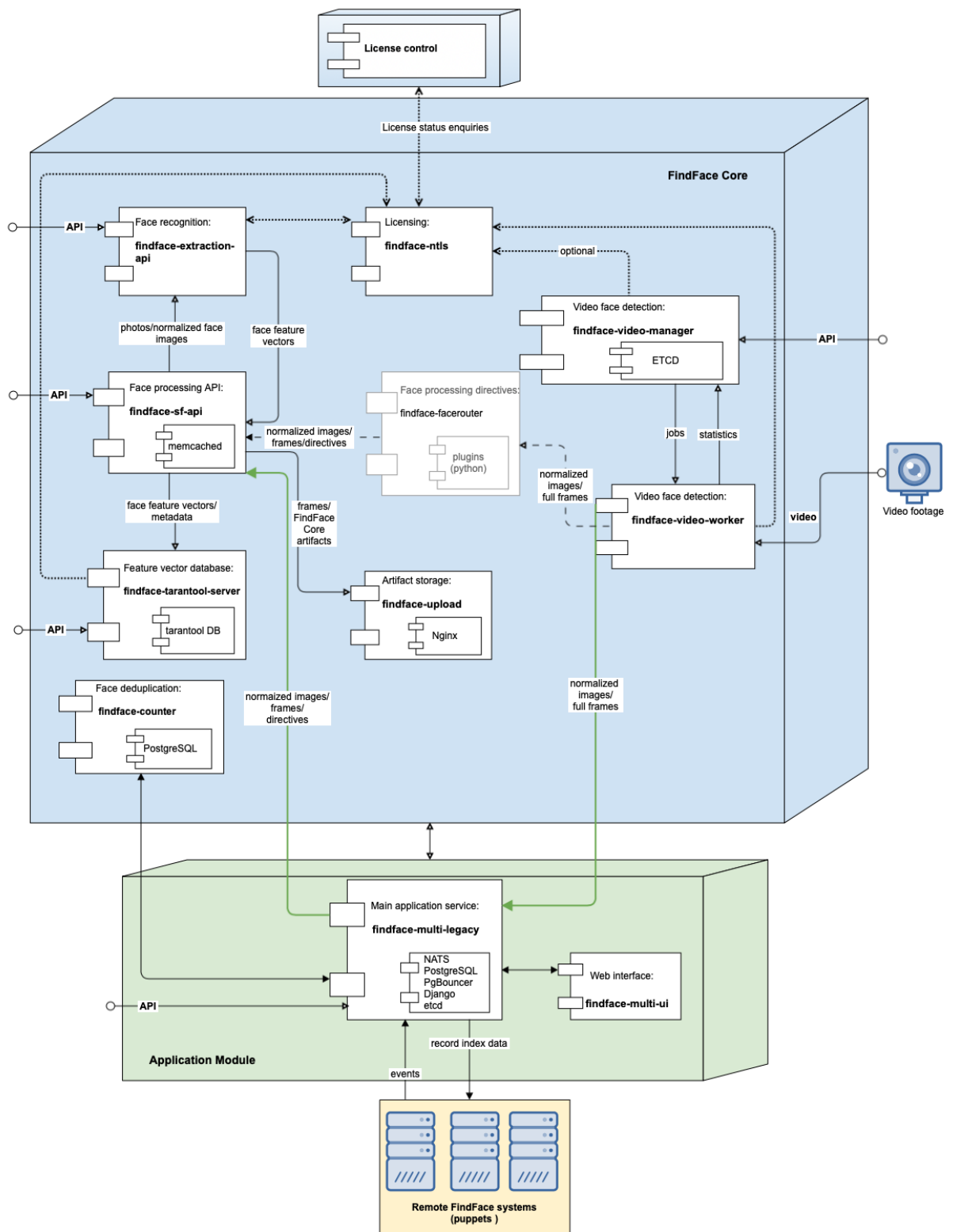
FindFace CIBR is deployed in Docker, a set of platform as a service (PaaS) products that use OS-level virtualization to deliver software in packages called containers. Each FindFace CIBR service runs in a Docker container.

2.1.3 Architectural Elements

FindFace CIBR consists of the following fundamental architectural elements:

- FindFace core, a cutting-edge AI-based recognition technology that can be used as a separate product [FindFace Enterprise Server](#).
- Application Module (FindFace CIBR), implementing a set of tools for criminal investigations based on video footage and photo files.

Architecture scheme



FindFace core

There is a separate product [FindFace Enterprise Server](#) inside the FindFace core. It includes the following components:

Component	Ports in use	Description	Vendor
findface-extraction-api	18666	Service that uses neural networks to detect a face in an image and extract its feature vector. It also recognizes face attributes, for example, gender, age, emotions, beard, glasses, etc. CPU- or GPU-acceleration.	NtechLab own deployment
findface-sf-api	18411	Service that implements the internal HTTP API for face detection and recognition.	
findface-tarantool-server	32001, shard ports (default 330xx, 81xx)	Service that provides interaction between the <code>findface-sf-api</code> service and the feature vector database (the Tarantool-powered database that stores face feature vectors).	
findface-upload	3333	NginX-based web server used as a storage for original images, thumbnails, and normalized face images.	
findface-facerouter	18820	Service used to define processing directives for detected faces. In FindFace CIBR, its functions are performed by <code>findface-multi-legacy</code> (see Application Module (FindFace CIBR)).	
findface-video-manager	18810, 18811	Service, part of the video face detection module, that is used for managing the video face detection functionality, configuring the video face detector settings and specifying the list of to-be-processed video files.	
findface-video-worker	18999	Service, part of the video face detection module, that recognizes a face in the video and posts its normalized image, full frame and metadata (such as detection time) to the <code>findface-facerouter</code> service for further processing according to given directives. CPU- or GPU-acceleration.	
findface-ntls	443 (TCP), 3133, 3185	License server that interfaces with the NtechLab Global License Server, a USB dongle, or hardware fingerprint to verify the license of your FindFace CIBR instance.	
findface-counter	18300	Service used for event deduplication.	
Tarantool	Shard ports (default 330xx, 81xx)	Third-party software that implements the feature vector database that stores extracted face feature vectors and identification events. The system data, records, user accounts are stored in PostgreSQL (part of the FindFace CIBR application module).	Tarantool
etcd	2379	Third-party software that implements a distributed key-value store for <code>findface-video-manager</code> . Used as a coordination service in the distributed system, providing the video face detector with fault tolerance.	etcd
NginX	80; SSL: 8002, 8003, 443, 80	Third-party software that implements the system web interfaces.	nginx
mem-cached	11211	Third-party software that implements a distributed memory caching system. Used by <code>findface-sf-api</code> as a temporary storage for extracted face feature vectors before they are written to the feature vector database powered by Tarantool.	mem-cached

Application Module (FindFace CIBR)

The FindFace CIBR application module includes the following components:

Component	Ports in use	Description	Vendor
findface-multi-legacy	Configurable	Service that serves as a gateway to the FindFace core. Provides interaction between the FindFace core and the web interface, the system functioning as a whole, HTTP and web socket, face monitoring, event notifications, etc.	Ntech-Lab own deployment
findface-multi-pause	n/a	Internal services that assist findface-multi-legacy. The findface-multi-audit service is created for the future. It will become full-fledged in the upcoming versions. As for now, it partly duplicates the findface-multi-legacy functionality. Use findface-multi-legacy to work with it. The findface-multi-identity-provider is the service for authentication, user management, and management of role-based model of accesses.	
findface-multi-audit	8012, 8013, 8014		
findface-multi-identity-provider	8022, 8023, 8024		
cleaner	n/a	The service is responsible for the data cleanup.	
findface-multi-ui	Configurable	Main web interface used to interact with FindFace CIBR. Based on the Django framework . Allows you to work with face recognition events, search for faces, manage cases, users, record index, watch lists, and many more.	
NATS	4222	Third-party software that implements a message broker in findface-multi-legacy.	NATS
etcd	2379	Third-party software that implements locks in the findface-multi-legacy service, such as locks in NTLS checker, reports, video processing, etc.	etcd
Pg-bouncer	5439	Third-party software, a lightweight connection pooler for PostgreSQL. Optional, used to increase the database performance under high load.	Pg-Bouncer
PostgreSQL	5432	Third-party software that implements the main system database. This database stores records of individuals and data for internal use. The face feature vectors and face identification events are stored in Tarantool (part of the FindFace core).	PostgreSQL

See also:

- [FindFace CIBR Data Storages](#)

2.1.4 CPU- and GPU-acceleration

The findface-extraction-api and findface-video-worker services can be either CPU- or GPU-based. You will have an opportunity to choose the acceleration type you need during the [installation](#).

Important: Refer to [Requirements](#) when choosing hardware configuration.

Important: If video resolution is more than 1280x720px, it is strongly recommended to use the GPU-accelerated package findface-video-worker.

2.2 Requirements

In this chapter:

- *System Requirements for Basic Configuration*
- *Required Administrator Skills*
- *Video File Formats*

2.2.1 System Requirements for Basic Configuration

To calculate the FindFace CIBR host(s) characteristics, use the requirements provided below.

Tip: Be sure to learn about the FindFace CIBR *architecture* first.

Important: If the video resolution is more than 1280x720px, it is strongly recommended to use the GPU-accelerated package `findface-video-worker-gpu`.

Important: On AMD CPU servers, the full functionality of the CPU-accelerated `findface-extraction-api` service is not guaranteed. Use the GPU-accelerated service `findface-extraction-api-gpu` along with the GPU-version of neural networks instead.

Note: In case of a high-load system (~> 15 events per second), be sure to use an SSD.

	Minimum	Recommended
CPU	Intel Core i5 CPU with 4+ physical cores 3+ GHz. AVX2 support	Intel Xeon Silver/Gold with 6+ physical cores
	The own needs of FindFace CIBR require 2 cores HT > 2.5 GHz. The characteristics also depend on the number of video files in process. A single video file 720p@25FPS requires 2 cores >2.5 GHz. AVX2 support.	
GPU (optional)	GeForce® RTX 3060 12 GB	NVIDIA A10
	Supported devices: NVIDIA, Pascal architecture and above. <i>Note: NVIDIA GeForce RTX 40 Series graphics cards are currently not supported.</i>	
RAM	16 Gb	32+ Gb
	The RAM consumption depends on: <ul style="list-style-type: none"> • the number of algorithms being used, • the number of selected attributes, • the number of faces being processed, etc. A single video file 720p@25FPS requires 2 GB RAM. Please, contact our support team for details (support@ntechlab.com).	
HDD (SSD for best performance)	65 Gb	65+ Gb
	The own needs of FindFace CIBR require 65 GB. The total volume is the subject to the required depth of the event archive in the database and in the log, at the rate of 1.5 Mb per 1 event.	
Operating system	Ubuntu Server / Desktop from 18 to 22, x64 only, RHEL, CentOS 7, Debian 11.	

Note: You can also use an Intel-based VM if there is AVX2 support, and eight physical cores are allocated exclusively to the VM.

Tip: For more accurate hardware selection, contact our support team by support@ntechlab.com.

2.2.2 Required Administrator Skills

A FindFace CIBR administrator must know and understand the OS, on which the product instance is deployed, at the level of an advanced user.

2.2.3 Video File Formats

FindFace CIBR supports a wide variety of file formats depending on the acceleration type, CPU or GPU.

Both CPU- and GPU-accelerated instances support all FFmpeg codecs. In addition to that, the following codecs are supported:

- *CPU-based acceleration:* FLV (both as a codec and as a container), H263, H264, H265, MJPEG, VP8, VP9, MPEG1VIDEO, MPEG2VIDEO, MSMPEG4v2, MSMPEG4v3.
- *GPU-based acceleration:* MJPEG, H264, H265, VP9, and others, depending on the list of codecs supported by the used video card. Apart from that, for instances with `video-worker-gpu`, you can expand the number of supported codecs by enabling video decoding on the CPU, which is not available by default.

To enable video decoding on the CPU for GPU-based acceleration, do the following:

1. Open the `/opt/findface-cibr/configs/findface-video-worker/findface-video-worker.yaml` file.

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

2. Set `cpu: true` in the `video_decoder` section.

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

3. Restart the `findface-cibr-findface-video-worker-1` container.

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

2.3 Deploy and Remove FindFace CIBR

Docker platform

Starting from version 2.1.1 FindFace CIBR is Docker-based. You must install and start a set of Docker products before proceeding with the FindFace CIBR deployment. For your convenience, this chapter contains [Ubuntu](#), [CentOS](#), and [Debian](#) server preparation sections covering the intricacies of installing Docker on the above-mentioned operating systems. For other platforms, please refer to the [Docker documentation](#).

NVIDIA driver and NVIDIA Container Runtime (GPU only)

If you intend to deploy FindFace CIBR with GPU-acceleration, you need to install the NVIDIA driver and NVIDIA Container Runtime. Again, you will find the relevant information in the server preparation sections.

Deployment options

After you are finished with the server preparation, you are all set for the FindFace CIBR deployment. You are provided with the following options here:

1. Automatically install FindFace CIBR standalone in one run. Being the simplest, this installation type is excellent to kick-start your experience with FindFace CIBR. We recommend choosing it if you are a first-timer. See [FindFace CIBR Standalone Automated Deployment](#) for guidance.
2. Fully customized installation. It requires fundamental understanding of the product architecture. See [Fully Customized Installation](#).

Note: If you select the installation type #2, keep in mind to *install necessary neural network models* along with the `findface-extraction-api` component.

Installer questions and automatic deployment from the installation file

Before starting the actual installation, the installer asks you a few questions and performs several automated checks to ensure that the host meets the system requirements. After filling out each prompt, press **Enter**.

To install the same configuration of FindFace CIBR on a different host, use the automatic deployment from the installation file. In this case, you won't have to answer the installation questions again. The exact path to the installation file is displayed right after the last question, before the installer starts off active progress:

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

Important: Be sure to remove fields `*.config`, `exp_ip`, and `int_ip` before installing FindFace CIBR on a host with a different IP address.

To launch the automatic deployment from the `/tmp/<findface-installer-*>.json` file, execute:

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

Post-development procedures and how-to's

Browse through the *Post-deployment Procedures and How-to's* section to learn how to set the time zone, license your instance, and configure logging. This section will also provide you with few basic commands that will help you to kick-start your work with FindFace CIBR containers, in case you are a newbie to Docker.

Important: Starting the GPU-accelerated services `findface-extraction-api` and `findface-video-worker` for the first time after deployment may take a considerable amount of time due to the caching process (up to 45 minutes).

Important: Although FindFace CIBR provides tools to ensure its protection from unauthorized access, they are not replacing a properly configured firewall. Be sure to use a firewall to heighten the FindFace CIBR network protection.

Instance removal

To remove your instance, you must run a set of commands. See the *Remove FindFace CIBR Instance* section.

2.3.1 Ubuntu Server Preparation

To prepare a server on Ubuntu for the FindFace CIBR deployment, follow the instructions below minding the sequence.

Note: For other platforms, please refer to the following resources:

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

In this section:

- *GPU: Install NVIDIA Drivers*
- *Install Docker Products*
- *GPU: Install NVIDIA Container Runtime*

GPU: Install NVIDIA Drivers

The first step of the server preparation is the NVIDIA driver installation. It's applicable only for the GPU configuration. Go straight to the [Docker installation](#) if your configuration is CPU-accelerated.

With the GPU-accelerated FindFace CIBR, you'll need the NVIDIA driver version 530 or above. Add the NVIDIA repository and install an applicable driver from it.

Warning: We do not recommend using a `.run` installer from [NVIDIA Driver Downloads](#) instead, as its drivers may conflict with the drivers installed via packages.

To install the 530 driver from a repository, do the following:

1. Install the repository signature key:

```
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. Install aptitude:

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

3. Install nvidia-driver-530:

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

4. Reboot the system:

```
sudo reboot
```

Install Docker Products

Docker products must be installed on both CPU and GPU servers. Do the following:

1. Update the apt package index and install packages to allow apt to use a repository over HTTPS.

```
sudo apt-get update

sudo apt-get install \
  ca-certificates \
  curl \
```

(continues on next page)

(continued from previous page)

```
gnupg \
lsb-release
```

Tip: You may receive the following errors when running `sudo apt-get install \`:

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

There are two ways to resolve them:

1. Forcibly terminate all the `apt-get` processes currently running in the system.

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

2. If the previous command didn't help, run the set of commands below. It's fine if some of the to-be-deleted directories do not exist — just keep going.

```
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-frontent
sudo dpkg --configure -a
```

2. Add the Docker's official GPG key (GNU Privacy Guard key) to the host.

```
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. Set up the Docker repository.

```
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. Update the apt package index one more time.

```
sudo apt-get update
```

Tip: If you have received a GPG error when running this command, try granting read permission for the Docker public key file before updating the package index.

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

5. Install the latest versions of Docker products.

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

Note: The most recent versions that have been tested are 26.* versions. Use later versions of Docker packages, if any, at your own discretion.

6. Check whether the Docker installation was a success. The following command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

```
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. Install 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. Perform the [Docker Engine post-installation procedures](#) to ease your future work with Docker and *FindFace CIBR containers*. Once you can manage Docker as a non-root user, you don't have to apply `sudo` in the commands related to Docker.

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

9. Configure the Docker network.

```

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

```

GPU: Install NVIDIA Container Runtime

To deploy containerized GPU-accelerated FindFace CIBR, you need NVIDIA Container Runtime. We recommend installing NVIDIA Container Toolkit that includes this runtime. Do the following:

1. Specify the repository and install NVIDIA Container Toolkit from it by executing the following commands.

```

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-ctl runtime configure --runtime=docker
sudo systemctl restart docker

```

2. Configure the Docker network. Configure usage of the NVIDIA Container Runtime installed along with 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. Restart Docker.

```
systemctl restart docker
```

Now you are all set to install FindFace CIBR. Refer to the following sections:

- *FindFace CIBR Standalone Automated Deployment*
- *Fully Customized Installation*

2.3.2 CentOS 7 Server Preparation

To prepare a server on CentOS 7 for the FindFace CIBR deployment, follow the instructions below minding the sequence.

Note: For other platforms, please refer to the following resources:

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

In this section:

- *Install Updates*
- *GPU: Install NVIDIA Drivers*
- *Install Docker Products*
- *GPU: Install NVIDIA Container Runtime*

Install Updates

1. Run updates and reboot the server.

```
sudo yum update  
sudo reboot
```

2. Install fuse by running the following command.

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

GPU: Install NVIDIA Drivers

The first step of the server preparation is the NVIDIA driver installation. It's applicable only for the GPU configuration. Go straight to the [Docker installation](#) if your configuration is CPU-accelerated.

With the GPU-accelerated FindFace CIBR, you'll need the NVIDIA driver version 530 or above. Download a relevant .run installer from [NVIDIA Driver Downloads](#).

Since you use a .run installer from NVIDIA, the following dependencies must be installed:

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

Install Docker Products

Docker products must be installed on both CPU and GPU servers. Do the following:

1. Install the yum-utils package (which provides the yum-config-manager utility) and set up the repository.

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

2. Install the latest versions of Docker products.

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

Note: The most recent versions that have been tested are 26.* versions. Use later versions of Docker packages, if any, at your own discretion.

3. Start and enable Docker.

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

4. Check whether the Docker installation was a success. The following command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

```
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)
```

(continues on next page)

(continued from previous page)

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. Install 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. Perform the [Docker Engine post-installation procedures](#) to ease your future work with Docker and *FindFace CIBR containers*. Once you can manage Docker as a non-root user, you don't have to apply `sudo` in the commands related to Docker.

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

7. Configure the Docker network and devicemapper usage.

```
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: Install NVIDIA Container Runtime

To deploy containerized GPU-accelerated FindFace CIBR, you need NVIDIA Container Runtime. We recommend installing NVIDIA Container Toolkit that includes this runtime. Do the following:

1. Specify the repository and install NVIDIA Container Toolkit from it by executing the following commands.

```
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
```

(continues on next page)

(continued from previous page)

```
sudo yum install -y nvidia-container-toolkit
sudo nvidia-ctl runtime configure --runtime=docker
sudo systemctl restart docker
```

2. Configure the Docker network, devicemapper usage, and usage of the NVIDIA Container Runtime installed along with 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. Restart Docker.

```
systemctl restart docker
```

Now you are all set to install FindFace CIBR. Refer to the following sections:

- [FindFace CIBR Standalone Automated Deployment](#)
- [Fully Customized Installation](#)

2.3.3 Debian 11 Server Preparation

To prepare a server on Debian 11 for the FindFace CIBR deployment, follow the instructions below minding the sequence.

Note: For other platforms, please refer to the following resources:

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

In this section:

- *Install FUSE*
- *GPU: Install NVIDIA Drivers*
- *Install Docker Products*
- *GPU: Install NVIDIA Container Runtime*

Install FUSE

1. Install FUSE (Filesystem in Userspace) by running the following command.

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

GPU: Install NVIDIA Drivers

The first step of the server preparation is the NVIDIA driver installation. It's applicable only for the GPU configuration. Go straight to the [Docker installation](#) if your configuration is CPU-accelerated.

With the GPU-accelerated FindFace CIBR, you'll need the NVIDIA driver version 530 or above. Download a relevant .run installer from [NVIDIA Driver Downloads](#).

Since you use a .run installer from NVIDIA, the following dependencies must be installed:

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

Install Docker Products

Docker products must be installed on both CPU and GPU servers. Do the following:

1. Update the apt package index and install packages to allow apt to use a repository over HTTPS.

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

2. Add the Docker's official GPG key (GNU Privacy Guard key) to the host.

```
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. Set up the Docker repository.

```
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. Update the apt package index one more time.

```
sudo apt-get update
```

5. Install the latest versions of Docker products.

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

Note: The most recent versions that have been tested are 26.* versions. Use later versions of Docker packages, if any, at your own discretion.

6. Check whether the Docker installation was a success.

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

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

7. Install 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. Perform the [Docker Engine post-installation procedures](#) to ease your future work with Docker and *FindFace CIBR containers*. Once you can manage Docker as a non-root user, you don't have to apply `sudo` in the commands related to Docker.

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

9. Configure the Docker network.

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

GPU: Install NVIDIA Container Runtime

To deploy containerized GPU-accelerated FindFace CIBR, you need NVIDIA Container Runtime. We recommend installing NVIDIA Container Toolkit that includes this runtime. Do the following:

1. Specify the repository and install NVIDIA Container Toolkit from it by executing the following commands.

```
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-ctl runtime configure --runtime=docker  
sudo systemctl restart docker
```

2. Switch to the superuser account.

```
sudo su
```

3. Configure the Docker network. Configure usage of the NVIDIA Container Runtime installed along with NVIDIA Container Toolkit.

```
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
```

4. Restart Docker.

```
systemctl restart docker
```

Now you are all set to install FindFace CIBR. Refer to the following sections:

- [FindFace CIBR Standalone Automated Deployment](#)
- [Fully Customized Installation](#)

2.3.4 FindFace CIBR Standalone Automated Deployment

To deploy FindFace CIBR as a standalone server in one run, follow the instructions below. Being the simplest, this installation type is excellent to start off your work with FindFace CIBR. Be sure to meet the *system requirements* and, depending on your OS, prepare the server first:

See:

- [Ubuntu Server Preparation](#)
- [CentOS 7 Server Preparation](#)
- [Debian 11 Server Preparation](#)

Important: The FindFace CIBR host must have a static IP address in order to be running successfully. To make the IP address static, open the `/etc/network/interfaces` file and modify the current primary network interface entry as shown in the case study below. Be sure to substitute the suggested addresses with the actual ones, subject to your network specification.

```
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
```

Restart networking.

```
sudo service networking restart
```

Be sure to edit the `etc/network/interfaces` file with extreme care. Please refer to the Ubuntu [guide on networking](#) before proceeding.

Do the following:

1. Download the installer file `findface-*.run`.
2. Put the `.run` file into some directory on the designated host (for example, `/home/username`).
3. From this directory, make the `.run` file executable.

Note: Be sure to specify the actual file name instead of `findface-*`.

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

4. Execute the `.run` file.

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

The installer will ask you a few questions and perform several automated checks to ensure that the host meets the system requirements. After filling out each prompt, press Enter. The questions and answers are the following:

1. Q: Which product should be installed?
A: 1

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

(default: cibr)

```
product> 1
```

2. Q: Please choose installation type:

A: 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. Q: Directory to install into:

A: Specify the FindFace CIBR installation directory. By default, it's /opt/findface-cibr. Press Enter to confirm it. Otherwise, specify the directory of your choice and press Enter.

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

4. Q: Found X interface(s). Which one should we announce as our external address?

A: Choose the interface that you are going to use as the instance IP address.

```
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. Q: Which variant of Video Worker should be installed?

A: Specify the findface-video-worker package type, CPU or 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. Q: Which variant of Extraction API should be installed?

A: Specify the findface-extraction-api package type, CPU or 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. Q: Do you want to configure face features right now?(y/n)

A: We recommend you to configure face attribute recognition functionality straightaway during the installation. Answer y to initiate the process. You can skip it, though, by answering n and perform the necessary steps later, following the instructions in the [Enable Face Attribute Recognition](#) section.

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

8. Q: Please select face features to install:

A: This question appears if you have requested configuration of face attributes. By default, all face attributes are subject to installation. Answer done to confirm it. If some attribute is not necessary, you can enter the keyword (number) related to it. For example, enter 3 to exclude recognition of emotions. Then enter 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. Q: Please set findface-cibr admin password

A: Set the Super Administrator (superuser) password.

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

After the final question you will see a path to a JSON file with your answers:

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

The installer will pull the FindFace CIBR images from the Ntechlab registry and start the following services in Docker containers:

Service	Container	Configuration
pause	findface-cibr-pause-1	Started
nats-jetstream	findface-cibr-nats-jetstream-1	Started
mon-godb	findface-cibr-mongodb-1	Started
findface-ntls	findface-cibr-findface-ntls-1	Started
nats	findface-cibr-nats-1	Started
post-gresql	findface-cibr-postgresql-1	Started
mem-cached	findface-cibr-memcached-1	Started
findface-upload	findface-cibr-findface-upload-1	Started
etcd	findface-cibr-etcd-1	Started
findface-sf-api	findface-cibr-findface-sf-api-1	Started
findface-extraction-api	findface-cibr-findface-extraction-api-1	Started (CPU/GPU-acceleration)
findface-tarantool-server-shard-*	findface-cibr-findface-tarantool-server-shard-*-1	Started. The number of shards is calculated using the formula: $N = \min(\max(\min(\text{mem_mb} // 2000, \text{cpu_cores}), 1), 16 * \text{cpu_cores})$. I.e., it is equal to the RAM size in MB divided by 2000, or the number of CPU physical cores (but at least one shard), or the number of CPU physical cores multiplied by 16, if the first obtained value is greater.
findface-video-manager	findface-cibr-findface-video-manager-1	Started
findface-counter	findface-cibr-findface-counter-1	Started
pg-bouncer	findface-cibr-pgbouncer-	Started

After the installation is complete, the following output is shown in the console:

Tip: Be sure to save this data: you will need it later.

```
#####
#                               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. Perform the *post-deployment procedures*.

Tip: To install the same configuration of FindFace CIBR on a different host, use the automatic deployment from the installation file. In this case, you won't have to answer the installation questions again. The exact path to a JSON file with answers is displayed right after the last question, before the installer starts active progress:

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

Be sure to remove fields `*.config`, `exp_ip`, and `int_ip` before installing FindFace CIBR on a host with a different IP address.

To launch the automatic deployment from the `/tmp/<findface-installer-*>.json` file, execute:

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

2.3.5 Fully Customized Installation

The FindFace CIBR developer-friendly installer provides you with a few installation options, including the fully customized installation. This option is mostly used when deploying FindFace CIBR in a highly distributed environment and requires a high level of knowledge and experience.

To initiate the fully customized installation, do the following:

1. Download the installer file `findface-*.run`.
2. Put the `.run` file into some directory on the designated host (for example, `/home/username`).
3. From this directory, make the `.run` file executable.

Note: Be sure to specify the actual file name instead of `findface-*`.

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

4. Execute the `.run` file.

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

The installer will ask you a few questions and perform several automated checks to ensure that the host meets the system requirements. After filling out each prompt, press Enter. The questions and answers are the following:

1. Q: Which product should be installed?

A: 1

```
Which product should be installed?

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

(default: cibr)
product> 1
```

2. Q: Please choose installation type:

A: 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. Q: Directory to install into:

A: Specify the FindFace CIBR installation directory. The default suggestion is `/opt/findface-cibr`. Press Enter to confirm it. Otherwise, specify the directory of your choice and press Enter.

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

4. Q: Please enter path to docker-compose binary:

A: Specify the actual path to the `docker-compose` binary. The default suggestion is `/usr/local/bin/docker-compose` and it is the path you get when you install `docker-compose` during *Ubuntu / CentOS / Debian* server preparation. Press Enter to confirm it. Otherwise, specify another path and press Enter.

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

5. Q: Found X interface(s). Which one should we announce as our external address?

A: Choose the interface that you are going to use as the instance IP address.

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

(continues on next page)

(continued from previous page)

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

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

6. Q: Found X interface(s). Which one should we announce as our inter-service communication address?

A: Choose the interface for inter-service communication.

```
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. Q: Please select FindFace CIBR components to install:

A: Choose FindFace CIBR components to install. By default, all components are subject to installation. You can leave it as is by entering `done`, or select specific components. Whenever you have to make a selection, first, deselect all the listed components by entering `-*` in the command line, then select required components by entering their sequence number (keyword), for example: `1 7 13`, etc. Enter `done` to save your selection and proceed to another step.

Warning: The pause component hosts a network namespace for other components. We do not recommend excluding it from installation, as this will leave the remaining components without a network namespace.

If you exclude the pause component intentionally, make sure to edit the `/opt/findface-cibr/docker-compose.yaml` file and provide a host name for each service in the `network_mode` parameter.

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. Specific questions related to the selected components: acceleration type, the required number of component instances, neural network models, etc. If you are experiencing a difficulty answering them, try to find an answer in this documentation, or submit your question to support@ntechlab.com.

9. Q: Please set findface-cibr admin password

A: Set the Super Administrator (superuser) password.

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

The installer will pull the FindFace CIBR images from the Ntechlab registry and start the associated services in Docker containers.

5. Perform the *post-deployment procedures*.

Tip: To install the same configuration of FindFace CIBR on a different host, use the automatic deployment from the installation file. In this case, you won't have to answer the installation questions again. The exact path to the installation file is displayed right after the last question, before the installer starts active progress:

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

Be sure to remove fields `*.config`, `exp_ip`, and `int_ip` before installing FindFace CIBR on a host with a different IP address.

To launch the automatic deployment from the `/tmp/<findface-installer-*>.json` file, execute:

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

2.3.6 Installation of Neural Network Models

To detect and recognize faces and face attributes, `findface-extraction-api` uses neural networks.

If you want to manually initiate the installation of neural network models, use the console installer as follows:

1. Execute the `findface-*.run` file.

Note: Be sure to specify the actual file name instead of `findface-*`.

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

2. Product to install: FindFace CIBR
3. Select the installation type: Fully customized installation.
4. Select a FindFace CIBR component to install: `findface-data`. To do so, first, deselect all the listed components by entering `-*` in the command line, then select the required component by entering its sequence number (keyword). Enter `done` to save your selection and proceed to the next step.
5. In the same manner, select models to install. After that, the installation process will automatically begin.

You can find installed models for the face and face attribute recognition at `/opt/findface-cibr/models/`. See *Neural Networks Summary*.

2.3.7 Post-deployment Procedures and How-to's

After you are finished with the FindFace CIBR deployment, perform the procedures below.

In this section:

- *Specify Time Zone*
- *License Instance*
- *Configure Logging*
- *Useful Docker Commands*

Specify Time Zone

The time zone on the FindFace CIBR server determines the time in reports, logs, and names of such FindFace CIBR artifacts as detection full frames and thumbnails, etc.

The time zone is specified in the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file, either in the Region/Country/City or Etc/GMT+H format. The best way to do so is to copy/paste your time zone from [this table](#) on Wikipedia.

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

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

Restart the `findface-cibr-findface-multi-legacy-1` container.

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

License Instance

FindFace CIBR provides several licensing options. Whichever option you choose, you upload the FindFace CIBR license file via the main web interface `http://<Host_IP_address>/#/license`. To access the web interface, use the superuser credentials.

Important: Do not disclose the superuser (Super Administrator) credentials to others. To administer the system, create a new user with administrator privileges. Whatever the role, the Super Administrator cannot be deprived of its rights.

Refer to the [Licensing](#) section to learn about the licensing options available.

Configure Logging

By default, the FindFace CIBR processes are logged to [Docker container logs](#), which can be accessed via the `docker logs` and `docker service logs` commands. In addition, Docker uses the [json-file logging driver](#), which caches container logs in JSON. You can configure Docker to use another logging driver, choosing from the [multiple logging mechanisms available](#). See [Logging](#) to learn how to do it.

Useful Docker Commands

In order to efficiently and easily administer FindFace CIBR, you must have extensive knowledge and skills with Docker. If you're new to Docker, get started with the commands below. Then explore the Docker documentation for additional skills.

- View all Docker containers, including the stopped ones:

```
docker ps -a
```

To get a more compact and understandable output, execute:

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

To extend the previous output, execute:

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

- Restart the Docker service:

```
sudo systemctl restart docker
```

- View a container log if the `journald` logging driver is *enabled*:

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

- Stop a Docker container:

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

Stop all Docker containers:

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

- Start a Docker container:

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

Start all Docker containers:

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

- The FindFace CIBR `docker-compose.yaml` file can be viewed as such:

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

- FindFace CIBR configuration files can be found [here](#):

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

Once you made changes to a configuration file, restart a relevant container by executing:

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

- Enter a running Docker container to execute a command in it:

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

- Stop and remove all FindFace CIBR containers:

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

- Build, recreate, and start FindFace CIBR containers:

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

2.3.8 Remove FindFace CIBR Instance

If you deployed FindFace CIBR *standalone from the installer*, use the `uninstall.sh` script integrated into the installer file to remove your instance.

Important: Make sure to *back up* your instance before uninstalling it if you plan to *restore* FindFace CIBR and its data later on.

Run the `uninstall.sh` script from the `/opt/findface-cibr/` folder:

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

You will be asked a question:

Q: This script will remove all docker containers, images, volumes, config files and directory `"/opt/findface-cibr/"`. Do you want to proceed(y/n)?:

A: Answer `y` to remove the instance.

In case your FindFace CIBR deployment is not typical, and you are unable to use the automatic uninstaller script, remove all product components manually. The following command replicates the `uninstall.sh` script behavior:

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

2.4 Administration and Configuration

2.4.1 Licensing

In this chapter:

- *Licensing Principles*
- *View and Update License*
- *Offline Licensing via USB dongle*
- *Offline Licensing via Hardware Fingerprint*

Licensing Principles

1. The overall number of extracted feature vectors.

Note: The feature vectors are extracted from objects detected in the video, from images in the record index, and user photos, and when building so-called cluster centroids.

The licensing scheme is the following:

- Events: 1 event of video object detection = 1 object in a license.
 - Record Index: 1 photo in a record = 1 object in a license.
 - Clusters: 1 person = 1 object in a license.
 - Users: 1 photo of a user = 1 object in a license.
2. The number of video sources currently in use (i.e., active video processing jobs for video files).
 3. The number of model instances in use in the `findface-extraction-api` service.
 4. Face attribute recognition: gender/age/emotions/glasses/beard/face mask/etc.

You can choose between the following licensing methods:

- The online licensing is provided by interaction with the NtechLab Global License Manager license. `ntechlab.com` and requires a stable internet connection, DNS, and open port 443 TCP. Upon being disconnected from the internet, the system will continue working off-grid for 4 hours.

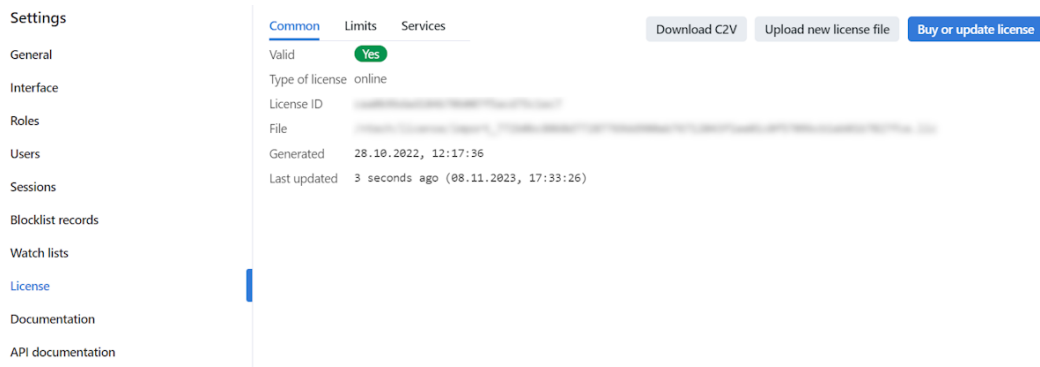
Note: It is possible to prolongate the off-grid period for up to 2 days. Inform your manager if you need that.

- The offline licensing via a USB dongle requires a USB port on the physical server with the `findface-ntls` service (license server in the *FindFace core*).
- The offline licensing via hardware fingerprint requires Sentinel drivers installed on the physical server with the `findface-ntls` service.

Important: For the system to function, a single instance of `findface-ntls` should be enough. If your system requires more license servers, contact your NtechLab manager beforehand to prevent your system from being blocked.

View and Update License

After installing FindFace CIBR, upload the license file you obtained from the manager into the system. To do so, navigate to *Settings* -> *License*.



Use the same tab to consult current licensing information and upgrade your license.

Offline Licensing via USB dongle

To implement the licensing via a USB dongle, do the following:

1. Inform your manager that you intend to apply this licensing method and request your USB dongle and a license file.
2. Open the `/opt/findface-cibr/docker-compose.yaml` configuration file.

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

3. Add the line `privileged: true`. Mount the `/dev` directory into the `findface-cibr-findface-ntls-1` container by listing it in the volumes of the `findface-ntls` section. As a result, the entire section will look as follows:

```
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. Create a new **udev** rule.
 1. Download the `95-grdnt.rules` file (e.g., into the `/home/username/tmp/` directory).
 2. Copy the `95-grdnt.rules` file into the `/etc/udev/rules.d/` directory.

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

5. Rebuild all FindFace CIBR containers.

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

6. Insert the USB dongle into a USB port.
7. Upload the license file on the *License* tab.

Offline Licensing via Hardware Fingerprint

Note: Sentinel is a type of offline licenses that, unlike guardant licenses, do not require any physical media for its work.

Glossary:

- Sentinel is a software protection and licensing system by [Thales](#). It allows you to implement offline licensing without access to a global server.
 - The C2V file is a file, containing data about a hardware fingerprint of the client's machine, for binding the license only to this machine. This file is generated by the sentinel library. The C2V file is generated on the client's machine where the license key will be installed later.
-

To implement the fingerprint licensing, do the following:

1. Inform your manager that you intend to apply this licensing method and request your unique license id. The manager will also supply you with the `findface-sentinel-lib_*.deb` package necessary for the FindFace CIBR and Sentinel integration.
2. Install the Sentinel drivers on the physical server with the `findface-ntls` component.

Do the following:

1. Download [Sentinel drivers](#) from the official website.
2. Unzip the downloaded archive and browse to it.

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

3. There is another archive `aksusbd-*.tar.gz` inside the archive. Unzip it and browse to the resulting directory.

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

4. Run the installation command.

```
sudo ./dinst
```

5. Run and check the statuses of the Sentinel services.

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

3. Mount the `/var/hasplm` and `/etc/hasplm` directories into the `findface-cibr-findface-ntls-1` container. To do so, open the `/opt/findface-cibr/docker-compose.yaml` configuration file and list them in the volumes of the `findface-ntls` section.

```
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. Rebuild all FindFace CIBR containers.

```
cd /opt/findface-cibr

sudo docker-compose down

sudo docker-compose up -d
```

5. Put the `findface-sentinel-lib_*.deb` package received from your manager into some directory on the same host. Install the package.

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

6. In the FindFace CIBR web interface, navigate to *Settings -> License*. Take a hardware fingerprint (C2V file) by clicking the *Download C2V for activation* button.

Tip: If you prefer working with the console, you can send the following API request to `findface-ntls` instead:

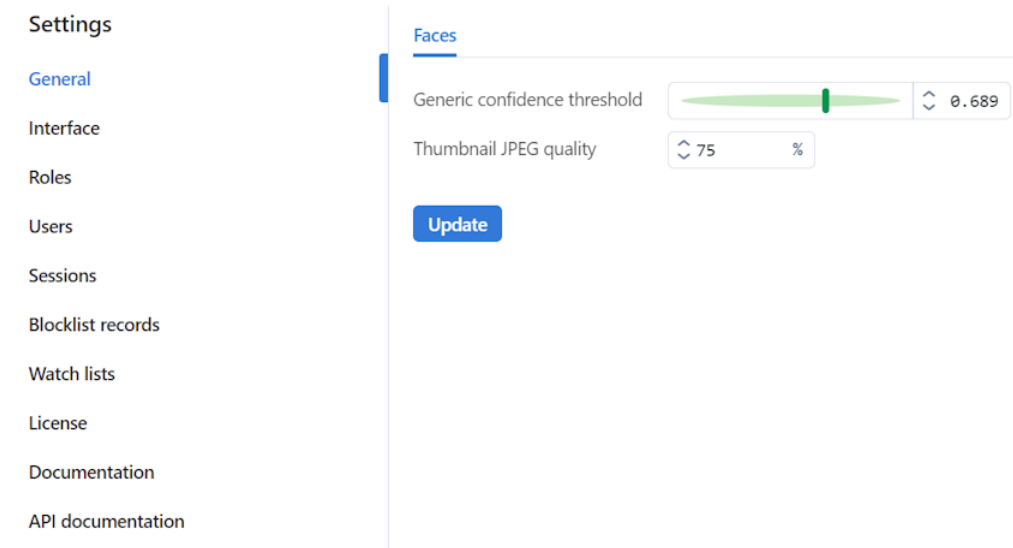
```
curl <findface-ntls-server-ip>:3185/c2v >my_pc.c2v
```

7. Send the License ID and the C2V file to your manager and receive your license file in return.
8. Upload the license file on the *License* tab.

2.4.2 General Settings

The FindFace CIBR general settings, such as generic confidence thresholds for face recognition and thumbnail JPEG quality, determine your system functioning and resource consumption.

To configure the general settings, navigate *Settings* → *General*. After you are finished with adjustments, click *Update*. Find the detailed explanation of each general setting below.



In this section:

- *Generic Confidence Threshold*
- *Thumbnail JPEG Quality*

Generic Confidence Threshold

FindFace CIBR verifies that selected faces belong to the same person (i.e. match), based on the pre-defined confidence threshold. The default threshold is set to the optimum value. If necessary, you can change it.

Note: The higher the threshold, the less a chance that a wrong person will be positively verified, however, some valid photos may also fail verification.

Tip: You can configure the confidence thresholds individually for each *watch list*.

Important: The default generic confidence threshold is optimal for the majority of recognition cases. We do not recommend changing it on your own. Be sure to consult with our technical experts prior (support@ntechlab.com).

Thumbnail JPEG Quality

Subject to JPEG quality, thumbnails may take up a significant amount of disc volume. Use the *General* tab to configure the parameter.

2.4.3 User Management and System Security

Important: Although FindFace CIBR provides tools to ensure its protection from unauthorized access, they are not replacing a properly configured firewall. Be sure to use a firewall to heighten the FindFace CIBR network protection.

Role and User Management

In this chapter:

- *Predefined Roles*
- *Create Custom Role in UI*
- *Primary and Additional User Privileges*
- *Create User Account Manually*
- *Work with Roles and Users via Console*
- *Deactivate or Delete Users*

Predefined Roles

FindFace CIBR provides the following predefined roles:

- Administrator is granted full access to the FindFace CIBR functionality, integrative and administrative tools.

Important: Whatever the role, the first administrator (Super Administrator) cannot be deprived of its rights.

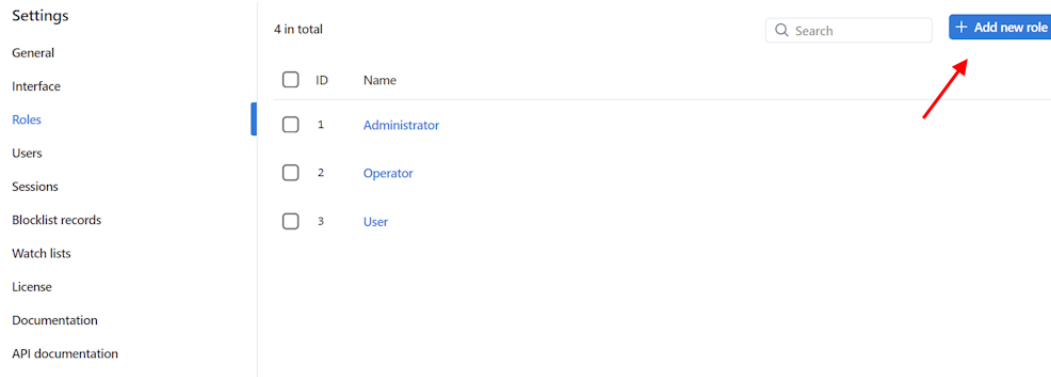
- Operator is granted full access to the FindFace CIBR functionality.
- User is granted rights to modify their profile and manage cases. The other functions are available read-only.

You can change the role privileges for operator and user, as well as create various custom roles, but you can't change the role privileges for administrator.

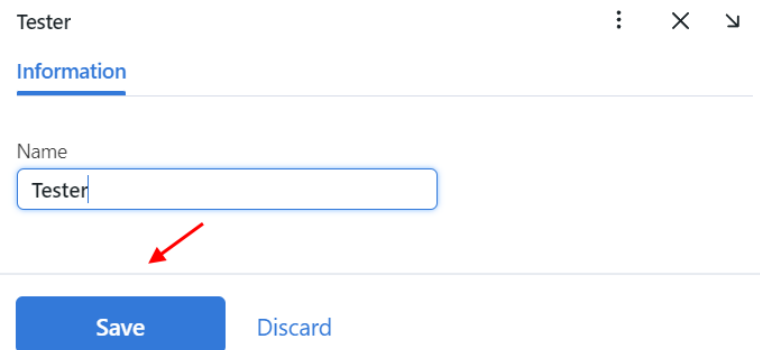
Create Custom Role in UI

To create a custom role in the web interface, do the following:

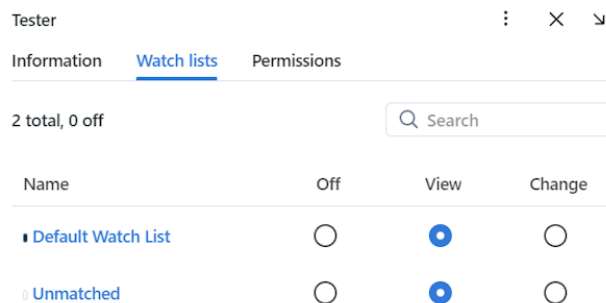
1. Navigate *Settings* -> *Roles*.
2. Click + *Add new role*.



3. On the *Information* tab, specify the role name. Save the role.



4. After saving the role, you will see the following tabs appear next to the *Information* tab:



- *Watch Lists*: role privileges for specific watch lists
- *Permissions*: role privileges for entire system functions and entities

Set role privileges, subject to your needs. Note that there is a distinction between role privileges for a specific watch list and a system entity with the name `watchlist`. For example, if you set Off for a certain watch list on the *Watch lists* tab, users with this role won't be able to work with **this** very watch list. Deselecting all checkboxes for the `watchlist` entity on the *Permissions* tab will prevent users from viewing and working with **all** watch lists.

The full list of the FindFace CIBR entities which are used in the current version is as follows:

- `all_own_sessions`: all *sessions* of the current user on different devices

Note: If relevant permissions for this entity are set, users will be able to view (*view*) and close (*delete*) all their sessions on different devices. Otherwise, users will be only allowed to view and close their session on the current device. Working with sessions takes place on the *Sessions* tab (*Settings*).

- `case`: case file
- `dailysearchevent`: daily search
- `deviceblacklistrecord`: *blocklist*
- `faceobject`: face photo in a *record*
- `group`: *roles*
- `humancard`: *record of an individual*
- `remotemonitoringrecord`: *remote monitoring*
- `report`: *report*
- `searchrequest`: remote search
- `upload`: item (photo) in batch photo upload
- `uploadlist`: list of photos in batch upload
- `user`: *user*
- `videoarchive`: object identification in video files
- `watchlist`: *watch list*

You can also enable and disable rights for the following functionality:

- `batchupload_cards`: *bulk record upload*
- `change_runtime setting`: changing the FindFace CIBR *general settings*
- `configure_ntls`: configuration of the `findface-ntls` *license server*
- `view_auditlog`: viewing and working with the *audit logs*

5. Save the changes.

Primary and Additional User Privileges

You can assign privileges to a user by using roles:

- *Primary role*: main user role, mandatory for assignment. You can assign only one primary role to a user.
- An additional user role, optional for assignment. You can assign several roles to one user. The rights associated with the additional roles will be added to the primary privileges.

All users belonging to a particular primary role automatically get access to video archives within the group and watch lists (and records in the watch list) created by a user with the same primary role, subject to the privileges defined by their additional role(s).

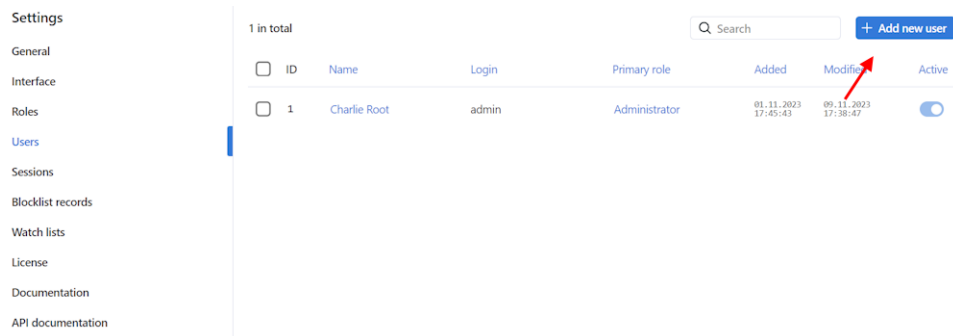
See also:

[Create User Account Manually](#)

Create User Account Manually

To create a user account manually, do the following:

1. Navigate *Settings* -> *Users*.
2. Click + *Add new user*.



3. On the *Information* tab, specify user data such as name, login, and password. If necessary, add a comment.

Note: When setting a password, mind password requirements:

- at least 8 characters long
 - not only numerals
 - not within the list of 20000 commonly used passwords
 - not similar to other user attributes
 - only Latin letters, numerals, and special characters are allowed
-

4. From the *Roles* drop-down menu, select one or several user roles. Set one of them as the *Primary role*.
5. On the *Photos* tab, attach user's photo(s).
6. Save the user account.

Marsha Johnes

Changed ×
⋮
×
↵

Information
Photos

Name

Marsha Johnes

Login

Marsha Johnes

[Change password](#)

Comment

Roles	Primary role
User	<input type="radio"/> <div>🗑️</div>
Tester	<input checked="" type="radio"/> <div>🗑️</div>

Add role ▼

ID 2

Created 09.11.2023 17:56:31

Save

Discard

Work with Roles and Users via Console

In case *predefined roles* have been removed from the system, use the following command to create them:

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

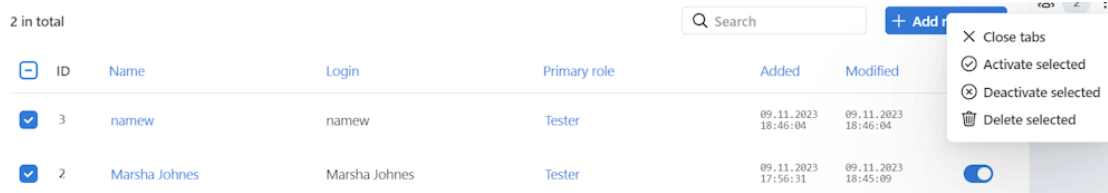
To create a user with Super Administrator rights (superuser), execute the following command, mind that password is a required argument:

```
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>
```

Deactivate or Delete Users

In order to deactivate a user, move the *Active* slider to inactive position on the user list (*Settings -> Users*).

If you are going to deactivate multiple users, select them on the user list and then click *Deactivate selected*.



To delete users from FindFace CIBR, select them on the user list and then click *Delete selected*.

Enable Data Encryption

To ensure data security, we recommend enabling SSL encryption. Do the following:

1. On the host system, create the nginx configuration directory with the subdirectory that will be used to store all the SSL data:

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

2. Create the SSL key and certificate files. When using self-signed certificate, use the following command:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/nginx/ssl/my-  
example-domain.com.key -out /etc/nginx/ssl/my-example-domain.com.crt
```

You will be asked a few questions about your server in order to embed the information correctly in the certificate. Fill out the prompts appropriately. The most important line is the one that requests the Common Name. You need to enter the domain name or public IP address that you want to be associated with your server. Both of the files you created (`my-example-domain.com.key` and `my-example-domain.com.crt`) will be placed in the `/etc/nginx/ssl/` directory.

3. When using CA-certificate, add the certificate path to **volumes** for the `findface-video-worker` service, add the CA-certificates installation and update the root certificate store in the service container.

1. Open the `/opt/findface-cibr/docker-compose.yaml` file:

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

2. Locate the `findface-video-worker` section and adjust it to make sure it looks as follows.

For 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/
  ↪ findface-video-worker.yaml"]
  depends_on: [findface-video-manager, findface-ntls, mongodb]
  image: docker.int.ntl/ntech/universe/video-worker-cpu:ffserver-9.230407.1
  logging: {driver: journald}
  network_mode: service:pause
  restart: always
```

(continues on next page)

(continued from previous page)

```

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']

```

For GPU, it will be enough to add the path to the certificate and update the root certificate store:

```

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, mongodb]
  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']

```

Important: For CPU version, the configuration requires internet access. If there is no access, please, contact our support team (support@ntechlab.com).

Warning: For CPU version, the startup time will increase by ~15 seconds for the findface-video-worker container.

3. Rebuild all FindFace CIBR containers.

```

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

```

4. Configure nginx to use SSL. Open the nginx configuration file `/opt/findface-cibr/configs/findface-multi-ui/nginx-site.conf`. Apply the following modifications to the file:

1. Add the new `server {...}` section that contains the URL replacement rule. In the `rewrite ^(.*) https://...` line, replace `ip_address_server_ffcibr` with IP address of the server where FindFace CIBR is installed.

```
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. Comment out the following lines in the existing `server {...}` section:

```
# listen 80 default_server;
# listen [::]:80 default_server;
```

3. Add the following lines, including the paths to the certificate and the key, to the existing `server {...}` section:

```
listen 443 ssl;

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

The example of the configuration file `/opt/findface-cibr/configs/findface-multi-ui/nginx-site.conf` with correctly configured SSL settings is shown below:

```
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";
}

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

(continues on next page)

(continued from previous page)

```

        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/;
        }
        location /doc/ {
            alias /opt/findface-security/doc/;
        }
        location /api-docs {

```

(continues on next page)

(continued from previous page)

```

        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;
        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;
    }

```

(continues on next page)

(continued from previous page)

```

        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;
#     }
}

```

4. Copy the generic nginx configuration file `nginx.conf` from the `findface-cibr-findface-multi-ui-1` container to the `/etc/nginx/` directory:

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

5. In the configuration file `/etc/nginx/nginx.conf`, find the SSL Settings section and append the following lines:

```
ssl_session_cache    shared:SSL:10m;  
ssl_session_timeout 1h;
```

5. In the `/opt/findface-cibr/docker-compose.yaml` file, mount the SSL-encryption data directory `/etc/nginx/ssl/` and the configuration file `/etc/nginx/nginx.conf` of the host system into the `findface-cibr-findface-multi-ui-1` container:

1. Open the `/opt/findface-cibr/docker-compose.yaml` file:

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

2. Locate the `findface-multi-ui` section and adjust it to make sure it looks like this:

```
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. Edit the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file.

1. In the `ROUTER_URL` and `IMAGE_CROP_URL` parameters, substitute the `http://` prefix with `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. If you use a CA-certificate, specify in the `ROUTER_URL` parameter the domain for which the certificate was created:

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

3. Add `https://my-example-domain.com` address to the `EXTERNAL_ADDRESS` parameter:

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

7. Open the `/etc/hosts` file on the server where FindFace CIBR is installed and add the following line:


```
sudo vi /etc/hosts

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

8. In the system where you use a browser to interact with FindFace CIBR navigate to the hosts file. Add IP address of the server that hosts FindFace CIBR instead of the `ip_address_server_ffcibr`. Replace `my-example-domain.com` with your domain address - the same way you did it in the previous steps.

1. For Linux OS do the following:

```
sudo vi /etc/hosts

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

2. If you use Windows OS, run `C:\Windows\System32\drivers\etc\hosts` as an administrator. Add the following line to the hosts file:

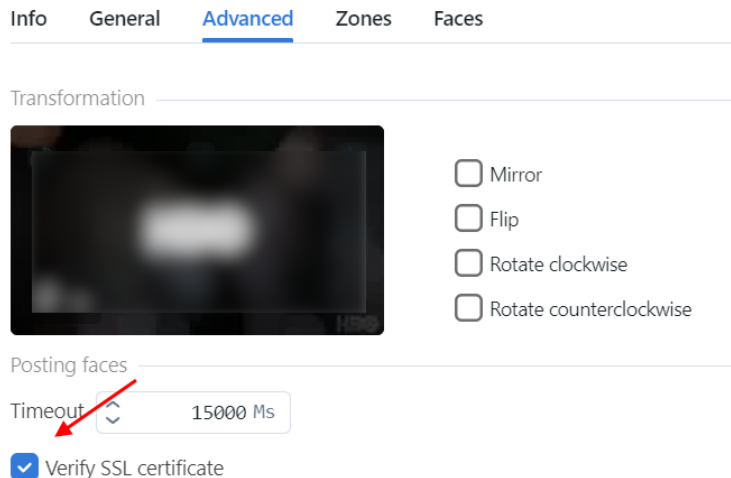
```
*ip_address_server_ffcibr* my-example-domain.com
```

9. Restart the containers:

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

10. If you use self-signed certificate, disable SSL certificate verification for uploaded videos:

1. Navigate to the *Videos* within a case.
2. Click on the uploaded video.
3. On the *Advanced* tab, deselect *Verify SSL certificate*.



List of User Sessions. Blocklist

In this chapter:

- *Grant Permissions to Work with Sessions*
- *View User Sessions*
- *Block Device*

FindFace CIBR allows you to monitor user sessions and learn associated data, such as the connected device UUID, type of user interface, IP address, last ping time, and so on.

If necessary, you can add a device to the blocklist without deactivating the user account. The device block may come in handy in various situations. For example, if you want users to access the system only from their workplaces. Use the blocklist functionality to take your system safety to the next level.

Grant Permissions to Work with Sessions

A user's access to the list of sessions depends on the granted *permissions*:

- Administrator: can view and close sessions of all users
- User with the `all_own_sessions` permissions: can view/close all sessions initiated with their username
- User without the `all_own_sessions` permissions: can only view/close their current session

View User Sessions

To view the list of user sessions, navigate *Settings* -> *Sessions*.

User	Device	Status	Last ping
<input type="checkbox"/> UUID			
<input type="checkbox"/> 788078b1-3d85-45c6-b863-2a68f12795c3	admin	Online	13.11.2023 11:28:36
<input type="checkbox"/> 2f570d87-8827-4a8f-bc76-875a36a3c273	admin	Offline	11.11.2023 13:27:58
<input type="checkbox"/> f2a6a81a-f4cc-4686-b9cd-ad7f5d23a98c	admin	Offline	10.11.2023 17:36:22
<input type="checkbox"/> 3a38a71e-ad7a-4023-b844-f70a08a3c282	admin	Offline	10.11.2023 10:39:34

Each session record provides the following data:

- device UUID
- username
- type of the user interface (mobile/web)
- device information
- IP address

- status (online, offline, blocked)
- last ping time

Use the filter panel above the list of sessions to set up the search conditions.

To close a session, select it in the list and click *Terminate*.

UUID	User	Device	IP address	Status	Last ping
<input checked="" type="checkbox"/> 788070b-5470-410c-b9d3-2a48f5c1705d	admin		172.28.128.6	Online	13.11.2023 11:28:36
<input checked="" type="checkbox"/> 2f719d87-8d07-4a07-b17d-87f0c868d071	admin		172.28.127.80	Offline	11.11.2023 13:27:58
<input type="checkbox"/> f70a6b6a-f0c0-4908-b9cd-a27f9c1c196c	admin		172.28.127.140	Offline	10.11.2023 17:36:22
<input type="checkbox"/> 3c3d4d76-af76-4023-b949-f7028816a08f	admin		172.28.128.80	Offline	10.11.2023 10:39:34
<input type="checkbox"/> c179a002-9047-4a76-b97d-7028a0f98047	admin		172.28.125.137	Offline	09.11.2023 21:00:33

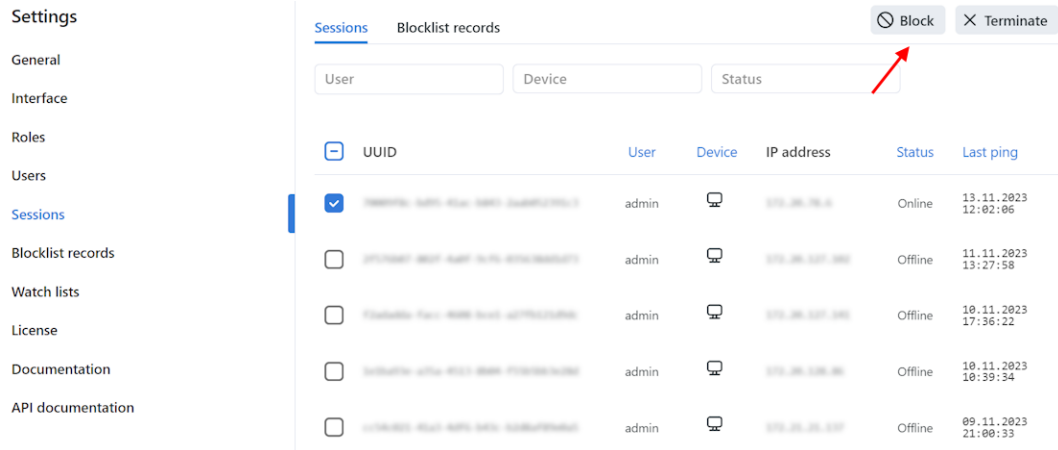
Block Device

The list of blocked devices is available on the *Blocklist Records* tab.

You can add a device to the blocklist on the *Sessions* tab. Blocking a device leads to the user's automatic log-out.

To block a device, do the following:

1. Select the relevant session record(s).
2. Click *Block*.



3. Specify the reason for the device to be blocked (mandatory) and the block expiry date (optional). If an expiration date is not specified, the block will be permanent.
4. Click *Save*.

Create blocklist record

×

UUID

70a0000a-5a05-474e-b843-2a6d05239f13

Reason

Reason

Expires

15.06.2036

12:00

Cancel

Save

Allowed File Extensions in Records

By default, you can attach a file of any extension to a *record*. It is possible to strengthen your system safety by creating the allowlist of file extensions. It will prevent your users from uploading files of unwanted formats, including those that might contain hidden malicious code, such as `.js`, `.swf`, and such.

To create the allowlist of file extensions, do the following:

1. Open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file.

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

2. In the `FFSECURITY` section, find the `CARD_ATTACHMENTS_FILENAME_REGEX` parameter. Set an expression with the allowed file extensions. Any valid [Python regular expression](#) will do.

Examples:

- `r'.*\.png'`: allows only files with the `.png` extension
- `r'.*(png|jpg)'`: allows the `.png` and `.jpg` extensions
- `r'.*'`: allows all file extensions
- `None`: allows all file extensions
- `'XXXXXX'`: uploading files of any extension is prohibited

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

Tip: Commenting out the `CARD_ATTACHMENTS_FILENAME_REGEX` parameter also allows all file extensions.

- Restart the `findface-cibr-findface-multi-legacy-1` container.

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

2.4.4 Watch Lists

The appearance of specific individuals in the video is monitored with a set of default and custom watch lists.

Records of individuals are assigned to watch lists. Once a watch list is activated, the system will be looking for each person in it during video processing or *remote monitoring*.

You can create as many custom watch lists as necessary: wanted, suspects, etc. — subject to your needs.

In this section:

- *Monitor Unmatched Faces*
- *Create Watch List*
- *Remove Watch List*

Monitor Unmatched Faces

FindFace CIBR features a special pre-configured watch list used for monitoring only unmatched faces (faces that do not match any record). This watch list cannot be removed from the system. To edit its settings, navigate to the *Settings* tab. Click *Watch Lists* and then click *Unmatched*.

Unmatched

Information Permissions

Name: Unmatched Color: #ffffff

Comment: Default list for unmatched events

☐ Confidence threshold

☒ Collect location data

☒ Active

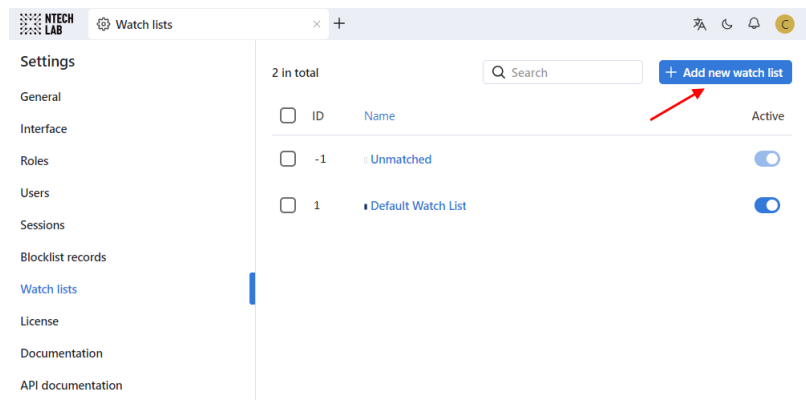
ID -1

Created 15.12.2023 17:52:10

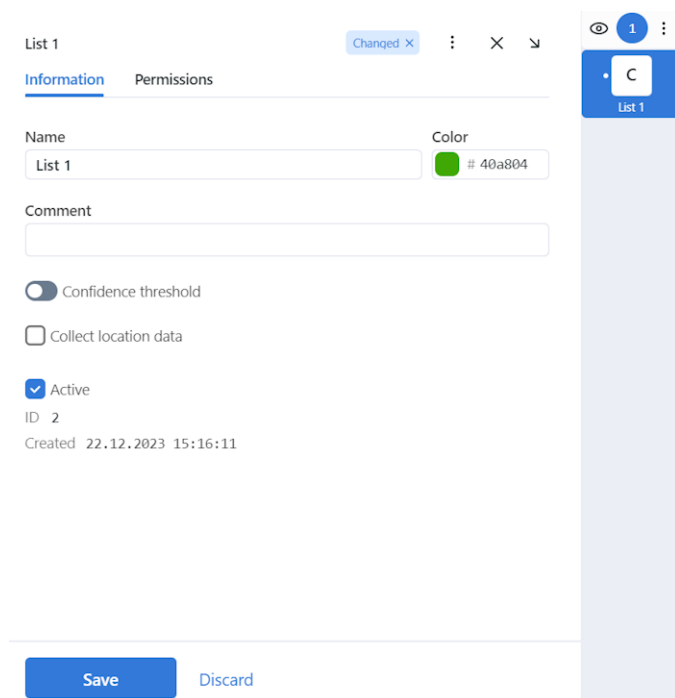
Create Watch List

To create a custom watch list, do the following:

1. Navigate *Settings* → *Watch Lists*.
2. Click + *Add new watch list*.



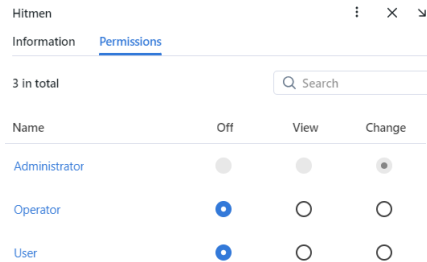
3. On the *Information* tab, specify the watch list name.
4. From the *Color* palette, select a color which will be shown in notifications for this list.



5. Describe the watch list in a comment, if needed.
6. By default, the *generic confidence threshold* is applied in all watch lists in the system. To set an individual threshold for the watch list, enable *Confidence threshold* settings and specify the threshold value.

Important: The default generic confidence threshold is optimal for the majority of recognition cases. We do not recommend changing it on your own. Be sure to consult with our technical experts prior (support@ntechlab.com).

7. (Optional) If you have enabled *daily search*, select *Collect data location*, so all records associated to this watch list may receive daily search events.
8. On the *Permissions* tab, assign privileges on the watch list, specifying which user roles are allowed to change/view the watch list settings.

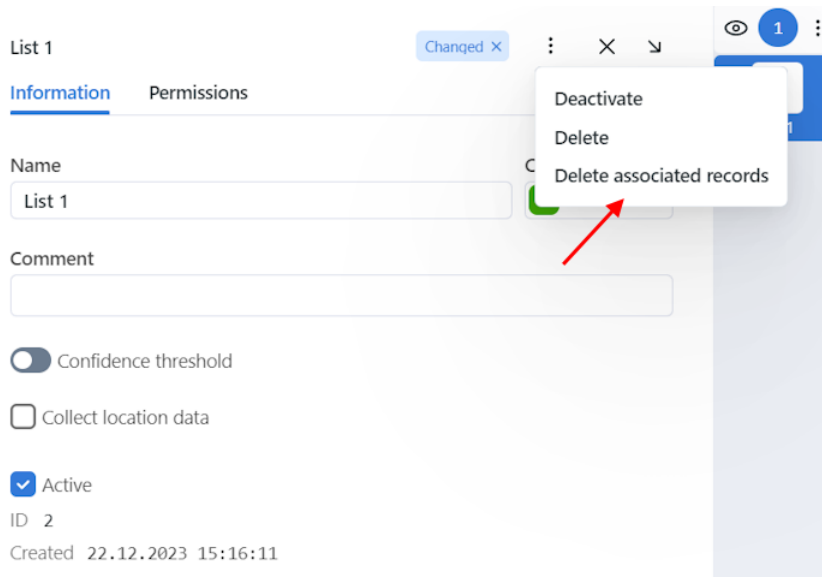


Name	Off	View	Change
Administrator	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>
Operator	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
User	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>

9. Activate and save the watch list.

Remove Watch List

To remove a custom watch list, delete records associated with it first. Otherwise, the system will return an error and will not delete the watch list.



List 1

Changed x

Information Permissions

Name
List 1

Comment

☒ Confidence threshold

☐ Collect location data

☒ Active

ID 2

Created 22.12.2023 15:16:11

Deactivate
Delete
Delete associated records

Default Watch List and *Unmatched* watch list cannot be removed from the system.

2.4.5 Custom Tabs, Fields, and Filters in Record Index

See also:

To create custom fields in the feature vector database, refer to *Custom Metadata in Tarantool*.

To add custom tabs and fields to the records of individuals, do the following:

1. Prepare the list of custom tabs and fields you want to add to the records.
2. Open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file.

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

3. Customize the records of individuals. To do so, uncomment the `FFSECURITY -> CUSTOM_FIELDS -> human_card` section and modify the exemplary content, considering the following:
 - `'items'`: the list of fields in a record. Describe each field with the following parameters:
 - `'name'`: field's internal name, string.
 - `'default'`: field's default value. If a default value exceeds `1e14 - 1`, use a string data type to specify it, for example, `"123123..."` instead of `123123...`
 - `'label'`: field's label in a record, string.
 - `'tab'`: tab that features the field.
 - `'display'`: display format (`form` or `list`), string or array.
 - `'description'`: field's description, string.
 - `'editable'`: field's editability, boolean.
 - `'type'`: field data type, string. Possible values:
 - * `list`: requires `items`, additional parameter for lists (see below), expects objects `{id, name}` in dictionaries;
 - * `valuelist`: expects elements of primitive types.
 - * `objectlist`: allows for creating arrays of objects of required types.
 - * `datetime`: primitive data type displayed as a datetime list.
 - * `date`: primitive data type displayed as a date picker.
 - * `boolean`: primitive data type displayed as a checkbox.
 - * `string`: primitive data type `string`.
 - additional parameters for lists (`type=list`, `type=valuelist`):
 - * `multiple`: possibility of selecting several items in the list, boolean.
 - * `items`: dictionary used as a data source for the list.
 - * `allow_create`: possibility of adding new items to the list.
 - * `custom_id`: custom field for id (`type=list`).
 - additional parameters for object lists (`type=objectlist`):
 - * `object`: objects used as a data source for the object list.
 - * `simple`: indicator that the field expects data of a primitive type instead of objects, for example, expects strings with phone numbers.

- 'filters': the list of search filters associated with the custom fields. Parameters:
 - 'name': filter's internal name,
 - 'label': filter's label in the web interface,
 - 'field': associated field in the format [field name].
- 'tabs': the list of tabs in a record.

```
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': [
```

(continues on next page)

(continued from previous page)

```

        {
            'name': 'personid',
            'label': 'Sigur person ID filter',
            'field': 'personid'
        }
    ]
},
...
},
}

```

4. Restart the `findface-cibr-findface-multi-legacy-1` container.

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

You will see the custom content appear in the records.

2.4.6 Custom Metadata in Tarantool

It is often necessary to assign additional metadata to the faces extracted from images uploaded to the record index and now stored in the feature vector database.

Customize Meta Fields of Face Objects

To assign custom meta fields to the face objects, do the following:

1. Prepare the list of custom meta fields to assign.
2. Open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file.

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

3. In the `FFSECURITY` section, uncomment the `CUSTOM_FIELDS -> face_object` section and modify the exemplary content, considering the following:
 - `field_name`: field's name;
 - `type`: data type (`uint`, `string` or `bool`);
 - `default`: field's default value. If a default value exceeds `1e14 - 1`, use a string data type to specify it, for example, `"123123..."` instead of `123123...`

```

FFSECURITY = {
    ...

    # -- Custom model fields --
    ...
    # Edit CUSTOM_FIELDS -> `face_object` section to customize face object fields.
    ...
    # 'CUSTOM_FIELDS': {
        ...
    }
}

```

(continues on next page)

(continued from previous page)

```

    '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
        }
      ]
    },
  },
}

```

4. *Add the new meta fields* to the feature vector database structure.
5. Restart the findface-cibr-findface-multi-legacy-1 container.

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

You can work with the new meta fields through *HTTP API* using the `objects/faces/` methods.

See also:

To create custom tabs, fields, and filters in records, refer to *Custom Tabs, Fields, and Filters in Record Index*.

2.4.7 Enable Face Attribute Recognition

Face attributes, such as age, gender, emotions, etc., are present in the filter set for detected face analysis during a *case investigation*.

Face attribute recognition can be automatically enabled and configured during the *FindFace CIBR installation*. If you skip this step, you can manually do it later. Face attribute recognition works on both GPU- and CPU-acceleration.

Do the following:

1. Open the `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml` configuration file.

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

2. Specify the relevant recognition models in the `extractors` section, as shown in the example below. Be sure to indicate the right acceleration type for each model, matching the acceleration type of `findface-extraction-api`: CPU or GPU. Be aware that `findface-extraction-api` on CPU can work only with CPU-models, while `findface-extraction-api` on GPU supports both CPU- and GPU-models.

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)

(continued from previous page)

```

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: ''

```

The following extraction models are available:

Extractor	Acceleration	Configure as follows
age	CPU	face_age: faceattr/age.v2.cpu.fnk
	GPU	face_age: faceattr/age.v2.gpu.fnk
beard	CPU	face_beard: faceattr/beard.v0.cpu.fnk
	GPU	face_beard: faceattr/beard.v0.gpu.fnk
face feature vector	CPU	face_emben: face/mango_320.cpu.fnk
	GPU	face_emben: face/mango_320.gpu.fnk
gender	CPU	face_gender: faceattr/gender.v2.cpu.fnk
	GPU	face_gender: faceattr/gender.v2.gpu.fnk
emotions	CPU	face_emotions: faceattr/emotions.v1.cpu.fnk
	GPU	face_emotions: faceattr/emotions.v1.gpu.fnk
glasses	CPU	face_glasses3: faceattr/glasses3.v0.cpu.fnk
	GPU	face_glasses3: faceattr/glasses3.v0.gpu.fnk
face liveness	CPU	face_liveness: faceattr/liveness.web.v0.cpu.fnk
	GPU	face_liveness: faceattr/liveness.web.v0.gpu.fnk
face mask	CPU	face_medmask3: faceattr/medmask3.v2.cpu.fnk
	GPU	face_medmask3: faceattr/medmask3.v2.gpu.fnk
face quality	CPU	face_quality: faceattr/quality_fast.v1.cpu.fnk
	GPU	face_quality: faceattr/quality_fast.v1.gpu.fnk

Tip: To leave a model disabled, pass the empty value '' to the relevant parameter. Do not remove the parameter itself. Otherwise, the system will be searching for the default model. E.g., to disable emotions and face mask recognition, pass the empty value to the corresponding extraction models:

```

extractors:
...
models:
...
face_emotions: ''
face_medmask3: ''

```

Note: You can find face attribute recognition models at `/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
```

- Restart the findface-cibr-findface-extraction-api-1 container.

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

- Enable recognition of face attributes in the /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py configuration file. In the FFSECURITY section, specify the face attributes that you want to display for the face recognition events.

```
# 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
↳ '],
```

- Restart the findface-cibr-findface-multi-legacy-1 container.

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

2.4.8 Integration with Remote Facial Recognition Systems

You can integrate your FindFace CIBR instance with remote facial recognition systems. In this case, the server known as a puppeteer will be pushing records designated for remote alerting to remote servers known as puppets. In return, it will be receiving recognition events matching with those records. You can set up a *daily search* that allows the puppeteer to receive scheduled events.

This functionality has a large scope of possible applications. One course is tracking offenders' location and routes and detecting alleged accomplices. Another one is finding missing people. The results are especially great if applied to Public and Transport Safety systems with thousands of cameras.

The current version supports only integration with facial recognition systems from the FindFace family.

In this section:

- *Sync Schedule*
- *Configure Puppeteer*
- *Configure Puppet*

Sync Schedule

The data between a puppeteer and a puppet are synced in the following way:

- The puppeteer delivers designated records to the puppet with an interval specified in the `REMOTE_MONITORING_SYNC_INTERVAL` parameter (see configuration below).
- The puppet delivers matching recognition events to the puppeteer as soon as they appear.

Configure Puppeteer

To configure your FindFace CIBR instance to be a puppeteer, do the following:

1. Open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file. Make sure that the `EXTERNAL_ADDRESS` parameter is filled.

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

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

Note: The value of `EXTERNAL_ADDRESS` is defined during deployment.

2. Find the Puppeteer section.

```
# ===== 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': [
```

(continues on next page)

(continued from previous page)

```

#      {
#          '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. Uncomment the section as shown in the example below and specify the following parameters:

- `REMOTE_MONITORING_SYNC_INTERVAL`: interval in seconds with which the puppeteer sends designated records to a puppet.
- `REMOTE_MONITORING_EVENTS_MAX_AGE`: remote alerts older than this number of days will be automatically deleted on the puppeteer (every night at 1:17 a.m. by default).
- `puppets` → `id`: a puppet ID.
- `puppets` → `url`: IP address and port of a puppet's principal server.

Specify the port as follows:

- Leave the default port as is if the puppet represents a public or transport safety system (i.e., it has the public-security service at its core).
- Switch the default port to `80` or do not specify it at all if the puppet is based on the `findface-security` service (i.e., with FindFace Security or FindFace Multi installed).
- `puppets` → `token`: token for mutual authentication between the puppeteer and a puppet.

Tip: Use the following command to generate a random token.

```
pwgen -s 64 1
```

- `puppets` → `facen_model`: neural network model used on a puppet for face recognition.
- `puppets` → `extractor`: IP address and port of the biometric data extraction service on a puppet if the neural network model for face recognition on the puppet differs from that on the puppeteer.

Leave other parameters in the section commented out.

```
# ===== 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)
            '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',
        },
    ],
}
```

(continues on next page)

(continued from previous page)

```
]
}
```

4. If necessary, enable the daily searching by uncommenting and setting 'ENABLE_DAILY_SEARCH': True. Set the schedule for sending records to the puppet and receiving events that matched the records that have been sent.

```
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
    ...
}
```

Navigate to the *Watch list* tab and check **Collect location data** for the watch list of your interest.

5. If necessary, enable daily search cleanup by uncommenting the following parameters:

```
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: the duration (in seconds) of storing location data of the object of interest. By default, 0, permanent storage.
- 'DAILY_SEARCH_EVENTS_FULLFRAME_MAX_AGE': 30*24*60*60: the storage duration (in seconds) of full frames of daily search events obtained from the puppets. By default, 30*24*60*60 seconds (30 days). Change 30 to the desired number of days. if needed.

Note: You can manually remove location data of the object and full frames of daily search events older than the given number of seconds by executing the commands below.

To set the number of seconds to remove location data of the object, use the `--daily-search-events-max-age` argument.

```
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
```

To remove full frames of daily search events obtained from the puppets older than 5 days (5*24*60*60 = 432000 seconds), use `--daily-search-events-fullframe-max-age` argument.

```
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
```

- Restart the findface-cibr-findface-multi-legacy-1 container.

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

Configure Puppet

To configure a remote **FindFace Multi 2.0+** instance to be a puppet, do the following:

- Open the /opt/findface-multi/configs/findface-multi-legacy/findface-multi-legacy.py configuration file. Make sure that the EXTERNAL_ADDRESS parameter is filled.

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

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

- Find the Vns section.

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

# VNS_CONFIG = {
#     'USERS': {
#         'user1': 'token1',
#         'user2': 'token2'
#     },
#     'MONITORING_THRESHOLD': 0.75,
#     'DAILY': {
#         'ENABLED': False,
#         'THRESHOLD': 0.75,
#         'START_TIME': "00:00:00"
#     }
# }
```

- Uncomment the section as shown in the example below and specify the following parameters:

- token:** token for mutual authentication between the puppet and a puppeteer. You can specify several users and tokens if the puppet is communicating with several puppeteers. You can leave the user names as is.
- MONITORING_THRESHOLD:** confidence threshold in face recognition events sent to a puppeteer.

Leave other parameters in the section commented out.

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

VNS_CONFIG = {
    'USERS': {
```

(continues on next page)

(continued from previous page)

```

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

```

4. If you have enabled the daily search in the Puppeteer (see step #4 above), uncomment and enable 'DAILY' parameters. Specify the time for matching the events with the records that were received from the puppeteer.

```

===== 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. Restart the findface-multi-findface-multi-legacy-1 container.

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

To configure a remote **FindFace Multi instance version 1.2 and early** to be a puppet, do the following:

1. Open the /etc/findface-security/config.py configuration file. Make sure that the EXTERNAL_ADDRESS parameter is filled.

```

sudo vi /etc/findface-security/config.py

EXTERNAL_ADDRESS = 'http://192.168.0.5'

```

2. Find the Vns *section*.
3. Uncomment the section as shown in *the example* and specify token and MONITORING_THRESHOLD parameters.
4. If you have enabled the daily search in the Puppeteer, uncomment and enable 'DAILY' parameters. Specify the time for matching the events with the records that were received from the puppeteer.
5. Restart the findface-security service.

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

6. Perform migration to sync with puppeteers.

```
sudo findface-security migrate
```

See also:*Remote Alerting and Remote Search*

2.4.9 Multiple Video Cards Usage

Should you have several video cards installed on a physical server, you can create additional `findface-extraction-api` or `findface-video-worker` instances on your GPU-based system and distribute them across the video cards, one instance per card.

If you have followed the instruction to prepare the server on *Ubuntu* (*CentOS*, *Debian*) you should already be all set to proceed.

However, before you take further action, make sure that you have a properly generated `/etc/docker/daemon.json` configuration file. The example is provided for the Ubuntu operating system and shows how to configure the Docker network and also configure usage of the NVIDIA Container Runtime. Note that in our example, we imply that you have already *installed* the NVIDIA Container Runtime.

```
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
```

Refer to the corresponding sections of the documentation to validate configuration of the `/etc/docker/daemon.json` file for *CentOS* and *Debian*.

In this section:

- *Distribute findface-extraction-api Instances Across Several Video Cards*
- *Allocate findface-video-worker to Additional Video Card*

Distribute findface-extraction-api Instances Across Several Video Cards

To distribute the findface-extraction-api instances across numerous video cards, create multiple instances of the findface-extraction-api service in the docker-compose.yaml configuration file. Then, for the findface-extraction-api GPU instances to work within one system, bind them via a load balancer, e.g., Nginx.

Do the following:

1. Configure the docker-compose.yaml file.
 1. Open the /opt/findface-cibr/docker-compose.yaml file and create multiple records of the findface-extraction-api configuration in the findface-extraction-api section. Each new instance of the findface-extraction-api should be configured similarly to another one so that there are no problems in query processing. Do the following:
 - Rename the default findface-extraction-api service section.
 - Create configuration copies for the new instances.
 - Configure the instances.

Below is an example of two configured services:

```
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
  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 — the readable name of an instance. Must be unique for each instance;

- `CUDA_VISIBLE_DEVICES=0` — GPU ID to run the service instance. Must be unique for each instance;
- `CFG_LISTEN=127.0.0.1:18660` — IP:Port to listening requests. Must be unique for each instance;
- `./cache/findface-extraction-api/models:/var/cache/findface/models_cache` — the volume for storing the model cache. It may be the same if you have the same GPU models. In other cases, the caches are stored separately.

The remaining configuration parameters can be the same for every `findface-extraction-api` service instance.

2. Configure the load balancing in the `/opt/findface-cibr/docker-compose.yaml` file. To do so, add a new service (`findface-extraction-api-lb` in our example) to the `docker-compose.yaml` file, e.g., to the end of the file.

```
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']
```

We have specified the `loadbalancer.conf` configuration file in the volumes of the `findface-extraction-api-lb` section. Now we need to create it.

2. Create the load balancer configuration file with the following content:

```
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;
    }
}
```

In the `upstream` section, configure the balancing policy and the list of the `findface-extraction-api` instances between which the load will be distributed.

You can choose among the sending policy options below, but we recommend using the `least_conn` sending policy.

- `round-robin` — requests are distributed evenly across the backend servers in a circular manner;
- `least_conn` — requests are sent to the server with the fewest active connections. This algorithm is useful when you have backend servers with different capacities;
- `weighted` — backend servers are assigned different weights, and requests are distributed based on those weights. It allows you to prioritize certain servers over others.

The server `127.0.0.1:18666 max_fails=3 fail_timeout=60s` line consists of the following meaningful parts:

- `server` — defines an upstream server in Nginx;
 - `127.0.0.1:18660` — represents the IP address (`127.0.0.1`) and port (`18660`) of the `findface-extraction-api` instance;
 - `max_fails=3` — this parameter sets the maximum number of failed attempts that Nginx will allow when connecting to the server before considering it as unavailable;
 - `fail_timeout=60s` — this parameter specifies the amount of time Nginx should consider the server as unavailable (in this case, 60 seconds) if it exceeds the maximum number of failed attempts specified by `max_fails`;
 - `weight=2` (not used in the provided example) — this parameter allows you to assign a relative weight or priority to each server in an upstream group. This means that for every request, the server with `weight=2` will receive twice as many requests. It might be useful in scenarios where GPUs of different performance are used.
3. Open the `/opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml` configuration file. If the address or port of the load balancer differs from the `127.0.0.1:18666` in the `extraction-api` section, you must set it to `127.0.0.1:18666`. Otherwise, don't change anything.

```
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. Rebuild all FindFace CIBR containers, removing at the same time orphan containers for services that are no longer defined in the `docker-compose.yaml` file.

```
cd /opt/findface-cibr/

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

To make sure that everything works as expected, check the logs of the services.

```
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
```


Allocate findface-video-worker to Additional Video Card

To create an additional findface-video-worker instance on your GPU-based system and allocate it to a different video card, do the following:

1. In the /opt/findface-cibr/docker-compose.yaml file, specify the findface-video-worker configuration for each running findface-video-worker instance. Copy your current findface-video-worker configuration.

```
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, mongodb]
  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']
```

Then, adjust it accordingly.

```
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
```

(continues on next page)

(continued from previous page)

```

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']

```

The main parameters here are as follows:

- `findface-video-worker-0` — the new name of the `findface-video-worker` instance (in our example, the `findface-video-worker` instance is assigned a number equal to the GPU ID on the device);
- `CUDA_VISIBLE_DEVICES=0` — the CUDA device ID to run a new instance;
- `CFG_STREAMER_PORT=18991` — the streamer port (must be unique);
- `CFG_STREAMER_URL=127.0.0.1:18991` — the URL to connect to and get stream from the `findface-video-worker` instance (must be unique).

All other parameters remain unchanged.

2. Rebuild all FindFace CIBR containers, removing at the same time orphan containers for services that are no longer defined in the `docker-compose.yaml` file.

```

cd /opt/findface-cibr/

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

```

To make sure that everything works fine, check the logs of the `findface-video-worker` services.

```

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

```

2.4.10 Configure Saving Images in Reports

When building *reports*, you will be able to choose to save the report images as links, thumbnails, or full frames. It is possible to configure the image parameters. To do so, open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file and alter the default JPEG quality and the maximum height of thumbnails and full frames, subject to your free disc space.

```

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,

```

Restart the `findface-cibr-findface-multi-legacy-1` container.

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

2.4.11 Console Bulk Record Upload

In addition to the *web interface upload*, you can bulk-upload records to the record index via the **uploader.py** console utility. Make choice in favor of this utility if the number of uploaded photos exceeds 10,000.

Tip: To view the **uploader.py** help, execute:

```
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)
--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
--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.
```

Do the following:

1. Write the list of photos and metastrings to a CSV or TSV file.

Important: The file used as a metadata source must have the following format: `path to photo | metastring`.

To prepare a TSV file, you can use a script, or generate it with python or other utility.

Note: Both the script and the command in the examples below create the `images.tsv` file. Each image in the list will be associated with a metastring coinciding with the image file name in the format `path to photo | metastring`.

To build a TSV file listing photos, upload the script to your home directory, for example `/home/ubuntu`, and run the following command:

```
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"
```

where `/home/ubuntu/photos` is a directory with your photos.

A TSV file may also be generated using the following command:

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

You can use a `images.tsv` file as example.

2. Create a job file out of a CSV or TSV file by using the add utility command. As a result, a file `enroll-job.db` will be created and saved in a current directory.

```
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"
```

The add utility options:

- `--format`: input file format, tsv by default,
- `--delimiter`: field delimiter, by default `"\t"` for TSV, and `","` for CSV.

Note: A job file represents a sqlite database which can be opened on the **sqlite3** console.

3. Process the job file by specifying the path to the photos (for example, `/home/ubuntu/photos`) and passing the necessary arguments. Use the following command:

```
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"
```

The run utility options:

- `--parallel`: the number of photo upload threads, 10 by default. The more threads you use, the faster the bulk upload is completed, however it requires more resources too.
 - `--api`: findface-security API URL, `http://127.0.0.1:80/` by default. Mandatory option.
 - `--user`: login. Mandatory option.
 - `--password`: password. Mandatory option.
 - `--watch-lists`: comma-separated list of the watch lists id's. Mandatory option.
 - `--inactive`: mark new records as inactive.
 - `--failed`: should an error occur during the job file processing, correct the mistake and try again with this option.
 - `--noface`: by default, images classified as having no faces will be assigned the NOFACE status and automatically excluded from the upload. To attempt re-detecting faces in such images, re-run the job file with this option. If the re-detection gives a negative result again, an image will be skipped and a relevant record will appear in the upload log.
 - `--all-faces`: upload all faces from a photo if it features several faces.
 - `--detect-timeout`: request timeout for detect photos.
 - `--logging-delta`: logging period delta.
4. (Optional) Print the job processing results as JSON. If necessary, you can print only failed images/ images without detected faces. Use the following command:

```
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/
```

(continues on next page)

(continued from previous page)

```
↪ findface-security/bin/python3 /tigre_prototype/ffsecurity/uploader.py print --
↪ noface --failed"
```

The print utility options:

- --failed: show only failed images.
- --noface: show only images without detection.

2.5 Maintenance and Troubleshooting

2.5.1 Update to FindFace CIBR 2.1.2

Update FindFace CIBR 1.3 to FindFace CIBR 2.1.2

To update FindFace CIBR 1.3 to FindFace CIBR 2.1.2, do the following:

1. Create a backup copy of the old schema of the Tarantool-based feature vector database:

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

Starting from version 2.1.1, one of the most significant differences between FindFace CIBR and earlier versions of the product is the structure of the Tarantool biometric database (so called meta-schema). The new structure is created as a set of spaces, while in previous versions of the product there was only one space by default in the structure of the Tarantool-based database.

2. Open the `/etc/findface-security/config.py` configuration file. Save the values of the following parameters for later use: `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN`, `ROUTER_URL`, `CUSTOM_FIELDS`.

```
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': "",
```

(continues on next page)

(continued from previous page)

```

#         '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'
#     }
# ]
# },
# '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",

```

(continues on next page)

(continued from previous page)

```
#           "type": "bool",
#           "default": True
#       },
#   ]
# }
# },
# }
```

3. Stop the findface-security service.

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

4. Create a backup of the Tarantool-based feature vector database in any directory of your choice, for example, /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. To avoid port conflicts, stop and disable all services before installing a new version.

Note: There are eight shards in the example below. If it differs with the number of shards in your system, adjust the below command accordingly. E.g., for the system with sixteen shards, replace `tarantool@shard-00{1..8}.service` with `tarantool@shard-0{01..16}.service`. Get the list of active shards with the `ls /etc/tarantool/instances.enabled/` command.

You may stop and disable the services one by one:

```
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 mongodb.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
```

(continues on next page)

(continued from previous page)

```

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
sudo systemctl disable mongod.service
sudo systemctl disable memcached.service
sudo systemctl disable nginx.service

```

Or you may use the following compact commands instead:

```

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 mongod.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
↪mongod.service memcached.service nginx.service

```

6. *Install* the FindFace CIBR 2.1.2 instance. Don't forget to prepare a server first:

See:

- *Ubuntu Server Preparation*
- *CentOS 7 Server Preparation*
- *Debian 11 Server Preparation*

7. After FindFace CIBR installation, stop all containers from the `/opt/findface-cibr/` directory.

```

cd /opt/findface-cibr/

sudo docker-compose stop

```

8. Open the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file and paste saved on step #2 values for the parameters `EXTERNAL_ADDRESS`, `SECRET_KEY`, `VIDEO_DETECTOR_TOKEN`, `ROUTER_URL`, and `CUSTOM_FIELDS` into it.

```

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.

```

(continues on next page)

(continued from previous page)

```
EXTERNAL_ADDRESS = 'http://172.20.77.58'
...
# findface-video-worker authorization token
'VIDEO_DETECTOR_TOKEN': '8977e1b0067d43f6c908d0bf60363255',
...
# findface-video-worker face posting address,
# it must be set to either FFSecurity EXTERNAL_ADDRESS (by default)
# or findface-facerouter url (in some specific cases)
'ROUTER_URL': 'http://127.0.0.1:80',
```

9. Open the old version of the findface-ntls configuration file available at /etc/findface-ntls.cfg and check it against the new version /opt/findface-cibr/configs/findface-ntls/findface-ntls.yaml. Move all the custom parameters from the old version to the new one. Do the same for other components, e.g., for findface-extract-api, check /etc/findface-extract-api.ini against /opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml, for findface-sf-api, check /etc/findface-sf-api.ini against /opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml, etc.

```
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
```

Important: Follow these rules to transfer parameters from the old configuration file to the new one:

- If a parameter had an empty value in the old configuration file, but has a certain value in the new configuration file, delete its value in the new configuration file.
- Keep as is those parameters that were not included in the old configuration file, but are present in the new configuration file.

10. Modify the Tarantool database structure by applying the `tnt_schema.lua` schema from 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. Purge data from all the directories relevant to active shards.

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

12. Copy the meta-schema of the `face_clusters_space` space from the `old_tnt_schema.lua` configuration file to the new `tnt_schema.lua` configuration file. An easy way to do it is to follow these steps:

- 12.1. In the new configuration file `/etc/findface-security/tnt_schema.lua`, replace the following line at the beginning of the file:

```
cfg_spaces = {
```

with this one:

```
spaces = {
```

12.2. Copy and add `face_clusters_space` space from the `old_tnt_schema.lua` old configuration file:

```
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. Remove the default configuration file `FindFace.lua`, generated by the `findface-tarantool-server` package, as it will block the restart, required on the next step.

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

14. Restart the `findface-tarantool-server` shards.

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

Upon completion of the above steps, the shards will still keep the old galleries created within the `default` space, but new spaces (e.g., `ffsec_face_case_events`, `ffsec_face_case_clusters` and so on) will also become available.

15. Restore old data from the backup. The data will be restored as it existed previously: all galleries will stay within the default space.

```
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. Start old services `findface-sf-api.service`, `nginx.service`, they will be required during postgresql migration.

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

17. Perform PostgreSQL database migrations for FindFace CIBR 2.1.2 compatibility. Do the following:

- 17.1. Navigate to the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file. In the DATABASES -> default section, temporarily replace PASSWORD with the old one, used in the `/etc/findface-security/config.py` configuration file.

Important: Make sure to write down the password from the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file. You will need it later.

```
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. On the host system, perform the database migration:

```
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. Back up an existing database with PostgreSQL, installed on the host system.

```
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_
ffsecurity_grouppermission -t ffsecurity_runtimesetting -t ffsecurity_
user -t ffsecurity_user_groups -t ffsecurity_user_user_permissions -t_
```

(continues on next page)

(continued from previous page)

```
↪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. Back up role permissions.

```
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. Change back the password, replaced in step #17.1

17.6. Stop all the services.

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

17.7. Open the /opt/findface-cibr/docker-compose.yaml file and copy POSTGRES_PASSWORD value to use it in further commands.

17.8. Recreate the ffsecurity database to clean it up from the default data. Paste {POSTGRES_PASSWORD} value that you previously copied in step #17.7 into the command below:

```
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. Restore data into the recreated ffsecurity database. Paste {POSTGRES_PASSWORD} value that you previously copied in step #17.7 into the command below:

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

17.10. Recreate the ffsecurity_identity_provider database to clean it up from the default data. Paste {POSTGRES_PASSWORD} value that you previously copied in step #17.7 into the command below:

```
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. Run migration.

```
sudo docker-compose up -d pgbouncer
sudo docker run --rm --network host --volume '/opt/findface-cibr/configs/
```

(continues on next page)

(continued from previous page)

```
↪ 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. Restore data into the recreated `ffsecurity_identity_provider` database. Paste `{POSTGRES_PASSWORD}` value that you previously copied in step #17.7 into the command below:

```
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. Start all the services.

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

17.14. Run the command to create records in the outbox table for watch lists, camera groups, and cases. Paste `{POSTGRES_PASSWORD}` value that you previously copied in step #17.7 into the command below:

```
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. Before you restore role permissions into the `identity_provider` service, examine the `/opt/findface-cibr/permissions.csv` file. Make sure to replace `*_ffsecauthtoken` with `*_authtoken` if any. This is mostly applicable to those cases when FindFace CIBR 1.3 installation was an upgrade from earlier versions of the product.

After that, restore role permissions into the `identity_provider` service.

```
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. Copy full frame photos, normalized images, and the license file. Copy files from the `/opt/ntech/license/` folder into the `/opt/findface-cibr/data/findface-ntls/` folder, from the `/var/lib/findface-security/uploads/` folder into the `/opt/findface-cibr/data/findface-multi-legacy/`

uploads/ folder, from the /var/lib/ffupload/uploads/ folder into the /opt/findface-cibr/data/findface-upload/uploads/ folder.

```
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/*
```

Alternatively, the above folders can be directly mounted into the relevant docker containers via the docker-compose.yaml file, like in the example below:

```
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:
  depends_on: [findface-multi-legacy]
```

(continues on next page)

(continued from previous page)

```

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. To move Tarantool data, do the following:

Stop all FindFace CIBR containers:

```
sudo docker-compose down
```

Start the old shards and the findface-sf-api service again:

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

Create a new backup of the feature vector database:

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

Stop the rest of the services, clear the instances.enabled directory, start the containers again, and perform the storage-api-restore operation:

```
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
```

The update has been completed. The new version includes new *neural network models*, but the migration of feature vectors to a different neural network model is not required. You can use old neural network models by moving them from the `/usr/share/findface-data/models/` directory to the `/opt/findface-cibr/models/` directory, specifying them in the `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml` file, if they are included in FindFace CIBR 2.1.2.

Note that face detection models in FindFace CIBR 1.3 are stored in the `/usr/share/findface-data/models/facedet/` directory. In FindFace CIBR 2.1.2 all face detection models can be found in the `/opt/findface-cibr/`

models/detector/ directory. When moving old neural network models from the /usr/share/findface-data/models/ directory to the /opt/findface-cibr/models/ directory, make sure to place all face detection models (facedet) to the /opt/findface-cibr/models/detector/ directory.

See for reference:

```
$ 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
```

Important: We highly recommend disabling the Ubuntu automatic update to preserve the FindFace CIBR compatibility with the installation environment. In this case, you will be able to update your OS manually, fully controlling which packages to update.

To disable the Ubuntu automatic update, execute the following commands:

```
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
```

Update FindFace CIBR 2.1.1 to FindFace CIBR 2.1.2

To update FindFace CIBR 2.1.1 to FindFace CIBR 2.1.2, do the following:

1. In a user home directory, create a folder for FindFace CIBR 2.1.1 backup data.

```
mkdir ~/backup_data/
```

2. Back up FindFace CIBR configuration files and the docker-compose.yaml file and switch to the /opt/findface-cibr/ directory.

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

cd /opt/findface-cibr/
```

3. Back up PostgreSQL data.

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

4. Back up Tarantool data.

```
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. List content of the backup_data/ folder, located in a user home directory. It will look like this.

```
ls ~/backup_data/

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

6. Stop all FindFace CIBR containers.

```
sudo docker-compose down
```

7. Remove the /opt/findface-cibr/data/findface-tarantool-server/ and /opt/findface-cibr/data/postgresql/ folders.

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

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

8. *Install* the FindFace CIBR 2.1.2 instance into the /opt/findface-cibr directory.

9. Restore Tarantool data from the backup.

1. Copy the ~/backup_data/ffmulti_dump/ folder to the findface-cibr-findface-sf-api-1 container.

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

2. Restore Tarantool data.

```
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. In the new configuration files, replace the NTECH USER password with the one, used in FindFace CIBR 2.1.1. The old password can be obtained from the configs/postgresql/40-init.sql file in the backup archive configs.tar.gz. The NTECH USER password should be replaced in the following configuration files:

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

11. In the new configuration file /opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py, replace the SECRET_KEY value in the GENERAL SETTINGS section and 'VIDEO_DETECTOR_TOKEN' value in the FINDFACE SECURITY SETTINGS section with the ones, used in FindFace CIBR 2.1.1. The old parameter values can be obtained from the configs/findface-multi-legacy/findface-multi-legacy.py file in the backup archive configs.tar.gz.

12. Compare previous configuration files against the new ones. Move all custom parameters from the previous version to the new one. Compare configuration files for the services `findface-ntls`, `findface-extract-api`, `findface-sf-api`, etc. You may need to apply changes to the `docker-compose.yaml` file if you previously configured it to bring or exclude services or made any other changes.

Important: Follow these rules to transfer parameters from the old configuration file to the new one:

- If a parameter had an empty value in the old configuration file, but has a certain value in the new configuration file, delete its value in the new configuration file.
 - Keep as is those parameters that were not included in the old configuration file, but are present in the new configuration file.
-

13. Stop all FindFace CIBR containers from the `/opt/findface-cibr` directory.

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

14. Copy data from the `~/backup_data/pg_bkp/data/` directory to the `/opt/findface-cibr/data/postgresql` directory.

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

15. Start all FindFace CIBR containers.

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

2.5.2 Back Up and Recover FindFace CIBR and Its Data

You can back up FindFace CIBR before uninstalling it to recover the product and its data later on.

In this section:

- *Back up FindFace CIBR*
- *Recover FindFace CIBR and Its Data*

Back up FindFace CIBR

To back up your FindFace CIBR instance and its data, run the following commands:

```
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 ~/
```

Recover FindFace CIBR and Its Data

To restore FindFace CIBR and its data from the backup, do the following:

1. Download the installer file `findface-*.run`.
2. Put the `.run` file into some directory on the designated host (for example, `/home/username`).
3. From this directory, make the `.run` file executable.

Note: Be sure to specify the actual file name instead of `findface-*`.

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

4. Execute the `.run` file.

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

5. Go through the installation process as described [here](#).
6. After you have finished the installation, to restore FindFace CIBR, its data and configuration files, stop all FindFace CIBR containers.

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

7. Remove new configuration files and data generated and created by the installer and restore them from the backup.

Important: Compare your old `~/docker-compose.yaml` file that you have copied during the backup process against the new one `/opt/findface-cibr/docker-compose.yaml`. You may need to apply changes to the old `~/docker-compose.yaml` file if you previously configured it to bring or exclude services or made any other changes. You must be fully aware of what you are doing by replacing the new `docker-compose.yaml` file with the old one.

```
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. Restart FindFace CIBR containers.

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

2.5.3 Migrate Feature Vectors to a Different Neural Network Model

Tip: Do not hesitate to contact our experts on migration by support@ntechlab.com.

Important: In case you are doing migration as part of the system update to a newer version, complete the *update* first. Only after that you can proceed to the migration.

This section is about how to migrate object feature vectors to another neural network model.

Do the following:

1. Create a backup of the Tarantool-based feature vector database in any directory of your choice, for example, `/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. Create new shards that will host regenerated feature vectors.
 1. Navigate to the `/opt/findface-cibr/data/findface-tarantool-server` directory and find out the number of shards by counting the number of directories.

Note: There are eight shards in the example below.

```
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. Create directories that will host files of the new shards.

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

3. Open the `/opt/findface-cibr/configs/findface-extraction-api/findface-extraction-api.yaml` configuration file and replace the face extraction model with the new one in the `face_emben` parameters.

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

extractors:
  ...
models:
```

(continues on next page)

(continued from previous page)

```
...
face_emben: face/<new_model_face>.cpu<gpu>.fnk
...
```

Restart the findface-cibr-findface-extraction-api-1 container.

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

4. In the docker-compose.yaml file, create a new service for each new shard. To do that, copy the existing service and replace the name of the shard, ports in CFG_LISTEN_PORT, TT_LISTEN and the path to the shard in volumes section.

```
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']
  ...
```

5. Start the new shards by starting the containers.

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

6. Create a configuration file with migration settings migration.yaml based on the example below.

```
sudo vi migration.yaml

extraction-api:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 0s
  extraction-api: http://127.0.0.1:18666
storage-api-from: # current location of the gallery
  timeouts:
    connect: 5s
```

(continues on next page)

(continued from previous page)

```

    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: ''
    workers_num: 3
    objects_limit: 1000
    extraction_batch_size: 8
    normalized_storage:
      type: webdav
      enabled: True
      webdav:

```

(continues on next page)

(continued from previous page)

```

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

```

In the `storage-api-from` section, specify the old shards to migrate the data from.

```

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: ''
...

```

In the `storage-api-to` section, specify the new shards that will host migrated data.

```

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/
  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/

```

(continues on next page)

(continued from previous page)

```

slave: ''
- master: http://127.0.0.1:8118/v2/
  slave: ''
...

```

7. Copy the migration.yaml file into the findface-cibr-findface-sf-api-1 container. Launch the sf-api-migrate utility with the -config option and provide the migration.yaml configuration file.

```

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

```

Note: The migration process can take up a significant amount of time if there are many events and records in the system.

8. After the migration is complete, remove services for the old shards from the docker-compose.yaml file and stop their containers.

```

sudo docker-compose up -d --remove-orphans

```

9. Open the /opt/findface-cibr/configs/findface-sf-api/findface-sf-api.yaml configuration file and adjust the shards ports, subject to the new shards settings. Restart the findface-cibr-findface-sf-api-1 container.

```

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/
      slave: ''

sudo docker-compose restart findface-sf-api

```

10. Migrate clusters as well. To do so, execute the following command:

```

sudo docker exec -it findface-cibr-findface-multi-legacy-1 /opt/findface-security/
↪bin/python3 /tigre_prototype/manage.py migrate_clusters --face --use-thumbnail

```

As a result, the system will regenerate feature vectors for the existing cluster events and automatically launch clustering to rebuild clusters.

2.5.4 Modify Feature Vector Database Structure

Sometimes it may be necessary to apply a new structural schema to your Tarantool-based feature vector database, for example, when updating to the latest version of the product, or when you want to enhance the default database structure with additional parameters, advanced face metadata, and so on.

In this section:

- *About Database Structure*
- *Structure Modification*

About Database Structure

In FindFace CIBR, the database structure is set via the `/opt/findface-cibr/configs/findface-tarantool-server/tnt-schema.lua` file.

The structure is created as a set of spaces and fields. Each field is described with the following parameters:

- `id`: field id;
- `name`: field name, must be the same as the name of a relevant object parameter;
- `field_type`: data type (`unsigned|string|set[string]|set[unsigned]`);
- `default`: field default value. If a default value exceeds `1e14 - 1`, use a string data type to specify it, for example, `"123123..."` instead of `123123...`.

You can find the default `tnt-schema.lua` file [here](#).

Structure Modification

To modify the database structure, do the following:

1. Create a backup of the Tarantool-based feature vector database in any directory of your choice, for example, `/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. Modify the database structure by applying the new schema to the `tnt-schema.lua` file.

```
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. Open the `docker-compose.yaml` file. Make sure each Tarantool shard service contains defined variable `CFG_EXTRA_LUA: loadfile("/tnt_schema.lua")` in the environment section.

```
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. Purge data from all the directories relevant to active shards.

```
sudo rm /opt/findface-cibr/data/findface-tarantool-server/shard-*/{index,snapshots,
↪xlogs}/*
```

5. Restart the findface-tarantool-server shards.

```
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. Restore the Tarantool database from the backup.

```
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;'
```

Important: If some fields were removed from the new database structure, you have to first manually delete the corresponding data from the backup copy.

See also:

Custom Metadata in Tarantool

2.5.5 Check Component Status

Check the status of containers once you have encountered a system problem, using the following commands:

```
docker ps
sudo docker container inspect <container_id>/<container_name>
sudo docker container stats <container_id>/<container_name>
```

2.5.6 Logging

Consulting logs is one of the first things you should do to identify a cause of a problem. By default, the FindFace CIBR processes are logged to [Docker container logs](#), which can be accessed via the `docker logs` and `docker service logs` commands. In addition, Docker uses the [json-file logging driver](#), which caches container logs in JSON. You can configure Docker to use another logging driver, choosing from the [multiple logging mechanisms available](#).

This section describes how to set up Docker to use the `journald` logging driver, which sends container logs to the `systemd` journal. In this case, log entries are retrieved using the `journalctl` command, through the `journal` API, or the `docker logs` command. You can configure the `systemd` journal as well by using the instructions below.

In this section:

- [Configure Journald](#)
- [Enabling Journald Logging Driver](#)
- [Consult Logs](#)

Configure Journald

To configure the `systemd-journal` service, do the following:

1. Check whether the `/var/log/journal` directory already exists. If not, create it by executing the following command:

```
sudo mkdir /var/log/journal
sudo chmod 755 /var/log/journal
```

2. Open the `/etc/systemd/journald.conf` configuration file. Enable saving `journald` logs to your hard drive by uncommenting the `Storage` parameter and changing its value to `persistent`. Disable filtering in `systemd-journal` as well:

```
sudo vi /etc/systemd/journald.conf

[Journal]
...
Storage=persistent
...
RateLimitInterval=0
RateLimitBurst=0
...
```

If necessary, uncomment and edit the `SystemMaxUse` parameter. This parameter determines the maximum volume of log files on your hard drive. Specify its value in bytes or use K, M, G, T, P, E as units for the specified size (equal to 1024 , 1024^2 , ... bytes).

```
...
SystemMaxUse=3G
```

3. Restart the `journald` service.

```
sudo systemctl restart systemd-journald.service
```

Enabling Journald Logging Driver

To enable Docker to use the `journald` logging driver, do the following:

1. Add the following line to the `/etc/docker/daemon.json` configuration file.

Tip: This file may not be present on your system. Check whether it exists and if it does not, create the `/etc/docker` directory first and then the file.

```
sudo mkdir /etc/docker
sudo touch /etc/docker/daemon.json
```

```
sudo vi /etc/docker/daemon.json

{
  "log-driver": "journald"
}
```

2. Stop all Docker containers.

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

3. Restart the Docker service.

```
sudo systemctl restart docker
```

4. Start all Docker containers.

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

Consult Logs

Use any convenient method to work with the `journald` logs. The following commands are a good place to start:

- Display all logs:

```
journalctl -fa
```

- Display logs by container ID:

```
journalctl CONTAINER_ID=<container_id> -f
```

- Display logs by container name:

```
journalctl CONTAINER_NAME=<container-name> -f
```

See also:

Audit Log

2.5.7 Troubleshoot Licensing and findface-ntls

When troubleshooting licensing and findface-ntls (see [Licensing](#)), the first step is to retrieve the licensing information and findface-ntls status. You can do so by sending an API request to findface-ntls. Necessary actions are then to be undertaken, subject to the response content.

Tip: Please do not hesitate to contact our experts on troubleshooting by support@ntechlab.com.

Note: The online licensing is done via the NtechLab Global License Manager license.ntechlab.com. Check its availability. A stable internet connection and DNS are required.

To retrieve the FindFace CIBR [licensing](#) information and findface-ntls status, execute on the findface-ntls host console:

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

The response will be given in JSON. One of the most significant parameters is `last_updated`. It indicates in seconds how long ago the local license has been checked for the last time.

Interpret the `last_updated` value as follows:

- [0, 5] — everything is alright.
- (5, 30] — there may be some problems with connection, or with the local drive where the license file is stored.
- (30; 120] — almost certainly something bad happened.
- (120; ∞) — the licensing source response has been timed out. Take action.
- "valid" -> "value": false: connection with the licensing source was never established.

```
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",

```

(continues on next page)

(continued from previous page)

```
"value": 9007199254740991,
"current": 0
},
{
  "type": "number",
  "name": "cameras",
  "value": 4294967295,
  "current": 0
},
{
  "type": "number",
  "name": "extraction_api",
  "value": 256,
  "current": 0
},
{
  "type": "boolean",
  "name": "gender",
  "value": true
},
{
  "type": "boolean",
  "name": "age",
  "value": true
},
{
  "type": "boolean",
  "name": "emotions",
  "value": true
},
{
  "type": "boolean",
  "name": "fast-index",
  "value": true
},
{
  "type": "boolean",
  "name": "sec-genetec",
  "value": false
},
{
  "type": "boolean",
  "name": "beard",
  "value": false
},
{
  "type": "boolean",
  "name": "glasses",
  "value": false
},
{
  "type": "boolean",
```

(continues on next page)

(continued from previous page)

```

        "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 Manually Purge Old Data from Database

To manually remove old data from the FindFace CIBR database, use the `cleanup` utility. You can separately remove the following data:

- audit-logs,
- remote monitoring events (applicable only if the *Integration with Remote Facial Recognition Systems* is enabled).

The `cleanup` utility runs in the `findface-cibr-findface-multi-legacy-1` container. To invoke the `cleanup` help message, execute:

```

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]

```

(continues on next page)

(continued from previous page)

```

[--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_KEEP_
↪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_SEND_
↪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]
[--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

```

(continues on next page)

(continued from previous page)

```

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
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.

```

(continues on next page)

(continued from previous page)

```

--traceback          "/home/djangoprojects/myproject".
--no-color           Don't colorize the command output.
--force-color        Force colorization of the command output.
--skip-checks        Skip system checks.

```

To remove audit logs, older than a given number of seconds, use the `--audit-logs-max-age` option. For example, to remove audit logs older than 5 days (432000 seconds), execute:

```

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

```

To remove remote monitoring events older than 5 days (432000 seconds), execute:

```

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

```

Important: You must provide at least one of the mentioned arguments.

2.5.9 Reset Password

To reset a user password to the FindFace CIBR web interface, execute the following command:

```

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 Migrate Data to Another Disk

High disk load may lead to delays in event arrivals. In severe cases, it might result in complete inoperability of FindFace CIBR. One of the means for reducing the disk load is to migrate the FindFace CIBR data storages to another disk.

In this section:

- *Prepare Disk*
- *Migrate Photo and Video Storage*
- *Migrate Main Database (PostgreSQL)*

Prepare Disk

To prepare a disk for the data migration, do the following:

1. Create a new mount point (/mnt/ffdata in our example).

```
sudo mkdir /mnt/ffdata
```

2. Create a partition.

```
sudo parted /dev/sdb
mklabel gpt
mkpart primary ext4 1MiB 100%
q
sudo mkfs.ext4 /dev/sdb1
```

3. Learn the UUID of the partition (sdb1 in our example).

```
sudo blkid | grep sdb1
/dev/sdb1: LABEL="data" UUID="0638ebe0-853e-43ea-8f35-bfae305695d1" TYPE="ext4"
↳ PARTUUID="8cebaacc-77d7-4757-b4c6-14147e92646c"
```

4. Add the partition to fstab to make it automatically mount on booting.

```
sudo vi /etc/fstab
-----
#DATA mount
UUID=0638ebe0-853e-43ea-8f35-bfae305695d1 /mnt/ffdata/ ext4 auto,user,rw
↳ 0 2
-----
```

5. Mount all the filesystems.

```
sudo mount -a
```

Migrate Photo and Video Storage

Note: We recommend that you get acquainted with *FindFace CIBR Data Storages* before starting storages migration.

To migrate FindFace CIBR photo and video storages, do the following:

1. Inside the mount point, create directories (e.g., /mnt/ffdata/ffupload and /mnt/ffdata/findface-security) to store photos and videos. Then, move contents of the /opt/findface-cibr/data/findface-upload/uploads/ and the /opt/findface-cibr/data/findface-multi-legacy/uploads/ directories to the newly created directories.

```
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/
```

(continues on next page)

(continued from previous page)

```
sudo rm -r /opt/findface-cibr/data/findface-multi-legacy/uploads/
sudo rm -r /opt/findface-cibr/data/findface-upload/uploads/
```

2. Mount the created directories (/mnt/ffdata/ffupload and /mnt/ffdata/findface-security in the example) into the corresponding containers. To do so, open the /opt/findface-cibr/docker-compose.yaml configuration file and list them in the volumes of the sections mentioned in the example instead of the default directories.

```
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-
↪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. Rebuild all FindFace CIBR containers.

```
cd /opt/findface-cibr

sudo docker-compose down

sudo docker-compose up -d
```

Migrate Main Database (PostgreSQL)

To migrate the PostgreSQL database, do the following:

1. Stop all FindFace CIBR containers:

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

2. Move the /opt/findface-cibr/data/postgresql directory to the /mnt/ffdata directory.

```
sudo mv /opt/findface-cibr/data/postgresql /mnt/ffdata
```

3. Specify the new path of the postgresql directory (/mnt/ffdata/postgresql in the example) to mount into the findface-cibr-postgresql-1 container. To do so, open the /opt/findface-cibr/docker-compose.yaml configuration file and list the /mnt/ffdata/postgresql directory in the volumes of the postgresql section instead of the default /opt/findface-cibr/data/postgresql directory.

```
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. Start all FindFace CIBR containers.

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

2.6 Appendices

2.6.1 Installation File

FindFace CIBR installation configuration is automatically saved to a file /tmp/<findface-installer-*.json. You can edit this file and use it to install FindFace CIBR on other hosts without having to answer the installation questions again.

Tip: See *FindFace CIBR Standalone Automated Deployment* to learn more about the FindFace CIBR installer.

Important: Be sure to remove fields *.config, exp_ip, and int_ip before installing FindFace CIBR on a host with a different IP address.

Here is an example of the installation file.

2.6.2 Neural Networks Summary

Here you can find a summary of neural network models created by our Lab and used in FindFace CIBR.

Find installed models at `/opt/findface-cibr/models/`.

Important: The default face biometrics model upon a clean install is `mango_320`.

Face detection

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

face.jasmine_fast.003.cpu.fnk  face.jasmine_fast.003.gpu.fnk
```

Face image normalization

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

bee.v3.cpu.fnk                crop2x.v2_maxsize400.cpu.fnk  facenorm.multicrop_full_
↪center_size400.cpu.fnk      crop2x.v2_maxsize400.gpu.fnk  facenorm.multicrop_full_
bee.v3.gpu.fnk                crop2x.v2_maxsize400.gpu.fnk  facenorm.multicrop_full_
↪center_size400.gpu.fnk      crop2x.v2_no_maxsize.cpu.fnk  facenorm.multicrop_full_
bee_fast.cpu.fnk              crop2x.v2_no_maxsize.gpu.fnk  facenorm.multicrop_full_
↪crop2x_size400.cpu.fnk      crop2x.v2_no_maxsize.gpu.fnk  facenorm.multicrop_full_
bee_fast.gpu.fnk              crop2x.v2_no_maxsize.gpu.fnk  facenorm.multicrop_full_
↪crop2x_size400.gpu.fnk      crop1x.v2_maxsize400.cpu.fnk
crop1x.v2_maxsize400.cpu.fnk  cropbbox.v2.cpu.fnk
crop1x.v2_maxsize400.gpu.fnk  cropbbox.v2.gpu.fnk
```

Face recognition

```
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
```

Face attribute recognition

```
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 FindFace CIBR Data Storages

In this section:

- *List of Storages*
- *Feature Vector Database Galleries*

List of Storages

FindFace CIBR uses the following data storages:

- Tarantool-based feature vector database that stores face feature vectors and recognition events: `/opt/findface-cibr/data/findface-tarantool-server`.
- PostgreSQL-based main system database `ffsecurity` that stores internal system data, records, and user accounts: `/opt/findface-cibr/data/postgresql`.
- The `/opt/findface-cibr/data/findface-upload/uploads` directory that stores normalized face images.
- The `/opt/findface-cibr/data/findface-multi-legacy/uploads` directory that stores photos uploaded to records, photos uploaded to user profiles, video and photo files uploaded to cases, full frames and thumbnails of clusters and remote monitoring events.

Feature Vector Database Galleries

There are the following FindFace CIBR-specific galleries in the Tarantool-based feature vector database:

- `ffsec_face_case_events`: feature vectors extracted from faces detected in the video or photo uploaded to a case.
- `ffsec_face_case_clusters`: centroids of case face clusters.

USER'S GUIDE

This chapter describes how to work with the FindFace CIBR web interface, including its advanced possibilities, and will be of interest to administrators, operators, and other users.

3.1 Getting Started

Once you have successfully *deployed and configured* FindFace CIBR, it's time to open the web interface, and get started. In this chapter, you can find a recommended sequence of steps that will help you harness your system's complete functionality.

Organize Watch Lists and Global Record Index

1. *Create a new watch list* or use the default one. A watch list is an entity that allows you to classify individuals by arbitrary criteria, e.g., wanted, suspects, etc.
2. Upload records of individuals to the global record index and add them to watch lists. See *Record Index*.

Connect FindFace CIBR to Remote Systems

Integrate FindFace CIBR with *remote facial recognition systems*. In this case, the server known as a puppeteer will be pushing record index data to remote servers known as puppets. In return, it will be pulling recognition events matching with the records.

FindFace CIBR in Action

1. *Create case files* and process video footages and photo files from crime scenes to define participants.
2. *Search for faces* across the system.
3. *Compare two faces* and verify that they belong to the same individual.
4. *Build* detailed reports on search results and record index.
5. *Harness* the FindFace CIBR comprehensive and searchable audit log to enhance your system protection.
6. View alerts from remote facial recognition systems and search remote systems for specific individuals. See *Remote Alerting and Remote Search*.

Basic Maintenance

1. Manually *purge* old data.
2. Regularly *back up* the database.

Go Further

Harness the FindFace CIBR functions through *HTTP API*.

3.2 Web Interface Basics

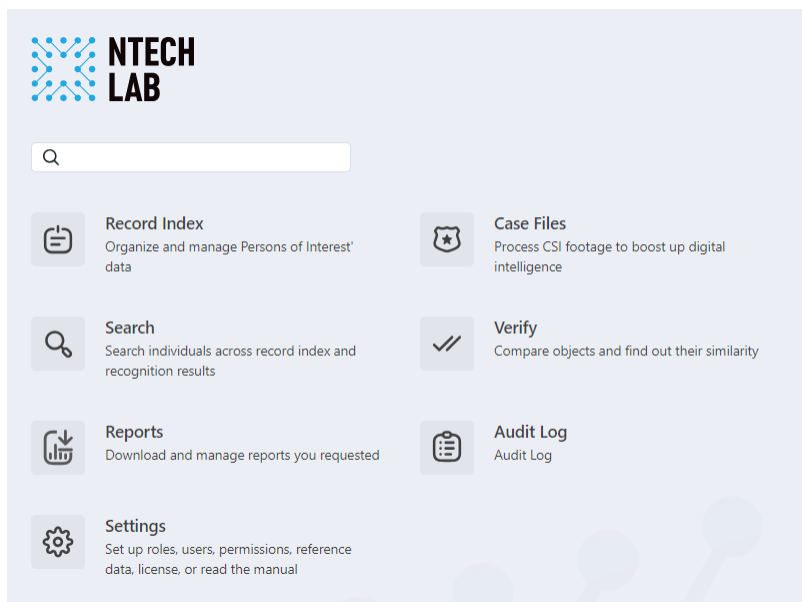
Use the web interface to interact with FindFace CIBR. To open the web interface, enter its basic address in the address bar of your browser, and log in.

Note: The basic address is set during *deployment*.


Important: To log in for the first time, use the admin account created during *deployment*. To create more users, refer to *Role and User Management*.

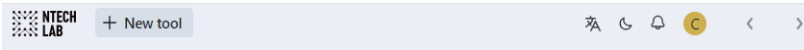
3.2.1 Navigation

There is a different number of items in the navigation bar depending on the user's role.



3.2.2 Web Interface Language and Theme

To change the system language and a theme, click  on a top panel.



3.3 Record Index

Record index stores records of individuals, including their biometric data, related documents, links to relevant cases, and other important data.

To create records in bulk, use the *console bulk record upload* functionality.

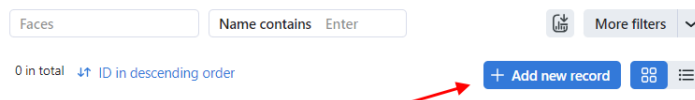
In this section:

- *Create Record*
- *Batch Record Upload*
- *Autopopulation with Criminal Investigation Artifacts*
- *Filter Records*
- *Purge Record Index*

3.3.1 Create Record

To create a record manually, do the following:

1. Navigate to the *Record Index* tab.
2. Click **+ Add new record**.



3. Specify the record name. If necessary, add a comment.
4. From the *Watch lists* drop-down menu, select a watch list for the record (or several watch lists, one by one).
5. Make sure that *Record active* is enabled. If a record is inactive, it is excluded from the *case analysis, remote alerting and search*.
6. Save the record. You will see additional tabs appear.
7. On the tab *Info*, attach related files.

Mrs. Smith

Info

Faces

Case Clusters

Related Cases

Related records

Record name

Mrs. Smith

Watch lists

Hitmen

Comment

☒ Record active

No files

Attach the first one

ID

1

Created

05.12.2023 10:16:56

8. On the *Faces* tab, upload images of the individual’s face. Supported formats: WEBP, JPG, BMP, PNG, HEIC.

Mrs. Smith

Info

Faces

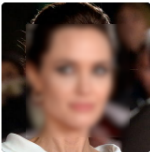
Case Clusters

Related Cases

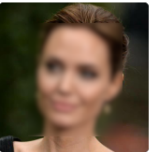
Related records

↑ ID in descending order

Upload photo



4568131716156140505



4568131702684881870

9. On the *Case Cluster* tab, *Related Cases* tab and *Related Records* tab, the data will be displayed if the record automatically matches the case cluster, or the connection is set manually during *people analysis*.

3.3.2 Batch Record Upload

If there are too many records to create, you can batch load records into the record index. Do the following:

- 1. Navigate to the *Record Index* tab.
- 2. Click + *Add new record* → Try *batch record upload*.
- 3. Select files or folder to drag and drop to upload.

Faces

Drag and drop files to upload
select files or select a folder

Name prefix

☐ Use file name

Name postfix

Comment prefix

☐ Use file name

Comment postfix

Watch lists

Default Watch List

Group photo

Reject

Parallel upload

2

5

10

20

Start



120

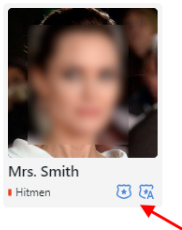
Chapter 3. User’s Guide

4. Specify a name prefix and postfix. If necessary, add a comment prefix and postfix.
5. Select a watch list (or several watch lists, one by one) from the drop-down menu.
6. Click *Start*.

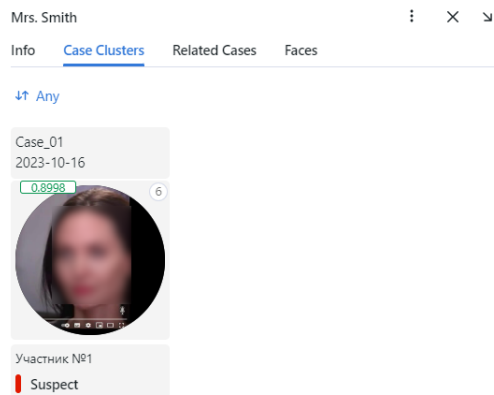
3.3.3 Autopopulation with Criminal Investigation Artifacts

While you are performing your duties using FindFace CIBR, investigating cases and analyzing CSI footage, a record is automatically populated with the case data.

If the record has an automatch with the face from the photo or video, uploaded in the case, then a sign  will be displayed on the record thumbnail. If the connection is set manually, then a sign  will be displayed.

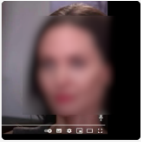


Click it to navigate to the *Case Clusters* tab or *Related Cases* of the record.




If the selected case cluster isn't a case participant, navigate to the connections wizard by clicking the face. See [Detected People Analysis](#).

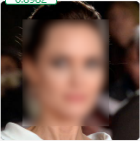
Clusters



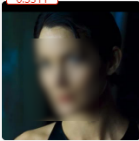
This is the individual who was automatically recognized while processing case file footage. If they are relevant to the case, mark them as a participant. If you are sure that there is a match with the other recognized individuals, or with the index records, or with the participants of other cases, establish a link by clicking the Add button




Record index



Mrs. Smith
Hitmen




Triniti
Matrix



Mr. Smith
Hitmen

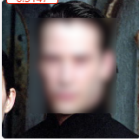
Clusters in Other Cases

Case_02



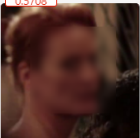
0.5356

Case_02

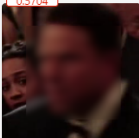


0.5147

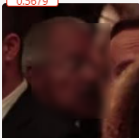
Clusters in Current Case



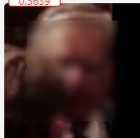
0.5703



0.5704



0.5679



0.5639

Mark as participant

Save and close

If the selected case cluster is a case participant, navigate to the *case participant record* by clicking the face.

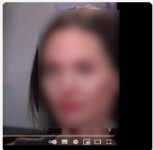
- On the *Info* tab you can modify participant information, if needed.

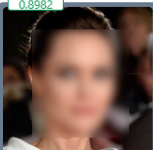
Participant №1

Info

Cluster events

Connections





0.8982

Mrs. Smith
Hitmen

Participant Info

Name

Participant №1

Type

Suspect

Comment

Case Info

Case Name

Case_01

Case ID

1

Incident Date

2023-10-16

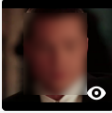
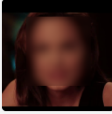
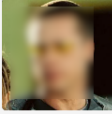

Case Date

2024-04-12

- On the *Cluster events* tab you can view detected face images of the participant and other face images of the case cluster.

Участник №1

Info Cluster events Connections

	<p>ID: 55</p> <p>Created: 2023-11-27 12:48:42</p> <p>42 years No mask</p> <p>No beard Neutral expression</p> <p>Man Head turn: -4°</p> <p>No glasses Head tilt: 8°</p>
	<p>ID: 35</p> <p>Created: 2023-11-27 12:48:28</p> <p>28 years No mask</p> <p>No beard Happiness</p> <p>Woman Head turn: 1°</p> <p>No glasses Head tilt: 1°</p>
	<p>ID: 5</p> <p>Created: 2023-11-27 12:38:50</p> <p>38 years No mask</p> <p>No beard Neutral expression</p> <p>Man Head turn: 23°</p> <p>Sunglasses Head tilt: 5°</p>
	<p>ID: 4</p> <p>Created: 2023-11-27 12:38:50</p> <p>30 years No mask</p> <p>No beard Neutral expression</p> <p>Woman Head turn: 10°</p> <p>Sunglasses Head tilt: 2°</p>

- On the *Connections* tab you can view connections of a participant with persons from other clusters.

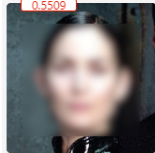
Участник №1

Info Cluster events Connections

Case_02

2023-11-27

0.5509



Triniti

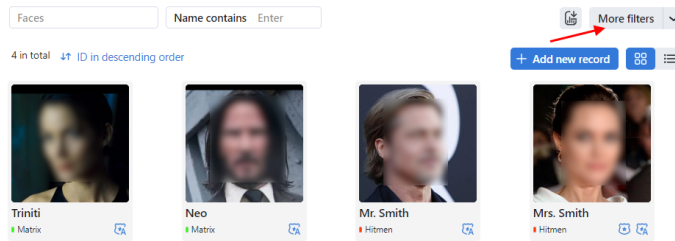
Matrix

3.3.4 Filter Records

The most frequently used filters for the record index are available in the upper part of the window.

To display the entire set of filters, click the *More filters* button. Here it is:

- *Watch lists*: display records from selected watch lists.
- *Faces*: filter records by presence of a face image.
- *Filling*: display only empty records, only filled ones, or any records.
- *Name contains*: filter records by name.
- *ID*: display a record with a given ID.
- *Status*: filter records by status (only active, only inactive, or any status).
- *Match type*: filter records by match type.

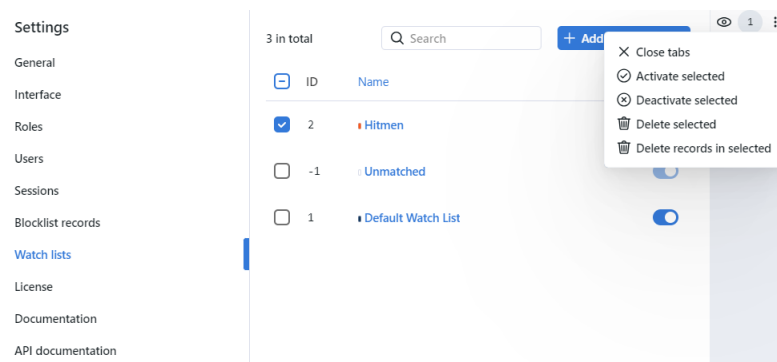


You can sort out records on the list by ID.

3.3.5 Purge Record Index

You can purge the record index entirely or by watch lists in one click. Do the following:

1. Navigate *Settings* → *Watch Lists*.
2. Select one or several watch lists.
3. Click *Delete records in selected*.



3.4 Case Files

FindFace CIBR allows for conducting case management and performing investigation by analyzing associated video footage and photo files. This functionality is available on the *Case Files* tab.

In this section:

- *Video File Formats*
- *Case Investigation Workflow*
- *Create Case File*
- *Case Access Permissions and Related Documents*
- *Upload and Process Video File*
- *Video processing parameters*

- *Upload and Process Photo File*
- *Detected People Analysis*
- *Case Participant Records*
- *Case File Archive*

3.4.1 Video File Formats

Video footage used for case investigations is accepted in a wide variety of formats. Click [here](#) for listing.

3.4.2 Case Investigation Workflow

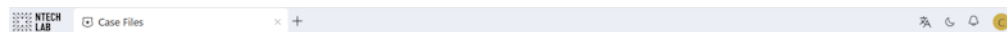
Case investigation is conducted in the following way:

1. Create a new case file. Specify the incident date and name, the case ID in a registry, and the incident date.
2. Specify case details and set up access permissions for it.
3. Upload a video footage and photo files from the criminal scene. Set up video processing parameters if needed. Process the video. The system will return human faces detected in it.
4. Parse the detected faces. Figure out which individual is likely to be a suspect, a victim, or a witness. Other individuals will be considered not relevant to the case. Link faces to matching records in the [global record index](#) and participants of other cases.
5. Fill in case participant records. Specify their names and attach related documents.
6. Keep returning to the case file to supplement it with new materials, as the investigation progresses.

3.4.3 Create Case File

To create a case file, do the following:

1. Navigate to *Case Files*.
2. Click + *New case file*.



New case file

You are opening a new case file. Be sure to specify its name and number (ID), and the incident date

+ New case file



3. Enter the descriptive name of the case. Specify the incident date.
4. Add a comment, if needed.

- 5. Specify the case ID and date in the registry, if needed.
- 6. Click *Add >*. As a result, the case file will be added to the case list.

New case file

You are opening a new case file. Be sure to specify its name and number (ID), and the incident date

Name

Case_01

Incident date

16.10.2023

Comment

Case ID in registry

1111

Case date

16.10.2023

Add >

3.4.4 Case Access Permissions and Related Documents

After you create a case, specify its details and set up access permissions. Do the following:

- 1. Click the case on the case list to open it.

All 2

Open 2

Archived 0

Unacknowledged clusters

More filters

2 in total

Search

+ New case file

<input type="checkbox"/>	Name	Creator	Added	Updated	Videos	Clusters	Participants	Status
<input type="checkbox"/>	Case_01	Charlie Root	2024-01-09 11:07:39	2024-01-09 11:07:39	0	3	0	Open
<input type="checkbox"/>	Case_01	Charlie Root	2023-12-29 11:32:56	2023-12-29 11:32:56	2	43	0	Open

- 2. Click *Set access permissions* to modify the default access permissions.
- 3. Attach one or several files relevant to the case.
- 4. Click *Save*.

Case_01 [Information](#) → [Videos](#) → [Photos](#) → [Clusters](#) → [Participants](#)

Name Incident date

Comment

Case ID in registry Case date

3 in total

Name	View	Change	Delete
Administrator <input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Operator <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
User <input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

No files [Attach the first one](#)

[Save](#)

3.4.5 Upload and Process Video File

To upload and process a CSI video footage, do the following:

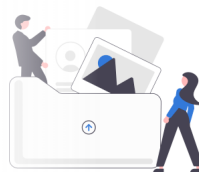
1. Navigate to the *Case files* tab.
2. Click the case on the case list to open it.
3. Navigate to the *Videos* tab.
4. Click the + *Add video* button.

Case_01 [Information](#) → [Videos](#) → [Photos](#) → [Clusters](#) → [Participants](#)

New video

Please upload your first video

[+ Add video](#)




5. Specify a URL or select a file. Click *Upload*.


File URLs ×

Enter file URLs

Use Enter to add multiple URLs



Drag and drop files to upload or
[Select files](#)


 [pittjolie.mp4](#) 2.26Mb ×

[Upload](#)


6. The video file will be added to the case video uploads. Click *Upload* to upload another video file.

Case_01 Information → **Videos 1** → Photos 0 → Clusters 0 → Participants 0

1 in total ⬆️ Upload

<input type="checkbox"/>	ID	Image	Name	Status	Face clusters
<input type="checkbox"/>	1		pittjolie.mp4 2.3 MB	Ready for processing	0


7. Click the video on the list to open the processing configuration wizard. Set up the *video processing parameters* and start face identification. With a progress bar in a status column, you can follow the progress of a video processing.

<input type="checkbox"/>	ID	Image	Name	Status	Face clusters
<input type="checkbox"/>	2		Towers, Ericson et al. (2018) - Interview in a forensic context, importance and benefits highlighted 52.7 MB	26% Processing	0

8. If you need to free disk space, but keep recognized faces and clusters, you can *Delete video files only*. To delete an uploaded file completely, select *Delete*.

Case_01 Information → **Videos 1** → Photos 1 → Clusters 3 → Participants 1

1 in total ⬆️ Upload


<input type="checkbox"/>	ID	Image	Name	Status	Face clusters
<input type="checkbox"/>	1		pittjolie.mp4 2.3 MB	Completed	<div> ⋮ <ul style="list-style-type: none"> Process Delete video files only Delete Reset to default > </div>

3.4.6 Video processing parameters

1. For each video, you will be provided with complete statistics such as current session duration, the number of objects processed with errors after the last job restart, the number of frame drops, and other data. To consult these data, click the name of the video on the list and go to the *Info* tab.

pittjolie.mp4 ⋮ × ↵

Info General Advanced Zones Faces



Info

ID	1
Status	Waiting
Process duration	
Frames dropped	
Job starts	
Objects statistics	
Face clusters	0

2. On the *General* tab, you can change name if needed.

pittjolie.mp4 ⋮ × ↵

Info General Advanced Zones Faces

Name	File size
<input type="text" value="pittjolie.mp4"/>	2 Mo

Start time

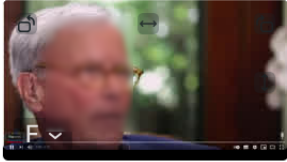
<input type="text" value="01.01.2020"/>	<input type="text" value="12:00"/>	<input type="button" value="Now"/>	<input type="button" value="Clear"/>
---	------------------------------------	------------------------------------	--------------------------------------

3. On the *Advanced* tab, fine-tune the video:

pittjolie.mp4 ⋮ × ↗

Info General **Advanced** Zones Faces

Transformation



☐ Mirror
☐ Flip
☐ Rotate clockwise
☐ Rotate counterclockwise

Posting faces

Timeout ⌵ 15000 Ms

☒ Verify SSL certificate

Timestamp

☐ Retrieve timestamps from stream Add to timestamps ⌵ 0 Sec

Other

FFmpeg parameters

Play speed limit ⌵ -1 Force input format Minimum motion intensity ⌵ 0

☒ Read frames from source without drops

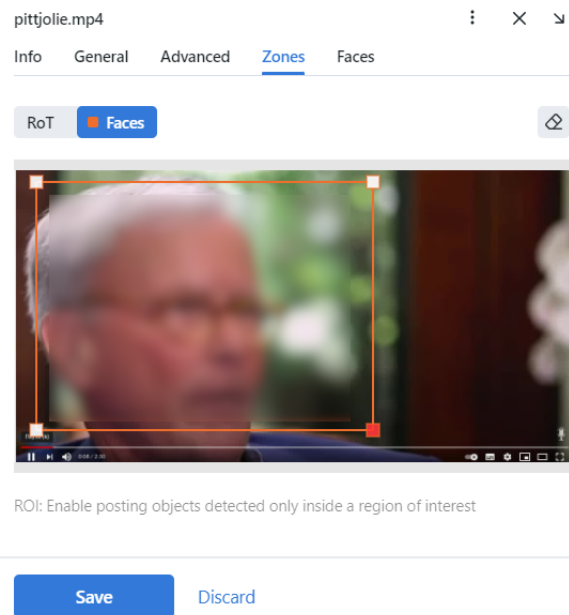
Router URL (router_url)

- If needed, change the video orientation.

Important: Be aware that the `findface-multi-legacy` server rotates the video using post-processing tools. It can negatively affect performance. Rotate the video via the camera functionality wherever possible.

- *Timeout:* Specify the timeout in milliseconds for posting detected objects.
- *Verify SSL certificate:* Select to enable verification of the server SSL certificate when the object tracker posts objects to the server over https. Deselect the option if you use a self-signed certificate.
- *Retrieve timestamps from stream:* Select to retrieve and post timestamps from the video stream. Deselect the option to post the current date and time.
- *Add to timestamps:* Add the specified number of seconds to timestamps from the stream.
- *FFMPEG parameters:* FFMPEG options for the video stream in the key-value format, for example, `["rtsp_transpotr=tcp", "ss=00:20:00"]`.
- *Play speed limit:* If less than zero, the speed is not limited. In other cases, the stream is read with the given `play_speed`. Not applicable for live-streaming.
- *Force input format:* Pass FFMPEG format (mxg, flv, etc.) if it cannot be detected automatically.
- *Minimum motion intensity:* Minimum motion intensity to be detected by the motion detector.
- *Read frames from source without drops:* If `findface-video-worker` does not have enough resources to process all frames with objects, it drops some of them. If this option is active (true) `findface-video-worker` puts excessive frames on a waiting list to process them later.
- *Router URL (router_url):* IP address for posting detected objects to external video workers from `findface-video-worker`. By default, `'http://127.0.0.1'`.

- Specify the region of tracking within the camera field and region of interest (*Zones*). Click *Save*.



The region of tracking enables detecting and tracking faces only inside a clipping rectangle. You can use this option to reduce the video object detector load. The region of interest enables posting objects detected only within its boundaries.

- On the *Faces* tab, specify settings for face detection.

pittjolie.mp4

Info General Advanced Zones **Faces**

☒ Enabled [Reset face parameters](#)

Filter

Size - Quality

Compression

JPEG quality

☐ Full frame in PNG

Tracking

Overlap threshold

Track maximum duration

Forced termination interval

☐ Send track history

ROT

☐ Crop full frame

Posting

☒ Offline mode (overall_only)

Interval

☐ Post first object immediately ☐ Post in every interval

☐ Post track first frame ☐ Post track last frame

[Save](#) [Discard](#)

- **Size:** Minimum object size in pixels to post and maximum object size in pixels to post.
- **Quality:** The minimum quality of the object image for detection. The allowed range is from 0 to 1. The recommended reference value is 0.5, which corresponds to object images of satisfying quality. Do not change the default value without consulting with our technical experts (support@ntechlab.com).
- **JPEG quality:** Full frame compression quality.
- **Full frame in PNG:** Send full frames in PNG and not in JPEG as set by default. Do not enable this parameter without consulting with our technical experts (support@ntechlab.com) as it can affect the entire system functioning.
- **Overlap threshold:** The percentage of overlap of bboxes between two serial frames so that these bboxes are considered as one track. The range of values is from 0 to 1. Do not change the default value without consulting with our technical experts (support@ntechlab.com).
- **Track maximum duration:** The maximum approximate number of frames in a track, after which the track is forcefully completed. Enable it to forcefully terminate endless tracks, for example, tracks with objects from advertising media.
- **Forced termination interval:** Terminate the track if no new object has been detected within the specified time (in seconds).
- **Send track history:** Send array of bbox coordinates along with the event. May be applicable for external integrations to map the path of an object.
- **Crop full frame:** Select to crop the full frame to the size of the ROT area before sending it for recognition. The size of the full frame will be equal to the size of the ROT area.
- **Offline mode (overall_only):** By default, the system uses the offline mode to process the video, i.e., it posts

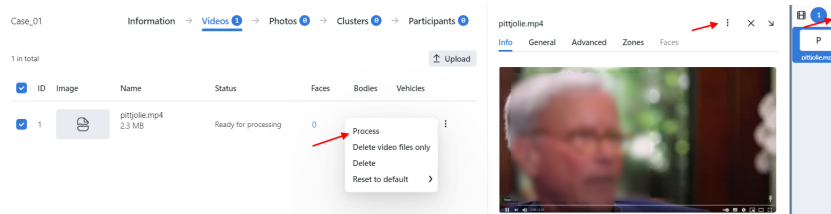
one snapshot of the best quality per track to save disk space. Disable it to receive more object snapshots if needed. If the offline mode is on, the parameters of the real-time mode are off.

Real-time mode parameters:

Note: These parameters are non-functional if the offline mode is on.

- *Interval*: Time interval in seconds (integer or decimal) within which the object tracker picks up the best snapshot in the real-time mode.
- *Post first object immediately*: Select to post the first object from a track immediately after it passes through the quality, size, and ROI filters, without waiting for the first `realtime_post_interval` to complete in real-time mode. Deselect the option to post the first object after the first `realtime_post_interval` completes.
- *Post track first frame*: At the end of the track, the first frame of the track will be additionally sent complimentary to the overall frame of the track. May be applicable for external integrations.
- *Post in every interval*: Select to post the best snapshot within each time interval (`realtime_post_interval`) in real-time mode. Deselect the option to post the best snapshot only if its quality has improved comparing to the posted snapshot.
- *Post track last frame*: At the end of the track, the last frame of the track will be additionally sent complimentary to the overall frame of the track. May be applicable for external integrations.

6. On the list of video uploads, click three dots → *Process* to start face identification.




The processing results will be available on the *Clusters* tab and on the *Cluster events* tab of the participant.

Important: When processing photo and video files, pay attention to the license limit on the number of images of a person's face.

The license limit for the number of images of a person's face takes into account not only the images uploaded to the *Record Index*, but also each cluster event generated during the processing of video and photo files in the case file.

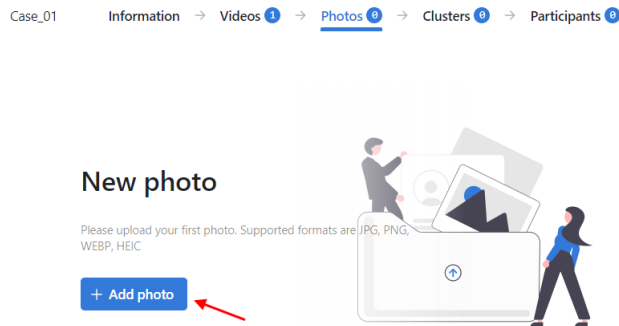
If a case file is deactivated (archived), the corresponding limit is released, and conversely: when a case file is extracted from the archive (reopened), all cluster events associated with it are taken into account in the limit.

Processing of a video or photo file within a case can be unsuccessful. This also applies to cases when the license limit has been reached. If a case file contains video and photo files that could not be processed, it is marked with an icon . In such case file, you can acknowledge clusters that automatically match the person's face in the record, but you can't save clusters and convert them into participants. The status of the case file will change either after successful processing of a video and / or photo file (to do this, restart file processing), or after deleting the file.

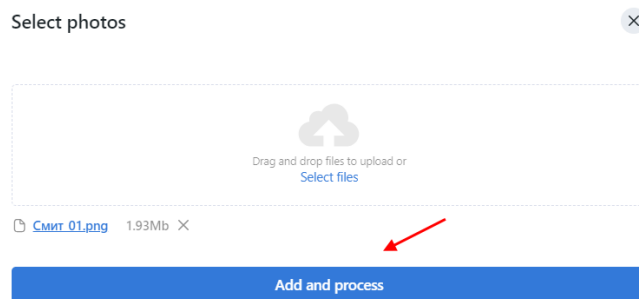
3.4.7 Upload and Process Photo File

To upload and process a CSI photo file, do the following:

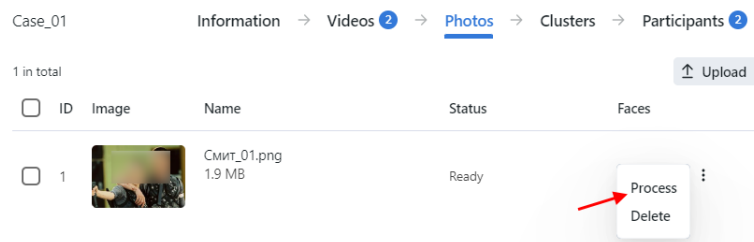
1. Navigate to *Case Files*.
2. Click + *Add photo*.



3. Select photos to upload.




4. Click *Add and process*. As a result, the photo file will be added to the list of photo uploads and processed.
5. To restart object identification on the photo file, click three dots → *Process*.



6. Click the name of an image in a list to see cluster events and modify the name of the photo file or add a comment, if needed.

Case_01.png





ID: 1
Created: 01.11.2023 10:31:28

Name
Case_01.png

Description

Status
Completed

Cluster events
2 in total [4*](#) [Any](#)

	ID: 50 Created: 2023-11-01 10:43:53 38 years No beard Man Sunglasses	No mask Neutral expression Head turn: 23° Head tilt: 5°
	ID: 49 Created: 2023-11-01 10:43:52 30 years No beard Woman Sunglasses	No mask Neutral expression Head turn: 10° Head tilt: 2°


3.4.8 Detected People Analysis

The clusters of faces, once detected in the video and photo files, are shown on the *Clusters* tab. Here you need to parse them subject to the role a person plays in the incident and establish links to relevant global records, other case files, participants of the same case, and other clusters.

If a detected object has a match in the record index, a case cluster thumbnail contains the following information:

- normalized object image
- record name
- the similarity between matched objects
- watch list(s)
- *Acknowledge* button

Click the *Acknowledge* button to acknowledge a case cluster.

Note: The sign  on the case list indicates that there is an unacknowledged case cluster in the case.

To analyze detected people, do the following:

1. Click on a face on the list.


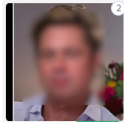
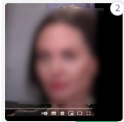
Case_01

Information → Videos [1](#) → Photos [4](#) → **Clusters [6](#)** → Participants [6](#)

Participant Acknowledged

More filters

3 in total

		
Unknown	Mr. Smith Hitmen Acknowledge	Mrs. Smith Hitmen Acknowledge

Tip: With the large number of clusters, use filters. Click the *More filters* button in the upper-right corner to display them.

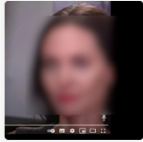
Note: Some filters from the list below may be hidden, depending on which recognition features are enabled.

- *Participant*: display only case clusters/only participants or any.
- *Acknowledged*: display only unacknowledged clusters/only acknowledged clusters or any.
- *Matches*: display case clusters only with/without matching the record or all events.
- *Watch lists*: display case clusters only for a selected watch list.
- *Date and time*: display only case clusters formed within a certain period.
- *ID*: display a case cluster with a given ID.
- *Photo ID*: display a case cluster with a given photo ID.
- *Video ID*: display a case cluster with a given video ID.


Specific filters for face clusters

- *Age*: display case clusters with people of a given age.
 - *Beard*: filter case clusters by the fact of having a beard.
 - *Emotions*: display case clusters with given emotions.
 - *Gender*: display case clusters with people of a given gender.
 - *Glasses*: filter case clusters by the fact of wearing glasses.
 - *Face mask*: filter case clusters by the fact of wearing a face mask.
2. Link the face to a matching record in the [global record index](#) and matching participants of other case files. Do so by clicking the *Add* button that appears when you hover over the photo. You can also link the face to participants of the same case and other clusters.
 3. Click the face in section *Clusters* to see detected faces in full screen and split some of them from a cluster, if needed.
 4. Click *Mark as participant*.

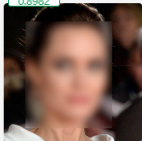
Clusters



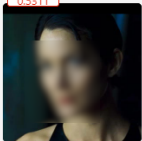
This is the individual who was automatically recognized while processing case file footage. If they are relevant to the case, mark them as a participant. If you are sure that there is a match with the other recognized individuals, or with the index records, or with the participants of other cases, establish a link by clicking the Add button.



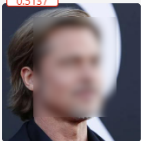
Record index



Mrs. Smith
Hitmen

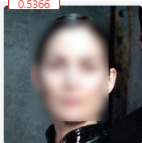


Triniti
Matrix

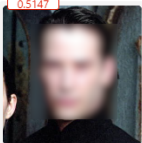


Mr. Smith
Hitmen

Clusters in Other Cases

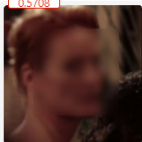


Case_02
0.5366

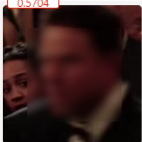


Case_02
0.5147

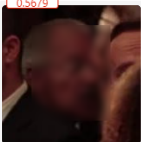
Clusters in Current Case



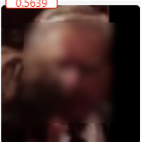
0.5703




0.5704



0.5679



0.5639


Mark as participant

Save and close

- Specify the participant name and type, and add a comment if necessary. This will create a new case participant record that you will be able to view on the *Participants* tab.

Save participant

Participant Info

Name

Participant 1

Type

Suspect

Comment

Case Info

Case Name

Case_01

Case ID

1

Incident Date

2023-10-16

Case Date

2024-04-12

Save and close

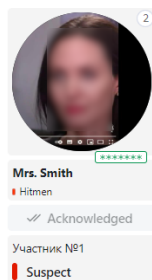
A participant can be one of the following types:

- victim
- witness
- suspect

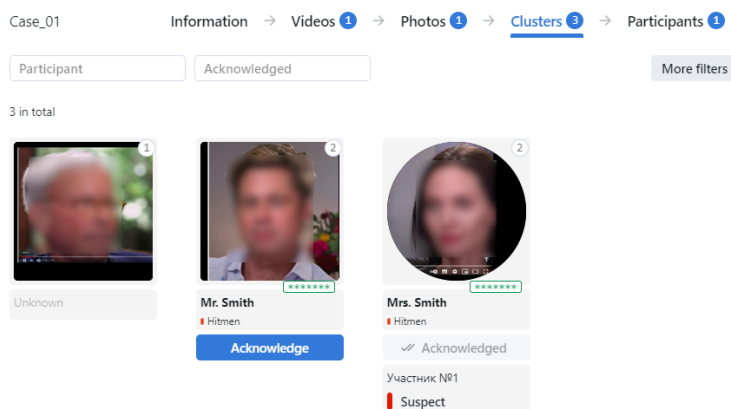
- Click *Save and close*.

As a result, the information will be added to a case cluster thumbnail and the face image will have a round frame. The number of cluster events is shown by the number in the upper right corner of the case cluster thumbnail.

Click the number to view full screen of the image in the new window. You can split the face from a cluster, if needed.



7. You can reopen the connections wizard on the *Clusters* tab by clicking on the participant's face again.

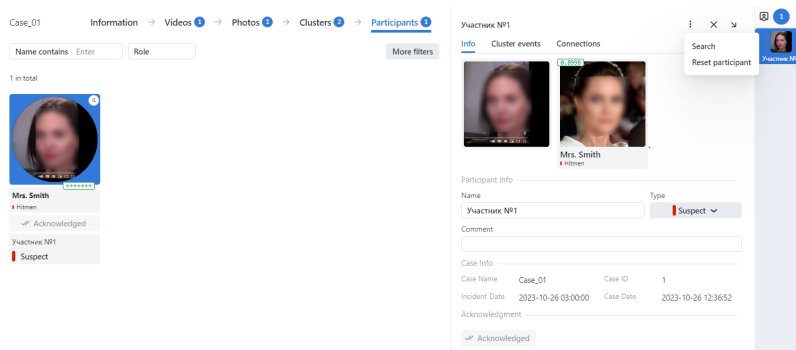


To access the participant record, navigate to the *Participants* tab.

3.4.9 Case Participant Records

The *Participants* tab provides access to all participant data collected so far.

To view a case participant record, click the relevant face on the list.



- On the *Info* tab you can modify participant information, if needed.
- On the *Cluster events* tab you can view detected face images of the participant and other face images of the case cluster.
- On the *Connections* tab you can view connections of a participant with persons from other clusters.

FindFace CIBR allows you to search detected objects from the case participant records. To navigate from the case participant record to the search tab, click three dots → *Search*.

If during a case investigation it is necessary to reset the participant record, click three dots → *Reset participant*.

With the large number of participants, use filters. Click the *More filters* button in the upper-right corner to display them.

- *Name contains*: display participants by name.
- *Role*: display participants for a selected role (Any/Victim/Witness/Suspect).
- *Date and time*: display participants by case date within a certain period.

3.4.10 Case File Archive

It's possible to label a case file as archived to indicate that the case is closed, or for another reason.

To archive/dearchive case files, do the following:

1. On the *Case Files* tab, select one or several case files.
2. Click *Archive* to archive the case files. Click *Delete* to delete them.

	Name	Creator	Added	Updated	Videos	Clusters	Participants	Status
<input type="checkbox"/>	Case_02 △	Charlie Root	2024-01-10 11:07:31	2024-01-10 11:07:31	0	3	0	Open
<input checked="" type="checkbox"/>	Case_01 △	Charlie Root	2024-01-10 11:03:49	2024-01-10 11:03:49	1	3	0	Open

If a case has been archived, you cannot edit it anymore (for example, change the name, add video or photo files, change processing settings, etc.).

- You cannot process or reprocess previously uploaded photo and video files.
- You cannot edit its clusters and mark them as participants.
- The search for case cluster events can no longer be performed.
- You can view clusters, participants and filter them.
- Cluster events from an archived incident will not be included in the license.
- There will be no rematching cluster events from an archived case file with the record index.
- However, the connections that are built on this case file in the records are saved.

If a case file is extracted from the archive (reopened), it can be edited again. The search for its cluster events becomes possible again.

To open the case again, select it and click *Extract from archive*.

You can filter the case file list by archived status: *All*, *Opened*, *Archived* or use additional filter by clicking *More filters*:

- *Unacknowledged clusters*: display cases by acknowledged status of the clusters.
- *Case ID in registry*: display a case with a given case ID in registry.
- *Date created*: display cases created within a certain period.
- *ID*: display a case with a given ID.

3.5 Search Faces in System

FindFace CIBR allows you to search for individuals throughout the entire system.

To find an individual, do the following:


1. Navigate to the *Search* tab.
2. Specify an object to search for in one of the following ways:
 - by record's URL or ID
 - by uploading a photo file

Define a search source

Enter a photo URL on the Internet or an internal entity ID/URL

Record
File

or upload a media file. Supported formats are JPG, PNG, WEBP, HEIC



Drag and drop file to upload or
[select file](#)

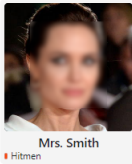
Note: The search by case cluster is available from the case participant records or by case cluster's URL or ID.

3. If you specified a record URL or ID, select a photo from it. If there are multiple photos, you can select some or all of them. Click the *Apply* button. When you select several objects from the record index, the search range will narrow. The search results will only contain the objects similar to those on the selected photos and within the preset search threshold.

Record
Case cluster
File

×
↑

Search



Mrs. Smith
Hitmen

Record

Case cluster

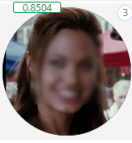
Participant

Acknowledged

5 in total

Most similar first

Case_01
2023-10-16
0.8504

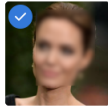
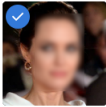


Mr. Smith
Hitmen

Участник 1

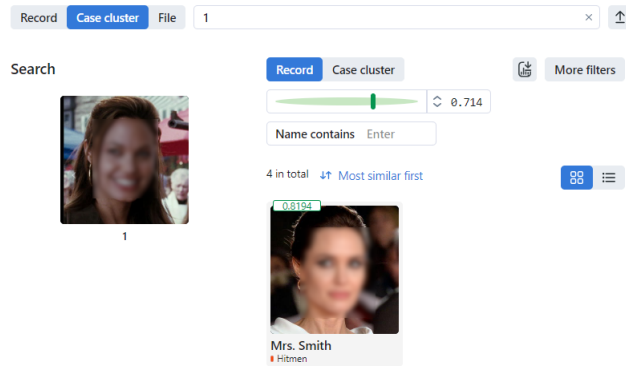
Suspect

✓ Deselect all

Apply

4. If you uploaded a photo file, it will be displayed in the new window. If there are multiple objects in the image, select the one of your interest. Click *Search*.



6. Define the search section – *Record* or *Case cluster*.

7. You will see the search results appear. If necessary, you can narrow down your search by specifying filters or lowering a similarity threshold.

The following filters are available for records:

- *Watch lists*: display records from selected watch lists.
- *Filling*: display only empty records, only filled ones, or any records.
- *Name contains*: filter records by name.
- *ID*: display a record with a given ID.
- *Status*: filter records by status (only active, only inactive, or any status).
- *Match type*: filter records by match type.
- *Limit*: display records within the specified limit.

The following filters are available for a case cluster:

- *Participant*: display only clusters/participants or any.
- *Acknowledged*: display only acknowledged/unacknowledged case clusters or all clusters.
- *Matches*: display case clusters only with/without matching the record or all events.
- *Watch lists*: display only case clusters that contain a record from a selected watch list.
- *Date and time*: display only case clusters that occurred within a certain period.
- *ID*: display a case cluster with a given ID.
- *Photo ID*: display a case cluster with a given photo ID.
- *Video ID*: display a case cluster with a given video ID.

The following specific filters are available for a case cluster:

Note: Some filters from the list below may be hidden, subject to enabled recognition features.

- *Age*: display case clusters with people of a given age.
- *Beard*: filter case clusters by the fact of having a beard.
- *Emotions*: display case clusters with given emotions.
- *Gender*: display case clusters with people of a given gender.
- *Glasses*: filter case clusters by the fact of wearing glasses.

- *Face mask*: filter case clusters by the fact of wearing a face mask.
- *Limit*: display case clusters within the specified limit.

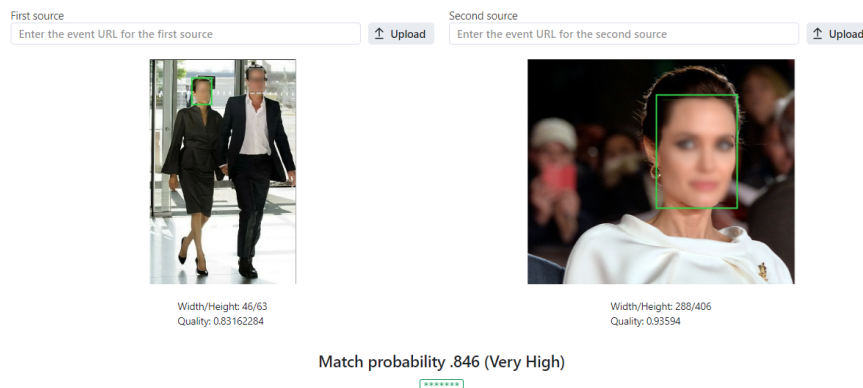
You can sort out results of a search by the following parameters:

- most similar first
- the newest first
- the oldest first

3.6 Compare Two Faces

FindFace CIBR allows you to compare two objects and verify that they match. Do the following:

1. Navigate to the *Verify* tab.
2. Specify two faces to verify in one of the following ways:
 - by event's URL
 - by uploading a photo
3. If there are multiple objects in the image, select the one of your interest.



4. You will see the match probability.

3.7 Reports

FindFace CIBR provides a possibility of building reports on the following system entities:


- *search results*
- *global records*
- *audit log*

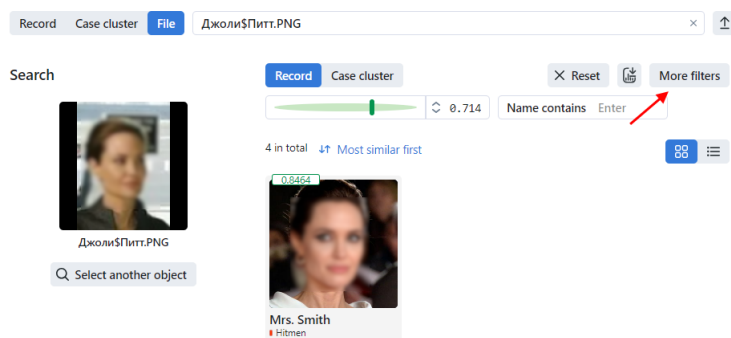
In this section:


- *Build Report*
- *Work with Reports*

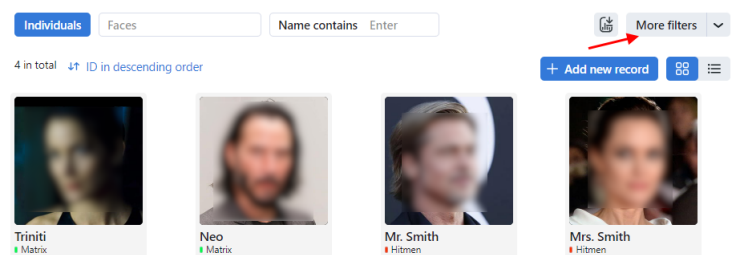
3.7.1 Build Report


To build a report on a system entity, do the following:

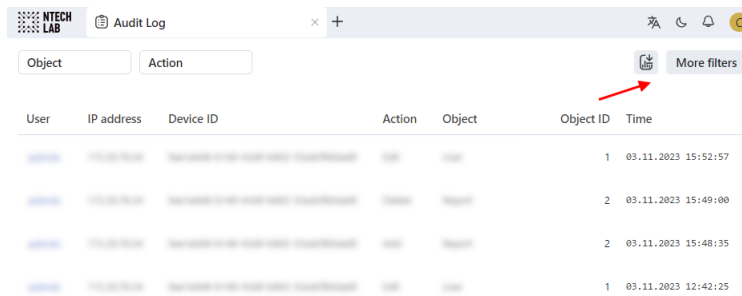
1. Navigate to the tab associated with the required entity: *Search*, *Record Index* or *Audit Log*.
2. Perform the search if you are on the *Search* tab. See *Search Faces in System*. Click the  button to build the report, or the *More filters* button to set filters first.



3. If you are on the *Record index* tab, click the  button to build the report, or the *More filters* button to set filters first.



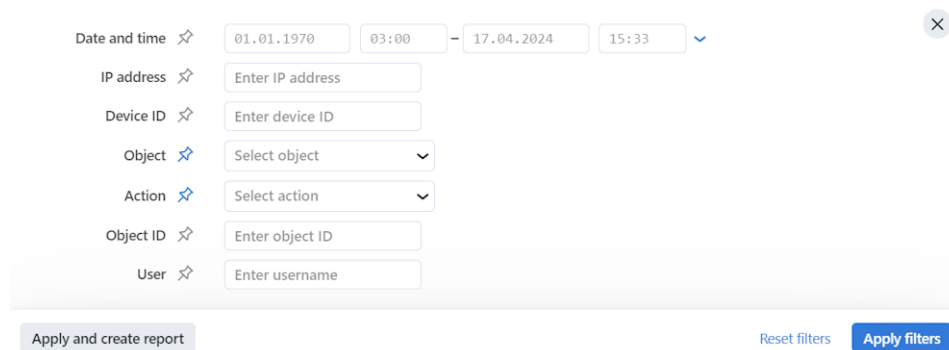
4. If you are on the *Audit Log* tab, click the  button to build the report, or the *More filters* button to set filters first.



User	IP address	Device ID	Action	Object	Object ID	Time
user	172.16.17.1	28c4b8b1-10-4000-0000-000000000000	add	user	1	03.11.2023 15:52:57
user	172.16.17.1	28c4b8b1-10-4000-0000-000000000000	delete	Report	2	03.11.2023 15:49:00
user	172.16.17.1	28c4b8b1-10-4000-0000-000000000000	add	Report	2	03.11.2023 15:48:35
user	172.16.17.1	28c4b8b1-10-4000-0000-000000000000	add	user	1	03.11.2023 12:42:25

5. Set filters for the report. Filters are different for each system entity.

6. Click *Apply and create report*.



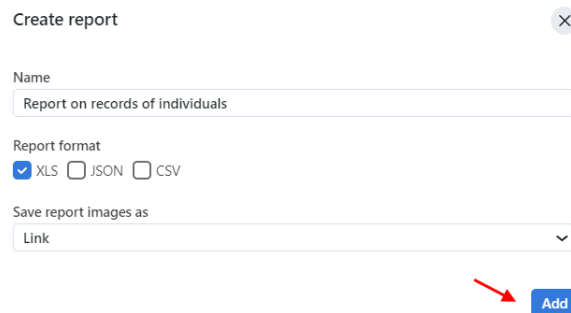
Date and time: 01.01.1970 03:00 - 17.04.2024 15:33
 IP address: Enter IP address
 Device ID: Enter device ID
 Object: Select object
 Action: Select action
 Object ID: Enter object ID
 User: Enter username

Apply and create report Reset filters Apply filters

7. Specify the report name.

8. Check one or several report formats: XLS, JSON, CSV.

9. Choose whether to save the report images as links, thumbnails, or full frames. (Only for search results and global records).



Name: Report on records of individuals
 Report format: ☒ XLS ☐ JSON ☐ CSV
 Save report images as: Link
 Add

10. Click *Add*. The report will be available for download on the *Reports* tab.

3.7.2 Work with Reports

You can access reports previously created in the system on the *Reports* tab. The following operations are available:

- Download selected reports.
- Update selected reports.
- Delete selected reports.

<

See also:

Configure Saving Images in Reports

3.8 Audit Log


The FindFace CIBR comprehensive and searchable audit log is an excellent complementary tool for user management that provides you with a thorough audit of the user actions and strengthens your system protection. You can access this functionality on the *Audit Log* tab.

NTECH LAB		Audit Log					
Object		Action				More filters	
User	IP address	Device ID	Action	Object	Object ID	Time	Image
	192.168.1.1	android-1000000000000000	Search	Record from image	192.168.1.1	29.12.2023 14:29:19	
	192.168.1.1	android-1000000000000000	Search	Video from image		29.12.2023 14:26:53	
	192.168.1.1	android-1000000000000000	Search	Video from image		29.12.2023 14:26:51	
	192.168.1.1	android-1000000000000000	Search	Video from image		29.12.2023 14:26:49	
	192.168.1.1	android-1000000000000000	Search	Video from image		29.12.2023 13:30:21	
	192.168.1.1	android-1000000000000000	Search	Video from image		29.12.2023 13:30:20	
	192.168.1.1	android-1000000000000000	Search	Video from image		27.12.2023 15:26:13	

Each record provides the following data:

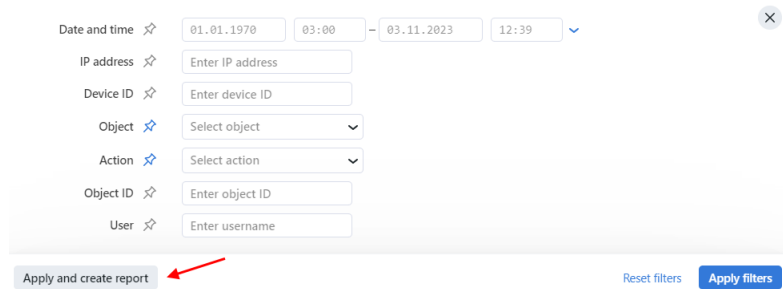
- username of the user who performed the action
- IP address where the request came from
- device ID: the unique identifier of the client device
- action type such as authorization, search, object modification, restart, and so on
- object type to which the action applies, for example, a record or a case
- object identifier

- details, subject to the action type
- timestamp
- image (if applicable).

For actions that involve usage of a photo, a log record also provides information about that photo (including a thumbnail and a full frame of the photo). This applies to such actions as searching for a face by an image, deleting a photo from a record, or adding a photo to a record. When you delete a record with a photo, the deletion of the image attached to a record is also logged. Clicking the  button displays that photo.

Use the filter panel above to set up the search conditions.

You can create a report by clicking the  button or setting filters → *Apply and create report*.



The filter panel includes the following fields:

- Date and time: 01.01.1970 03:00 - 03.11.2023 12:39
- IP address: Enter IP address
- Device ID: Enter device ID
- Object: Select object
- Action: Select action
- Object ID: Enter object ID
- User: Enter username

Buttons at the bottom: Apply and create report (highlighted with a red arrow), Reset filters, and Apply filters.

The report will be available for download on the *Reports* tab.

3.9 Remote Alerting and Remote Search

This section covers an additional but very useful functionality which is a possibility of pulling face recognition events, matching with records on the local server, from remote facial recognition systems. This functionality has a large scope of possible applications. One course is tracking offenders' location and routes and detecting alleged accomplices. Another one is finding missing people. The results are especially great if applied to Public and Transport Safety systems with thousands of cameras.

The remote alerting is disabled by default, so if you haven't configured it yet, click [here](#) for instructions.

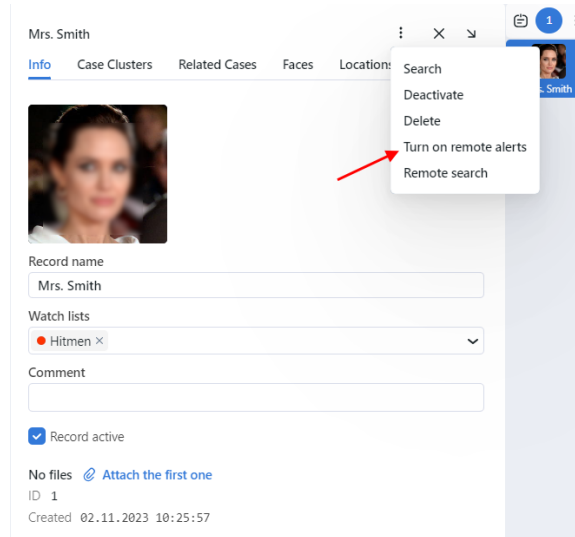
In this section:

- *Turn On/Off Remote Alerts for Individuals*
- *View Remote Alerts*
- *Search Individuals in Remote Systems*
- *Daily Search*

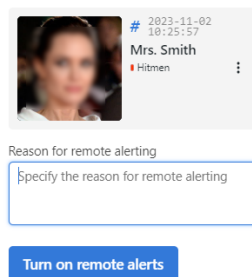
3.9.1 Turn On/Off Remote Alerts for Individuals

To turn on/off remote alerts for a specific individual, do the following:

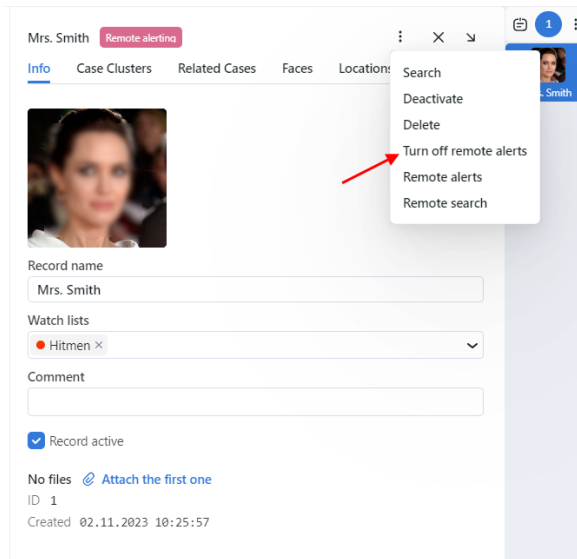
1. Navigate to *Record Index*.
2. Open the individual's record.
3. Click three dots → *Turn on remote alerts* to enable remote alerting.



4. In case you are turning on remote alerts, specify the reason for that. Click *Turn on remote alerts*.



5. Click three dots → *Turn off remote alerts* to disable remote alerting.

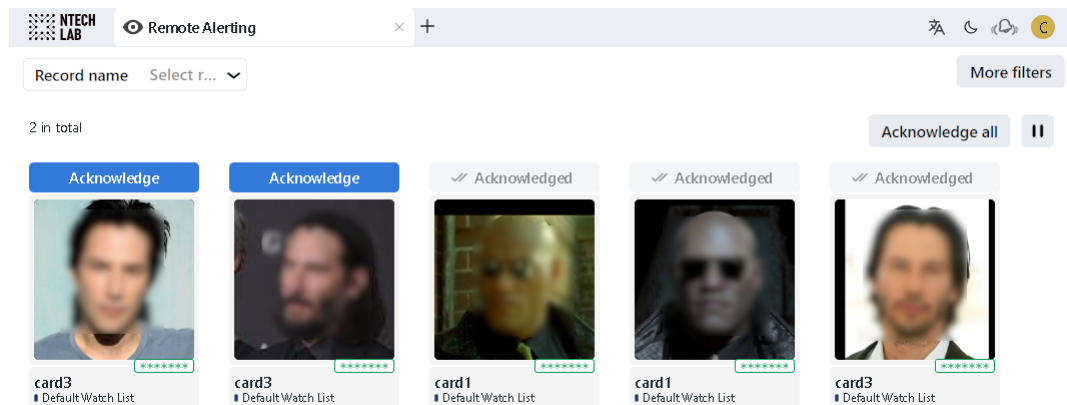


3.9.2 View Remote Alerts

You can view alerts from remote facial recognition systems on the *Remote Alerting* tab on the navigation bar. When a new event occurs an audio alert is triggered, and after the event is acknowledged the alert stops. You also can navigate to remote monitoring events by clicking the bell on a top panel, then *Show events*. You can turn off the alert sound by clicking *Mute*.

Do the following:


1. Navigate to the *Remote Alerting* tab.

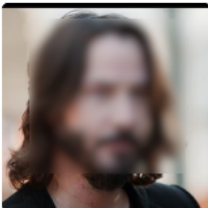


2. Filter the alerts by date and time and a record name by clicking *More filters*, if needed.
3. Click the face to navigate to the sidebar of matched event.

Matched event

✕ ↗




card3

Watch lists _____

Default Watch List

Cameras _____

Camera ID: 4568659578746293807 - Camera 1

Source _____

first_puppet

Info _____

ID 4568659578746293807

Created 07.12.2023 20:12:00

Reason _____

Найти и обезвредить

Acknowledgment _____

Acknowledge

4. Click the text bar of the thumbnail to navigate to the face record.
5. Click *Acknowledge* to acknowledge the event.

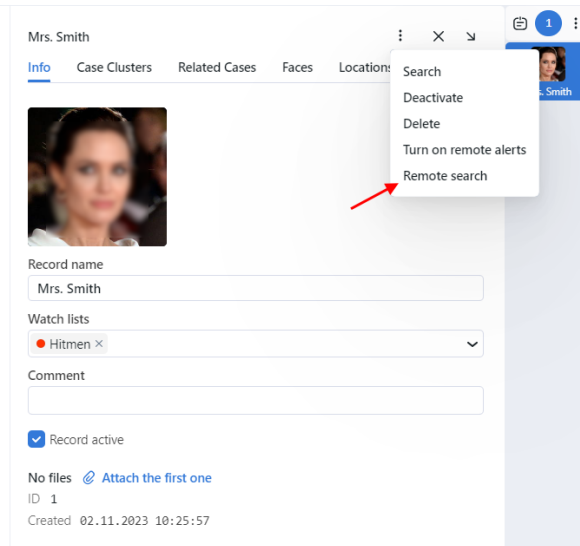
Note: Even if one event on the puppets' server matches multiple photos associated with the same record, there will be no more than one event in the remote monitoring list – the one with a higher match probability (and the corresponding photo from the record will be displayed in the list of monitoring results).

If the same event on the puppets' server matches multiple photos, but these photos are associated with different records, there will be as many events in the remote monitoring list as there are matches with the records.

3.9.3 Search Individuals in Remote Systems

To search an individual in remote systems, do the following:

1. Navigate to *Record Index*.
2. Open the individual's record.
3. Click three dots → *Remote search*.



4. Specify the search conditions, such as search reason, the confidence threshold, date and time of the individual's appearance, and the maximum number of search results. Click *Done*.

Search reason

Enter a search reason

Confidence threshold

0.65

Interval

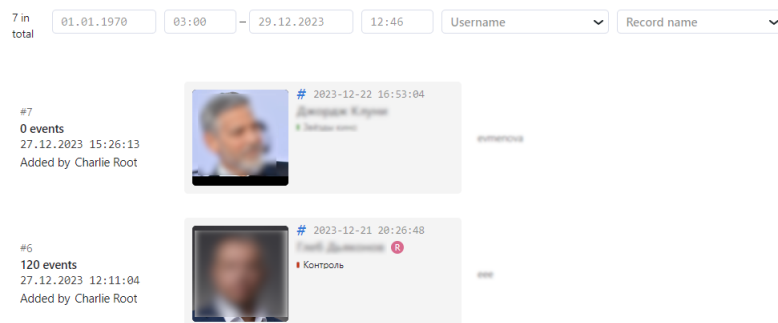
01.11.2023 00:00 - 04.11.2023 23:59

Results limit

120

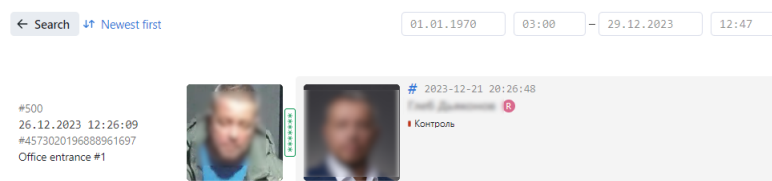
Done

5. The search results will be shown on the *Remote Search* tab.



You can filter search results by date and time, a record name and a user who initiated the search.

6. If you click the left part with information of the event, you will navigate to the puppet events.



The following information is available on the results of searching:

- puppet event date
- puppet camera name
- puppet name
- event's ID
- thumbnail of face event
- record that was turned on remote alerts
- the match probability

You can filter search events by the level of novelty.

3.9.4 Daily Search

Daily search allows the puppeteer to receive scheduled events from the puppet with the matching of the records.

To use this functionality, see [integration with remote facial recognition systems](#) and do the following:

1. Enable daily search in the `/opt/findface-cibr/configs/findface-multi-legacy/findface-multi-legacy.py` configuration file of the puppeteer.
2. Enable 'DAILY' parameters and specify the time for matching the events with the records that were received from the puppeteer in the configuration file of the puppet.
3. Navigate to the *Watch list* tab and check `Collect location data` for the watch list of your interest. For all records included in this list, all connected puppets return location data of objects of interest for the last 24 hours daily according to a set schedule, and if the flag is unchecked the puppets stop daily monitoring of the records in this list.

4. The results will be shown on the *Location Data* tab of the individual's record and are sorted in descending order of novelty by default.

The screenshot displays the FindFace application interface. On the left, a search bar contains 'Faces' and a 'Name contains' field with an 'Enter' button. Below the search bar, it shows '2 in total' and a sort option 'ID in descending order'. There are two cards, 'card2' and 'card1', each showing a face image and a 'Default Watch List' button. On the right, a detailed view for 'card2' is shown, featuring tabs for 'Case Clusters', 'Related Cases', 'Related Records', and 'Location Data'. The 'Location Data' tab is active, displaying a timeline with a date range from '01.01.1970' to '07.12.2023' and a time range from '06:00' to '19:40'. A specific entry is highlighted with the date '04.12.2023 22:07:27' and the label 'first_puppet', accompanied by two face images.

INTEGRATIONS

This chapter is all about integration with FindFace. In the current version of FindFace, you can only integrate your system via HTTP API.

4.1 HTTP API

Detailed interactive documentation on the FindFace HTTP API is available after installation at <http://<findface-ip:port>/api-docs>. Learn and try it out.

Tip: You can also find it by navigating to *Settings -> API Documentation* in the web interface.
