FindFace Security

Release 2.0

NtechLab

Jun 16, 2023

Contents

FindFace Security is a video-based biometric identification system that automates Security and Hospitality Operations Management. Based on a cutting-edge AI face recognition technology, it can be harnessed in such areas as retail, banking, social networking, entertainment, sports, event management, dating services, video surveillance, public safety, homeland security, etc.

FindFace Security detects and identifies faces of the unwanted persons and VIP guests in video, and notifies security and hospitality managers about their arrival. It can also recognize such face features as gender, age, emotions, glasses, and beard, and display this information in a face recognition event.

Early recognition of the arrival of unwanted persons and VIP guests allows for solving the following problems:

- Operational losses due to fraudulent activity
- Reputational losses and conflicts
- Better catering to the needs of VIP guests
- Prevention of life threatening situations

FindFace Security supports the integration of third-party solutions via *HTTP API* and *webhooks*, so you can enhance your current system or application with face recognition functionality.

This guide is intended for the FindFace Security administrators, security and hospitality managers, maintenance engineers, as well as for the system integration engineers who are going to integrate face recognition services into their systems.

CHAPTER 1

Administrator's Guide

1.1 Architecture

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

In this chapter:

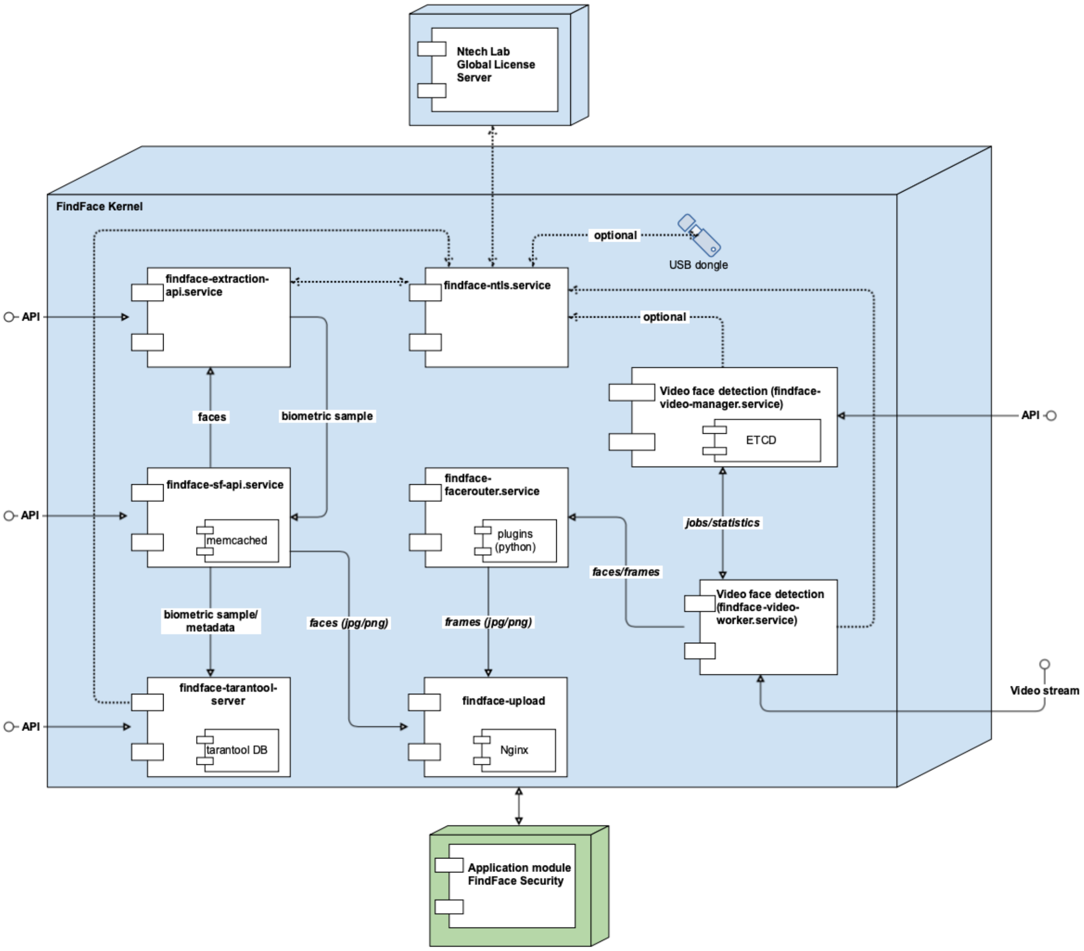
1.1.1 Architectural Elements

FindFace Security consists of the following basic architectural elements:

- FindFace core,
- FindFace Security application module.

FindFace Core

The FindFace core includes the following components:


Ntech Lab
Global License
Server
FindFace Kernel
optional
USB dongle
findface-extraction-api.service
findface-ntls.service
API
optional
faces
biometric sample
Video face detection (findface-video-manager.service)
ETCD
API
findface-sf-api.service
memcached
API
findface-facerouter.service
plugins (python)
jobs/statistics
faces/frames
biometric sample/ metadata
faces (jpg/png)
frames (jpg/png)
Video face detection (findface-video-worker.service)
findface-tarantool-server
tarantool DB
findface-upload
Nginx
API
Video stream
Application module
FindFace Security

Component	Description	Vendor
findface-extraction-api	A service which uses neural networks to detect a face in an image and extract a face biometric sample (feature vector). CPU- or GPU-acceleration.	Ntech Lab own deployment
findface-sf-api	A service that implements HTTP API for face detection and face recognition.	
findface-tarantool-server	A service that provides interaction between the `findface-sf-api` service and the biometric database (database that stores face biometric samples) powered by Tarantool.	
findface-upload	An NginX-based web server used as a storage for original images, thumbnails and normalized face images.	
findface-facerouter	A service used to define processing directives for detected faces. In FindFace Security, `findface-facerouter` functions are performed by `findface-security` (see *FindFace Security Application Module*).	
findface-video-manager	A 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 streams.	
findface-video-worker	A service, part of the video face detection module, which recognizes a face in video and posts its normalized image, full frame and metadata (such as the camera ID and detection time) to the `findface-facerouter` service for further processing according to given directives. CPU- or GPU-acceleration.	
findface-ntls	A license server which interfaces with the NtechLab Global License Server or a USB dongle to verify the license of your FindFace Security instance.	
Tarantool	Third-party software which implements the biometric database that stores extracted biometric samples (feature vectors). The system data, dossiers, face identification events, user accounts and camera settings are stored in PostgreSQL (part of the FindFace Security application module).	Tarantool
etcd	Third-party software that implements a distributed key-value store for `findface-video-manager`. Used as a coordination service in the distributed system, providing the video face detector with fault tolerance.	etcd
NginX	Third-party software which implements the system web interfaces.	nginx
memcached	Third-party software which implements a distributed memory caching system. Used by `findface-extraction-api` as a temporary storage for extracted face biometric samples before they are written to the biometric database powered by Tarantool.	memcached

FindFace Security Application Module

The FindFace Security application module includes the following components:

Component	Description	Vendor
ffsecurity	A component which serves as a gateway to the FindFace core. Provides interaction between the FindFace Core and the web interface, and the system functioning as a whole. The `ffsecurity` component includes 4 services: `findface-security-proto` (provides HTTP and web socket), `findface-security-worker` (provides interaction with the other system components), `findface-security-monitoring-updater` (updates the monitoring gallery in the biometric database), and `findface-security-webhook-updater` (pulls webhooks).	NtechLab own deployment
ffsecurity-ui	A main web interface that is used to interact with FindFace Security. Allows you to work with face identification events, search for faces, manage cameras, users, dossiers, and watch lists.	
PostgreSQL	Third party software which implements the main system database that stores detailed and categorized dossiers on particular persons, face identification events, as well as data for internal use such as user accounts and camera settings. The face biometric data are stored in Tarantool (part of the FindFace core).	PostgreSQL
Redis	Third-party software which implements a message broker between the `findface-security-proto` and `findface-security-worker` services (parts of `ffsecurity`).	Redis
Django	Third-party software which implements a web framework for the FindFace Security web interface.	Django

See also:

Components in Depth

1.1.2 Single- and Multi-Host Deployment

You can deploy FindFace Security on a single host or in a cluster environment. If you opt for the latter, we offer you one of the following deployment schemes:

- Deploy FindFace Security standalone and distribute additional `findface-video-worker` components across multiple hosts.
- Distribute the FindFace Security components across multiple hosts. If necessary, set up load balancing.

1.1.3 CPU- and GPU-acceleration

The `findface-extraction-api` and `findface-video-worker` services can be either CPU- or GPU-based. During installation from the developer-friendly *installer*, you will have an opportunity to choose the acceleration type you need.

If you opt to install FindFace Security from the *repository package*, deploy the `findface-extraction-api` and `findface-video-worker` packages on a CPU-based server, and the `findface-extraction-api-gpu` and/or `findface-video-worker-gpu` packages on a GPU-based server.

Important: Refer to *System Requirements* when choosing hardware configuration.

Important: If the resolution of a camera(s) in use is more than 1280x720p, it is strongly recommended to use the GPU-accelerated `findface-video-worker-gpu` package.

1.2 System Requirements

To calculate the FindFace Security host(s) characteristics, use the requirements provided in this chapter.

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

In this chapter:

1.2.1 Basic Configuration

Important: If the resolution of a camera(s) in use is more than 1280x720p, it is strongly recommended to use the GPU-accelerated `findface-video-worker-gpu` package.

	Minimum	Recommended
CPU	Intel Core i5 CPU with 4 physical cores 2.8 GHz	Intel Xeon E5v3 with 6 physical cores, or higher or similar CPU
	The own needs of FindFace Security require 2 cores HT > 2.5 GHz. The characteristics also depend on the number of cameras in use. A single camera 720p@25FPS requires 2 cores >2.5 GHz. AVX support	
GPU (optional)	Nvidia Geforce® GTX 1050 Ti with 4Gb RAM (only for running `findface-extraction-api` gpu-package with `elderberry_160`, not enough for `findface-video-worker-gpu`).	Nvidia Geforce® GTX 1080+ with 8+Gb RAM
	Supported series: GeForce (from Pascal architecture), Tesla (from Pascal architecture and Volta v100)	
RAM	8 Gb	16+ Gb
	The own needs of FindFace Security require 8 Gb. The RAM consumption also depends on the number of cameras in use. A single camera 720p@25FPS requires 2 GB RAM	
HDD	16 Gb	16+ Gb
	The own needs of the operating system and FindFace Security require 15 GB. The total volume is 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 16.04 x64 only	

Tip: For more accurate hardware selection, consult the FindFace Security resource consumption and performance *benchmark results*.

1.2.2 Benchmark Results

Here you can see the FindFace Security resource consumption and performance benchmark results. Use these data to select your hardware configuration.

Note: RAM usage and performance may slightly vary from test to test.

Warning: Strictly not recommended to use `face/elderberry_160` for work.

Testing Setup

Package versions:

- findface-extraction-api-cpu 2.6.999.1910+261.gebb8df6
- findface-extraction-api-gpu
- findface-video-worker-cpu 2.6.999.1910+261.gebb8df6
- findface-video-worker-gpu
- findface-tarantool-server 2.6.999.1910+261.gebb8df6

Hardware:

- Processor: Intel Core i5-8400 @ 3.60GHz (6 Cores)
- Motherboard: ASUS PRIME H370M-PLUS
- Chipset: Intel Cannon Lake PCH Shared SRAM
- Memory: 2 x 8192 MB DDR4-2400MHz
- Disk: 2000GB Seagate ST2000NM0008-2F3
- Graphics: Gigabyte NVIDIA GeForce GTX 1060 6GB

Software:

- OS: Ubuntu 16.04, Kernel: 4.15.0-29-generic (x86_64), Desktop: Unity 7.4.5
- Display Server: X Server 1.19.6
- Display Driver: NVIDIA 410.48, OpenGL: 4.5.0
- Screen Resolution: 1920x1200

RAM consumption is calculated by:

- CPU: htop;
- GPU: nvidia-smi

CPU performance:

```
sysbench 0.4.12:  multi-threaded system evaluation benchmark

Running the test with following options:
Number of threads: 1

Doing CPU performance benchmark

Threads started!
Done.

Maximum prime number checked in CPU test: 10000


Test execution summary:
    total time:                          9.1128s
    total number of events:              10000
    total time taken by event execution: 9.1112
    per-request statistics:
         min:                                  0.82ms
         avg:                                  0.91ms
         max:                                  1.47ms
         approx.  95 percentile:               1.02ms

Threads fairness:
    events (avg/stddev):           10000.0000/0.00
    execution time (avg/stddev):   9.1112/0.00
```

GPU performance:

```
Unigine Heaven 4.0:
    pts/unigine-heaven-1.6.4 [Resolution: 1920 x 1080 - Mode: Windowed - Renderer:␣
↪OpenGL]
    Test 1 of 2
    Estimated Trial Run Count:    3
    Estimated Test Run-Time:      15 Minutes
    Estimated Time To Completion: 29 Minutes
        Started Run 1 @ 17:54:37
        Started Run 2 @ 17:59:15
        Started Run 3 @ 18:03:52  [Std. Dev: 0.29%]

    Test Results:
        86.6473
        86.1475
        86.4553

    Average: 86.42 Frames Per Second


Unigine Heaven 4.0:
    pts/unigine-heaven-1.6.4 [Resolution: 1920 x 1080 - Mode: Fullscreen - Renderer:␣
↪OpenGL]
    Test 2 of 2
    Estimated Trial Run Count:    3
    Estimated Time To Completion: 15 Minutes
        Started Run 1 @ 18:08:33
        Started Run 2 @ 18:13:09
        Started Run 3 @ 18:17:45  [Std. Dev: 1.37%]

    Test Results:
        87.7017
        89.5186
        90.023

    Average: 89.08 Frames Per Second
```

Resource Consumption: `findface-extraction-api` and `findface-extraction-api-gpu`

RAM usage: `findface-extraction-api`

Model	# instances	RAM, MB	# instances	RAM, MB	# instances	RAM, MB
face/elderberry_576-cpu	1	3730	2	7450	3	11000
face/elderberry_160-cpu		1590		2800		4050
face/elderberry_576-cpu + features (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)		5568		10800		•
face/elderberry_160-cpu + features (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)		3473		6250		9400
Features only (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)		2270		3900		5800

RAM usage: `findface-extraction-api-gpu`

Note: `findface-extraction-api-gpu` allows only 1 model instance.

Model	RAM, MB
face/elderberry_576-gpu	~2200 (up to 4.5 Gb on initialization)
face/elderberry_160-gpu	~850 (up to 1.8 Gb on initialization)
face/elderberry_576-gpu + features (faceattr/age.v1-gpu, faceattr/beard.v0-gpu, faceattr/emotions.v1-gpu, faceattr/gender.v2-gpu, faceattr/glasses3.v0-gpu)	~3100 (up to 6.3 Gb on initialization)
face/elderberry_160-gpu + features (faceattr/age.v1-gpu, faceattr/beard.v0-gpu, faceattr/emotions.v1-gpu, faceattr/gender.v2-gpu, faceattr/glasses3.v0-gpu)	~1871 (up to 4 Gb on initialization)
Features only (faceattr/age.v1-gpu, faceattr/beard.v0-gpu, faceattr/emotions.v1-gpu, faceattr/gender.v2-gpu, faceattr/glasses3.v0-gpu)	1200

Performance: `findface-extraction-api` and `findface-extraction-api-gpu`

Speed: `findface-extraction-api`

Model	Time, ms (i5-8400)
face/elderberry_576-cpu	620
face/elderberry_160-cpu	350
face/elderberry_576-cpu + features (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)	655
face/elderberry_160-cpu + features (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)	380
Features only (faceattr/age.v1-cpu, faceattr/beard.v0-cpu, faceattr/emotions.v1-cpu, faceattr/gender.v2-cpu, faceattr/glasses3.v0-cpu)	300

Speed: `findface-extraction-api-gpu`

Model	Time, ms (1060TI)
face/elderberry_576-gpu	240
face/elderberry_160-gpu	225
face/elderberry_576-gpu + features (faceattr/age.v1-gpu, faceattr/beard.v0-gpu, faceattr/emotions.v1-gpu, faceattr/gender.v2-gpu, faceattr/glasses3.v0-gpu)	260
face/elderberry_160-gpu + features (faceattr/age.v1-gpu, faceattr/beard.v0-gpu, faceattr/emotions.v1-gpu, faceattr/gender.v2-gpu, faceattr/glasses3.v0-gpu)	235

Performance: `findface-video-worker` and `findface-video-worker-gpu`

CPU/RAM consumption and speed: `findface-video-worker`

Stream	RAM, MB	CPU utilization,% (i5-8400 6 cores)	Processing speed, FPS* (i5-8400)
1x 720p25FPS	370	230	62
2x 720p25FPS	755	500	56
3x 720p25FPS	1040	580	45
4x 720p25FPS	1437	600	36
5x 720p25FPS	1900	600	24
8x 720p25FPS	2650	600	18
1x 1080p25FPS	502	250	41
2x 1080p25FPS	1023	508	37
3x 1080p25FPS	1529	590	30
4x 1080p25FPS	2031	594	23
1x 720p25FPS + 1x 1080p25FPS	890	453	38
2x 720p25FPS + 2x 1080p25FPS	1750	590	21

Important: If video processing speed is less than the number of FPS in video, it means that the system is running low on resources and the lack of resources causes the video face detector to drop frames. Avoid this situation as it can lead to missing out on faces, instability and potential failures.

To check your resource consumption, execute:

```
sudo journalctl -f -a -u findface-video-worker | grep dropped
```

The following lines indicate that the system has less resources than necessary:

```
findface-video-worker[28882]: [2] 2 frames dropped!
findface-video-worker[28882]: [1] 6 frames dropped!
```

In this case, consider to change component settings or hardware configuration.

GPU RAM consumption and speed: `findface-video-worker-gpu`

Stream	GPU RAM, MB	Processing speed, FPS* (1060TI)
Without streams	600	•
1x 720p25FPS	656	254
2x 720p25FPS	738	126
4x 720p25FPS	858	63
8x 720p25FPS	1117	30
1x 1080p25FPS	735	202
2x 1080p25FPS	935	96
4x 1080p25FPS	1185	48
8x 1080p25FPS	2650	48
1x 720p25FPS + 1x 1080p25FPS	803	453
2x 720p25FPS + 2x 1080p25FPS	1100	54
4x 720p25FPS + 4x 1080p25FPS	1500	26
8x 720p25FPS + 8x 1080p25FPS	2300	11

Important: If video processing speed is less than the number of FPS in video, it means that the system is running low on resources and the lack of resources causes the video card to accumulate frames in its memory. Avoid this situation as it can lead to instability and potential failures.

To view the current processing speed, execute the following command on the `findface-video-manager` host console:

```
curl -s http://127.0.0.1:18810/jobs | jq -r '.[]|("id="+(.id|tostring)+" url="+.
↪stream_url+" FPS="+(.statistic.processing_fps|tostring))'
```

In the response, you will find each video stream processing speed. For example, enough amount of resources when processing 7 video streams with characteristics **h264 (High) ([27][0][0][0] / 0x001B), yuvj420p(pc, bt709), 1920x1080, 25 fps, 25 tbr, 90k tbn, 180k tbc** will result in the following response:

```
id=10 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.189745
id=5 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.189854
id=6 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.589714
id=3 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.189857
id=4 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.189857
id=7 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.389784
id=9 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=27.189857
```

Lack of resources when processing 8 video streams with the same characteristics will give FPS (processing speed) less than the video's 25 fps:

```
id=8 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.772333
id=10 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.772415
id=5 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.772339
id=6 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.372803
id=3 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.772339
id=4 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=23.772339
id=7 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.775822
id=9 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=22.573729
```

Even smaller values will be registered when processing 10 video streams with the same characteristics:

```
id=7 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380642
id=9 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380642
id=2 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380646
id=8 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380642
id=10 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=9.984919e-05
id=5 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380642
id=6 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380642
id=1 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.380651
id=3 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=20.180836
id=4 url=http://restreamer.int.ntl/hls/openspace.m3u8 FPS=19.581406
```

Important: If `findface-video-worker-gpu` processes video streams of equal FPS, the number of processed streams doesn't severely affect the GPU memory consumption, as all the streams are processed by the same worker. On the other hand, if `findface-video-worker-gpu` processes video streams of different FPS, it severely increases the memory consumption as different streams have to be processed by different workers.

1.2.3 Examples of Hardware Configuration

Important: The exemplary hardware configurations in this section are only for reference. Do not use these data to select your production instance configuration. To select the optimal configuration, ask advice from our experts by support@ntechlab.com.

Resource consumption may vary depending on the following factors:

- The number of HTTP requests per second, sent to `findface-extraction-api` (depends on the number of faces in a camera field of view, the number of user search requests, etc.).
- Video quality (video interference, colourful video background take up more resources).
- Motion intensity in video.

The following examples are given for standard component configuration.

Important: Changes in component settings may result in significant changes in resource consumption.

CPU-based Server

Cameras	CPU	RAM, GB	Extraction
1x720p25FPS	Intel Core i5 - 6400 (4 cores 2700MHz)	8	elderberry_160 + features* model_instances = 1 or elderberry_576 model_instances = 1
2x720p25FPS	min: Intel Core i7 - 6700 (4 core 3400MHz); recommended: Intel Core i7 - 6850K (6 cores 3600MHz)	12	elderberry_160 + features* model_instances = 2 or elderberry_576 + features* model_instances = 2
4x720p25FPS	min: Intel Core i7 - 8700K (6 cores 3700MHz); recommended: Intel Core i9 - 9900K (8 cores 3600MHz)	16	elderberry_576 + features* model_instances = 2 or elderberry_576 model_instances = 3
1x1080p25FPS	min: Intel Core i7 - 6700 (4 cores 3400MHz); recommended: Intel Core i7 - 6850K (6 core 3600MHz)	32	elderberry_576 + features* model_instances = 1 or elderberry_576 model_instances = 2

GPU-based Server

Cameras	CPU	RAM, GB	GPU	Installation	Extraction	Video
1x720p25FPS	Intel Core i5 - 6400 (4 cores 2700MHz)	8	nVidia GeForce GTX1060 6Gb	extraction-api on CPU video-worker on GPU	elderberry_160 + features* model_instances = 1 or elderberry_576-cpu model_instances = 1	basic
				extraction-api on GPU video-worker on CPU	basic	basic
2x720p25FPS	Intel Core i5 - 6400 (4 cores 2700MHz)	12	nVidia GeForce GTX1060 6Gb	extraction-api on CPU video-worker on GPU	elderberry_160 + features* model_instances = 2 or elderberry_576-cpu + features model_instances = 1 or elderberry_576-cpu model_instances = 2	basic
				extraction-api on GPU video-worker on CPU	basic	basic
4x720p25FPS	Intel Core i5 - 8400 (4 cores 2800MHz)	16	nVidia GeForce GTX1060 6Gb	extraction-api on CPU video-worker on GPU	elderberry_576-cpu + features* model_instances = 2	basic
8x720p25FPS	Intel Core i5 - 8400 (4 cores 2800MHz) Intel Core i7 - 6700 (4 cores 3400MHz)	16	nVidia GeForce GTX1060 TI 6Gb	extraction-api on CPU video-worker on GPU	elderberry_576-cpu + features* model_instances = 2	basic
16x720p25FPS	Intel Core i7 - 6700 (4 cores 3400MHz) Intel Core i7 - 8700K (6 cores 3700MHz) Intel Core i9 - 9900K (8 cores 3600MHz)	32	2x nVidia GeForce GTX1060 TI 6Gb	extraction-api on CPU video-worker on GPU	elderberry_576-cpu + features* model_instances = 4 or	basic

1.3 Deploy FindFace Security

For your convenience, we offer you several deployment options:

- Deploy from a console installer

- Deploy step-by-step from an APT repository.

1.3.1 Deploy from Console Installer

To deploy FindFace Security, use a developer-friendly console installer.

Tip: Before deployment, be sure to consult the *system requirements*.

Important: The FindFace Security 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.

To deploy FindFace Security from the console installer, do the following:

1. Download the installer file `findface-security-2.0.0-server-3.0.0.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.

   ```
   chmod +x findface-security-2.0.0-server-3.0.0.run
   ```

4. Execute the `.run` file.

   ```
   sudo ./findface-security-2.0.0-server-3.0.0.run
   ```

   The installer will ask you a few questions and perform several automated checks to ensure that the host meets the system requirements. Fill out the prompts appropriately once requested. The questions are the following:

   1. Product to install: FindFace Security.
   2. Installation type:
      - `1`: install FindFace Security standalone.
      - `2`: install FindFace Security and configure it to interact with additional remote `findface-video-worker` instances.

> **Tip:** To install only `findface-video-worker` on a host, refer to *Additional findface-video-worker deployment on remote hosts*.

- `3`: install only the apt repository that can be further used for the *step-by-step deployment*.
- `4`: fully customized installation.

3. Type of `findface-video-worker` package: CPU or GPU.
4. Type of `findface-extraction-api` package: CPU or GPU.

Once all the questions answered, the answers will be saved to a file `/tmp/<findface-installer-*>.json`. You can edit this file and use it to install FindFace Security on other hosts without having to answer the questions again.

After that, the FindFace Security components will be automatically installed, configured and/or started in the following configuration:

Service	Configuration
postgresql-9.5	Installed and started.
redis-server	Installed and started.
etcd	Installed and started.
memcached	Installed and started.
nginx	Installed and started.
django	Installed and started as a web framework for the FindFace Security web interface.
findface-ntls	Installed and started.
findface-tarantool-server	Installed and started. The number of instances (shards) is calculated using the formula: `N = max(min(mem_mb // 2000, cpu_cores), 1)`, i.e. it is equal to the RAM size in MB divided by 2000, or the number of CPU physical cores (but at least 1 shard).
findface-extraction-api	Installed and started.
findface-sf-api	Installed and started.
findface-upload	Installed.
findface-video-manager	Installed and started.
findface-video-worker(-gpu)	Installed and started.
findface-data-*	Neural network models for face and face features recognition (gender, age, emotions, glasses, beard). Installed.
findface-gpudetector-data/	NTechLab gpudetector data. Installed.
python3-ntech.ffsecurity-client	NtechLab FindFace Security API python client library. Installed.
findface-security-proto	Installed and started.
findface-security-worker	Installed and started, 4 instances.
findface-security-monitoring-updater	Installed and started.
findface-security-webhook-updater	Installed and started.
jq	Installed. Used to pretty-print API responses from FindFace Security.

Once the installation is complete, the following output will be shown on the console:

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

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

5. Upload the FindFace Security license file via the main web interface `http://<Host_IP_address>/#/license`. To access the web interface, use the provided `admin` credentials.

Note: The host IP address is shown in the links to FindFace web services in the following way: as an external IP address if the host belongs to a network, or `127.0.0.1` otherwise.

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

6. To automatically install FindFace Security on another host without answering the installation questions, use the `/tmp/<findface-installer-*>.json` file. Execute:

```
sudo ./findface-security-2.0.0-server-3.0.0.run -f /tmp/<findface-installer-*>.
↪json
```

Tip: You can find an example of the installation file in *Installation File*.

1.3.2 Deploy Step-by-Step from Repository

This section will guide you through the FindFace Security step-by-step deployment process. Follow the instructions below minding the sequence.

In this section:

- *Deploy FindFace Security Application Module and Biometric Database*

Get Distributable Packages

If you opt to deploy FindFace Security from an APT repository, you will be provided with the following packages:

- `<ffsecurity-repo*.deb>`: a deb-package that installs a local repository with components.
- `<findface-data*.deb>`: a deb-package(s) that installs neural network models for face detection and face features recognition.

You receive the packages from your Ntech Lab manager.

Note: You can deploy step-by-step by using the FindFace Security console *installer*.

Prepare Packages for Installation

To prepare the distributable packages for installation, do the following:

1. Unpack the package with components on each designated host.

```
sudo dpkg -i <ffsecurity-repo>.deb
```

2. Add a signature key on each designated host.

```
sudo apt-key add /var/ffsecurity-repo/public.key
sudo apt update
```

3. Unpack the packages with *models* (face, gender, age, emotions, etc.). In the cluster environment, models are installed only on the `findface-extraction-api` hosts.

```
sudo dpkg -i findface-data*
```

Prerequisites

FindFace Security requires such third-party software as PostgreSQL, Redis, etcd, and memcached. Do the following:

1. Install the prerequisite packages as such:

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

2. Open the `memcached` configuration file. Set the maximum memory to use for items in megabytes: `-m 512`. Set the max item size: `-I 16m`. If one or both of these parameters are absent, simply add them in the file.

```
sudo vi /etc/memcached.conf

-m 512
-I 16m
```

3. Enable the prerequisite services autostart and launch the services:

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

Provide Licensing

You receive a license file from your NTechLab manager. If you opt for the on-premise licensing, we will also send you a USB dongle.

The FindFace Security licensing is provided as follows:

1. Deploy `findface-ntls`, license server in the FindFace core.

   Important: There must be only one `findface-ntls` instance in each `FindFace Security` installation.

   Tip: In the `findface-ntls` configuration file, you can change the license folder and specify your proxy server IP address if necessary. You can also change the `findface-ntls` web interface remote access settings. See *findface-ntls* for details.

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

2. Upload the license file via the `findface-ntls` web interface in one of the following ways:

   - Navigate to the `findface-ntls` web interface `http://<NTLS_IP_address>:3185/#/`. Upload the license file.

     Tip: Later on, use the FindFace Security main web interface to consult your license information, and upgrade or extend your license (*Settings* -> *License*).

   - Directly put the license file into the license folder (by default, `/ntech/license`, can be changed in the `/etc/findface-ntls.cfg` configuration file).

3. For the on-premise licensing, insert the USB dongle into a USB port.

4. If the licensable components are installed on remote hosts, specify the IP address of the `findface-ntls` host in their configuration files. See *findface-extraction-api*, *findface-tarantool-server*, *Video face detection: findface-video-manager and findface-video-worker* for details.

See also:

View and Update License

Deploy Main Database

In FindFace Security, the main system database is based on PostgreSQL. To deploy the main database, do the following:

1. Using the **`PostgreSQL`** console, create a new user `ntech` and a database `ffsecurity` in PostgreSQL.

```
sudo -u postgres psql

postgres=# CREATE ROLE ntech WITH LOGIN;

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

Tip: To quit from the **PostgreSQL** console, type `\q` and press `Enter`.

2. Allow authentication in **PostgreSQL** by UID of a socket client. Restart **PostgreSQL**.

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

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

Deploy FindFace Core

To deploy the FindFace core, do the following:

Tip: You can find the description of the FindFace core components and their configuration parameters in *Architecture* and *Components in Depth*.

1. Install the FindFace core components:

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

Note: To install the GPU-accelerated `findface-extraction-api` component, use `findface-extraction-api-gpu` instead of `findface-extraction-api` in the command.

Note: To install the GPU-accelerated `findface-video-worker` component, use `findface-video-worker-gpu` instead of `findface-video-worker` in the command.

2. Open the `findface-extraction-api` configuration file (CPU or GPU service). Enable the `quality_estimator` to be able to estimate the face quality in a dossier.

Note: The *minimum face quality* in a dossier photo is set as `MINIMUM_DOSSIER_QUALITY` in `/etc/ffsecurity/config.py`.

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

quality_estimator: true
```

3. In the `findface-extraction-api` configuration file, enable recognition models for face features such as gender, age, emotions, glasses3, and/or beard, subject to your needs. Be sure to choose 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.

```
models:
  age: faceattr/age.v1.cpu.fnk
  emotions: faceattr/emotions.v1.cpu.fnk
  face: face/elderberry_576.cpu.fnk
  gender: faceattr/gender.v2.cpu.fnk
  beard: faceattr/beard.v0.cpu.fnk
```

The following models are available:

Face feature	Acceleration	Configuration file parameter
face (biometry)	CPU	`face: face/elderberry_576.cpu.fnk`
	GPU	`face: face/elderberry_576.gpu.fnk`
age	CPU	`age: faceattr/age.v1.cpu.fnk`
	GPU	`age: faceattr/age.v1.gpu.fnk`
gender	CPU	`gender: faceattr/gender.v2.cpu.fnk`
	GPU	`gender: faceattr/gender.v2.gpu.fnk`
emotions	CPU	`emotions: faceattr/emotions.v1.cpu.fnk`
	GPU	`emotions: faceattr/emotions.v1.gpu.fnk`
glasses3	CPU	`glasses3: faceattr/glasses3.v0.cpu.fnk`
	GPU	`glasses3: faceattr/glasses3.v0.gpu.fnk`
beard	CPU	`beard: faceattr/beard.v0.cpu.fnk`
	GPU	`beard: faceattr/beard.v0.gpu.fnk`

Tip: To disable a recognition model, simply pass an empty value to a relevant parameter. Do not remove the parameter itself as in this case the system will be searching for the default model.

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

4. Open the /etc/findface-video-worker.ini (/etc/findface-video-worker-gpu.ini) configuration file. In the mgr-static parameter, specify the findface-video-manager host IP address, which provides findface-video-worker with settings and the video stream list. In the capacity parameter, specify the maximum number of video streams to be processed by findface-video-worker.

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

mgr-static=127.0.0.1:18811

capacity=10
```

5. Enable the FindFace core services autostart and launch the services.

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

Deploy FindFace Security Application Module and Biometric Database

To deploy the FindFace Security application module, do the following:

1. Install the `ffsecurity` and `ffsecurity-ui` components from the **`<ffsecurity-repo>.deb`** package.

```
sudo apt update
sudo apt install -y ffsecurity ffsecurity-ui
```

2. Migrate the database architecture from FindFace Security to **`PostgreSQL`**, create user groups with *predefined* rights and the first user with administrator rights (a.k.a. Super Administrator).

   Important: Super Administrator cannot be deprived of its rights, whatever the role.

```
sudo findface-security migrate
sudo findface-security create_groups
sudo findface-security createsuperuser --username admin --email root@localhost
```

3. Create a structure of the Tarantool-based biometric database.

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

4. Import the `meta_scheme` variable from the `tnt-schema.lua` file. Open the `/etc/tarantool/instances.enabled/FindFace.lua` configuration file. Before the `FindFace.start` section, add a line `dofile("/etc/ffsecurity/tnt-schema.lua")`. In the `FindFace.start` parameters, define `meta_scheme=meta_scheme`.

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

dofile("/etc/ffsecurity/tnt-schema.lua")

FindFace.start("@TNT_FF_LISTEN_IP@", @TNT_FF_LISTEN_PORT@, {
    license_ntls_server="@TNT_FF_NTLS@",
    facen_size=576,
    meta_scheme = meta_scheme
 })
```

5. Enable the `findface-tarantool-server` service autostart and launch the service.

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

6. Open the `/etc/ffsecurity/config.py` configuration file. Specify the following parameters:

   - `EXTERNAL_ADDRESS`: external IP address or URL that will be used to access the FindFace Security web interface.
   - `VIDEO_DETECTOR_TOKEN`: to authorize the video face detection module, come up with a token and specify it here.
   - `VIDEO_MANAGER_ADDRESS`: IP address of the `findface-video-manager` host.
   - `NTLS_HTTP_URL`: IP address of the `findface-ntls` host.
   - `ROUTER_URL`: IP address of the `ffsecurity` host that will receive detected faces from the `findface-video-worker` instance(s). Specify either external or internal IP address, subject to the network through which `findface-video-worker` interacts with `ffsecurity`.

- SF_API_ADDRESS: IP address of the findface-sf-api host.

Tip: If necessary, ensure data security by enabling *SSL*.

Tip: If necessary, set 'IGNORE_UNMATCHED': True to disable logging events for faces which have no match in the dossiers (negative verification result). Enable this option if the system has to process a large number of faces. The face similarity threshold for verification is defined by the CONFIDENCE_THRESHOLD parameter.

Tip: It is recommended to change the MINIMUM_DOSSIER_QUALITY default value. This parameter determines the minimum quality of a face in a dossier photo. Photos containing faces of worse quality will be rejected when uploading to a dossier. Upright faces in frontal position are considered the best quality. They result in values around 0, mostly negative (such as -0.00067401276, for example). Inverted faces and large face angles are estimated with negative values some -5 and less. By default, 'MINIMUM_DOSSIER_QUALITY': -2 which is the average quality.

Important: If you enabled recognition models in the findface-extraction-api configuration file, add the following line in the FFSECURITY section: 'EVENTS_FEATURES': ['gender', 'age', 'emotions', 'beard', 'glasses'], subject to the list of enabled models.

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

MEDIA_ROOT="/var/lib/ffsecurity/uploads"
STATIC_ROOT="/var/lib/ffsecurity/static"

EXTERNAL_ADDRESS="http://172.20.77.23"

DEBUG = False

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'ffsecurity',
    }
}

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

FFSECURITY = {
    'VIDEO_DETECTOR_TOKEN': 'GOOD_TOKEN',
    'CONFIDENCE_THRESHOLD': 0.75,
    'MINIMUM_DOSSIER_QUALITY': -2,
    'IGNORE_UNMATCHED': False,
    'VIDEO_MANAGER_ADDRESS': 'http://127.0.0.1:18810',
    'EVENTS_MAX_AGE': 30,
    'NTLS_HTTP_URL': 'http://127.0.0.1:3185',
```

(continues on next page)

(continued from previous page)

```
    'ROUTER_URL': 'http://172.20.77.23',
    'MONITORING_UPDATE_INTERVAL': 60,
    'SF_API_ADDRESS': 'http://127.0.0.1:18411',
    'EVENTS_FEATURES': ['gender', 'age', 'beard', 'glasses'],
}

FFSECURITY_UI_CONFIG = {
}

# integration plugins
INSTALLED_APPS.append('ffsecurity_genetec') # remove or comment out this line to␣
↪disable genetec integration```
```

7. Generate a signature key for the session encryption (used by Django) by executing: `pwgen -sncy 50 1|tr “’” “.”`. Specify this key as `SECRET_KEY`.

8. Start the services.

Important: The `ffsecurity` service includes `findface-security-proto` (provides HTTP and web socket), `findface-security-worker` (provides interaction with the other system components), `findface-security-monitoring-updater`, and `findface-security-webhook-updater`.

Important: The number of the `findface-security-worker` instances is calculated using the formula: `N=(number of CPU cores-1)`, and specified after the `@` character, for example, `findface-security-worker@{1,2,3}` indicates 3 instances.

In the example below, `findface-security-worker` has 4 instances.

```
sudo systemctl enable findface-security-proto findface-security-worker@{1,2,3,4}␣
↪findface-security-monitoring-updater findface-security-webhook-updater
sudo systemctl start findface-security-proto findface-security-worker@{1,2,3,4}␣
↪findface-security-monitoring-updater findface-security-webhook-updater
```

Important: For high load projects, start more instances of `findface-security-worker`. The main indicator that allocated resources are not enough and you should start more `findface-security-worker` instances is errors in the web interface.

9. Disable the default nginx server and add the `ffsecurity` server to the list of enabled servers. Restart nginx.

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

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

sudo nginx -s reload
```

1.3.3 Additional `findface-video-worker` deployment on remote hosts

To install only the `findface-video-worker` service, do the following:

Tip: Before deployment, be sure to consult the *system requirements*.

1. Download the installer file `findface-security-2.0.0-server-3.0.0.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.

   ```
   chmod +x findface-security-2.0.0-server-3.0.0.run
   ```

4. Execute the `.run` file.

   ```
   sudo ./findface-security-2.0.0-server-3.0.0.run
   ```

   The installer will ask you a few questions and perform several automated checks to ensure that the host meets the system requirements. Fill out the prompts appropriately once requested. The questions are the following:

   1. Product to install: FindFace Video Worker.
   2. Type of `findface-video-worker` package: CPU or GPU.
   3. IP address of the `ffsecurity` host.

   After that, the installation process will automatically begin.

   Note: The answers will be saved to a file `/tmp/<findface-installer-*>.json`. You can edit this file and use it to install FindFace Security on other hosts without having to answer the questions again.

Note: If you chose to install `findface-ntls` and/or `findface-video-manager` on different hosts than that with `ffsecurity`, specify their IP addresses in the `/etc/findface-video-worker.ini` configuration file after the installation.

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

In the `ntls-addr` parameter, specify the `findface-ntls` host IP address.

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

In the `mgr-static` parameter, specify the `findface-videomanager-api` host IP address, which provides `findface-video-worker` with settings and the video stream list.

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

Tip: To automatically install `findface-video-worker` on another host without answering the installation questions, use the `/tmp/<findface-installer-*>.json` file. Execute:

```
sudo ./findface-security-2.0.0-server-3.0.0.run -f /tmp/<findface-installer-*>.json
```

You can find an example of the installation file in *Installation File*.

1.4 Work with FindFace Security

Use the web interface to interact with FindFace Security. 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 *User Management*.

The web interface has a highly intuitive and handy design and provides the following functionality:

- *Camera Management*. Group cameras subject to their location. Add and configure a camera.
- *Dossier Database*. Manage dossier classification lists (watch lists). Create dossiers manually and in bulk.
- *User Management*. Manage FindFace Security users and their roles.
- *General Preferences*. Configure the confidence threshold for face verification. Set up automatic cleanup of the event database.
- *Compare faces*. Verify that 2 given faces belong to the same person.
- Operator's Guide. *Real time face identification* in live streams and video files. *Search for faces* in the event list and dossier database.

1.4.1 Camera Management

To configure video-based biometric identification, add cameras to FindFace Security, grouping them subject to their location.

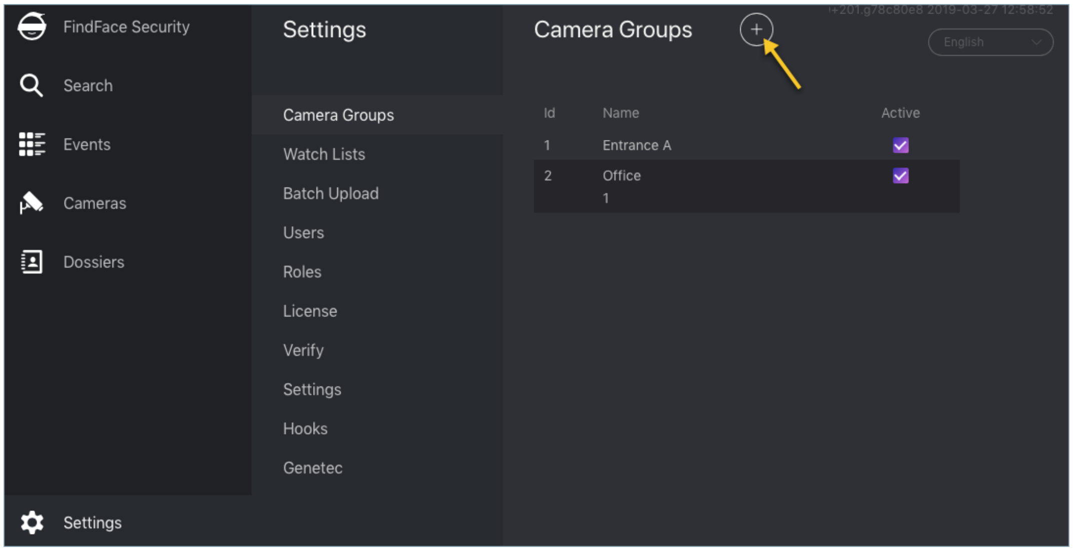
Note: Privileges to create camera groups and cameras are managed in user's permissions (see *User Management*).

In this chapter:

Create Camera Group

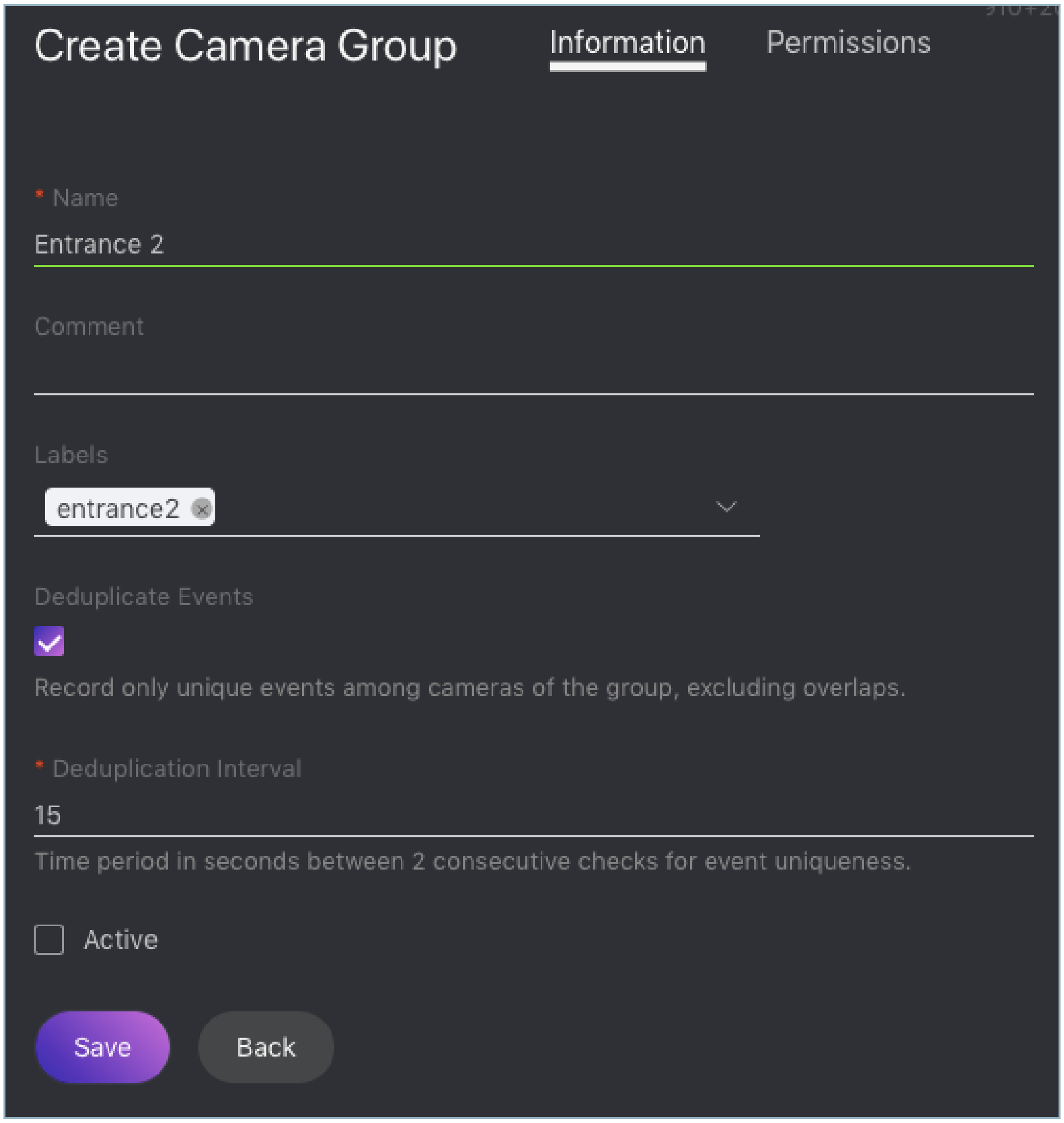
To create a group of cameras, do the following:

1. Navigate to the *Preferences* tab. Click *Camera Groups*.
2. Click +.



Id	Name	Active
1	Entrance A	☑
2	Office 1	☑

3. On the *Information* tab, specify the group name. Add a comment if needed.



Create Camera Group

* Name: Entrance 2

Comment:

Labels: entrance2

Deduplicate Events: ☑

Record only unique events among cameras of the group, excluding overlaps.

* Deduplication Interval: 15

Time period in seconds between 2 consecutive checks for event uniqueness.

□ Active

4. If you want to allocate a certain `findface-video-worker` instance to process video streams from the group, create or select one or several allocation labels.

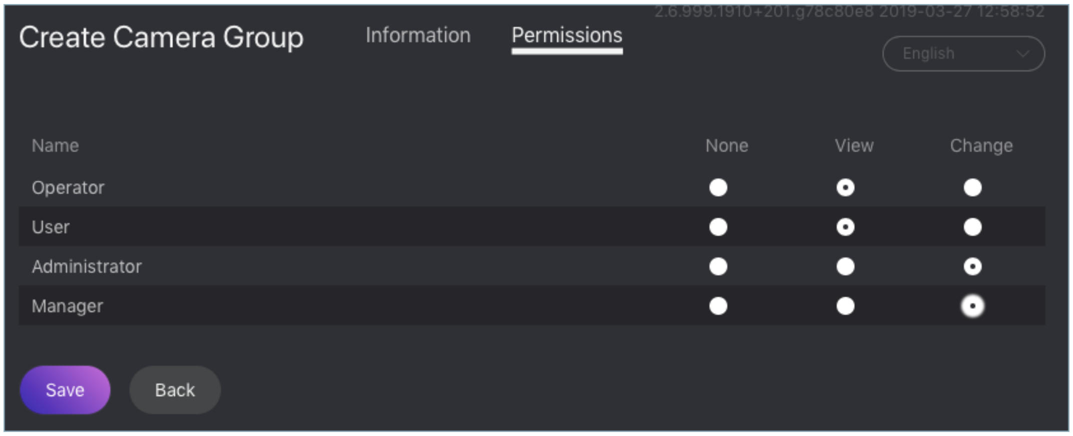
   Note: To complete the allocation, list the labels in the `findface-video-worker` configuration file. See *Allocate findface-video-worker to Camera Group* for details.

5. If you want to deduplicate events from cameras that belong to the same group, i. e. exclude coinciding events, check *Deduplicate Events* and specify the deduplication interval (interval between 2 consecutive checks for event uniqueness).

   Warning: Use deduplication with extreme caution, as if cameras within a group observe different scenes, some faces may be skipped. See *Deduplicate Events* for details.

6. Check *Active*.
7. Click *Save*.

8. On the *Permissions* tab, assign privileges on the camera group, specifying which user roles are allowed to change/view the camera group settings.



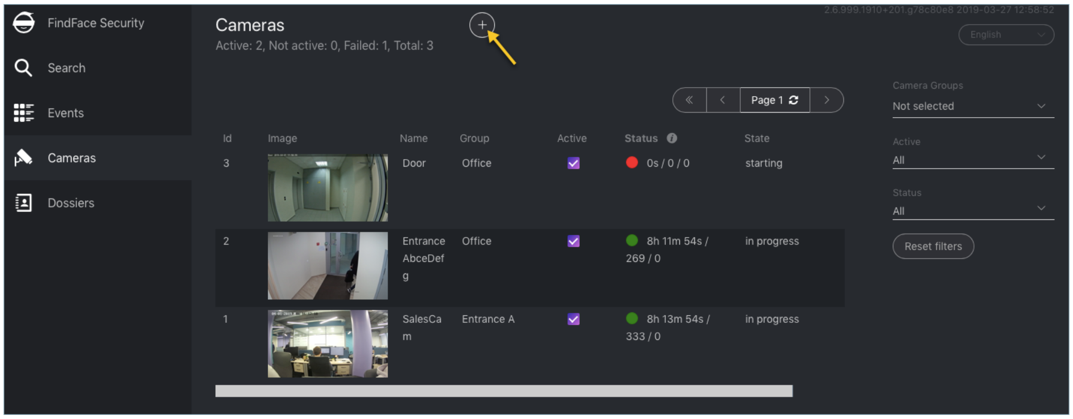
Name	None	View	Change
Operator	○	●	○
User	○	●	○
Administrator	○	○	●
Manager	○	○	●

9. Click *Save*.

Add Camera

To add a camera, do the following:

1. Navigate to the *Cameras* tab.
2. Click +.



Active: 2, Not active: 0, Failed: 1, Total: 3

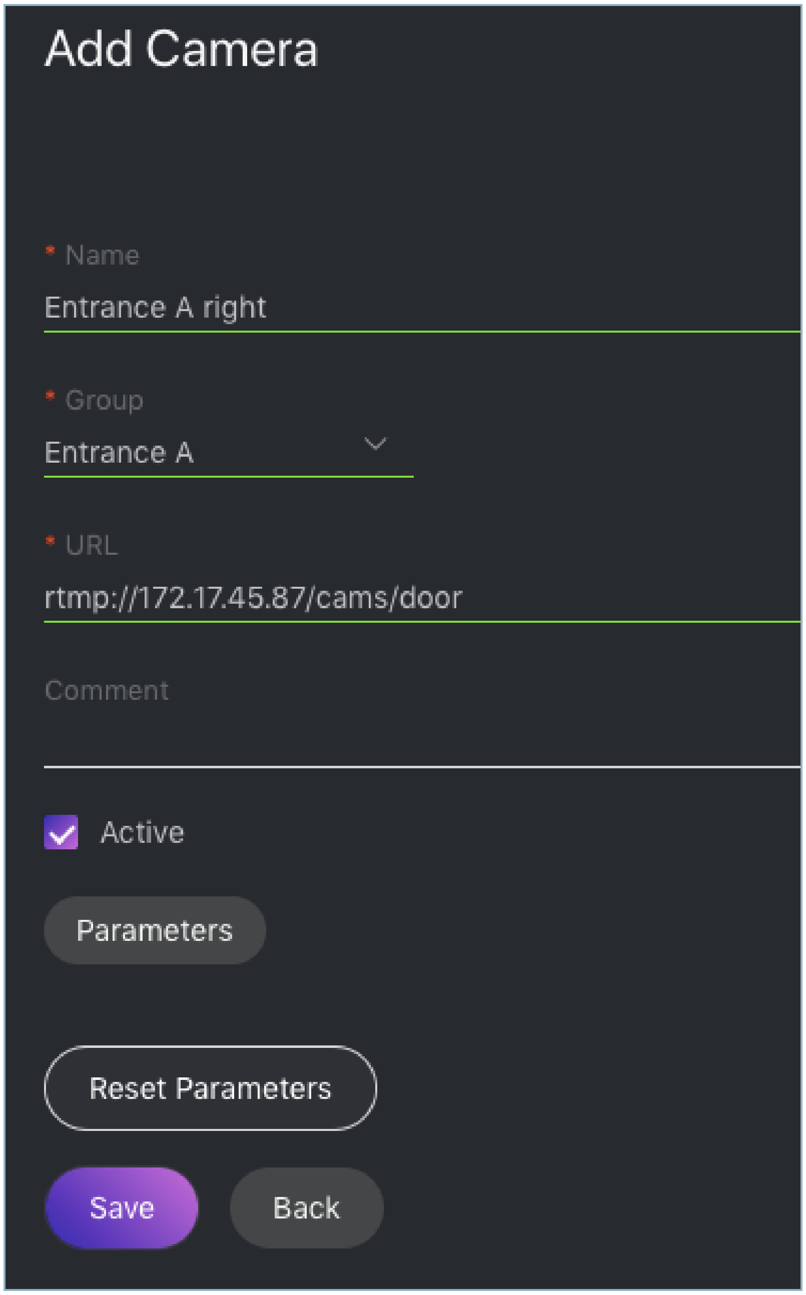
Id	Image	Name	Group	Active	Status	State
3		Door	Office	☑	🟥 0s / 0 / 0	starting
2		Entrance AbceDef g	Office	☑	🟩 8h 11m 54s / 269 / 0	in progress
1		SalesCa m	Entrance A	☑	🟩 8h 13m 54s / 333 / 0	in progress

Camera Groups: Not selected

Active: All

Status: All

3. Specify the name of a camera and add it to a group. If necessary, add a comment.



Add Camera

* Name

Entrance A right

* Group

Entrance A

* URL

rtmp://172.17.45.87/cams/door

Comment

☑ Active

4. Specify the camera URL or path to the video file, for example, `file:///data/some.mp4`.
5. Check *Active*.
6. To configure CPU-based video processing, click *Parameters* and navigate to the *CPU* tab.
   - *Minimum face quality* (`min_score`): Minimum quality of a face snapshot when searching for the best one. To be fitted empirically: negatives values around `0` = high quality faces, `-1` = good quality, `-2` = satisfactory quality, `-5` = inverted faces and large face angles, face recognition may be inefficient.
   - *Maximum face deviation* (`min_d_score`): Maximum deviation of a face from its frontal position. To be fitted empirically: `-3.5` = large face angles, face recognition may be inefficient, `-2.5` = satisfactory deviation, `-0.05` = close to the frontal position, `0` = frontal face.
   - *Minimum face size* (`min_face_size`): Minimum face size in pixels. The less the value, the longer it

takes to detect and track faces. Optimum value: 80-100-120. If 0, the filter is off.

- *Maximum face size* (`max_face_size`): Maximum face size in pixels. If 0, the filter is off.
- *Realtime mode* (`realtime`): Realtime mode. Pick up the best snapshot within each Snapshot picking interval time interval. If Post each best snapshot: true, the best snapshot is posted at the end of each interval; if false, the best snapshot is posted only if its quality has improved comparing to the previously posted snapshot.
- *Time interval* (`realtime_dly`): Time interval in milliseconds within which the face tracker picks up the best snapshot in realtime mode.
- *Post best snapshot* (`realtime_post_perm`): If true, post the best snapshot obtained within each Snapshot picking interval time interval in realtime mode. If false, post the best snapshot only if its quality has improved comparing to the previously posted snapshot.
- *Offline mode* (`overall`): Offline mode. Enable posting one snapshot of the best quality for each face.
- *Region of Tracking* (`ROT`): Enable detecting and tracking faces only inside a clipping rectangle. Use this option to reduce the video face detector load.
- *Region of Interest* (`ROI`): Enable posting faces detected only inside a region of interest.

Tip: To specify ROT/ROI, use the visual wizard. First, create a camera without ROT/ROI. Then open it for editing and click *Parameters*. You will see the visual wizard appear.

If necessary, specify optional parameters for CPU-based video processing. Click *Advanced Parameters*.

- *FFMPEG options* (`ffmpeg_params`): FFMPEG options for a video stream in the key-value format ["rtsp_transpotr=tcp", "ss=00:20:00"].
- *Video frame height* (`fd_frame_height`): Video frame height in pixels for the face tracker. Negative values correspond to the initial size. Optimum value to reduce load: 640-720.
- *Maximum number of faces* (`npersons`): Maximum number of faces simultaneously tracked by the face tracker. This parameter severely affects performance.
- *Tracking threads number* (`tracker_threads`): Number of tracking threads for the face tracker. This value should be less or equal to the maximum number of tracked faces. Recommended to set them equal. If the number of tracking threads is less than the maximum number of tracked faces, resource consumption is reduced but so is the tracking speed.
- *Compression quality* (`jpeg_quality`): Full frame compression quality.
- *Face motion track drawing* (`draw_track`): Enable drawing a face motion track in a bbox.
- *Response timeout* (`api_timeout`): Response timeout in milliseconds for an API request.
- *Minimum motion intensity* (`md_threshold`): Minimum motion intensity to be detected by the motion detector. To be fitted empirically: 0 = detector disabled, 0.002 = default value, 0.05 = minimum intensity is too high to detect motion.
- *Video frame scaling coefficient* (`md_scale`): Video frame scaling coefficient for the motion detector from 0 to 1. Scale down in the case of high resolution cameras, or close up faces, or if the CPU load is too high, to reduce the system resources consumption.

7. To configure GPU-based video processing, click *Parameters* and navigate to the *GPU* tab.

   - *Minimum face quality* (`min_score`): Minimum quality of a face snapshot to post. To be fitted empirically: negatives values around 0 = high quality faces, -1 = good quality, -2 = satisfactory quality, -5 = inverted faces and large face angles, face recognition may be inefficient.

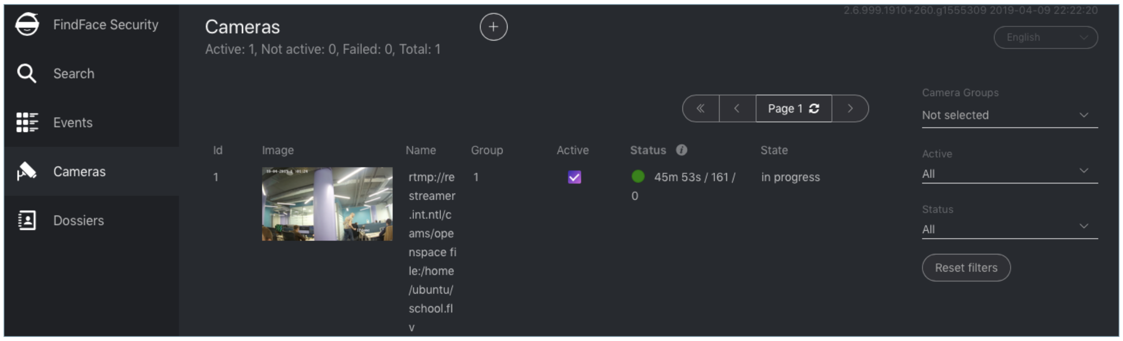
- *Minimum face size* (`min_face_size`): Minimum face size in pixels to post. If 0, the filter is off.
- *Maximum face size* (`max_face_size`): Maximum face size in pixels in post.
- *Compression quality* (`jpeg_quality`): Full frame compression quality.
- *FFMPEG options* (`ffmpeg_params`): FFMPEG options for a video stream in the key-value format [“rtsp_transpotr=tcp”, “ss=00:20:00”].
- *Post best snapshot* (`realtime_post_perm`): Offline mode. Enable posting one snapshot of the best quality for each face.
- *Posting timeout* (`router_timeout_ms`): Response timeout in milliseconds for an API request.
- *Retrieve timestamps from stream* (`use_stream_timestamp`): If true, retrieve and post timestamps from a video stream. If false, post the actual date and time.
- *Add to timestamps* (`start_stream_timestamp`): Add the specified number of seconds to timestamps from a stream.

8. Click *Save*.

Note: Each created camera is associated with a so called job, a video processing task which contains configuration settings and stream data and assigned to `findface-video-worker`. This task can be restarted (see *Monitor Camera Operation*).

Monitor Camera Operation

To monitor the operation of cameras, navigate to the *Cameras* tab.



2.6.999.1910+260.g1555309 2019-04-09 22:22:20

Cameras

Active: 1, Not active: 0, Failed: 0, Total: 1

Id	Image	Name	Group	Active	Status	State
1		rtmp://restreamer.int.ntl/cams/openspace file:/home/ubuntu/school.flv	1	☑	● 45m 53s / 161 / 0	in progress

Camera Groups: Not selected

Active: All

Status: All

Camera statuses:

- Green: the video stream is being processed without errors.
- Yellow: the video stream is being processed for less than 30 seconds, or one or more errors occurred when posting a face.
- Red: the video stream cannot be processed.

For each camera, you will be provided with the following statistics: current session duration/ the number of successfully posted faces/ the number of faces processed with errors after the last job restart.

To restart a job, click ⟳ in the *Action* column. In this case, the number of errors will be reset to `0`.

With a large number of cameras in the system, use the following filters:

- *Camera groups*,

- *Active*,
- *Status*.

See also:

- *Allocate findface-video-worker to Camera Group*
- *Deduplicate Events*

1.4.2 Face Monitoring and Dossier Database

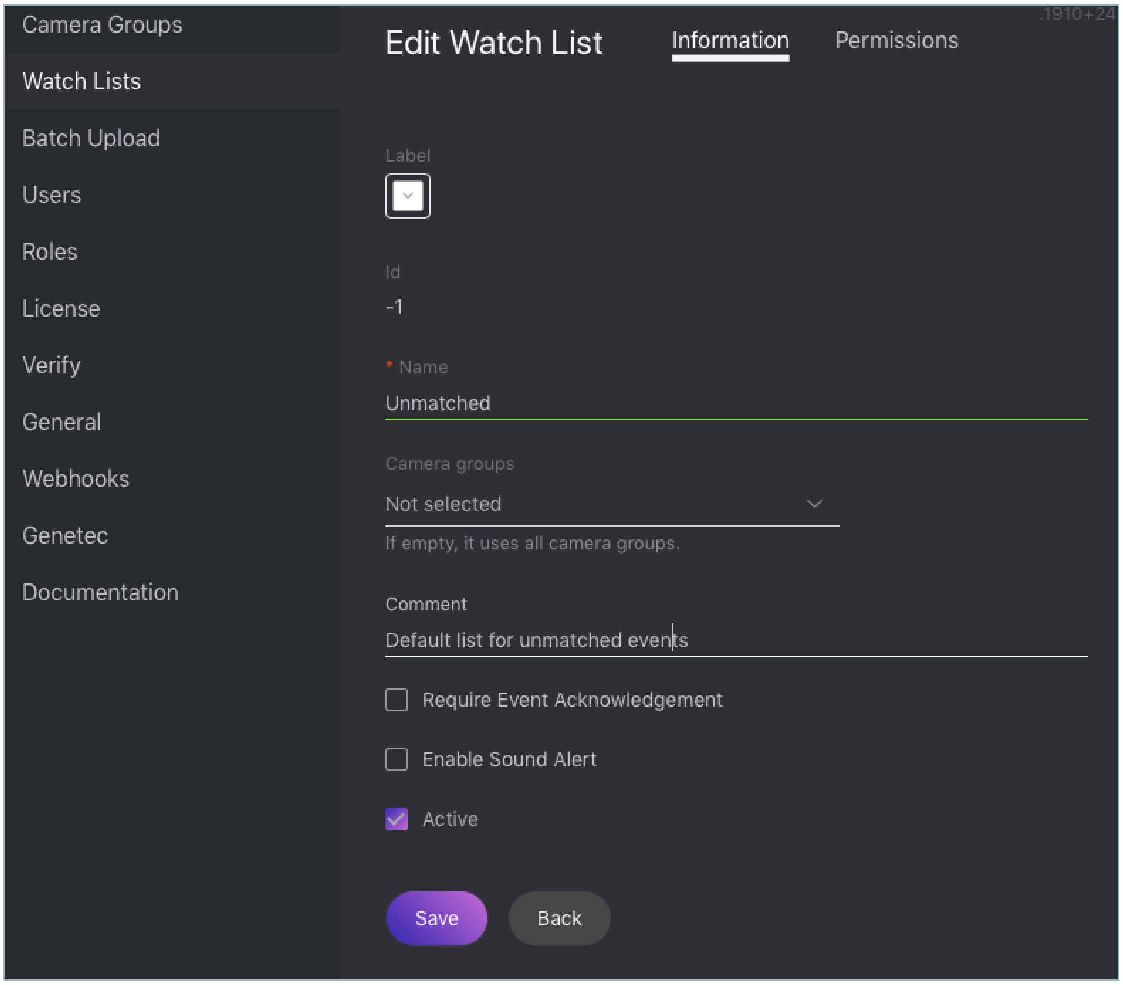
This chapter is all about monitoring detected faces and creating the dossier database. Each dossier has to contain one or several photos of a person and belong to a certain classification list (watch list), black or white in the simplest case. You can create several watch lists, subject to a person status or hazard level.

Tip: To create dossiers in bulk, use the batch photo upload functionality.

In this section:

Monitoring Unmatched Faces

FindFace Security features one pre-configured watch list that is used for monitoring only unmatched faces. This watch list cannot be removed from the system. To edit its settings or deactivate it, navigate to the *Preferences* tab. Click *Watch Lists* and then click *Unmatched* in the table.



Edit Watch List

Label

Id
-1

* Name
Unmatched

Camera groups
Not selected
If empty, it uses all camera groups.

Comment
Default list for unmatched events

□ Require Event Acknowledgement

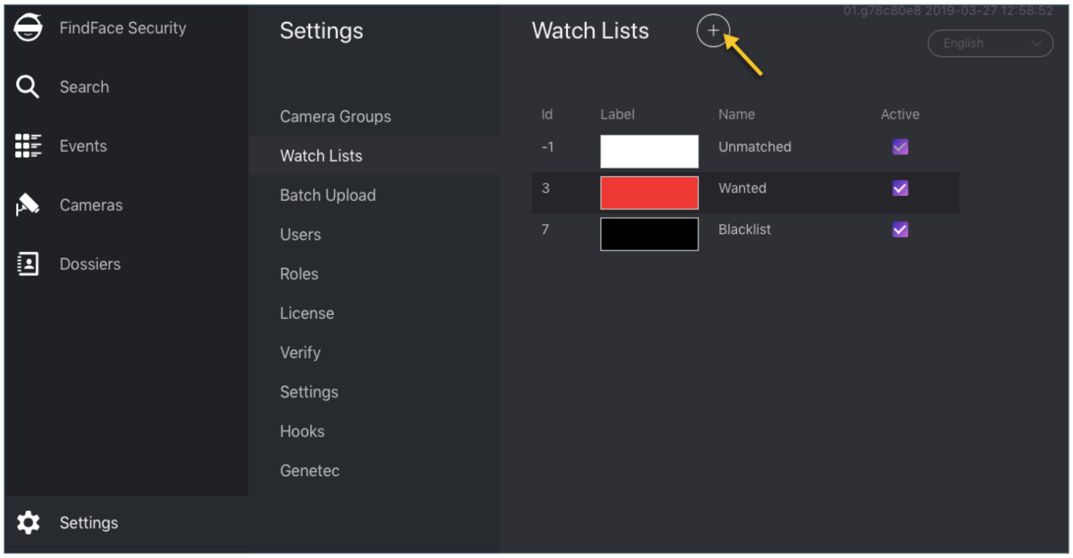
□ Enable Sound Alert

☑ Active

Create Watch List

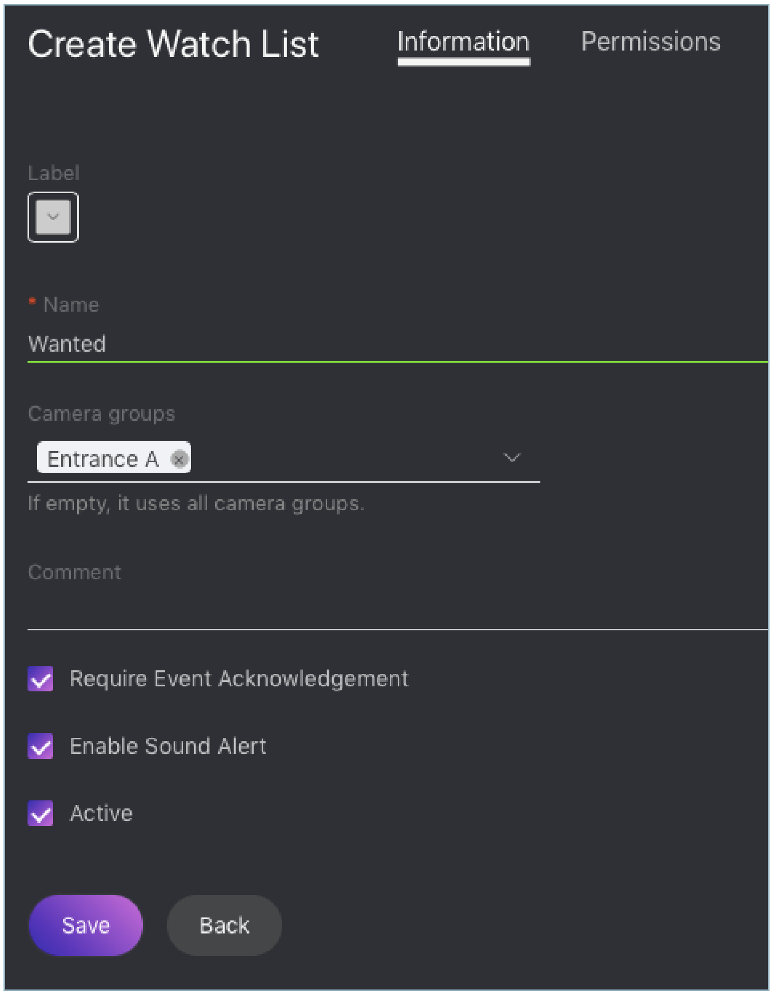
To create a custom watch list, do the following:

1. Navigate to the *Preferences* tab. Click *Watch Lists*.
2. Click +.



Id	Label	Name	Active
-1	⬜	Unmatched	☑
3	🟥	Wanted	☑
7	⬛	Blacklist	☑

3. From the *Label* palette, select a color which will be shown in notifications for this list. Keep in mind that the right color makes for quicker response of security and hospitality managers.



Label

* Name: Wanted

Camera groups: Entrance A

If empty, it uses all camera groups.

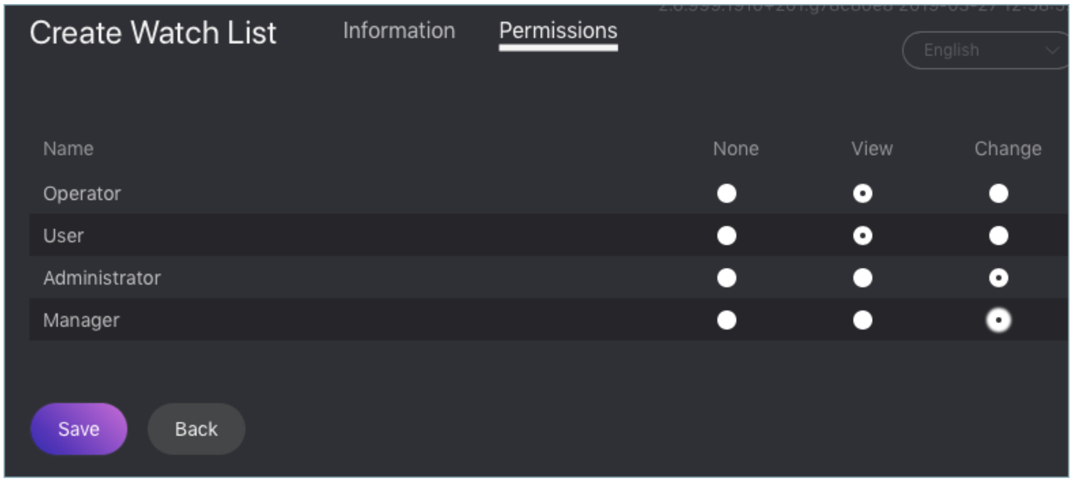
Comment

☑ Require Event Acknowledgement

☑ Enable Sound Alert

☑ Active

4. Specify the watch list name. Add a comment if needed.
5. Select a camera group(s) which will be used to monitor the watch list. If no groups specified, the watch list will be monitored by all active cameras in the system.
6. Check *Require acknowledgment* if it is mandatory that a manager acknowledge events for the list.
7. Check *Enable sound alert* to turn on sound notifications for the list if needed.
8. Check *Active*.
9. Click *Save*.
10. On the *Permissions* tab, assign privileges on the watch list, specifying which user roles are allowed to change/view the watch list settings.



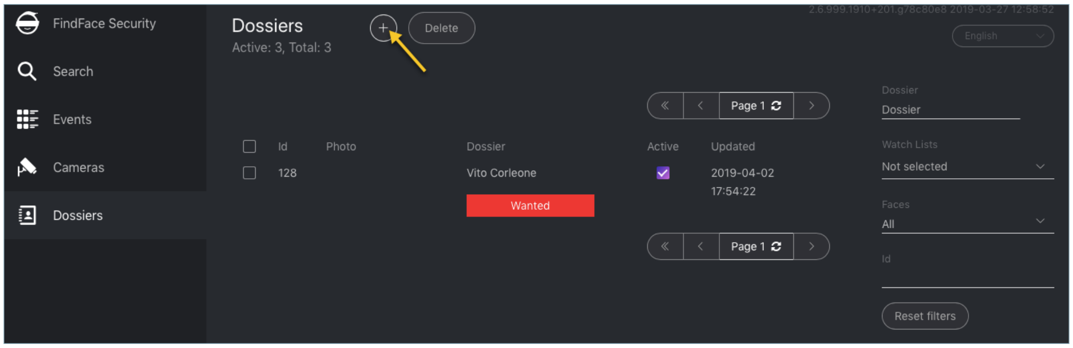
Name	None	View	Change
Operator	○	◉	○
User	○	◉	○
Administrator	○	○	◉
Manager	○	○	◉

11. Click :*Save*.

Create Dossier Manually

To create a dossier manually, do the following:

1. Navigate to the *Dossiers* tab.
2. Click +.



Dossiers
Active: 3, Total: 3

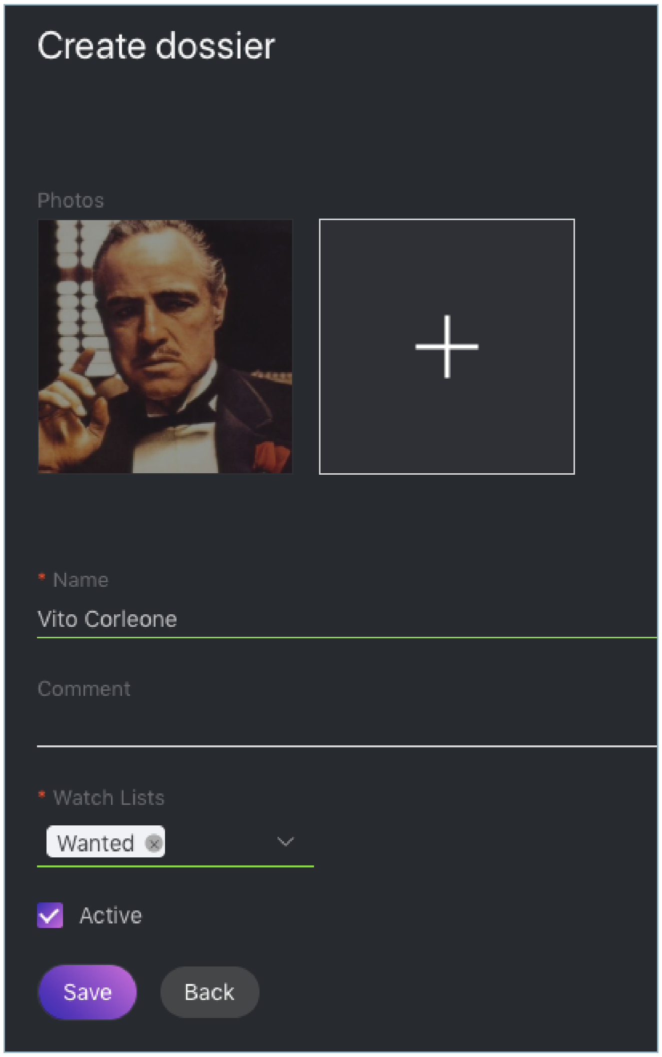
□	Id	Photo	Dossier	Active	Updated
□	128		Vito Corleone Wanted	☑	2019-04-02 17:54:22

Dossier: Dossier
Watch Lists: Not selected
Faces: All
Id:

3. Attach a photo and specify the name of a person. If necessary, add a comment.

Important:

> A face in the photo must be of high quality, i.e. close to a frontal position. Photos that do not meet the requirement will be rejected with a detailed error description.



Create dossier

Photos

* Name

Vito Corleone

Comment

* Watch Lists

Wanted

☑ Active

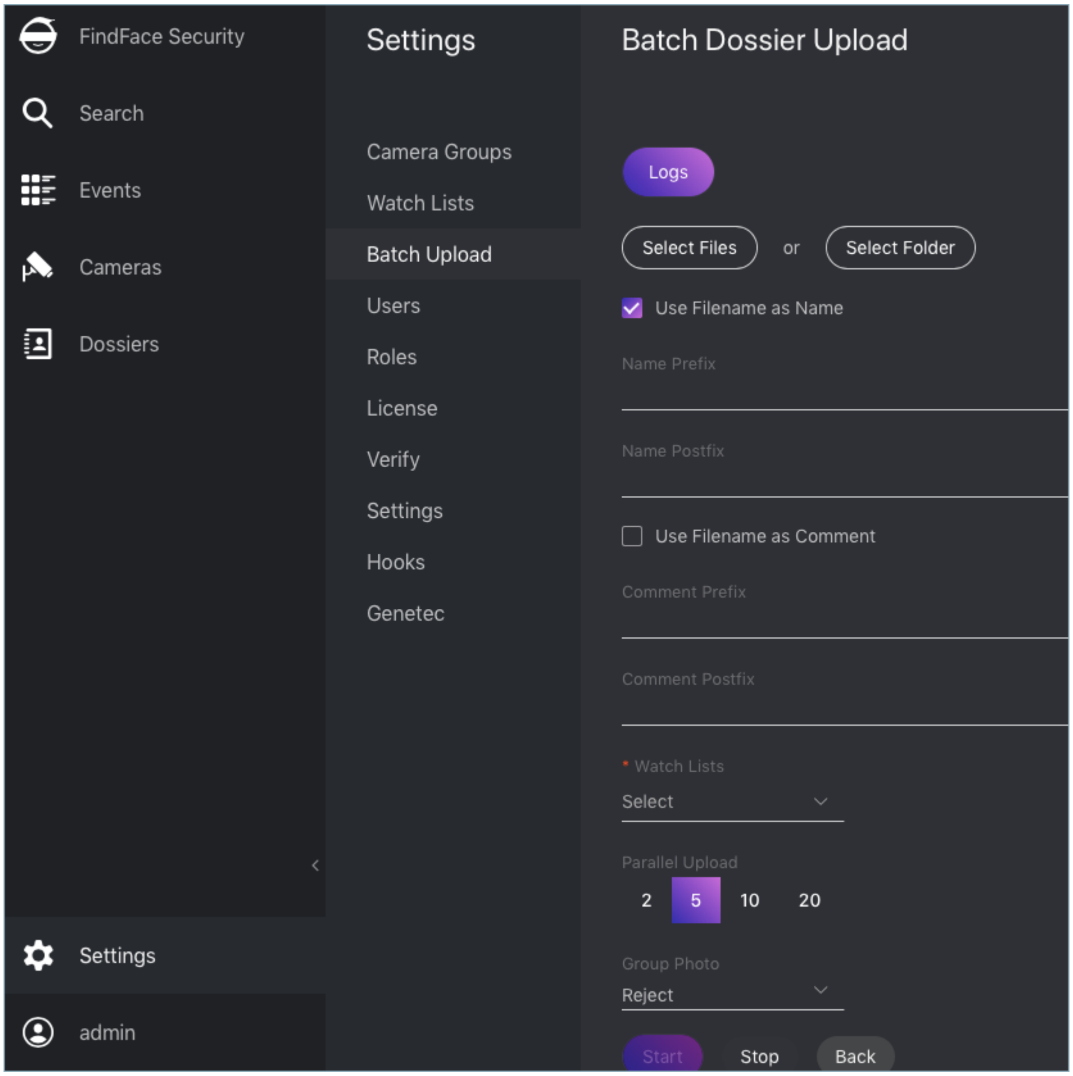
4. From the *Watch lists* drop-down menu, select a classification list (or several lists, one by one) for the dossier.
5. Check *Active*. If a dossier is inactive, it is excluded from the real time face identification.
6. Click *Save*.

Batch Photo Upload

To create dossiers in bulk, use the batch photo upload. Do the following:

Tip: If you need to upload a large number of photos (more than 10,000), use *Console Bulk Photo Upload*.

1. Navigate to the *Preferences* tab. Click *Batch Upload*.



2. Select multiple image files, or a folder.

3. You can use image file names as a basis for names and/or comments in dossiers to be created. Select the necessary option(s). Then configure the automatic name/comment generation rule by appending a custom prefix and/or postfix to the file name.

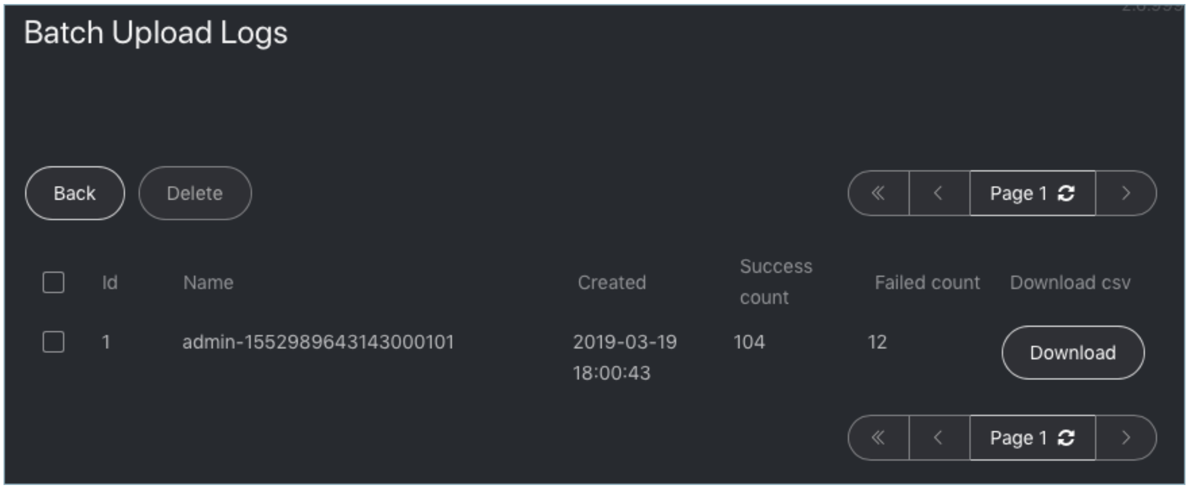
   Tip: To avoid merging the 3 words into one, use underscore or another symbol in the prefix and postfix.

4. From the *Watch lists* drop-down menu, select a classification list for the dossiers.

5. Use the *Parallel Upload* option to specify the number of photo upload streams. The more streams you use, the faster it takes to complete the upload, however it requires more resources as well.

6. From the *Group Photo* drop-down menu, select the system behavior upon detecting several faces in a photo: reject the photo, or upload the biggest face.
7. Click *Start* to launch the photo upload.

Important: To view the batch photo upload log, click *Logs*. You can then download the log in the `.csv` format if needed.

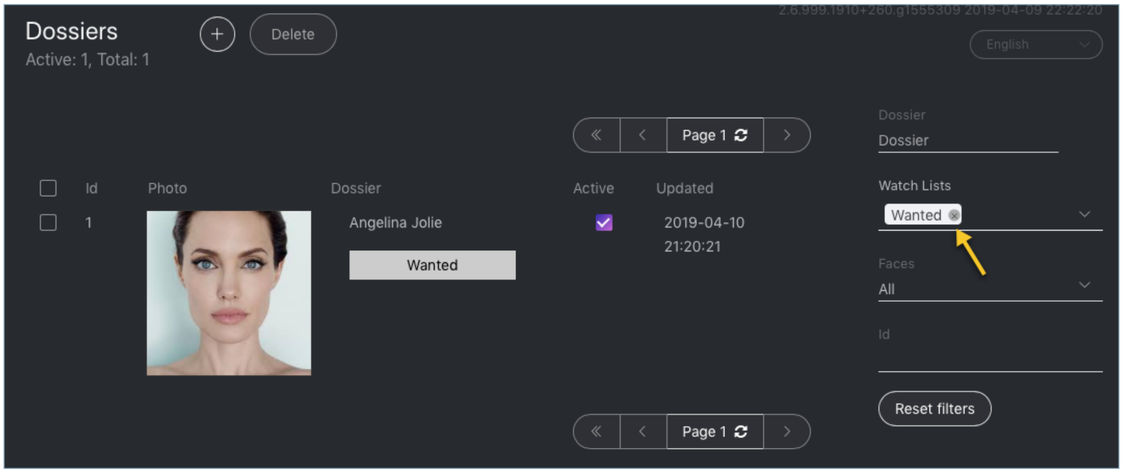


Batch Upload Logs

☐	Id	Name	Created	Success count	Failed count	Download csv
☐	1	admin-1552989643143000101	2019-03-19 18:00:43	104	12	Download

Filter Dossiers by Watch List

You can find all dossiers created in FindFace Security on the *Dossiers* tab. Use the *Watch lists* filter to filter dossiers by list.



Dossiers
Active: 1, Total: 1

☐	Id	Photo	Dossier	Active	Updated
☐	1		Angelina Jolie Wanted	☑	2019-04-10 21:20:21

Dossier: Dossier
Watch Lists: Wanted
Faces: All
Id:

1.4.3 User Management

In this chapter:

Predefined Roles

FindFace Security provides the following predefined roles:

- Administrator has rights to *manage cameras*, events, FindFace Security users, the *dossier database*, and full access to all other functions.

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

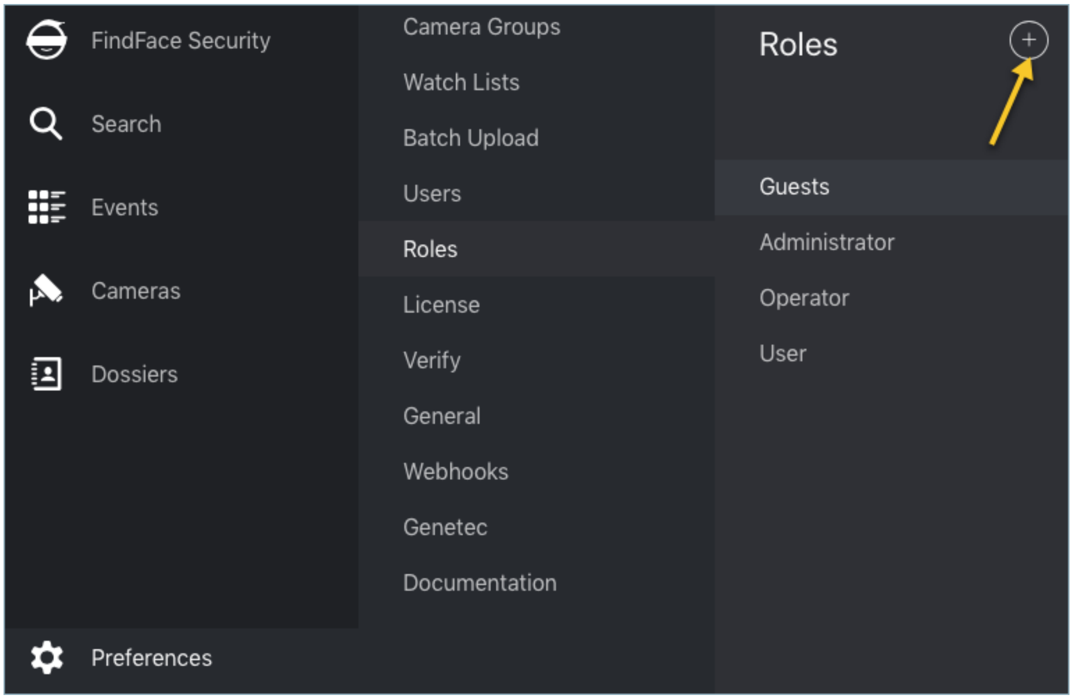
- Operator can *create dossiers manually*, receive and acknowledge events, and search for faces in the event list. The other data is available read-only. The *batch dossier creation* is unavailable.
- User has a right to receive and acknowledge events, and to search for faces in the event list. The other data is available read-only.

You can change the predefined roles privileges, as well as create various custom roles.

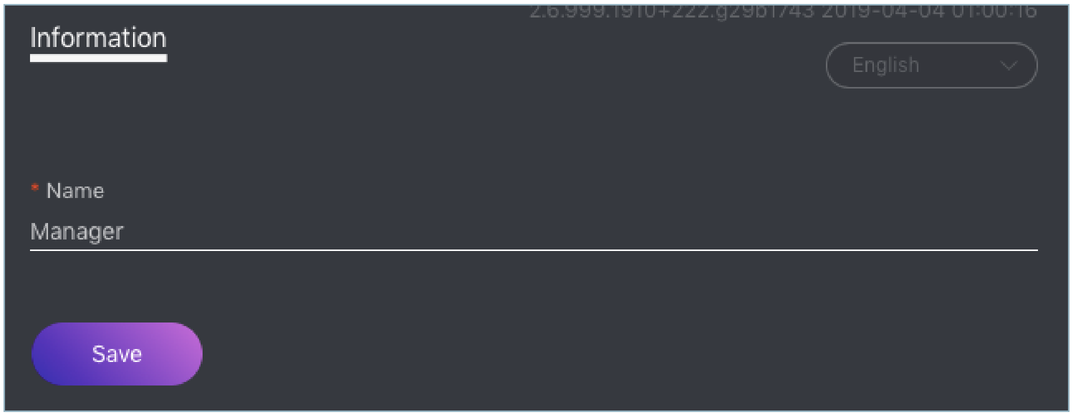
Create Custom Role

To create a custom role, do the following:

1. Navigate to the *Preferences* tab. Click *Roles*.
2. Click +.

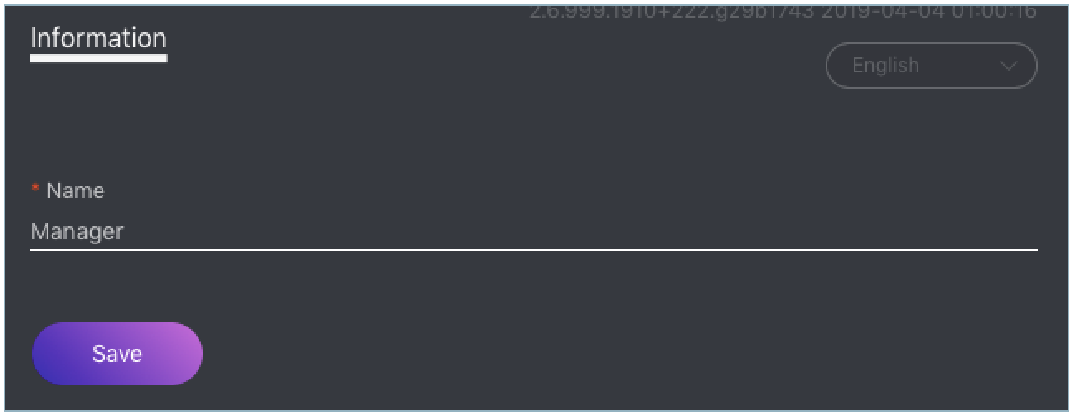


3. On the *Information* tab, specify the role name.



* Name
Manager

4. Click *Save*. You will see the *Watch Lists*, *Camera Groups*, and *Permissions* tabs appear in addition to the *Information* tab. You can use these tabs to assign the role privileges for specific watch lists and camera groups, as well as for various system functions.



* Name

Manager

Primary and Additional User Privileges

You assign privileges to a user by assigning one primary and (optional) few additional roles:

- *Primary role*: main user role, mandatory for assignment. You can assign only one primary role to one user. A user's primary role will be automatically added with `write` permissions to Access Control List for all objects newly created by this user.
- *Role*: 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.

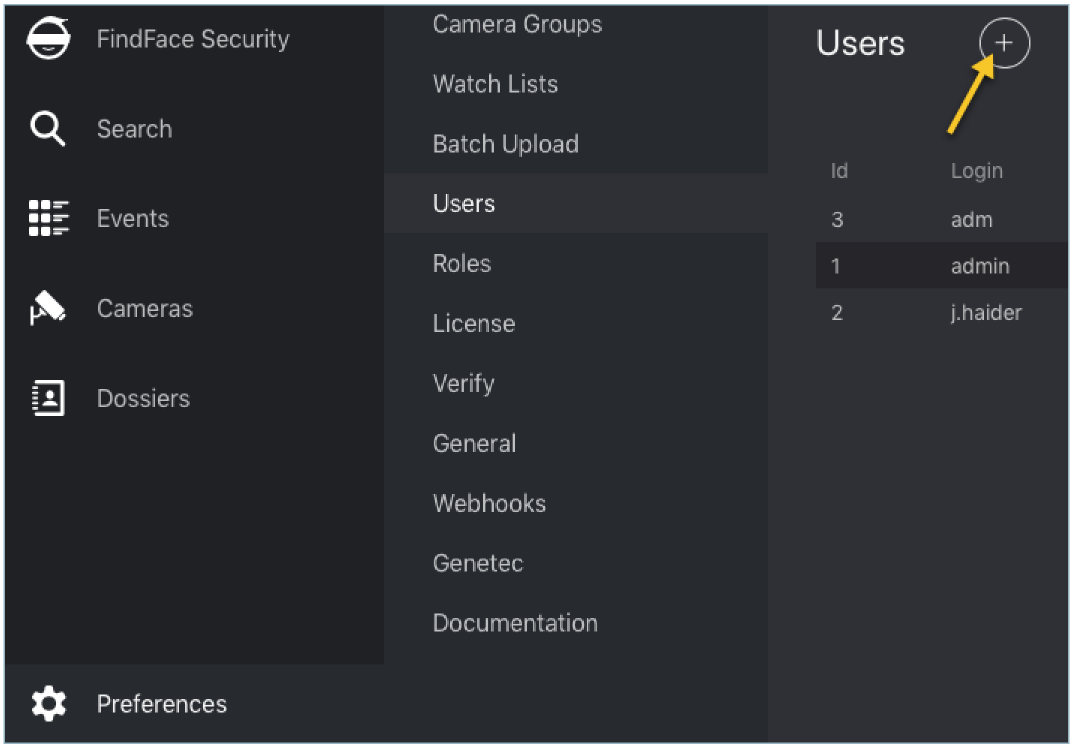
See also:

Create User

Create User

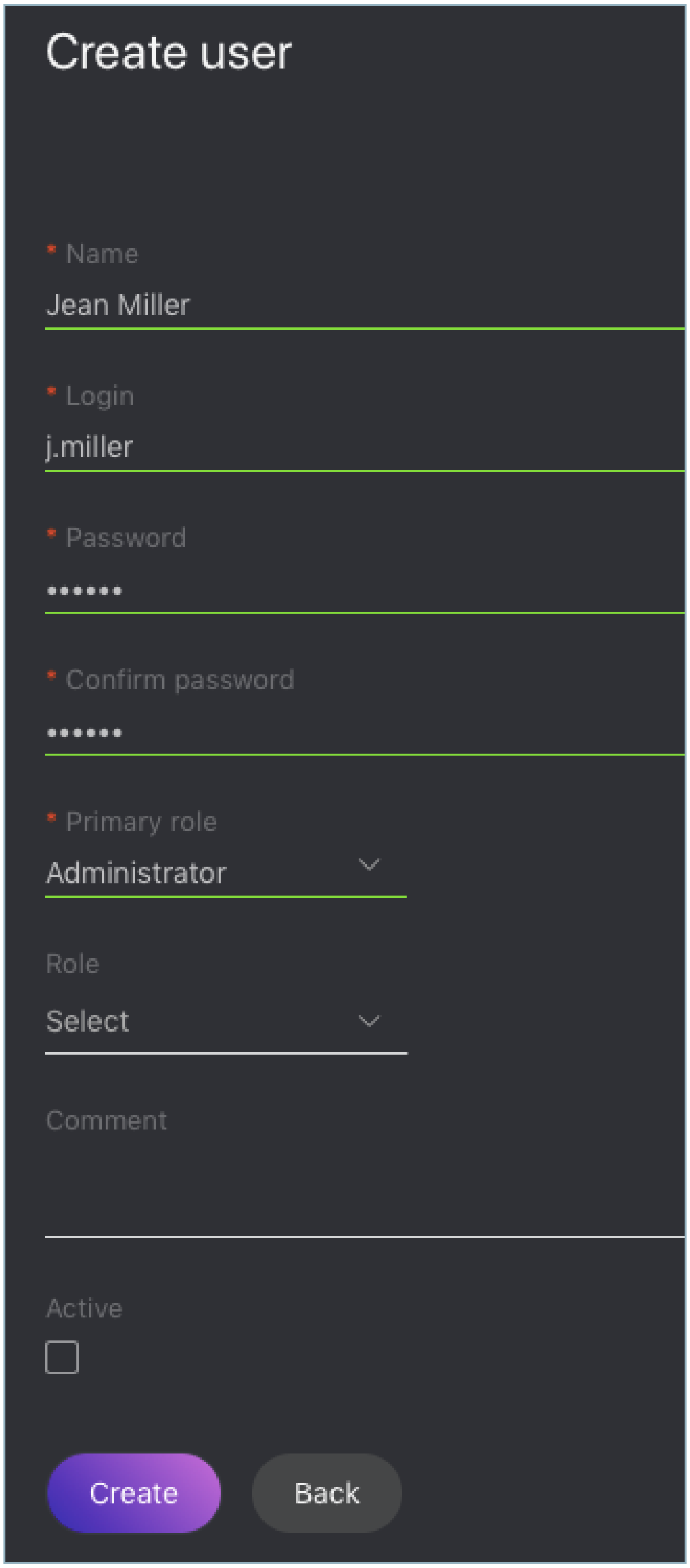
To create a user, do the following:

1. Navigate to the *Preferences* tab. Click *Users*.
2. Click +.



Id	Login
3	adm
1	admin
2	j.haider

3. Specify such user data as name, login and password. If necessary, add a comment.
4. From the *Primary role* drop-down menu, select the user primary role. If necessary, add one or several additional roles by selecting them from the *Role* drop-down menu. The rights associated with the additional roles will be added to the primary privileges. The user's primary role will be automatically added with `write` permissions to Access Control List for all objects newly created by this user.



Create user

* Name

Jean Miller

* Login

j.miller

* Password

••••••

* Confirm password

••••••

* Primary role

Administrator

Role

Select

Comment

Active

□

5. Check *Active*.
6. Click *Create*.

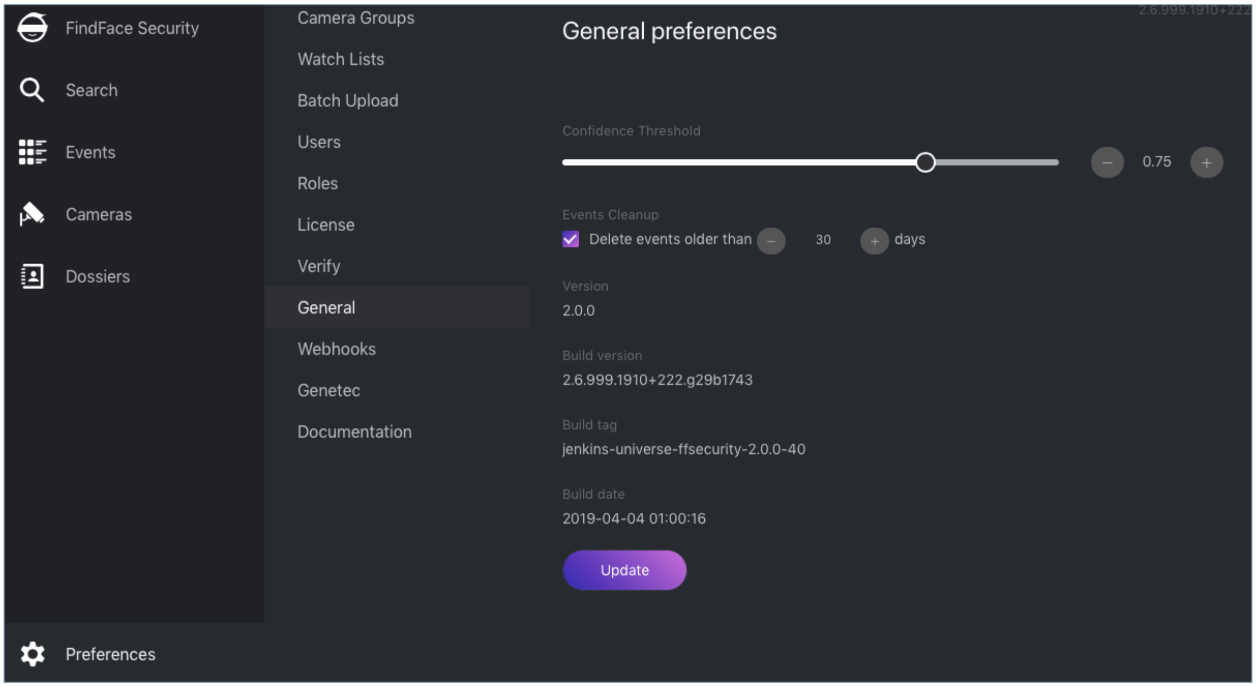
Deactivate or Delete User

In order to deactivate a user, simply uncheck *Active* in the user list (*Preferences* -> *Users*).

To delete a user from FindFace Security, click on the user login in the list. Click *Delete*.

1.4.4 General Preferences

To configure the confidence threshold for face verification and automatic events cleanup, navigate to the *Preferences* tab. Click *General*. After you are finished, click *Update*.



General preferences

Confidence Threshold

0.75

Events Cleanup

☑ Delete events older than 30 days

Version

2.0.0

Build version

2.6.999.1910+222.g29b1743

Build tag

jenkins-universe-ffsecurity-2.0.0-40

Build date

2019-04-04 01:00:16

Confidence Threshold

FindFace Security verifies that a detected face and some face from the dossiers belong to the same person (i. e. the faces match), based on the pre-defined similarity threshold. The default threshold is set to 0.75 which can be considered as optimum. If necessary, you can change the threshold.

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

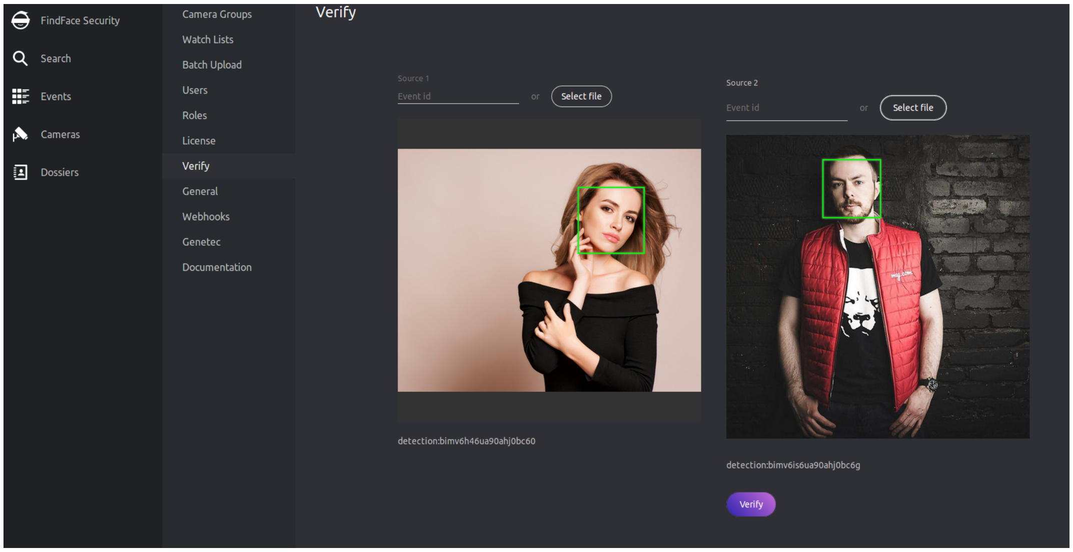
Automatic Events Cleanup

Use the same tab to schedule purging old events from the database on a regular basis.

1.4.5 Compare Faces

FindFace Security allows you to compare 2 faces. Do the following:

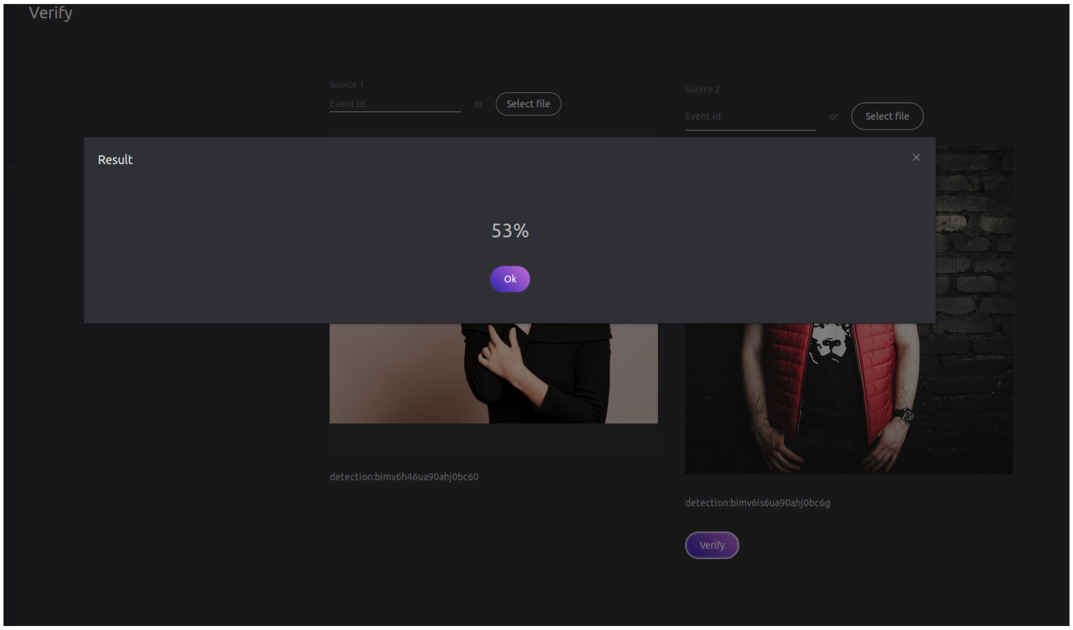
1. Navigate to the *Preferences* tab. Click *Verify*.



2. Specify the IDs of events that feature the faces you want to compare, and/or upload photos with the faces.

Tip: You can find event IDs on the *Events* tab.

3. Click *Verify*. You will see the probability of the faces belonging to the same person appear.



1.5 Advanced Functionality

1.5.1 Allocate `findface-video-worker` to Camera Group

In distributed architectures, it is often necessary that video streams from a group of cameras be processed *in situ*, without being redistributed across remote `findface-video-worker` instances by the main server. Among typical use cases are hotel chains, chain stores, several security checkpoints in the same building, etc. In this case, simply allocate the local `findface-video-worker` to the camera group.

Do the following:

1. Navigate to the *Preferences* tab. Click *Camera Groups*.
2. Open the camera group settings.
3. In the *Labels*, create or select one or several allocation labels. Save changes.
4. Open the `findface-video-worker` configuration file and specify the allocation labels in the following format: `label_name=true` (label `terminal_1` in the example below).

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

   wrk-labels=terminal_1=true
   ```

5. Restart `findface-video-worker`.

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

Note: If a camera is assigned an allocation label, its video stream can be processed by a `findface-video-worker` instance with the same label, as well as by all unlabeled `findface-video-worker` instances.

Warning: If a labeled camera is processed by an unlabeled `findface-video-worker` instance and a free similar-labeled instance appears, the camera won't automatically switch to the latter. To switch the camera, restart the similar-labeled `findface-video-worker` instance.

1.5.2 Console Bulk Photo Upload

To bulk-upload photos to the dossier database, you can use the **`findface-security-uploader`** utility from the FindFace Security package (in addition to the web interface upload functionality). Use this utility when you need to upload a large number of photos (more than 10,000).

Tip: To view the **`findface-security-uploader`** help, execute:

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

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, use either a `script` or the `find` command.

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 from a specified directory (`/home/user/25_celeb/` in the example below), run the following command:

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

The `find` usage example:

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

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

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

The `add` 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 using `run`.

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

The `run` 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.
- `--user`: login.
- `--password`: password.
- `--dossier-lists`: comma-separated list of the watch lists id's.
- `--failed`: should an error occur during the job file processing, correct the mistake and try again with this option.

1.5.3 Deduplicate Events

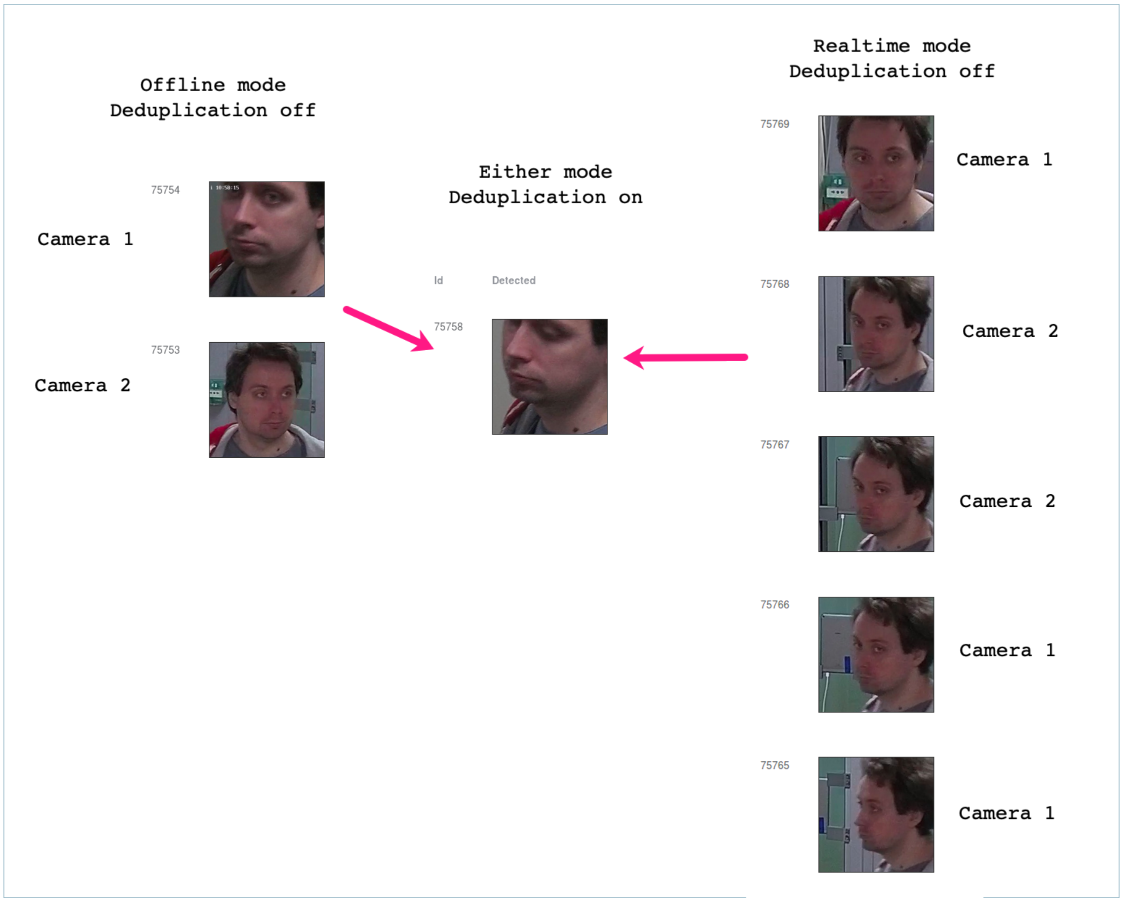
In this section:

If observation scenes of cameras within one group overlap, consider to enable Deduplication. This feature allows you to exclude coinciding facial recognition events among cameras belonging to the same group.

Warning: Use deduplication with extreme caution, as if cameras within a group observe different scenes, some faces may be skipped.

How It Works

The deduplication algorithm's infographics are shown in the diagram below:



1. If the video face detector is working in the offline mode without deduplication, the server receives one best face snapshot per camera. We recommend to use this mode if cameras in the same group observe different scenes.

2. If the video face detector is working in the online mode without deduplication, the server receives several images from each camera of a group. This mode is the most storage intensive. In the case of large number of visitors, security operators may also experience difficulties dealing with a large number of identical face recognition events.
3. With enabled deduplication, the server receives only one face snapshot per group, the best one in the current tracking session whatever the video face detector mode. Use deduplication only if the observation scenes of cameras within a group overlap.

Enable Deduplication

To enable event deduplication, do the following:

1. Navigate to the *Preferences* tab. Click *Camera Groups*.
2. Open the camera group settings.
3. Check *Deduplicate Events* and specify the deduplication interval in seconds (interval between 2 consecutive checks for event uniqueness).

1.5.4 Direct API Requests to Tarantool

You can use HTTP API to extract data directly from the Tarantool Database.

In this section:

General Information

API requests to Tarantool are to be sent to `http://<tarantool_host_ip:port>`.

Tip: The port for API requests can be found in the `FindFace.start` section of the Tarantool configuration file:

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

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

Note: In the case of the standalone deployment, you can access Tarantool by default only locally (127.0.0.1). If you want to access Tarantool remotely, *alter* the Tarantool configuration file.

API requests to Tarantool may contain the following parameters in path segments:

- `:ver`: API version (v2 at the moment).
- `:name`: gallery name.

Tip: To list gallery names on a shard, type in the following command in the address bar of your browser:

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

The same command on the console is as such:

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

You can also list gallery names by using a direct request to Tarantool:

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

Note that if there is a large number of shards in the system, chances are that a randomly taken shard does not contain all the existing galleries. In this case, just list galleries on several shards.

Add Face

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

Parameters in body:

JSON-encoded array of faces with the following fields:

- `"id"`: face id in the gallery, uint64_t,
- `"facen"`: raw feature vector, base64,
- `"meta"`: face metadata, dictionary.

Returns:

- HTTP 200 and empty body on success.
- HTTP 404 if a gallery with the given name doesn't exist.
- HTTP with a status other than 200 and error description in the body on failure.

Example

Request

```
curl -D - -s 'http://localhost:8001/v2/faces/add/testgal' --data '
[
  {
    "id": 9223372036854776000,
    "facen": "qgI3vZRv/z...NpO9MdHavW1WuT0=",
    "meta": {
"cam_id": "223900",
"person_name": "Mary Ostin",

    }
  }
]
```

Response

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

Remove Face

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

Parameters in body:

JSON-encoded array of face ids to be removed

Returns:

- HTTP 200 and empty body on success.
- HTTP 404 if a face with the given id is not found in the gallery.
- HTTP with a status other than 200 and error description in the body on failure.

Example

Request

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

Response

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

Face Search

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

Parameters in body:

JSON-encoded search request with the following fields:

- `limit`: maximum number of faces in the response.
- `sort`: sorting order. Pass one of the following values: `id`: increasing order by id, `-id`: decreasing order by id, `-score`: decreasing order by face similarity (only if you search for faces with similar feature vectors).
- `filter` (filters): * `facen`: (optional) search for faces with similar feature vectors. Pass a dictionary with the following fields: `data`: raw feature vector, base64; `score`: range of similarity between faces [threshold similarity; 1], where `1` is 100% match. * `id` and `meta/<meta_key>`: search by face id and metastring content. To set this filter, use the following operators:
  - `range`: range of values, only for numbers.
  - `set`: id or metastring must contain at least one value from a given set, for numbers and strings.
  - `subset`: id or metastring must include all values from a given subset, for numbers and strings.
  - `like`: by analogy with `like` in SQL requests: only ‘aa%’, ‘aa%’, and ‘%aa%’ are supported. Only for strings and set[string]. In the case of set[string], the filter will return result if at least one value meets the filter condition.
  - `ilike`: by analogy with `like` but case-insensitive, only for strings and set[string].

Returns:

- JSON-encoded array with faces on success. The value in the `X-search-stat` header indicates whether the fast index was used for the search: `with_index` or `without_index`.

  Note: Fast index is not used in API v2.

- HTTP with a status other than 200 and error description in the body on failure.

Example

Request

```
curl -D - -s 'http://localhost:8001/v2/testgal/search' --data '
{
        "limit": 2,
    "sort": {
                "score": -1
        },
        "filter": {
                "facen": {
                        "data": "qgI3vZRv/z0BQTk9rcirOyZrNpO9MdHavW1WuT0=",
                        "score": [0.75, 1]
                },
                "id": {
                        "range": [9223372036854000000, 9223372036854999000]
                },
                "meta": {
                        "person_id": {
                                "range": [444, 999]
                        },
                        "cam_id": {
                                "set": ["12767", "8632", "23989"]
                        }
                }
        }
}'
```

Response

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

{
  "results": [
    {
      "facen": " qgI3vZRv/z0BQTk9rcirOyZrNpO9MdHavW1WuT0=",
      "meta": {
        "timestamp": 0,
        "photo_hash": "",
        "person_id": 777,
        "cam_id": "8632"
      },
      "score": 0.9964,
      "id": 9223372036854776000
    }
  ]
}
```

Edit Face Metadata and Feature Vector

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

Parameters in body:

JSON-encoded array with faces with the following fields:

- `"id"`: face id, uint64_t.
- `"facen"`: (optional) new feature vector, base64. If omitted or passed as `null`, the relevant field in the database won't be updated.
- `"meta"`: dictionary with metadata to be updated. If some metastring is omitted or passed as `null`, the relevant field in the database won't be updated.

Returns:

- HTTP 200 and dictionary with all face parameters, including not updated, on success.
- HTTP 404 and error description if a face with the given id doesn't exist.
- HTTP with a status other than 200 and error description in the body on failure.

Example

Request

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

Response

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

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

List Galleries

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

Returns:

JSON-encoded array with galleries with the following fields: `name`: gallery name, `faces`: number of faces in a gallery.

Example

Request

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

Response

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

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

Get Gallery Info

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

Returns:

- HTTP 200 and dictionary with gallery parameters on success.
- HTTP 404 and error description if a gallery with the given name doesn't exist.
- HTTP with a status other than 200 and error description in the body on failure.

Example

Request

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

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

{"faces":2}
```

Create Gallery

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

Returns:

- HTTP 200 and empty body on success.
- with a status other than 200 and error description in the body on failure.

Example

Request

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

Response

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

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

Remove Gallery

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

Returns:

- HTTP 200 and empty on success.
- with a status other than 200 and error description in the body on failure.

Example

Request

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

Response

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

1.6 Maintenance and Troubleshooting

1.6.1 Update FindFace Security from 1.x to 2.x

To update FindFace Security from 1.x to 2.x, do the following:

1. Install a new version according to your architecture outline, following instructions in *Deploy FindFace Security*.
2. *Create* the following galleries in the Tarantool-based biometric database:
   - `ffsec_dossier_face`: biometric samples extracted from dossier photos.
   - `ffsec_events`: biometric samples extracted from faces in video.
   - `ffsec_monitoring`: biometrics samples from all dossiers under watch (active).

   ```
   curl -i -X POST 'http://127.0.0.1:18411/v2/galleries/ffsec_dossier_face'
   curl -i -X POST 'http://127.0.0.1:18411/v2/galleries/ffsec_events'
   curl -i -X POST 'http://127.0.0.1:18411/v2/galleries/ffsec_monitoring'
   ```

3. Migrate data drom PostgreSQL to Tarantool.

   ```
   sudo findface-security tnt_migrate
   ```

   Note: To purge PostgreSQL after migration is completed, execute the command with the option `--purge-sql`.

   ```
   sudo findface-security tnt_migrate --purge-sql
   ```

   Note: It is absolutely data-safe to interrupt the migration process and resume it later.

1.6.2 Migrate to Different Facen Model

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

Sometimes you have to migrate your face biometric data (facens) to another facen model. This usually happens when you decide to update to the latest version of the product.

To migrate to a different facen model, use the `sf-api-migrate` utility. To pass migration settings, launch it with the `-config` option and provide a configuration file shown in the example below.

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

Example of the configuration file:

```
extraction-api:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 0s
```

(continues on next page)

(continued from previous page)

```
  extraction-api: http://127.0.0.1:18666
storage-api-from: #
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  max-idle-conns-per-host: 20
  shards:
    - master: http://127.0.0.1:8001/v2/
      slave: ""
storage-api-to:
  timeouts:
    connect: 5s
    response_header: 30s
    overall: 35s
    idle_connection: 10s
  max-idle-conns-per-host: 20
  shards:
    - master: http://127.0.0.1:8002/v2/
      slave: ""
workers_num: 100
faces_limit: 1000
extraction_batch_size: 8
normalized_storage:
   type: webdav
  enabled: True
  webdav:
    upload-url: http://127.0.0.1:3333/uploads/
  s3:
    endpoint: 172.20.77.75:9000
    bucket-name: sf-api-normalized
    access-key: W0G6EQT6MC3BZC8136DW
    secret-access-key: XnottrdxRFp70wfEGdkvKgkzKZ3mEa2Y9bYmob4I
    secure: False
    region: ""
    operation-timeout: 10
    public-url: 123```
```

Parameter	Description
`extraction-api -> extraction-api`	`findface-extraction-api` with a new facen model in its configuration file.
`storage-api-from`	Previous facen storage
`storage-api-to`	Storage for re-generated facens
`normalized_storage -> upload-url`	Storage of normalized face images.

1.6.3 Remove FindFace Security Instance

You can automatically remove FindFace Security along with the database by using the `ffsec_1x-2.0_uninstall.sh` script. The FindFace Security configuration files and database will be backed up.

Do the following:

1. Download the `ffsec_1x-2.0_uninstall.sh` script to some directory on a designated host (for example,

to /home/username/).

2. From this directory, make the script executable.

```
chmod +x ffsec_1x-2.0_uninstall.sh
```

3. Run the script.

```
sudo ./ffsec_1x-2.0_uninstall.sh
```

4. Answer **all** to completely remove FindFace Security along with the database.

1.6.4 Checking Component Status

Check the status of components once you have encountered a system problem.

Component	Command to view service status
findface-extraction-api	sudo systemctl status findface-extraction-api.service
findface-sf-api	sudo systemctl status findface-sf-api.service
findface-tarantool-server	sudo systemctl status tarantool@FindFace.service
findface-video-manager	sudo systemctl status findface-video-manager.service
findface-video-worker	sudo systemctl status findface-video-worker.service
findface-ntls	sudo systemctl status findface-ntls
findface-security-worker	sudo systemctl status findface-security-worker
findface-security-proto	sudo systemctl status findface-security-proto
findface-security-monitor-updater	sudo systemctl status findface-security-monitor-updater
findface-security-webhook-updater	sudo systemctl status findface-security-webhook-updater
etcd	sudo systemctl status etcd.service
NginX	sudo systemctl status nginx.service

1.6.5 Analyze Log Files

Log files provide a complete record of each FindFace Enterprise Server component activity. Consulting logs is one of the first things you should do to identify a cause for any system problem.

Component	Command to view log
findface-extraction-api	sudo tail -f /var/log/syslog | grep extraction-api
findface-sf-api	sudo tail -f /var/log/syslog | grep sf-api
findface-tarantool-server	sudo tail -f /var/log/tarantool/FindFace.log
findface-video-manager	sudo tail -f /var/log/syslog | grep video-manager
findface-video-worker	sudo tail -f /var/log/syslog | grep video-worker
findface-security services	sudo tail -f /var/log/syslog | grep findface-security
findface-ntls	sudo tail -f /var/log/syslog | grep ntls
findface-security-worker	sudo tail -f /var/log/syslog | grep security-worker
findface-security-proto	sudo tail -f /var/log/syslog | grep security-proto
findface-security-monitor-updater	sudo tail -f /var/log/syslog | grep security-monitor-updater
findface-security-webhook-updater	sudo tail -f /var/log/syslog | grep security-webhook-updater
etcd	sudo tail -f /var/log/syslog | grep etcd

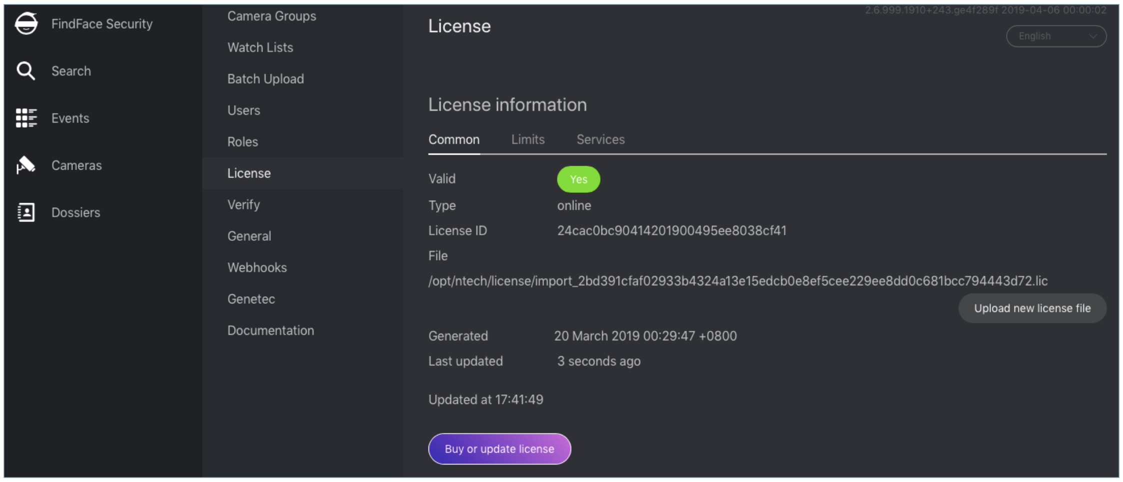
To view log, you can also use the journalctl -u <component> command, for example:

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

1.6.6 Licensing

View and Update License

To view your current licensing information or upload a new license file, navigate to *Preferences -> License*.



License

License information

Valid Yes
Type online
License ID 24cac0bc90414201900495ee8038cf41
File
/opt/ntech/license/import_2bd391cfaf02933b4324a13e15edcb0e8ef5cee229ee8dd0c681bcc794443d72.lic

Generated 20 March 2019 00:29:47 +0800
Last updated 3 seconds ago

Updated at 17:41:49

Troubleshoot Licensing and `findface-ntls`

When troubleshooting licensing and `findface-ntls` (see *Provide 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 info@ntechlab.com.

To retrieve the FindFace Enterprise Server *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": false`: connection with the licensing source was never established.

```
curl http://localhost:3185/license.json -s | jq
{
  "name": "NTLS",
  "time": 1520844897,
  "type": "offline (extended)",
  "license_id": "001278983",
  "generated": 487568400,
  "last_updated": 4,
  "valid": {
    "value": true,
    "description": ""
  },
  "source": "/ntech/license/001278983.lic",
  "limits": [
    {
      "type": "time",
      "name": "end",
      "value": 25343
    },
    {
      "type": "number",
      "name": "faces",
      "value": 90071,
      "current": 230258
    },
    {
      "type": "number",
      "name": "cameras",
      "value": 9007,
      "current": 3
    },
    {
      "type": "number",
      "name": "extraction_api",
      "value": 900,
      "current": 8
    },
    {
      "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
    }
```

(continues on next page)

(continued from previous page)

```
  ],
  "services": [
    {
      "name": "fkvideo-detector",
      "ip": "127.0.0.1:58970"
    },
    {
      "name": "FindFace-tarantool",
      "ip": "127.0.0.1:58978"
    },
    {
      "name": "findface-extraction-api",
      "ip": "127.0.0.1:52376"
    }
  ]
}
```

1.6.7 Automatic Tarantool Recovery

If your system architecture doesn't imply uninterrupted availability of Tarantool servers, it is recommended to enable automatic database recovery. In this case, each time an error occurs while reading a snapshot or xlog file, Tarantool will skip invalid records, read as much data as possible, and re-build the file.

> **Warning:** The automatic recovery process may result in MongoDB and Tarantool being out of sync.

To enable automatic database recovery, do the following:

1. Open the Tarantool configuration file.

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

2. Uncomment `force_rcovery = true`.

   ```
   box.cfg{

       force_recovery = true,
   }
   ```

1.7 Appendices

1.7.1 Enable Data Encryption

To ensure data security, it is recommended to enable SSL encryption. Do the following:

1. Under the nginx configuration directory, create a directory that will be used to hold all of the SSL data:

   ```
   sudo mkdir /etc/nginx/ssl
   ```

2. Create the SSL key and certificate files:

```
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. Configure nginx to use SSL. Open the nginx configuration file. Copy the code from the example below into the file.

```
sudo vi /etc/nginx/nginx.conf

upstream ffsecurity {
        server 127.0.0.1:8002;
}

# redirect from http to https version of the site
server {
        listen 80;
        server_name domain.ru www.domain.ru;
        rewrite ^(.*) https://domain.ru$1 permanent;
        access_log off;
}

server {
        listen 443 ssl;

        ssl_certificate     /etc/nginx/ssl/domain.pem;
        ssl_certificate_key /etc/nginx/ssl/domain.key;

        root /var/lib/ffsecurity;

        autoindex off;

        server_name domain.ru;

        location @ffsec {
                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 http://ffsecurity;
        }

         location /static/ {
        }
        location /uploads/ {
                add_header 'Access-Control-Allow-Origin' '*';
                add_header 'Access-Control-Allow-Methods' 'GET';
                add_header 'Access-Control-Allow-Headers' 'DNT,User-Agent,X-
↪Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range,Authorization
↪';
                add_header 'Access-Control-Expose-Headers' 'Content-Length,
↪Content-Range';
```

(continues on next page)

(continued from previous page)

```
                add_header 'Access-Control-Max-Age' 2592000;
        }
        location /ui-static/ {
                alias /usr/share/ffsecurity-ui/ui-static/;
        }
        location /doc/ {
                alias /opt/ffsecurity/doc/;
        }
        location / {
                try_files $uri $uri/ @ffsec;
                client_max_body_size 100m;
                alias /usr/share/ffsecurity-ui/;

        }
}
```

4. Restart nginx.

```
sudo service nginx restart
```

5. Edit the `ffsecurity` configuration file. In the `EXTERNAL_ADDRESS` parameter, substitute the `http://` prefix with `https://`.

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

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

6. If there are running video-worker services in the system, you need to either recreate cameras in the web interface, or change the `router_url` parameter in relevant video processing jobs, substituting the `http://` prefix with `https://`. This can be done with the following command:

```
curl -s localhost:18810/jobs | jq -r '.[]["id"]' | xargs -I {} curl -X PATCH -d '{
↪"router_url": "https://domain.ru/video-detector/frame"}' http://localhost:18810/
↪job/{}
```

1.7.2 Components in Depth

`findface-extraction-api`

The `findface-extraction-api` service uses neural networks to detect a face in an image, extract face biometric data (feature vector), and recognize gender, age, emotions, and other features.

It interfaces with the `findface-sf-api` service as follows:

- Gets original images with faces and normalized face images.
- Returns the coordinates of the face bounding box, and (optionally) feature vector, gender, age and emotions data, should these data be requested by `findface-sf-api`.

Functionality:

- face detection in an original image (with return of the bbox coordinates),
- face normalization,
- feature vector extraction from a normalized image,
- face feature recognition (gender, age, emotions, beard, glasses3, etc.).

The findface-extraction-api service can be based on CPU (installed from the findface-extraction-api package) or GPU (installed from the findface-extraction-api-gpu package). For both CPU- and GPU-accelerated services, configuration is done through the /etc/findface-extraction-api.ini configuration file. Its content varies subject to the acceleration type.

CPU-service configuration file:

```
allow_cors: false
detector_instances: 0
dlib:
  model: /usr/share/findface-data/normalizer.dat
  options:
    adjust_threshold: 0
    upsample_times: 1
extractors:
  instances: 1
  max_batch_size: 16
  models:
    age: ''
    emotions: ''
    face: face/elderberry_576.cpu.fnk
    gender: ''
  models_root: /usr/share/findface-data/models
fetch:
  enabled: true
  size_limit: 10485760
license_ntls_server: 127.0.0.1:3133
listen: 127.0.0.1:18666
max_dimension: 6000
nnd:
  model: /usr/share/nnd/nnd.dat
  options:
    max_face_size: .inf
    min_face_size: 30
    o_net_thresh: 0.9
    p_net_max_results: 0
    p_net_thresh: 0.5
    r_net_thresh: 0.5
    scale_factor: 0.79
  quality_estimator: true
  quality_estimator_model: /usr/share/nnd/quality_estimator_v2.dat
ticker_interval: 5000
```

GPU-service configuration file:

```
listen: :18666
dlib:
  model: /usr/share/findface-data/normalizer.dat
  options:
    adjust_threshold: 0
    upsample_times: 1
nnd:
  model: /usr/share/nnd/nnd.dat
  quality_estimator: false
```

(continues on next page)

(continued from previous page)

```
  quality_estimator_model: /usr/share/nnd/quality_estimator_v2.dat
  options:
    min_face_size: 30
    max_face_size: .inf
    scale_factor: 0.79
    p_net_thresh: 0.5
    r_net_thresh: 0.5
    o_net_thresh: 0.9
    p_net_max_results: 0
detector_instances: 0
extractors:
  models_root: /usr/share/findface-data/models
  max_batch_size: 16
  instances: 0
  models:
    age: faceattr/age.v1.cpu.fnk
    emotions: faceattr/emotions.v1.cpu.fnk
    face: face/elderberry_576.cpu.fnk
    gender: faceattr/gender.v2.cpu.fnk
license_ntls_server: 127.0.0.1:3133
fetch:
  enabled: true
  size_limit: 10485760
max_dimension: 6000
allow_cors: false
ticker_interval: 5000
```

When configuring `findface-extraction-api` (on CPU or GPU), refer to the following parameters:

Parameter	Description
`nnd -> quality_estimator`	Enables face quality estimation. In this case, `findface-extraction-api` returns a face quality score in the `detection_score` field. Interpret the quality score further in analytics. Upright faces in frontal position are considered the best quality. They result in values around 0, mostly negative (such as -0.00067401276, for example). Inverted faces and large face angles are estimated with negative values some -5 and less.
`nnd -> min_face_size`	The minimum size of a face (bbox) guaranteed to be detected. The larger the value, the less resources required for face detection.
`nnd -> max_face_size`	The minimum size of a face (bbox) guaranteed to be detected.
`license_ntls_server`	The ntls license server IP address and port.

You will also have to enable recognition models for face features such as gender, age, emotions, glasses3, and/or beard, subject to your needs. Be sure to choose 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.

```
models:
  age: faceattr/age.v1.cpu.fnk
  emotions: faceattr/emotions.v1.cpu.fnk
  face: face/elderberry_576.cpu.fnk
  gender: faceattr/gender.v2.cpu.fnk
  beard: faceattr/beard.v0.cpu.fnk
```

The following models are available:

Face feature	Acceleration	Configuration file parameter
face (biometry)	CPU	`face: face/elderberry_576.cpu.fnk`
	GPU	`face: face/elderberry_576.gpu.fnk`
age	CPU	`age: faceattr/age.v1.cpu.fnk`
	GPU	`age: faceattr/age.v1.gpu.fnk`
gender	CPU	`gender: faceattr/gender.v2.cpu.fnk`
	GPU	`gender: faceattr/gender.v2.gpu.fnk`
emotions	CPU	`emotions: faceattr/emotions.v1.cpu.fnk`
	GPU	`emotions: faceattr/emotions.v1.gpu.fnk`
glasses3	CPU	`glasses3: faceattr/glasses3.v0.cpu.fnk`
	GPU	`glasses3: faceattr/glasses3.v0.gpu.fnk`
beard	CPU	`beard: faceattr/beard.v0.cpu.fnk`
	GPU	`beard: faceattr/beard.v0.gpu.fnk`

Tip: To disable a recognition model, simply pass an empty value to a relevant parameter. Do not remove the parameter itself as in this case the system will be searching for the default model.

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

`findface-sf-api`

The `findface-sf-api` service implements HTTP API for the FindFace core main functionality such as face detection and face recognition (the mentioned functions themselves are provided by *findface-extraction-api*‘). It interfaces with the biometric database powered by Tarantool via the `findface-tarantool-server` service, as well as with `findface-extraction-api` (provides face detection and face recognition) and `findface-upload` (provides a storage for original images and FindFace core artifacts).

To detect a face in an image, you need to send the image in an API request to `findface-sf-api`. The `findface-sf-api` will then redirect the request to `findface-extraction-api` for face detection and recognition.

If there is a configured video face detection module in the system (like in FindFace Security), `findface-sf-api` also interfaces with the `findface-facerouter` service. It receives data of detected in video faces along with processing directives from `findface-facerouter`, and then executes the received directives, for example, saves faces into a specific database gallery.

Note: In FindFace Security, `findface-facerouter` functions are performed by `ffsecurity`.

Functionality:

- HTTP API implementation (face detection and face recognition methods, performed via `findface-extraction-api`).
- saving face data to the biometric database (performed via `findface-tarantool-server`),
- saving original images, face thumbnails and normalized face images to an NginX-powered web server (via `findface-upload`).

- provides interaction between all the FindFace core components.

The findface-sf-api configuration is done through the /etc/findface-sf-api.ini configuration file.

```
cache:
  inmemory:
    size: 16384
   memcache:
    nodes:
    - 127.0.0.1:11211
    timeout: 100ms
  redis:
    addr: localhost:6379
    db: 0
    network: tcp
    password: ''
    timeout: 5s
  type: memcache
extraction-api:
  extraction-api: http://127.0.0.1:18666
  timeouts:
    connect: 5s
    idle_connection: 10s
    overall: 35s
    response_header: 30s
limits:
  allow-return-facen: false
  body-image-length: 33554432
  deny-networks: 127.0.0.0/8,192.168.0.0/16,10.0.0.0/8,::1/128,fe00::/8
  url-length: 4096
listen: 127.0.0.1:18411
normalized-storage:
  enabled: true
  s3:
    access-key: ''
    bucket-name: ''
    endpoint: ''
    operation-timeout: 30
    public-url: ''
    region: ''
    secret-access-key: ''
    secure: true
  type: webdav
  webdav:
    timeouts:
      connect: 5s
      idle_connection: 10s
      overall: 35s
      response_header: 30s
    upload-url: http://127.0.0.1:3333/uploads/
storage-api:
  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: ''
  timeouts:
```

(continues on next page)

(continued from previous page)

```
    connect: 5s
    idle_connection: 10s
    overall: 35s
    response_header: 30s
```

When configuring `findface-sf-api`, refer to the following parameters:

Parameter	Description
`extraction-api -> extraction-api`	IP address of the `findface-extraction-api` host.
`storage-api -> shards -> master`	IP address of the `findface-tarantool-server` master shard.
`storage-api -> shards -> slave`	IP address of the `findface-tarantool-server` replica shard.
`limits -> body-image-length`	The maximum size of an image in an API request, bytes.
`upload_url`	WebDAV NginX path to send original images, thumbnails and normalized face images to the `findface-upload` service.

findface-tarantool-server

The `findface-tarantool-server` service provides interaction between the `findface-sf-api` service and the Tarantool-based biometric database in the following way:

Tip: See Tarantool official documentation for details.

- From `findface-sf-api`, `findface-tarantool-server` receives data, such as information of detected in video faces, to write into the biometric database.
- By request from `findface-sf-api`, `findface-tarantool-server` performs database searches and returns search results.

To increase search speed, multiple `findface-tarantool-server` shards can be created on each Tarantool host. Their running concurrently leads to a remarkable increase in performance (70x-100x).

Functionality:

- saving face data to the biometric database,
- database search,
- implementation of direct API requests to the database (see *Direct API Requests to Tarantool*).

The `findface-tarantool-server` configuration is done through the `/etc/tarantool/instances.enabled/<FindFace>.lua` configuration file. In a cluster environment, configuration has to be done for each shard.

```
--
-- Please, read the tarantool cfg doc:
-- https://tarantool.org/doc/reference/configuration/index.html#box-cfg-params
--

box.cfg{
    --port to listen, direct tarantool access
```

(continues on next page)

(continued from previous page)

```
    --Only need for admin operations
    --THIS IS NOT PORT YOU NEED FOR facenapi/sf-api
    listen = '127.0.0.1:33001',

    --Directory to store data
    vinyl_dir = '/opt/ntech/var/lib/tarantool/shard-001',
    work_dir = '/opt/ntech/var/lib/tarantool/shard-001',
    memtx_dir = '/opt/ntech/var/lib/tarantool/shard-001/snapshots',
    wal_dir = '/opt/ntech/var/lib/tarantool/shard-001/xlogs',

    --Maximum mem usage in bytes
    memtx_memory = 200 * 1024 * 1024,

    checkpoint_interval = 3600*4,
    checkpoint_count = 3,

    --uncomment only if you know what you are doing!!! and don't forget box.snapshot()
    -- wal_mode = 'none',

    --if true, tarantool tries to continue if there is an error while reading a␣
↪snapshot/xlog files:  skips invalid records, reads as much data as possible and re-
↪builds the file
    -- force_recovery = true,
}

pcall(function() box.schema.user.grant('guest', 'execute,read,write', 'universe') end)

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

-- host,port to bind for http server
-- this is what you need for facenapi
FindFace = require("FindFace")
FindFace.start("127.0.0.1", 8101, {
    license_ntls_server="127.0.0.1:3133",
    facen_size=576,
    meta_scheme = meta_scheme
})
```

When configuring `findface-tarantool-server`, refer to the following parameters:

Parameter	Description
`memtx_memory`	Maximum RAM that can be used by a Tarantool shard.
`force_recovery`	Enables automatic database recovery. In this case, each time an error occurs while reading a snapshot or xlog file, Tarantool will skip invalid records, read as much data as possible, and re-build the file.
`license_ntls_server`	IP address and port of the `findface-ntls` license server.
`facen_size`	Feature vector size. Before editing this parameter, be sure to consult NTechLab experts.
`meta_scheme`	A database structure to store the face recognition results. The structure is created as a set of fields. Describe each field with the following parameters: `id`: field id; `name`: field name, must be the same as the name of a relevant face parameter; `field_type`: data type; `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...`

Default database structure is passed from `/etc/ffsecurity/tnt_schema.lua` to the `meta_scheme` parameter.

findface-upload

The findface-upload component is an NginX-based web server used as a storage for original images, thumbnails and normalized face images which it receives from the findface-sf-api component.

By default the original images, thumbnails and normalized images are stored at /var/lib/ffupload/uploads/ .

The findface-upload component is automatically configured upon installation. Custom configuration is not supported.

Video face detection: findface-video-manager and findface-video-worker

Note: The findface-video-worker is delivered in a CPU-accelerated (findface-video-worker) and a GPU-accelerated (findface-video-worker-gpu) packages.

In this section:

Functions of findface-video-manager

The findface-video-manager service is the part of the video face detection module that is used for managing the video face detection functionality.

The findface-video-manager service interfaces with findface-video-worker as follows:

- It supplies findface-video-worker with settings and the list of to-be-processed video streams. To do so, it issues a so called job, a video processing task which contains configuration settings and stream data.
- In a distributed system, it distributes video streams (jobs) across vacant findface-video-worker instances.

Note: Configuration settings passed via jobs have priority over the findface-video-manager configuration file.

The findface-video-mananger service functioning requires ETCD, third-party software that implements a distributed key-value store for findface-video-manager. In the FindFace core, ETCD is used as a coordination service, providing the video face detector with fault tolerance.

Functionality:

- allows for configuring video face detection parameters,

- allows for managing the list of to-be-processed video streams,
- implements video face detection management.

Functions of `findface-video-worker`

The `findface-video-worker` (or `findface-video-worker-gpu`) service is the part of the video face detection module, which recognizes faces in video. It can work with both live streams and files, and supports most video formats and codecs that can be decoded by FFmpeg.

The `findface-video-worker` service interfaces with the `findface-video-manager` and `findface-facerouter` services as follows:

- By request, `findface-video-worker` gets a job with settings and the list of to-be-processed video streams from `findface-video-manager`.
- The `findface-video-worker` posts extracted normalized face images, along with the full frames and meta data (such as bbox, camera ID and detection time) to the `findface-facerouter` service for further processing.

Note: In FindFace Security, the `findface-facerouter` functions are performs by `findface-security`.

Functionality:

- detects faces in video,
- extracts normalized face images,
- searches for the best face snapshot,
- snapshot deduplication (only one snapshot per face detection event).

When processing video, `findface-video-worker` consequently uses the following algorithms:

- **`Motion detection`**. Used to reduce resource consumption. Only when the motion detector recognizes motion of certain intensity that the face tracker can be triggered.
- **`Face tracking`**. The face tracker tracks, detects and captures faces in video. It can simultaneously be working with several faces. It also searches for the best face snapshot, using an embedded neural network. After the best face snapshot is found, it is posted to `findface-facerouter`.

The best face snapshot can be found in one of the following modes:

- Real-time
- Offline

Real-Time Mode

In the real-time mode, `findface-video-worker` posts a face immediately after it appears in the camera field of view.

- If `rt-perm=True`, the face tracker searches for the best face snapshot within each time period equal to `rt-delay` and posts it to `findface-facerouter`.
- If `rt-perm=False`, the face tracker searches for the best face snapshot dynamically:
    1. First, the face tracker estimates whether the quality of a face snapshot exceeds a pre-defined threshold value. If so, the snapshot is posted to `findface-facerouter`.

2. The threshold value increases after each post. Each time the face tracker gets a higher quality snapshot of the same face, it is posted.
3. When the face disappears from the camera field of view, the threshold value resets to default.

By default, the real-time mode is disabled (`realtime=false` in the `/etc/findface-video-manager.conf` file).

Offline Mode

The offline mode is less storage intensive than the real-time one as in this mode `findface-video-worker` posts only one snapshot per track, but of the highest quality. In this mode, the face tracker buffers a video stream with a face in it until the face disappears from the camera field of view. Then the face tracker picks up the best face snapshot from the buffered video and posts it to `findface-facerouter`.

By default, the offline mode is enabled (`overall=true` in the `/etc/findface-video-manager.conf` file).

Configure Video Face Detection

The video face detector configuration is done through the following configuration files:

1. The `findface-video-manager` configuration file `/etc/findface-video-manager.conf`:

```
etcd:
  dial_timeout: 3s
  endpoints: 127.0.0.1:2379
exp_backoff:
  enabled: false
  factor: 2
  flush_interval: 2m0s
  max_delay: 1m0s
  min_delay: 1s
job_scheduler_script: ''
kafka:
  enabled: false
  endpoints: 127.0.0.1:9092
listen: 127.0.0.1:18810
master:
  lease_ttl: 10
  self_url: 127.0.0.1:18811
  self_url_http: 127.0.0.1:18811
ntls:
  enabled: false
  update_interval: 1m0s
  url: http://127.0.0.1:3185/
prometheus:
  jobs_processed_duration_buckets:
  - 1
  - 30
  - 60
  - 500
  - 1800
  - 3600
  - 21600
  - .inf
router_url: http://127.0.0.1:18820/v0/frame
```

(continues on next page)

(continued from previous page)

```
rpc:
 heart_beat_timeout: 4s
 listen: 127.0.0.1:18811
stream_settings:
  additional_body: []
  additional_headers: []
  api_ssl_verify: true
  api_timeout: 15000
  det_period: 8
  disable_drops: false
  draw_track: false
  fd_frame_height: -1
  ffmpeg_format: ''
  ffmpeg_params: []
  image_arg: photo
  jpeg_quality: 95
  max_candidates: 0
  max_face_size: 0
  md_scale: 0.3
  md_threshold: 0.002
  min_d_score: -1000
  min_face_size: 0
  min_score: -2
  npersons: 4
  only_norm: false
  overall: true
  parse_sei: false
  post_uniq: true
  realtime: false
  realtime_dly: 500
  realtime_post_perm: false
  roi: ''
  rot: ''
  send_track: 0
  tracker_threads: 4
  uc_max_avg_shift: 10
  uc_max_dup: 3
  uc_max_time_diff: 30
stream_settings_gpu:
  ffmpeg_format: ''
  ffmpeg_params: []
  filter_max_face_size: 8192
  filter_min_face_size: 1
  filter_min_quality: -2
  imotion_threshold: 0
  jpeg_quality: 95
  normalized_only: false
  overall_only: false
  play_speed: -1
  realtime_post_every_interval: false
  realtime_post_interval: 1
  roi: ''
  rot: ''
  router_body: []
  router_headers: []
  router_timeout_ms: 15000
  router_verify_ssl: true
```

(continues on next page)

(continued from previous page)

```
  start_stream_timestamp: 0
  use_stream_timestamp: false
```

When configuring findface-video-manager, refer to the following parameters:

Option	Description
router_url	IP address and port of the findface-facerouter host to receive detected faces from findface-video-worker. In FindFace Security, findface-facerouter functions are performed by findface-security. Default value: http://127.0.0.1:18820/v0/frame.
etcd -> endpoints	IP address and port of the etcd service. Default value: 127.0.0.1:2379.
ntls -> enabled	If true, findface-video-manager will send a job to findface-video-worker only if the total number of processed cameras does not exceed the allowed number of cameras from the license. Default value: false.
ntls -> url	IP address and port of the findface-ntls host. Default value: http://127.0.0.1:3185/.

2. If you opt for the CPU-accelerated package findface-video-worker, use the /etc/video-worker.ini configuration file:

```
ntls-addr=127.0.0.1:3133
mgr-static=127.0.0.1:18811
capacity=10
#mgr-exec=shell command with arguments
```

If you opt for the GPU-accelerated package findface-video-worker-gpu, use the etc/findface-video-worker-gpu.ini configuration file.

```
## cuda device number
device_number = 0

## read streams from file, do not use VideoManager
input =

## exit on first finished job, only when --input specified
exit_on_first_finished = false

## models directory
models_dir = /usr/share/findface-gpudetector/models

## batch size
batch_size = 1

## http server port for metrics, 0=do not start server
metrics_port = 0

## resize scale, 1=do not resize
resize_scale = 1.0

## maximum number of streams
capacity = 100

## videomanager grpc ip:port
```

(continues on next page)

(continued from previous page)

```
#mgr_static = 172.17.47.14:18811
mgr_static = 127.0.0.1:18811

## ntls server ip:port
#ntls_addr = 172.17.47.14:3133
ntls_addr = 127.0.0.1:3133

## debug: save faces to dir
save_dir =

## minimum face size
min_face_size = 60

## preinit detector for specified resolutions: "640x480;1920x1080"
resolutions =

## worker labels: "k=v;group=enter"
labels =

#------------------------------
[video_decoder]
#------------------------------
## decode video on cpu
cpu = false

#------------------------------
[send]
#------------------------------
## posting faces threads
threads = 8

## posting faces maximum queue size
queue_limit = 256

#------------------------------
[tracker]
#------------------------------
## max face miss duration, sec
miss_interval = 1.0

## overlap threshold
overlap_threshold = 0.25

## use timestamps from SEI packet (Netris only)
use_time_from_sei = false
```

Whether you configure `findface-video-worker` or `findface-video-worker-gpu`, refer to the following parameters:

Parameter	Description
ntls-addr	IP address and port of the findface-ntls host.
mgr-static	IP address of the findface-video-manager host to provide findface-video-worker with settings and the list of to-be-processed streams.
capacity	Maximum number of video streams to be processed by findface-video-worker.
mgr-exec	(Optional, instead of the mgr-static parameter) A script to describe dynamic IP address of the findface-video-manager host.

findface-ntls

The findface-ntls service is to be installed on a designated host to verify the FindFace license. For verification purposes, findface-ntls uses one of the following sources:

- Ntech Lab global license center if you opt for the online licensing, direct or via a proxy server.
- USB dongle if you opt for the on-premise licensing.

Use the main web interface to manage findface-ntls:

- view the list of purchased features,
- view license limitations,
- upload a license file,
- view the list of currently active components.

The following components are licensable:

- findface-tarantool-server,
- findface-extraction-api,
- findface-video-manager,
- findface-video-worker.

Important: After connection between findface-ntls and a licensable component, or between findface-ntls and the global license server is broken, you will have 6 hours to restore it before the licensable components will be automatically stopped.

The findface-ntls configuration is done through a configuration file /etc/findface-ntls.cfg.

```
# Listen address of NTLS server where services will connect to.
# The format is IP:PORT
# Use 0.0.0.0:PORT to listen on all interfaces
# This parameter is mandatory and may occur multiple times
# if you need to listen on several specific interfaces or ports.
listen = 127.0.0.1:3133


# Directory with license files.
# NTLS use most recently generated one.
# Note: "recentness" of a license file is detected not by
#       mtime/ctime but from its internal structure.
#
# This parameter is mandatory and must occur exactly once.
```

(continues on next page)

(continued from previous page)

```
license-dir = /opt/ntech/license

# You can specify proxy which NTLS will use to access
# global license server. The syntax is the same that is used by curl.
# Proxy is optional
#proxy = http://192.168.1.1:12345

# This is bind address for NTLS web-interface.
# Note: there're no authorization or access restriction mechanisms
#       in NTLS UI. If you need one, consider using nginx as proxy
#       with .htaccess / ip-based ACLs.
# This parameter may be specified multiple times.
ui = 127.0.0.1:3185
```

When configuring `findface-ntls`, refer to the following parameters:

Parameter	Description
`listen`	IP address from which licensable services access `findface-ntls`. To allow access from any IP address, use `0.0.0.0:3133`.
`license_dir`	Directory to store a license file.
`proxy`	(Optional) IP address and port of your proxy server.
`ui`	IP address from which accessing the `findface-ntls` web interface must originate. To allow access from any remote host, set "0.0.0.0".

ffsecurity

The `ffsecurity` component serves as a gateway to the FindFace core. It provides interaction between the FindFace Core and the web interface, and the system functioning as a whole. The `ffsecurity` component includes the following services:

- `findface-security-proto`: provides HTTP and web socket (along with Django).
- `findface-security-worker`: provides interaction with the other system components.
- `findface-security-monitoring-updater`: updates the `ffsec_monitoring` gallery in the biometric database (see :*galleries*).
- `findface-security-webhook-updater`: pulls webshooks (see *Webhooks*).

The `ffsecurity` component also performs the functions of `findface-facerouter` (part of the FindFace Core), setting processing directives for detected faces. It accepts a face bbox and normalized image along with the original image and other data (for example, the detection date and time) from the `findface-video-worker` service and redirect them to `findface-sf-api` for further processing.

The `ffsecurity` configuration is done through the `/etc/ffsecurity/config.py` configuration file.

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

MEDIA_ROOT="/var/lib/ffsecurity/uploads"
STATIC_ROOT="/var/lib/ffsecurity/static"

EXTERNAL_ADDRESS="http://127.0.0.1:8000"

DEBUG = False

LANGUAGE_CODE = 'en-us'
```

(continues on next page)

(continued from previous page)

```
TIME_ZONE = 'UTC'

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'ffsecurity',
    }
}

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

FFSECURITY = {
    'VIDEO_DETECTOR_TOKEN': 'GOOD_TOKEN',
    'CONFIDENCE_THRESHOLD': 0.75,
    'MINIMUM_DOSSIER_QUALITY': -2,
    'IGNORE_UNMATCHED': False,
    'VIDEO_MANAGER_ADDRESS': 'http://127.0.0.1:18810',
    'EVENTS_MAX_AGE': 30,
    'NTLS_HTTP_URL': 'http://127.0.0.1:3185',
    'ROUTER_URL': 'http://127.0.0.1:8000',
    'MONITORING_UPDATE_INTERVAL': 60,
    'SF_API_ADDRESS': 'http://127.0.0.1:18411',
    'EVENTS_FEATURES': ['gender', 'age', 'beard', 'glasses'],
}

FFSECURITY_UI_CONFIG = {
}

# integration plugins
INSTALLED_APPS.append('ffsecurity_genetec') # remove or comment out this␣
↪line to disable genetec integration```
```

When configuring `ffsecurity`, refer to the following parameters:

Parameter	Description
EXTERNAL_ADDRESS	External IP address or URL that will be used to access the FindFace Security web interface.
VIDEO_DETECTOR_TOKEN	To authorize the video face detection module, come up with a token and specify it here.
VIDEO_MANAGER_ADDRESS	IP address of the findface-video-manager host.
NTLS_HTTP_URL	IP address of the findface-ntls host.
ROUTER_URL	IP address of the ffsecurity host that will receive detected faces from the findface-video-worker instance(s). Specify either external or internal IP address, subject to the network through which findface-video-worker interacts with ffsecurity.
SF_API_ADDRESS	IP address of the findface-sf-api host.
IGNORE_UNMATCHED	Disable logging events for faces which have no match in the dossiers (negative verification result). Set true if the system has to process a large number of faces.
CONFIDENCE_THRESHOLD	Face similarity threshold for verification
MINIMUM_DOSSIER_QUALITY	Minimum quality of a face in a dossier photo. Photos containing faces of worse quality will be rejected when uploading to a dossier. Upright faces in frontal position are considered the best quality. They result in values around 0, mostly negative (such as -0.00067401276, for example). Inverted faces and large face angles are estimated with negative values some -5 and less. By default, 'MINIMUM_DOSSIER_QUALITY': -2 which is the average quality.
EVENTS_FEATURES	If you enabled recognition models in the findface-extraction-api configuration file, list them here.

1.7.3 Installation File

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

Tip: See *Deploy from Console Installer* to learn more about the FindFace Security installer.

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

Here is an example of the installation file:

```
{
  "findface-security.config": {
    "EXTERNAL_ADDRESS": "http://172.20.77.17"
  },
  "product": "security",
  "ext_ip.bind": "0.0.0.0",
  "findface-ntls.config": {
    "NTLS_LISTEN": "127.0.0.1:3133",
    "NTLS_LISTEN_UI": "127.0.0.1:3185",
    "NTLS_LICENSE_DIR": "/opt/ntech/license"
  },
  "components": [
    "findface-data",
    "memcached",
    "etcd",
```

(continues on next page)

(continued from previous page)

```
    "redis",
    "postgresql",
    "findface-ntls",
    "findface-extraction-api",
    "findface-sf-api",
    "findface-upload",
    "findface-video-manager",
    "findface-video-worker",
    "findface-security",
    "findface-tarantool-server"
  ],
  "memcached.config": {
    "max_memory": 1024,
    "listen_host": "127.0.0.1",
    "item_size": 16
  },
  "findface-video-manager.config": {
    "listen": "127.0.0.1:18810",
    "master": {
      "self_url_http": "127.0.0.1:18811",
      "self_url": "127.0.0.1:18811"
    },
    "rpc": {
      "listen": "127.0.0.1:18811"
    },
    "ntls": {
      "url": "http://127.0.0.1:3185/",
      "enabled": false
    }
  },
  "findface-video-worker.variant": "cpu",
  "findface-extraction-api.variant": "cpu",
  "ignore_lowmem": true,
  "findface-video-worker.config": {
    "FKVD_WRK_CAP": "10",
    "FKVD_MGR_ADDR": "127.0.0.1:18811",
    "FKVD_NTLS_ADDR": "127.0.0.1:3133"
  },
  "findface-extraction-api.config": {
    "listen": "127.0.0.1:18666",
    "extractors": {
      "instances": 1,
      "models": {
        "gender": "",
        "face": "face/elderberry_576.cpu.fnk",
        "age": "",
        "emotions": ""
      }
    },
    "nnd": {
      "quality_estimator": true
    },
    "license_ntls_server": "127.0.0.1:3133"
  },
  "ext_ip.advertised": "172.20.77.17",
  "findface-tarantool-server.config": {
    "shard-002": {
```

(continues on next page)

(continued from previous page)

```
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_LISTEN": "127.0.0.1:33002",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-002",
      "TNT_FF_LISTEN_PORT": "8102"
    },
    "shard-001": {
      "TNT_META_SCHEME": "meta_scheme",
      "TNT_LISTEN": "127.0.0.1:33001",
      "TNT_FF_LISTEN_IP": "127.0.0.1",
      "TNT_EXTRA_LUA": "\ndofile(\"/etc/ffsecurity/tnt_schema.lua\")\n",
      "TNT_FF_NTLS": "127.0.0.1:3133",
      "TNT_DATA_DIR": "/opt/ntech/var/lib/tarantool/shard-001",
      "TNT_FF_LISTEN_PORT": "8101"
    }
  },
  "tnt_instances": 2,
  "inter_ip.bind": "127.0.0.1",
  "type": "stand-alone",
  "findface-sf-api.config": {
    "listen": "127.0.0.1:18411",
    "extraction-api": {
      "extraction-api": "http://127.0.0.1:18666"
    },
    "storage-api": {
      "shards": [
        {
          "master": "http://127.0.0.1:8101/v2/",
          "slave": ""
        },
        {
          "master": "http://127.0.0.1:8102/v2/",
          "slave": ""
        }
      ]
    }
  },
  "findface-facerouter.config": {
    "plugin_source": "dir",
    "port": "18820",
    "plugin_dir": "/etc/findface-facerouter-plugins",
    "sfapi_url": "http://127.0.0.1:18411",
    "host": "127.0.0.1"
  },
  "inter_ip.advertised": "127.0.0.1"
}
```

1.7.4 Neural Network Models

Here you can see a summary for neural network models created by our Lab and used in FindFace Security:

Note: The CPU and GPU benchmark setup is the following:

- CPU - Intel® Core™ i7-5930K CPU @ 3.50GHz × 12
- GPU - GeForce GTX 1080

Warning: Strictly not recommended to use `face/elderberry_160` for work.

Model	CPU, FPS	GPU, FPS	Type
face/elderberry_160	14.99	204.98	Face biometrics
face/elderberry_576	2.07	71.14	
faceattr/age.v1	14.99	529.35	Age recognition
faceattr/beard.v0	15.03	532.05	Beard recognition
faceattr/emotions.v1	10.99	235.59	Emotions recognition
faceattr/gender.v2	15.01	523.22	Gender recognition
faceattr/glasses3.v0	15.01	529.64	Glasses recognition

1.7.5 Biometric Database Galleries

There are 3 galleries the Tarantool-based biometric database:

- `ffsec_dossier_face`: biometric samples extracted from dossier photos.
- `ffsec_events`: biometric samples extracted from faces in video.
- `ffsec_monitoring`: biometrics samples from all dossiers under watch (active).

CHAPTER 2

Operator's Guide

2.1 Web Interface

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

Note: Request credentials from administrator.

The web interface has a highly intuitive and handy design and provides the following functionality:

- *Search Databases*.
- *Real-time Face Identification*.
- *Dossier* (only for users with operator privileges).

2.2 Search Databases

FindFace Security allows you to search for faces in the following databases:

- Database of detected faces (the *Events* tab).
- Dossier database (the *Dossiers*). Contains face reference images.

To find a face in a database, navigate to the *Search* tab.

In this chapter:

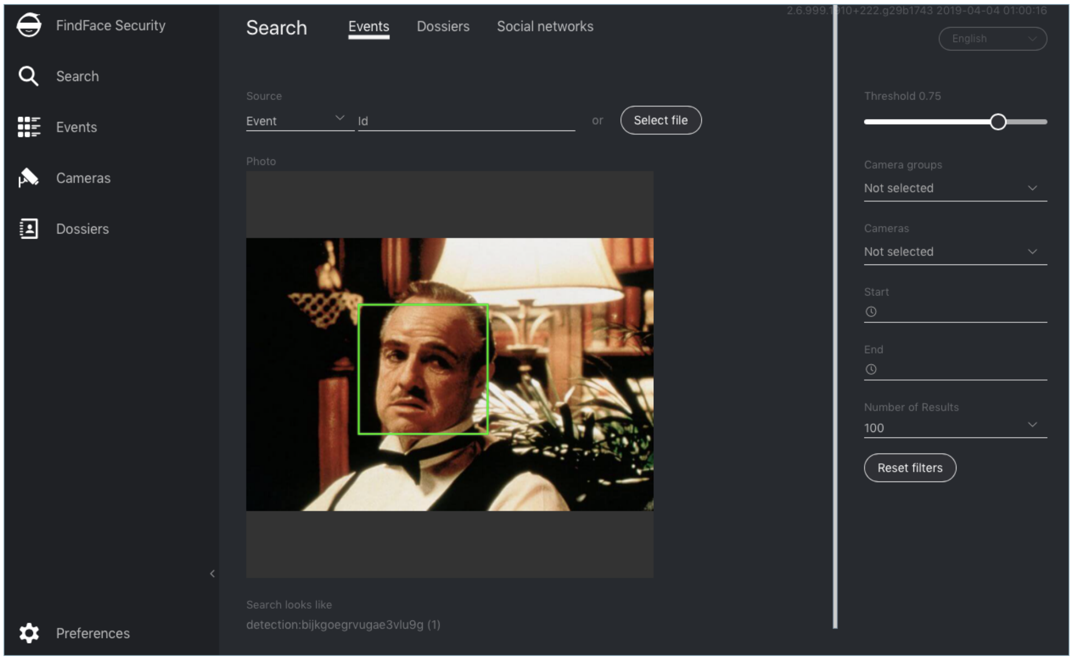
2.2.1 Search for Faces in Event List

FindFace Security allows you to search the database of detected faces.

Note: You can access this database by navigating to the event list (the *Events* tab).

To find a face, do the following:

1. Navigate to the *Search* tab.



2.6.999.1910+222.g29b1743 2019-04-04 01:00:16

Source: Event

Id

or

Photo

Search looks like
detection:bijkgoegrvugae3vlu9g (1)

Threshold 0.75

Camera groups: Not selected

Cameras: Not selected

Start

End

Number of Results: 100

2. Specify a database to search: *Events*.
3. Upload a photo. It will be displayed in the *Photo* area. If there are multiple faces in the image, select the one you want.

   Note: Instead of a photo, you can specify the ID of an event that features the face you want to find.

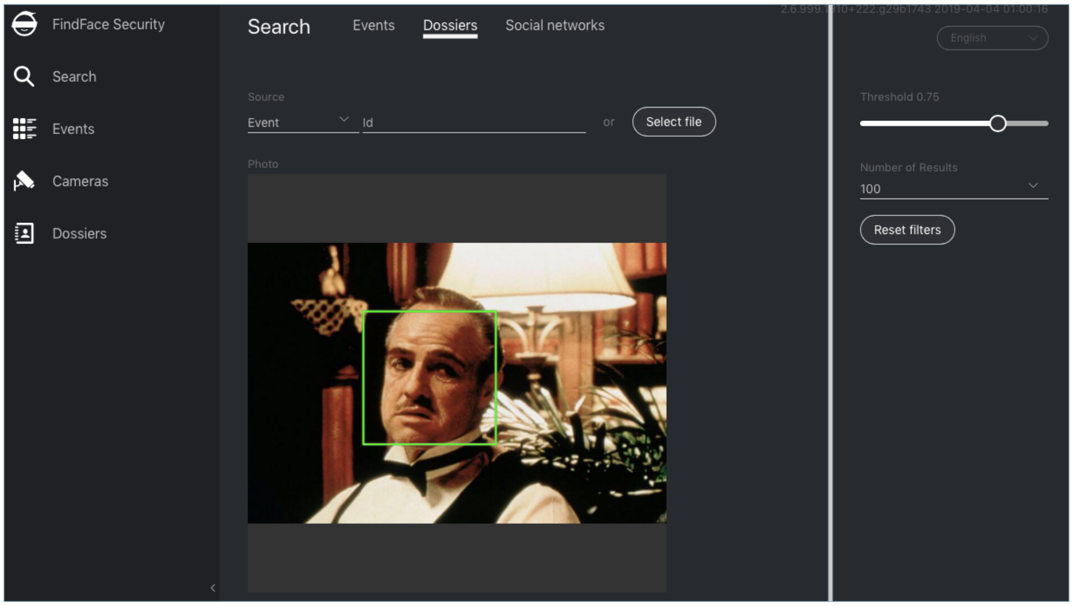
4. By default, the system searches for faces using the identification threshold 0.75. If necessary, set your own value using the *Threshold* filter.
5. (Optional) Specify a group of cameras, camera and a time period within which the event occurred.
6. Specify the maximum number of dossiers in the search results.
7. Click *Search*. You will see the search results appear below. For each face found, the matching confidence level is provided.

2.2.2 Search for Faces in Dossier List

FindFace Security allows you to search the database of dossiers containing face reference images.

To find a face, do the following:

1. Navigate to the *Search* tab.



2.6.999.1910+222.g29b1743 2019-04-04 01:00:16

Source: Event

Id

Photo

Threshold 0.75

Number of Results: 100

2. Specify a database to search: *Dossiers*.
3. Upload a photo. It will be displayed in the *Photo* area. If there are multiple faces in the image, select the one you want.

   Note: Instead of a photo, you can specify the ID of an event that features the face you want to find.

4. By default, the system searches for faces using the identification threshold 0.75. If necessary, set your own value using the *Threshold* filter.
5. Specify the maximum number of dossiers in the search results.
6. Click *Search*. You will see the search results appear below. For each face found, the matching confidence level is provided.

2.3 Real-time Face Identification

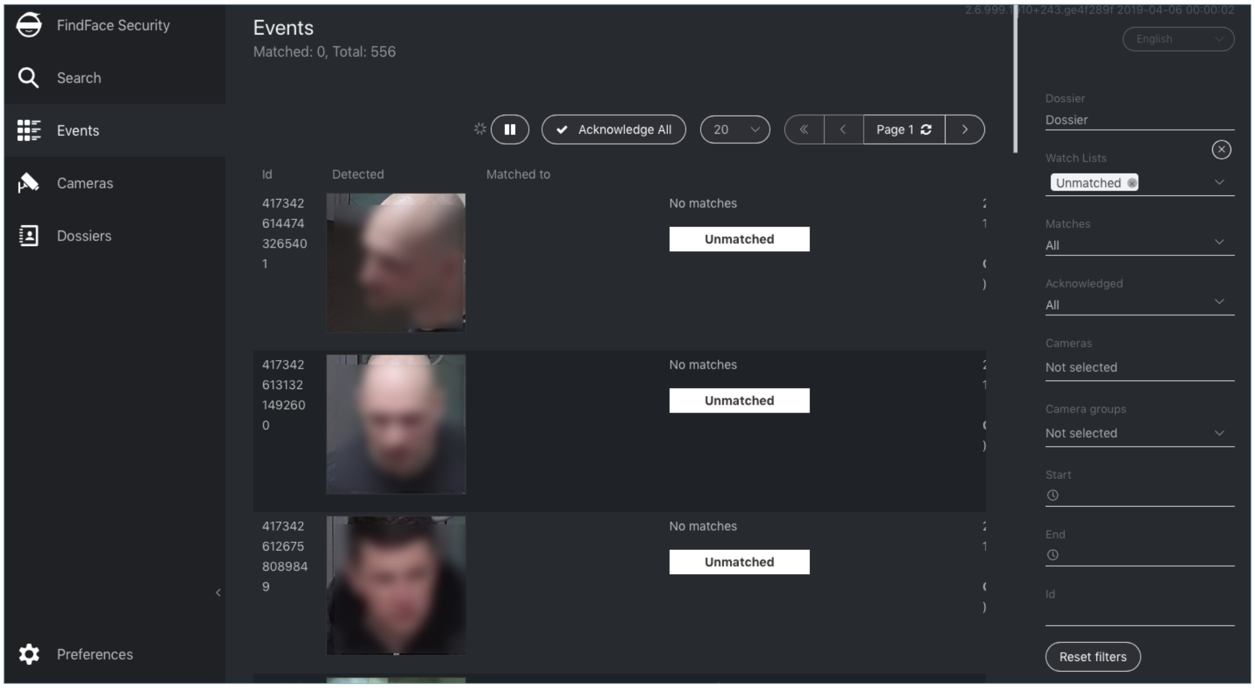
To monitor the real-time face identification, go to the *Events* tab. The system can identify faces in both live video streams and archived videos. Besides monitoring, the *Events* tab also allows you to access the history of identification events.

Tip: Search for faces through the event database and dossier database on the *Search* tab.

In this chapter:

2.3.1 View Identification Events

Once a face detected, you will see a notification in the event list.



Events

Matched: 0, Total: 556

Id	Detected	Matched to
4173426144743265401		No matches Unmatched
4173426131321492600		No matches Unmatched
4173426126758089849		No matches Unmatched

Dossier: Dossier

Watch Lists: Unmatched

Matches: All

Acknowledged: All

Cameras: Not selected

Camera groups: Not selected

Start

End

Id

A notification can feature different pieces of information, depending on whether a detected face has a match in the database:

- Match not found: a normalized face image, detection date and time, the name of a camera group.
- Match found: a normalized face image, the photo from a dossier, the name of a person, similarity between faces, the comment from a dossier, the name of a dossier list, detection date and time, the name of a camera group.

Note: You can configure the system in such a way that you will get notifications only for the faces with a match.

Important: In order to pause the notifications thread, click ⏸ above the list of events.

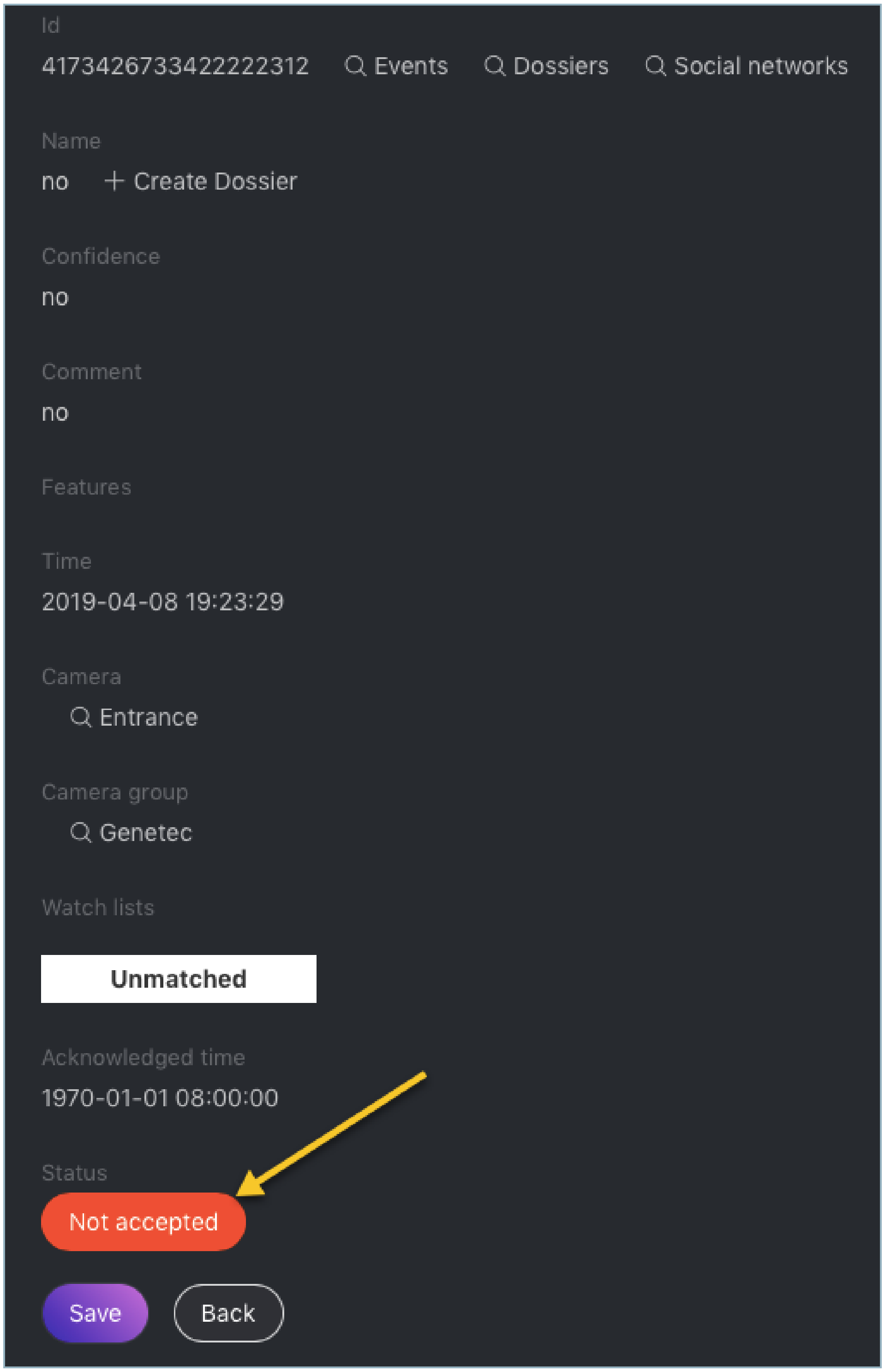
When working with events, the following filters may come in handy:

Dossier: display events only for a selected dossier. *Watch lists*: display events only for a selected dossier category (watch list). *Matches*: display events only with/without matches, or all events. *Acknowledged*: display only acknowledged/unacknowledged events, or all events. *Cameras*: display only events from a selected camera. *Camera groups*: display only events from a selected group of cameras. *Start*, *End*: display only events occurred within a certain time period. *id*: display an event with a given ID.

2.3.2 Event Ticket. Acknowledging Event

In order to navigate to an event ticket from the list of events, click on the face recognition result in a notification (*No matches* or the name of a matching person).

An event ticket contains the same data as a relevant *notification*. It also allows for acknowledging the event. To do so, click *Not accepted* to change the event aknowledgment status. Click *Save*.



Id

4173426733422222312

Name

no

Confidence

no

Comment

no

Features

Time

2019-04-08 19:23:29

Camera

Entrance

Camera group

Genetec

Watch lists

Unmatched

Acknowledged time

1970-01-01 08:00:00

Status

Not accepted

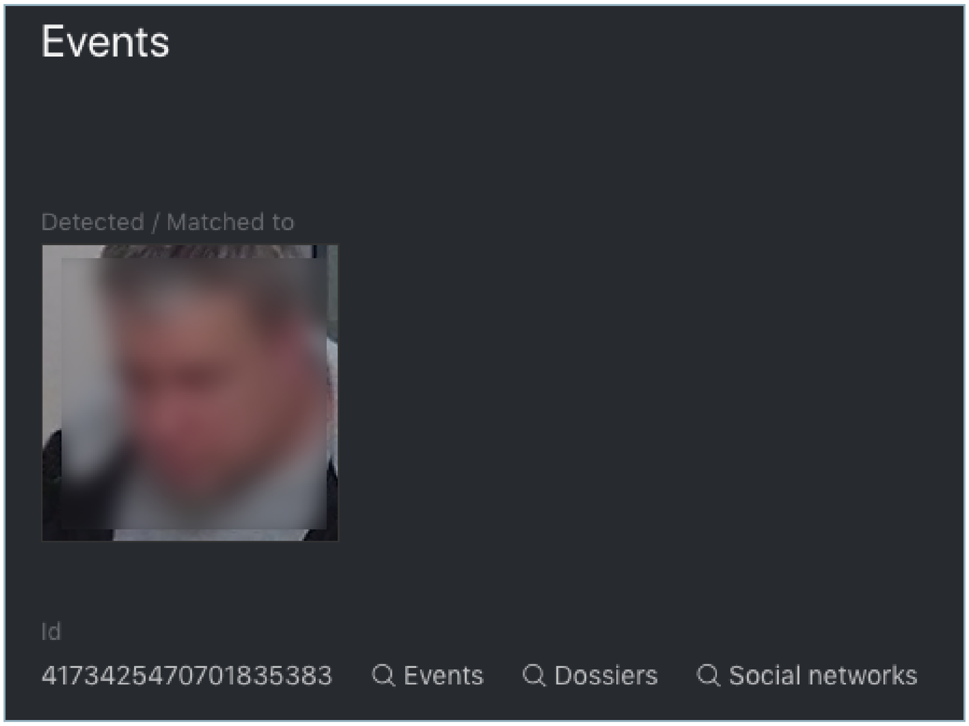
Tip: If a detected face has a match in the dossiers, you can navigate into a relevant one by clicking on the person's name in the event ticket.

Tip: In order to acknowledge all the events, click ✓ above the list of events.

Note: Event acknowledgment can be automated for selected watch lists.

2.3.3 Event Ticket. Face Search

FindFace Security allows you to search faces detected in video in the list of events and dossier database. To navigate from an event ticket to the search tab, click *Events* or *Dossiers* respectively.



Id
4173425470701835383

See also:

- *Search Databases*.

2.4 Dossier

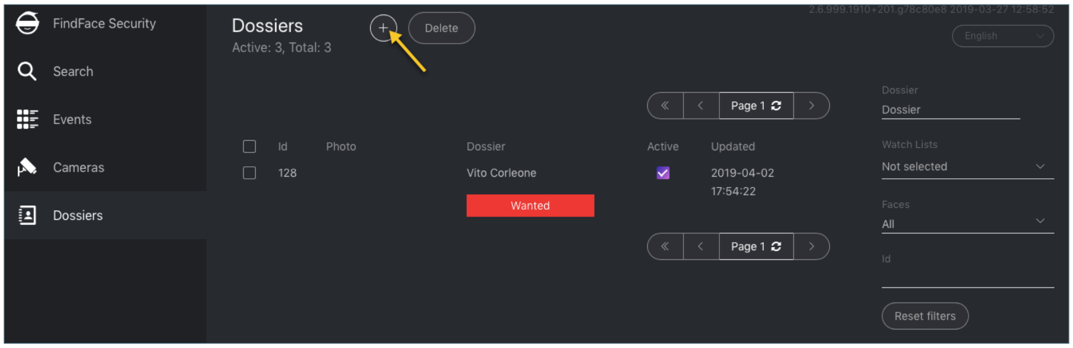
The dossier database contains dossiers on the unwanted persons and VIP guests. A dossier has to contain one or several photos of a person and belong to a certain classification list (watch list).

In this section:

2.4.1 Create Dossier

To create a dossier, do the following:

1. Navigate to the *Dossiers* tab.
2. Click +.



□	Id	Photo	Dossier	Active	Updated
□	128		Vito Corleone Wanted	☑	2019-04-02 17:54:22

Dossier: Dossier

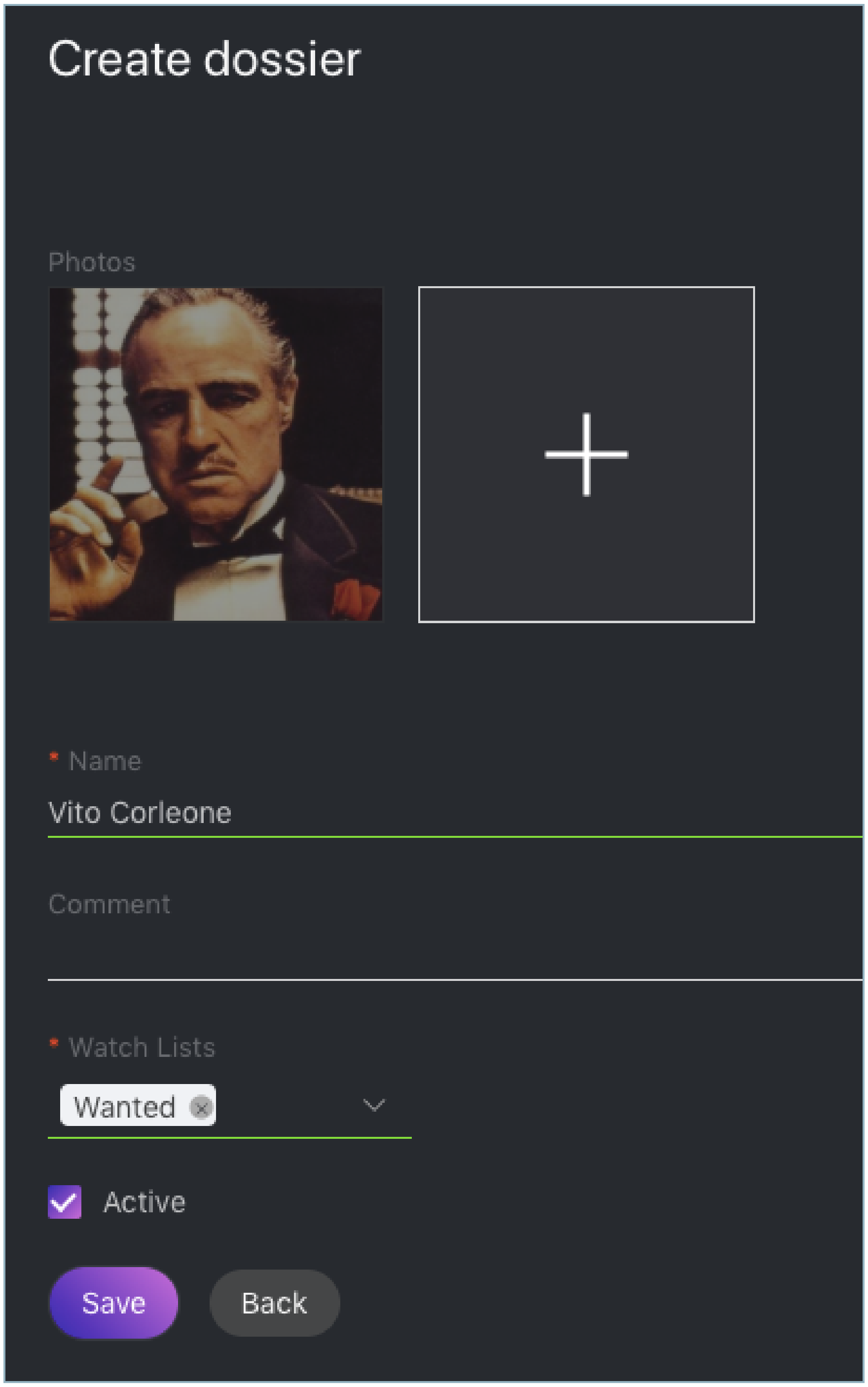
Watch Lists: Not selected

Faces: All

Id:

3. Attach a photo and specify the name of a person. If necessary, add a comment.

> **Important:** A face in the photo must be of high quality, i.e. close to a frontal position. Photos that do not meet the requirement will be rejected with a detailed error description.



Create dossier

Photos

* Name

Vito Corleone

Comment

* Watch Lists

Wanted

☑ Active

4. From the *Watch lists* drop-down menu, select a classification list (or several lists, one by one) for the dossier.

> **Note:** If you cannot find an appropriate watch list for the dossier, *create* a new one, or ask an administrator to do so.

5. Check *Active*. If a dossier is inactive, it is excluded from the *real time face identification*.
6. Click *Save*.

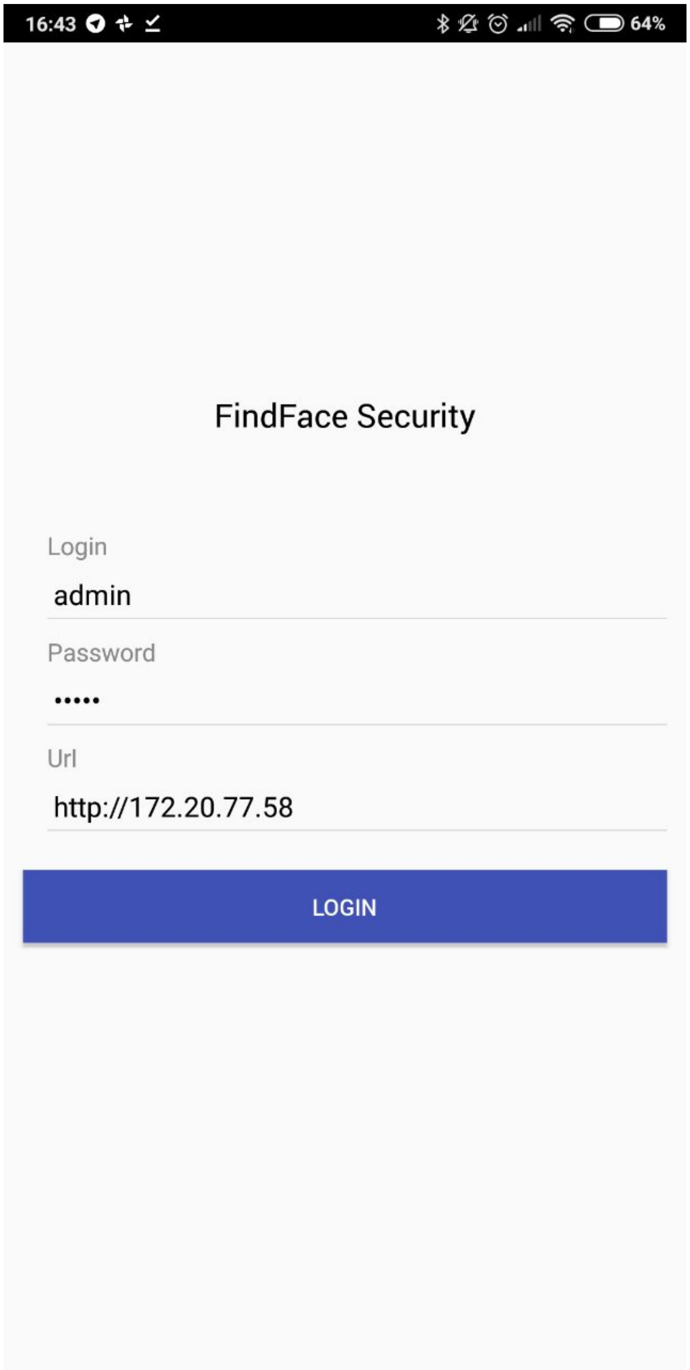
2.4.2 View Dossier

You can find all dossiers created in FindFace Security on the *Dossiers* tab. Use the *Watch lists* filter to filter dossiers by list.

2.5 Mobile App

To interact with FindFace Security on the go, use the mobile app. The FindFace Security app is available on request for Android.

In the app, specify your login and password, as well as the FindFace Security URL address, and log in.



FindFace Security

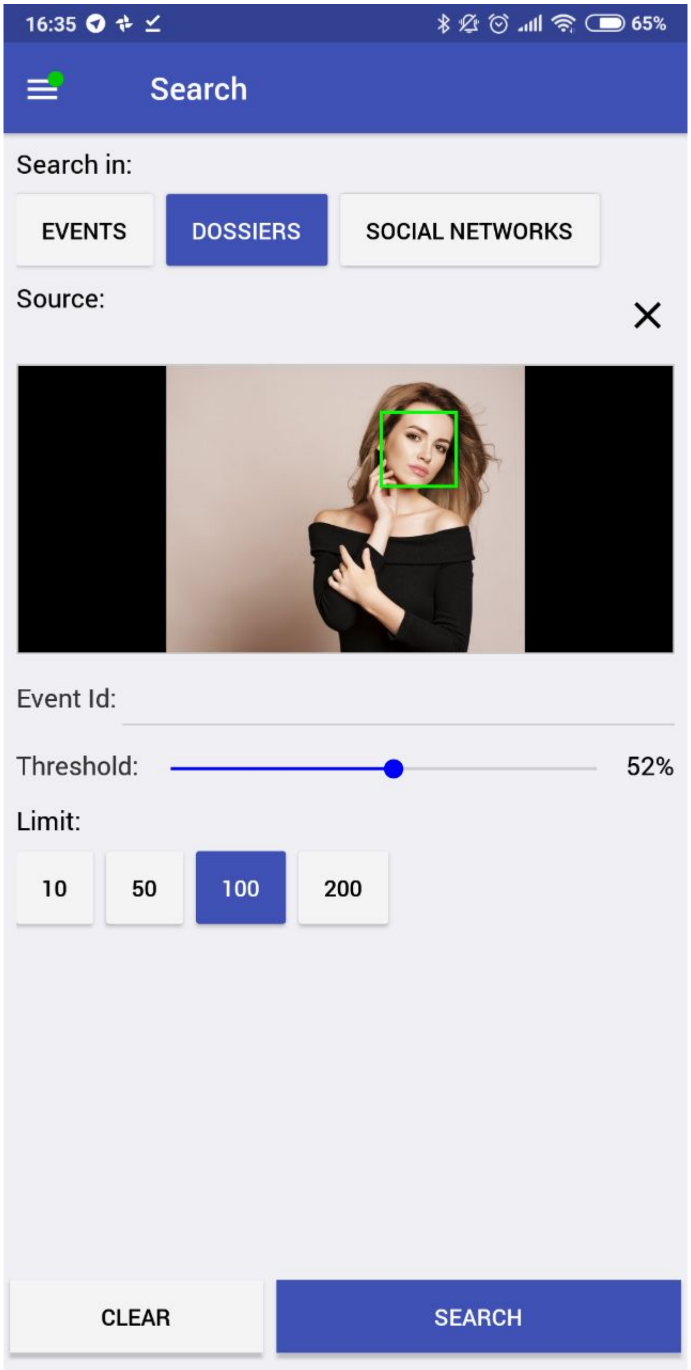
Login
admin

Password
•••••

Url
http://172.20.77.58

The mobile app has a highly intuitive and handy design and provides the following functionality:

- Search for faces in the event list and dossier database.



Search in: DOSSIERS

Source:

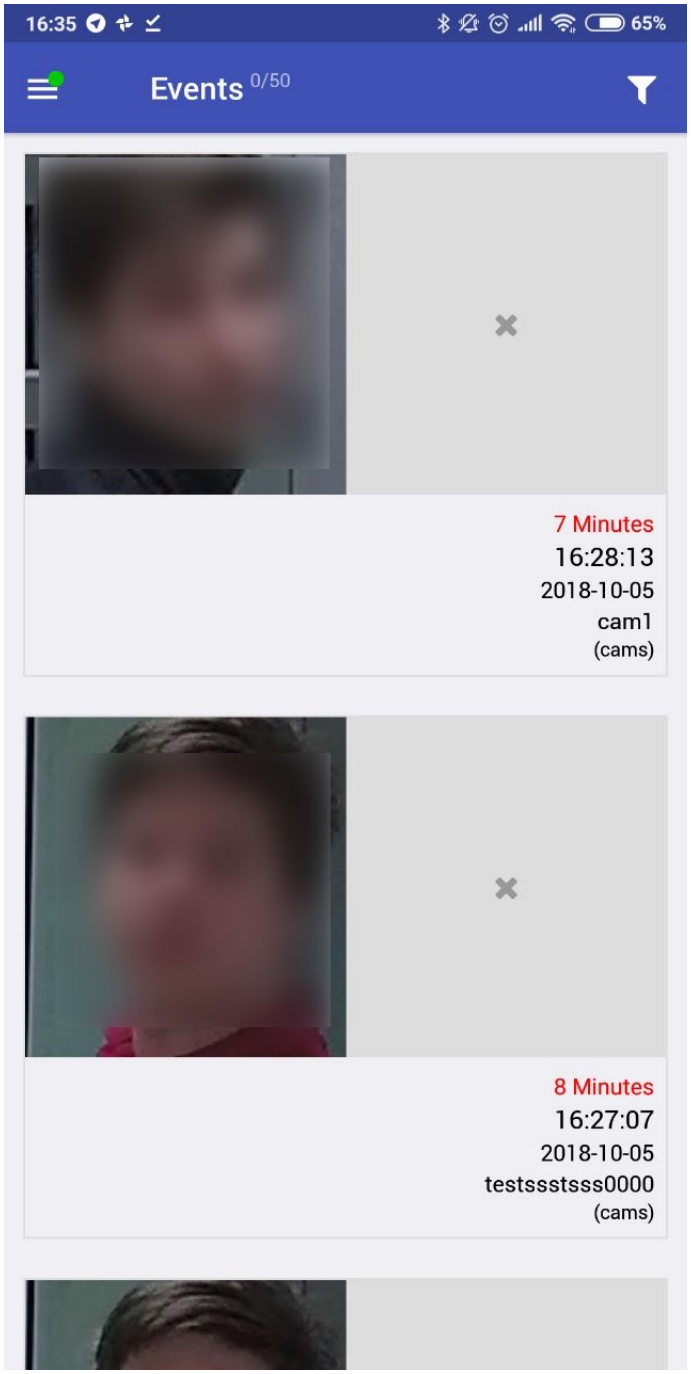
Event Id:

Threshold: 52%

Limit: 10 50 100 200

- Real time face identification in live streams and video files

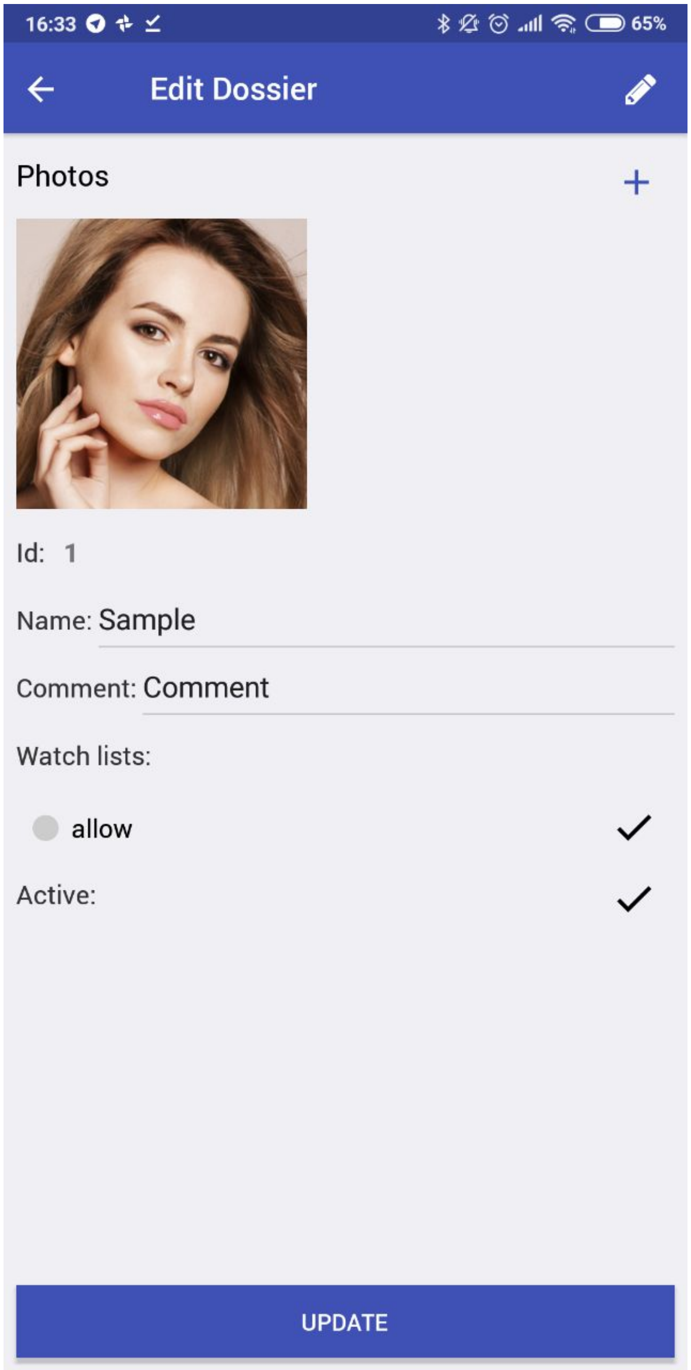
Important: To receive push notifications of events in the mobile version, open a relevant watch list settings in the full version, and check *Require Event Acknoledgment* and *Enable Sound Alert*.



7 Minutes
16:28:13
2018-10-05
cam1
(cams)

8 Minutes
16:27:07
2018-10-05
testssstsss0000
(cams)

- View and create a dossier on a person.



Edit Dossier

Photos

Id: 1

Name: Sample

Comment: Comment

Watch lists:

● allow ✓

Active: ✓

Working with the mobile app is similar to the full version.

CHAPTER 3

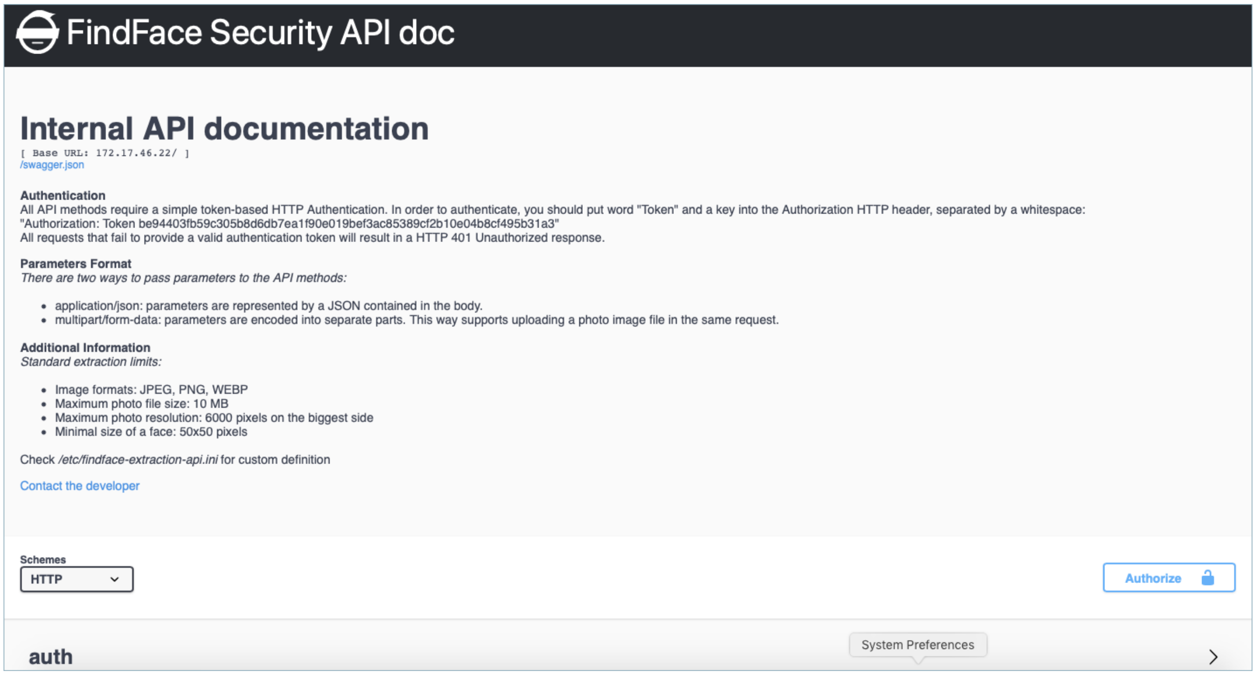
Integrations

This chapter is all about integration with FindFace Security. Integrate your system through HTTP API and webhooks, or check out our turnkey partner integrations.

3.1 HTTP API

Detailed interactive documentation on the FindFace Security HTTP API is available after installation at `http://<ffsecurity_ip:port>/api-docs`. Learn and try it out.

Tip: You can also find it by navigating to *Preferences* -> *Documentation* in the web interface.



Internal API documentation

[Base URL: 172.17.46.22/]
/swagger.json

Authentication
All API methods require a simple token-based HTTP Authentication. In order to authenticate, you should put word "Token" and a key into the Authorization HTTP header, separated by a whitespace:
"Authorization: Token be94403fb59c305b8d6db7ea1f90e019bef3ac85389cf2b10e04b8cf495b31a3"
All requests that fail to provide a valid authentication token will result in a HTTP 401 Unauthorized response.

Parameters Format
There are two ways to pass parameters to the API methods:

- application/json: parameters are represented by a JSON contained in the body.
- multipart/form-data: parameters are encoded into separate parts. This way supports uploading a photo image file in the same request.

Additional Information
Standard extraction limits:

- Image formats: JPEG, PNG, WEBP
- Maximum photo file size: 10 MB
- Maximum photo resolution: 6000 pixels on the biggest side
- Minimal size of a face: 50x50 pixels

Check */etc/findface-extraction-api.ini* for custom definition

Contact the developer

Schemes
HTTP

3.2 Webhooks

You can set up FindFace Security to automatically send notifications about certain events to a given URL. To do so, create and configure a webhook. In this case, when such an event occurs, FindFace Security will send an HTTP request to the URL configured for the webhook.

You can use webhooks for various purposes, for example, to notify a user about a certain event, invoke required behaviour on a target website, solve security tasks such as automated access control, etc.

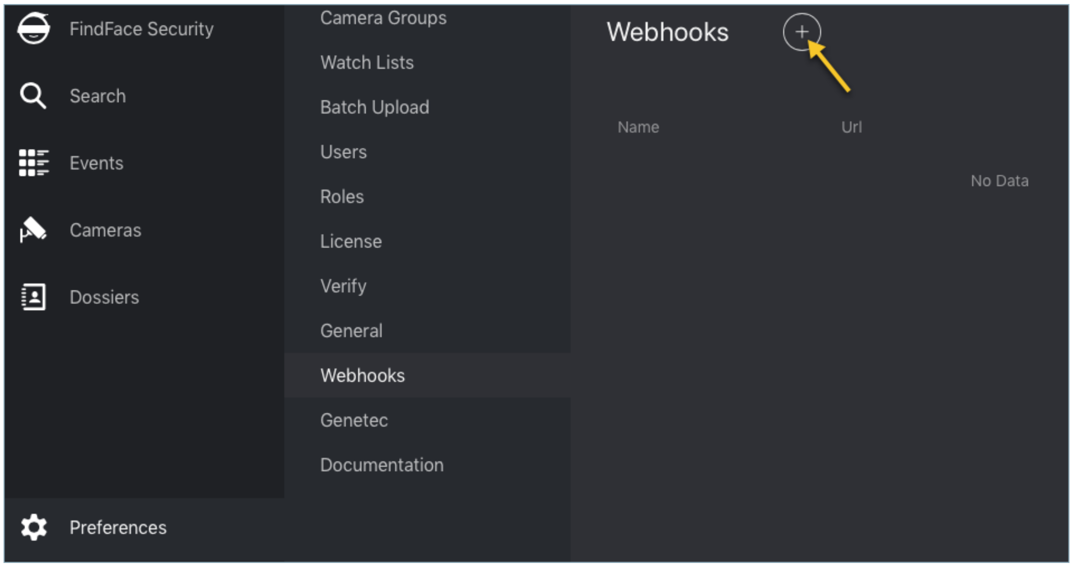
In this section:

3.2.1 Configure Webhook

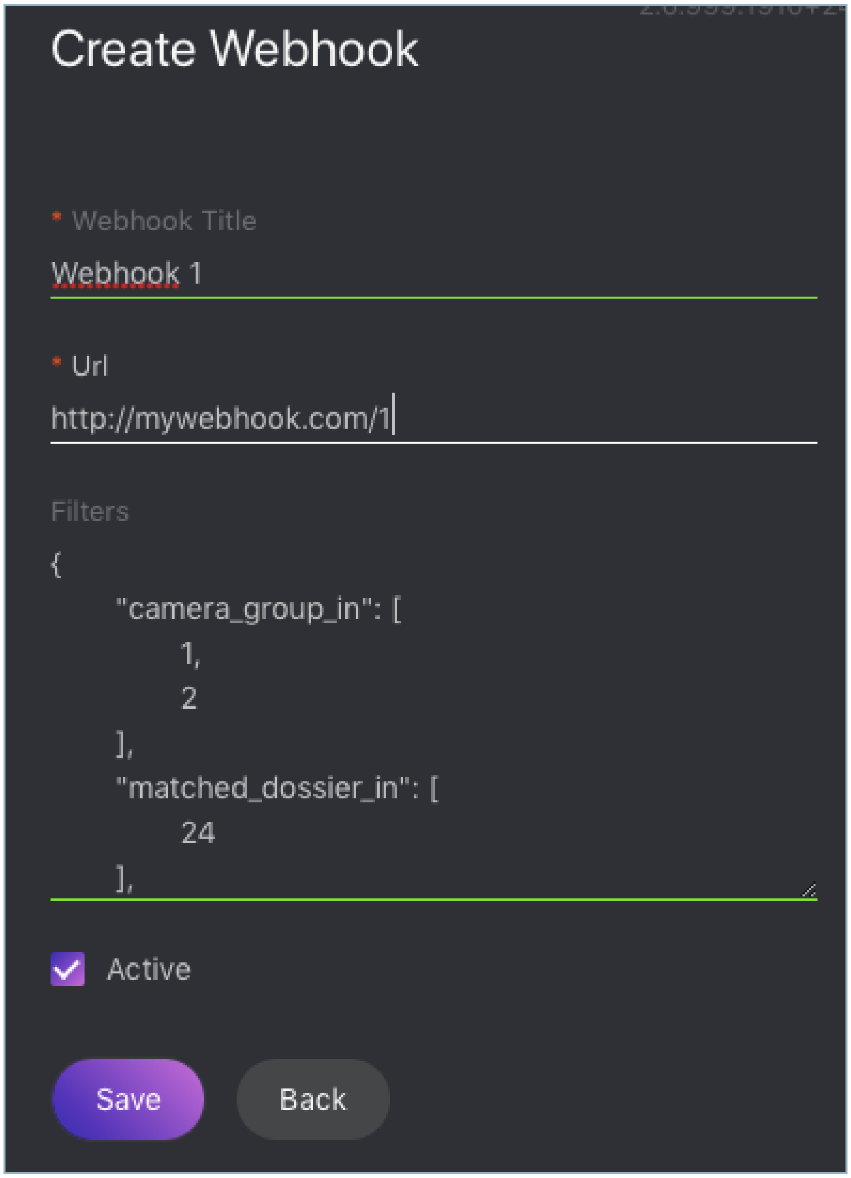
Important: You need Administrator privileges to create a webhook.

To create and configure a webhook, do the following:

1. Navigate to the *Preferences* tab. Click *Webhooks*.
2. Click +.



3. Specify the webhook title.



Create Webhook

* Webhook Title

Webhook 1

* Url

http://mywebhook.com/1

Filters

```
{
    "camera_group_in": [
        1,
        2
    ],
    "matched_dossier_in": [
        24
    ],
```

☑ Active

4. Specify URL to automatically send notifications to.
5. FindFace Security will be automatically sending notifications on events which match given filters. You can filter events by the following event parameters:
   - `camera_group_in`: camera group id, number.
   - `matched_dossier_in`: matched dossier id, number.
   - `matched`: event `matched` status (`true` or `false`), boolean.
   - `camera_in`: camera id, number.

Important: Use only filters which match your search needs. To turn off a filter, remove it from a webhook. Do not leave a filter empty (`[]`) as in this case the result of filtration will be empty as well.

Note: To get notifications about all matched events, pass only curly braces without any enclosed filters:

```
{}
```

Note: You can specify several values for each filter (except `matched`). In this case, the web hook will be triggered once one of the values from this filter has been matched. In the example below, you will get an event from the camera group `1` or `3` if a matched dossier is `12` or `25`.

```
{
  "camera_group_in": [1, 3],
  "matched_dossier_in": [12,25],
}
```

6. Check *Active*.
7. Click *Save*.

3.2.2 Webhook in Action

Try out a webhook by capturing event notifications with a simple web server in Python:

```
from pprint import pprint
from aiohttp import web


async def handle(request):
    pprint(await request.json())
    return web.Response(status=200)


app = web.Application()
# for aiohttp v 3.x
# app.add_routes([web.post('/', handle)])

# for aiohttp v 2.x
app.router.add_post('/', handle)

web.run_app(app, port=8888)
```

If no filters are configured for a webhook, this web server will be getting notifications about each event that occurs in the system. The notifications have the following format:

```
======== Running on http://0.0.0.0:8888 ========
(Press CTRL+C to quit)
[{'acknowledged': True,
  'acknowledged_by': None,
  'acknowledged_date': '2019-04-09T12:29:23Z',
  'acknowledged_reaction': None,
  'camera': 2,
  'confidence': 0.9098,
  'created_date': '2019-04-09T12:29:23Z',
  'face': 'http://172.20.77.17/uploads/2019/04/09/event/122955_face_aT3ZZh.jpg',
```

(continues on next page)

(continued from previous page)

```
  'features': {'age': None,
               'beard': None,
               'emotions': None,
               'gender': None,
               'glasses': None,
               'liveness': None},
  'frame': 'http://172.20.77.17/uploads/2019/04/09/event/122955_image_3msdHH.jpg',
  'frame_coords_bottom': 981,
  'frame_coords_left': 1630,
  'frame_coords_right': 1911,
  'frame_coords_top': 701,
  'id': '4173669353687265180',
  'looks_like_confidence': None,
  'matched': True,
  'matched_dossier': 1,
  'matched_face': '4173665826982243136',
  'matched_lists': [1],
  'normalized_photo': 'http://172.20.77.17/uploads/2019/04/09/event/122955_face0_
↪E638aW.png',
  'quality': -0.000158,
  'scores': {'direction_score': -2.62964,
             'frame_no': 800,
             'score': -0.000158435,
            'tracking_duration': 34000}}]
```

To view the webhook pulling status, execute:

```
sudo journalctl -u findface-security-webhook-updater.service
```

Success:

```
`Apr 09 16:02:28 ubuntu ffsecurity[1524]: INFO     [-] hook 1 was pulled on http://172.
↪20.77.70:8888`
```

Failure:

```
`Apr 09 15:59:02 ubuntu ffsecurity[1524]: INFO     [-] While working on hook 1␣
↪Exception occured: Cannot connect to host 172.20.77.70:8888 ssl:False [Connection␣
↪refused]`
```

3.3 Partner Integrations

3.3.1 Genetec Security Center

FindFace Security integration with Genetec Security Center allows you to expand the capabilities of your Genetec-based security system with face recognition functionality.

Configure Integration

Integration with Genetec Security Center is implemented via the `findface-genetec` plugin. By default, the plugin is enabled, and the FindFace Security *Preferences* features the *Genetec* tab.

Note: If it is not so, open the `ffsecurity` configuration file, and check whether it features the enabled line `INSTALLED_APPS.append(‘ffsecurity_genetec’)`.

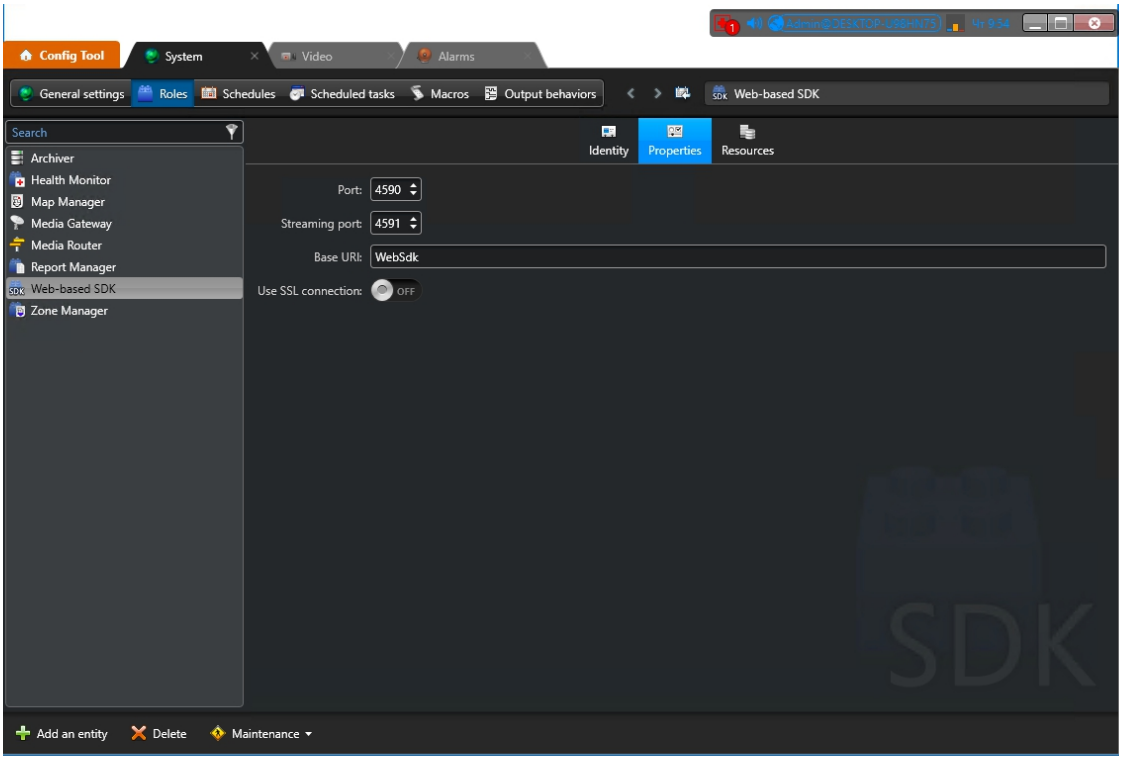
```
sudo vi /etc/ffsecurity/config.py

...

FFSECURITY_UI_CONFIG = {
}

# integration plugins
INSTALLED_APPS.append('ffsecurity_genetec') # remove or comment out this line to␣
↪disable
```

Before getting started with the integration on the FindFace Security side, deploy the Genetec Web SDK and Media Gateway packages, and create an `Alarm` entity that will be triggered in Genetec Security Center when a face recognition event occurs in FindFace Security.

In this chapter:

Configure Genetec Web SDK and Media Gateway

To enable and configure Web SDK, use Genetec Config Tool. For details, refer to *Security Center Administrator Guide -> Chapter 52: Role Types -> Web-based SDK configuration tabs*.



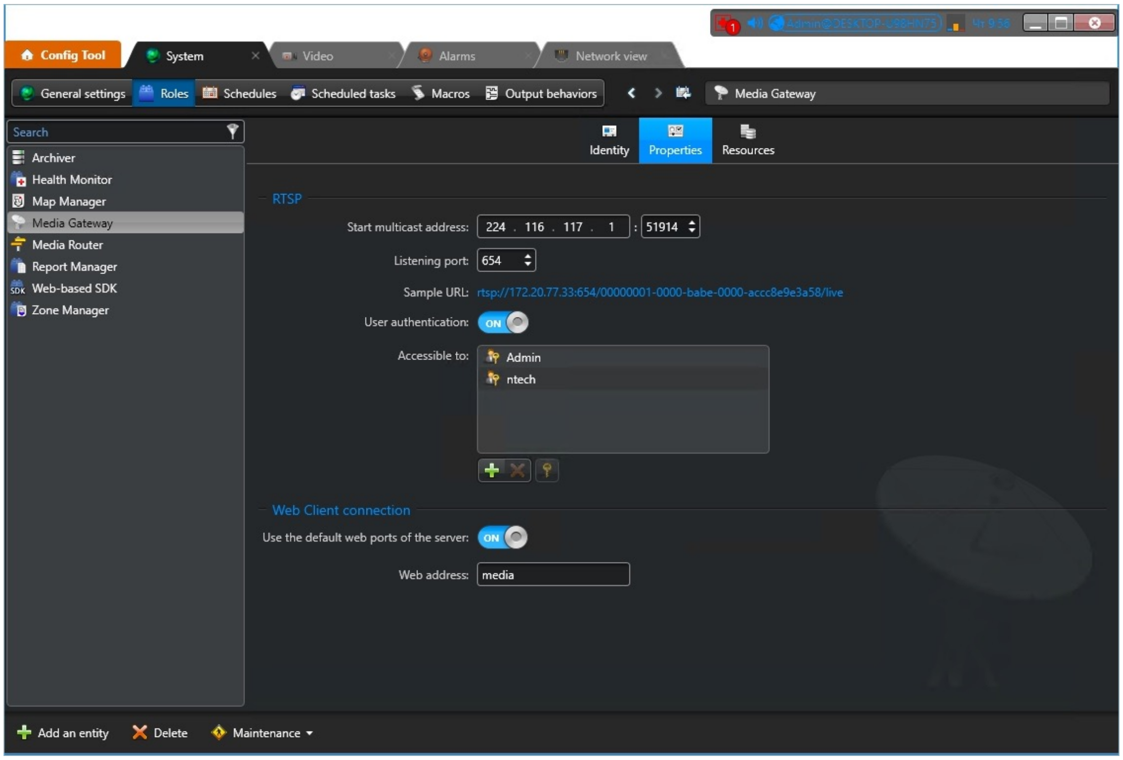
Port: 4590

Streaming port: 4591

Base URI: WebSdk

Use SSL connection: OFF

When enabling and configuring Media Getaway in Genetec Config Tool, refer to *Security Center Administrator Guide -> Chapter 24: Video Deployment*.



RTSP

Start multicast address: 224 . 116 . 117 . 1 : 51914

Listening port: 654

Sample URL: rtsp://172.20.77.33:654/00000001-0000-babe-0000-accc8e9e3a58/live

User authentication: ON

Accessible to: Admin, ntech

Web Client connection

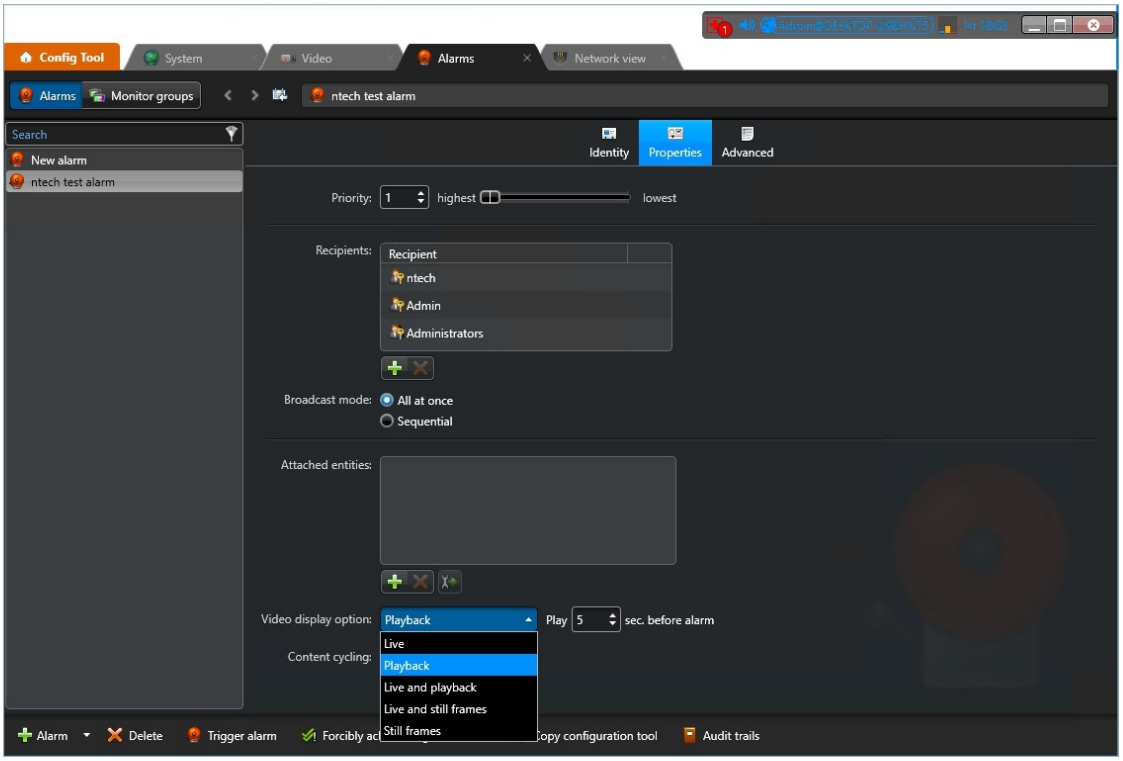
Use the default web ports of the server: ON

Web address: media

Important: Make sure that the firewall is configured so that the ports for the WebSDK and Media Gateway are open.

Create Alarm in Genetec Security Center

Create and configure a new `Alarm` entity in Genetec Config Tool. Refer to *Security Center Administrator Guide -> Chapter 48: Alarms -> Creating Alarms* for details.



Priority: 1 highest lowest

Recipients:

Recipient
ntech
Admin
Administrators

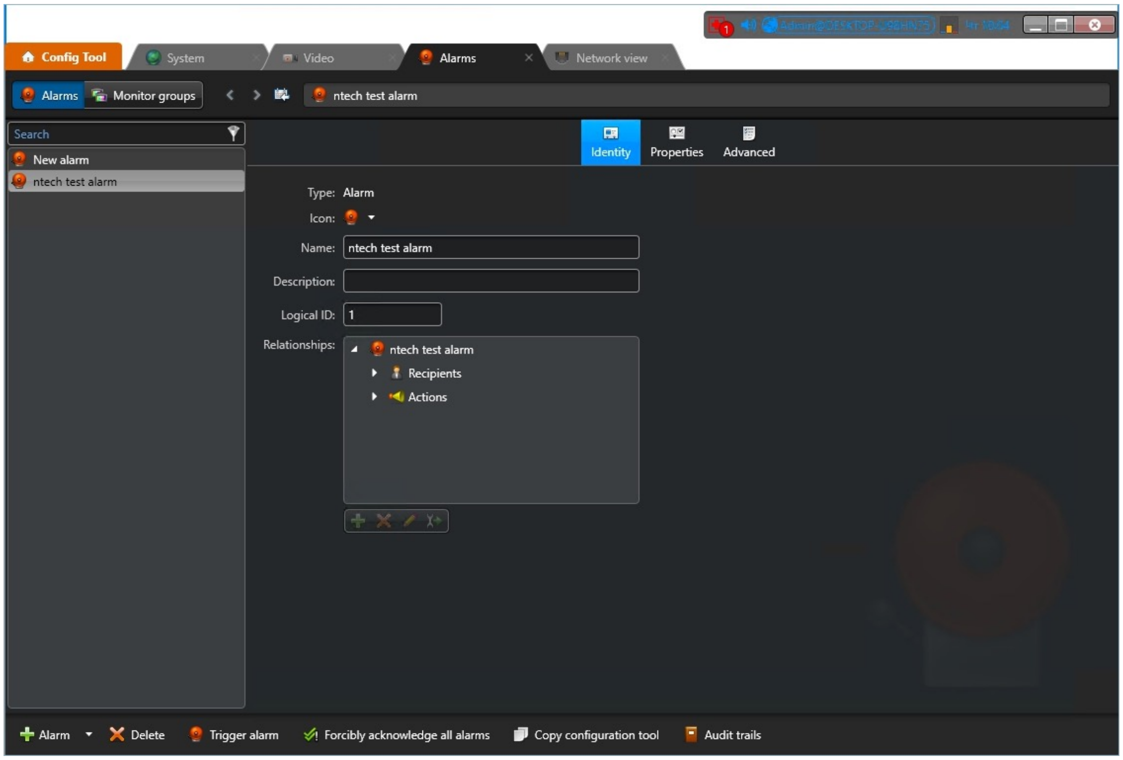
Broadcast mode: ● All at once ○ Sequential

Video display option: Playback Play 5 sec. before alarm

Tip: On the *Properties* tab, select the *Video display option* that suits your needs the best. Available options are *Live*, *Playback*, etc.

Tip: To enable alarm procedures and auto rotation of video right within the alarm pop-up window, enable *Content cycling*.

When configuring the integration in FindFace Security, you will have to enter the alarm logical id that is specified on the *Identity* tab.



Type: Alarm

Icon:

Name: ntech test alarm

Description:

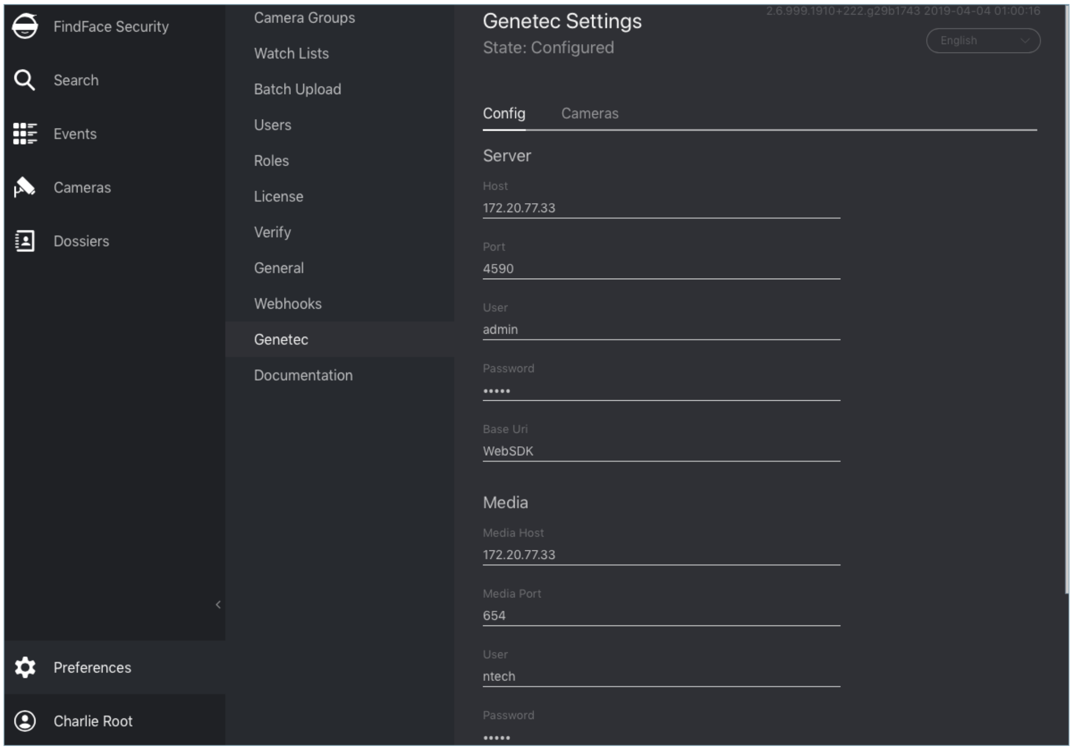
Logical ID: 1

Relationships:
- ntech test alarm
  - Recipients
  - Actions

Configure Endpoints in FindFace Security

To establish connection between FindFace Security and Genetec Security Center, do the following:

1. Navigate to the *Preferences* tab. Click *Genetec*.



Genetec Settings

State: Configured

Server

Host: 172.20.77.33

Port: 4590

User: admin

Password: •••••

Base Uri: WebSDK

Media

Media Host: 172.20.77.33

Media Port: 654

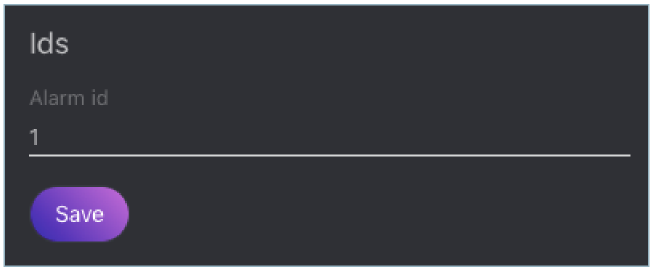
User: ntech

Password: •••••

2. In the *Server Media* sections, specify *settings* of the Web SDK and Media Gateway endpoints.

   Important: The ports for the WebSDK and Media Gateway need to be open.

3. In the *Ids* section, specify the *logical id* of the `Alarm` entity that will be triggered in Genetec Security Center when a face recognition event occurs in FindFace Security.



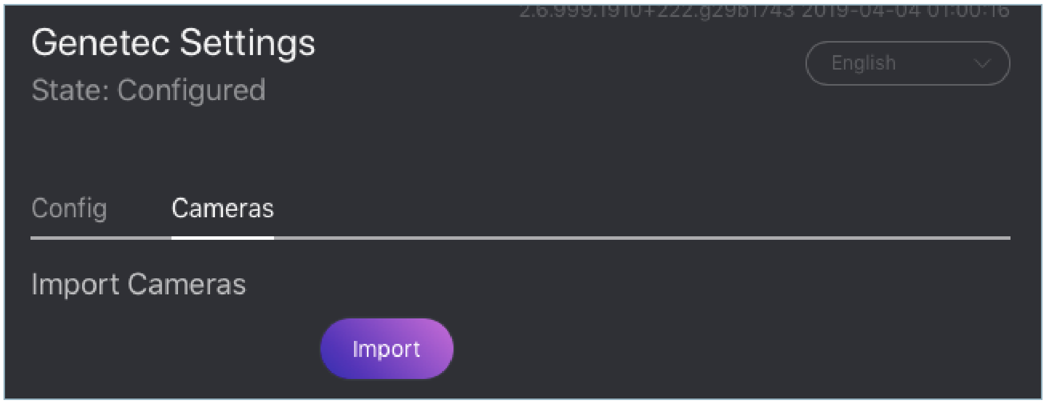
Ids

Alarm id: 1

4. Click *Save*. If the connection to Genetec Security Center is successfully established, you will see the *State* change to *Configured*.

Import Cameras from Genetec Security Center

Once the connection to Genetec Security Center is established, import cameras. To do so, click *Cameras* on the *Genetec* tab. Click *Import*.

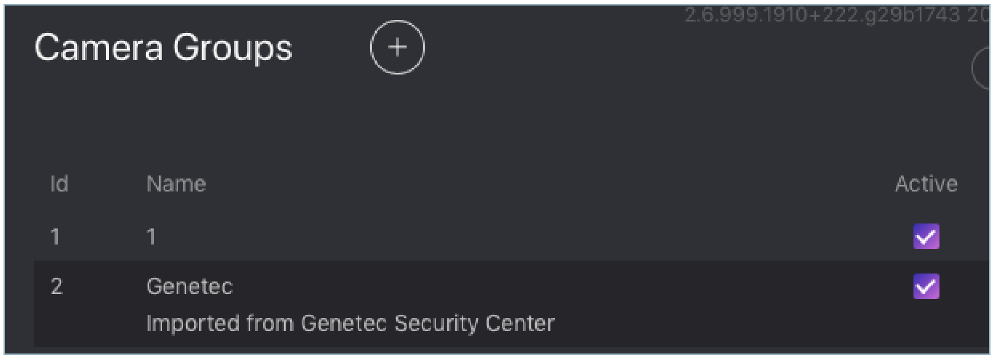


Genetec Settings

State: Configured

Import Cameras

This action will create a *group of cameras* `Genetec` listing all the cameras from Genetec Security Center.



Camera Groups

Id	Name	Active
1	1	☑
2	Genetec Imported from Genetec Security Center	☑

To view this list of cameras, navigate to the *Cameras* tab on the FindFace Security navigation bar. If you want to exclude a camera from face recognition, simply deactivate it in the list.

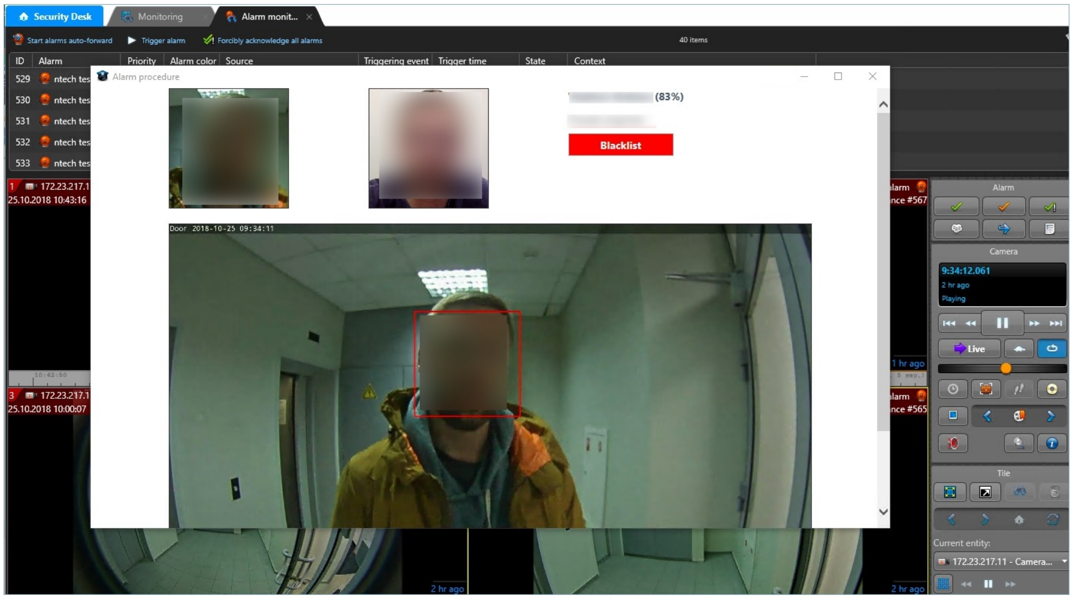
Create Watch Lists and Dossiers in FindFace Security

After you have configured the endpoints and camera settings, finish the integration by creating a *dossier database*. Notifications about face recognition events will be automatically sent to Genetec Security Center. See *Notifications in Genetec Security Center*.

Notifications in Genetec Security Center

Each face recognition event from a Genetec camera, that has a match with a dossier, triggers a relevant alarm in Genetec Security Center. Every alarm triggered by FindFace Security is associated with a relevant camera (source of the face recognition event) so you can instantly watch live or playback video within the `Alarm Monitoring` task in Genetec Security Desk. FindFace Security also utilizes `Alarm Procedures` to provide a user with additional content related to the alarm, such as:

- face detected in video
- matching face from the dossier database
- person's name and comment from the dossier
- matching confidence
- watch list's name
- full frame



After you receive a face recognition alarm, process it as you usually do with other alarms in Genetec Security Center.

3.3.2 Axxon Next

FindFace Security integration with Axxon Next allows you to detect and identify faces in video streams from an Axxon-based security system.

Important: One FindFace Security instance supports interaction with only one Axxon Next server.

Integration with Axxon Next is implemented via the `ffsecurity_axxon` plugin.

To configure the FindFace Security integration with Axxon Next in Ubuntu, do the following:

1. Activate the plugin by appending the `INSTALLED_APPS.append(‘ffsecurity_axxon’)` line to the `/etc/ffsecurity/config.py` configuration file.

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

...

# integration plugins
INSTALLED_APPS.append('ffsecurity_axxon') # remove or comment out this line to␣
↪disable
```

2. Add the `FFSECURITY->AXXON` section to the configuration file. Fill it out as shown in the example below. In the api parameter, specify the IP address of the Axxon Next server that will provide FindFace Security with Axxon API and HLS-archive streams. In the `rtsp` parameter, specify the common segment of Axxon video stream addresses.

```
 FFSECURITY = {
'AXXON': {
   'api': 'http://user:password@example.com/',
   'rtsp': 'rtsp://user:password@example.com:554/',
         }
}
```

3. (Optional). If facial recognition events are required to contain video from Axxon Next, edit the `FFSECURITY_UI_CONFIG` section as shown in the example below.

```
FFSECURITY_UI_CONFIG = {
 'dossier': {
   'video': True,
 }
}
```

4. Create representations of Axxon Next cameras in FindFace Security (see *Camera Management*). A camera representation URL must be specified in the format `axxon:<friendlyNameLong>`, where `friendlyNameLong` is a camera name on the Axxon Next server. Find out this name in the Axxon user interface, or via Axxon API by executing:

```
curl http://user:password@127.0.0.1/video-origins/

{
   "OLOLOE-DEV/DeviceIpint.vhod_1/SourceEndpoint.video:0:0" : {
      "friendlyNameLong" : "vhod_1.Vhod_1",
      "friendlyNameShort" : "Vhod_1",
      "origin" : "OLOLOE-DEV/DeviceIpint.vhod_1/SourceEndpoint.video:0:0",
      "state" : "signal_restored"
   }
}
```

   For the camera from the example above, URL must be specified as `axxon:vhod_1.Vhod_1`.

The configuration is now finished. If the integration is properly configured, FindFace Security will be detecting and identifying faces in Axxon Next video streams, and facial recognition events will be featuring video clips from Axxon Next (upon relevant settings).