Watcher manual

17 May 2019

Introduction to Flussonic Watcher

Flussonic Watcher

is a complete video surveillance software system that works with a distributed IP camera network. This system can be used for video streaming, recording and managing video archives.

Flussonic Watcher is a scalable and ready-to-use system with flexible integration options. This web and mobile-oriented solution can solve various business tasks: from launching a corporate cloud video monitoring to a municipal video surveillance system that covers the entire city or state.

Flussonic Watcher presentation

Internet providers — to launch a dedicated cloud-based video surveillance service and offer it to their customers for an additional fee. Watcher can be easily integrated with any provider and its existing billing system.

Management companies — for live video broadcasts from socially significant facilities and construction monitoring.

Production — to perform audio-video monitoring on factories, courts, polling stations, etc.

Municipal and federal projects — to provide free access to public cameras, as well as limited access to security organizations.

The control module is a component that has a database of all users and cameras, user access management, archive quotas, cluster servers, provisioning and camera activation, as well as APIs for integration with client billing or custom module development.

Web interface is the main interface for users and administrators. It works in all modern browsers and mobile devices. The interface includes a dashboard with cameras and archive, favorites, groups, map, user management tools and cameras, settings, interface branding tools and much more.

iOS and Android mobile applications give you access to cameras and user archive from mobile devices.

Firmware for cameras (agent) is an optional component. It can be installed on the cameras to provide access behind NAT with stable encryption and direct video delivery to Watcher.

Flussonic Watcher comes in two editions: Single and Cluster.

Single is good for smaller projects that have no more than 500 cameras and do not need interface branding (company logo, custom colors, etc.).

The Cluster edition is truly scalable with no limitations on the number of cameras. It also includes branding tools for adding a logo, corporate colors, captions, as well as the ability to organize a server cluster with backup tools.

System requirements for Watcher

Flussonic Watcher works in two versions:

- Single - Suitable for those who do not need branding of the interface (their logo, colors), for projects of up to 500 cameras, not more than 1 server.
- Cluster - Advanced version. Includes branding tools (changing logo, color, text), the ability to expand to tens of thousands of cameras, including the ability to build a cluster of dozens of servers with redundant streams. This configuration requires a minimum of 2 servers. Suitable for those who do subscriber service or a large video surveillance system.

Management + streaming server:

Operating system: Ubuntu 14.04 or later, Debian 7 or later;

Database: PostgreSQL;

CPU: CPU: Xeon E-3 1230v5 3.4 GHz and higher;

Memory: 32GB RAM;

Dedicated server: Yes.

Hard drive type: HDD / SSD;

Hard drive size: depends on the camera network size and video archive storage requirements.

10 GB of free disk space per 1 Mbps camera per day.
20 GB of free disk space per 2 Mbps camera per day.
70 GB of free disk space per 1 Mbps camera 7 days archive;

These system requirements are suitable for a network of:
500 1Mbps cameras
500 users
Failover turned off
Mosaic turned off

Or

250 2Mbps cameras
500 users
Failover turned off
Mosaic turned off

Servers should be dedicated, so no other software should run on them.

Management server (endpoint — Runs Flussonic Watcher and a DB):

Operating system: Ubuntu 14.04 or later, Debian 7 or later;

CPU: CPU: 2-core CPU;

Memory:8Gb RAM;

Virtual server support: Yes;

Database: PostgreSQL;

Hard drive type: SSD;

Hard drive size: 64GB of free disk space;

Dedicated server: Yes.

Video streaming server (streamer — Video streaming and video archive storage):

Operating system: Ubuntu 14.04 or later, Debian 7 or later;

CPU: CPU: Xeon E-3 1230v5 3.4 GHz and higher;

Memory: 32GB RAM;

Dedicated server: Yes.

Hard drive type: HDD / SSD;

Hard drive size: depends on the camera network size and video archive storage requirements.

10 GB of free disk space per 1 Mbps camera per day.

20 GB of free disk space per 2 Mbps camera per day.

70 GB of free disk space per 1 Mbps camera 7 days archive;

Important! For correct operation, Flussonic Watcher requires open ports 80, 443, 1935, 554 on all hosts, and the control server must have a real hostname, which is resolved from the Internet.

Installing Watcher

Flussonic Watcher can work both in cluster (multi-server) mode and in single-server mode. Watcher is installed in almost the same way regardless of the mode (single or cluster).

Watcher supports small to medium to large projects with unlimited number of cameras. The number of cameras is limited only by hardware.

When working in a cluster of servers, Watcher ensures the fault-tolerance of stream ingest (failover).

Watcher uses the PostgreSQL database engine.

Important! Watcher is installed from the package flussonic-watcher and Flussonic Media Server and the database are automatically installed with Watcher.

Comparing Watcher cluster with Watcher single

- Watcher Cluster supports UI branding tools (adding your custom logo, choosing your custom colors, etc.)
- Watcher Cluster can work with special firmware for cameras — Flussonic Agent, or simply Agent. Agent makes cameras accessible from behind NAT and greatly simplifies setting up the entire infrastructure. With Agent, you can add cameras in plug-and-play mode. This firmware improves the stability of video delivery and implements data encryption directly from cameras to streaming servers. To start using this firmware, contact our manager who leads your project.
- Single-server mode is suitable for small and medium-size projects, where the maximum number of cameras does not exceed 500.

Servers' configuration

If your poject is small (less than 500 IP cameras) and you don't need a cluster of streaming servers, just install Watcher on a single server. In single-server mode, all cameras are connected to a single server, where Flussonic Watcher, Flussonic Media Server and the database are installed, the web interfaces works, streams are ingested, and the archive is written.

The cluster requires at least two servers:

- The managing server (endpoint). It has the web interface to Watcher, Flussonic Media Server, the business logic, and the PostgreSQL database engine. Watcher works the managing server only.
- Streaming servers (streamers). A streamer is a server machine that has Flussonic Media Server installed. It stores DVR archives and handles camera streams. You can add from 1 to 100 streaming servers.

All servers must have public IP addresses and the same cluster key (specified in the Flussonic settings). In addition, the host name of the management server must resolve to the IP address.

The following picture shows the parts of cluster and the flow of video streams:

To install Watcher:

- Install Flussonic Watcher on the managing server. PostgreSQL and Flussonic Media Server are installed automatically together with Watcher.
- Specify the path to PostgreSQL in the administrator UI of Flussonic Media Server.
- Create a user with Watcher administrator privileges by using the Watcher's administrator UI.

That's all for a single-server Watcher.

If you plan to create a cluster, after you've done steps 1-3 continue the installation with the following steps:

Install Flussonic Media Server on all streaming servers.

Create a cluster and register streamers in Watcher by using the Watcher's administrator UI.

All the steps are described later on this page.

On how to update Watcher, see the section Updating Flussonic Watcher.

Installing Flussonic Watcher

On the server where you plan to run Watcher execute the command:

```
curl -sSf https://flussonic.com/raw/install_watcher.sh | sh
```

After successful installation, the system advises you to start PostgreSQL and suggests the command to do so. Do not start PostgreSQL yet, go to the next step - user creation.

Create the user and the database. First, create the user vsaas by typing this command:

```
sudo -u postgres -i createuser -P vsaas
```

The system will prompt you to enter the password that will be used for the user vsaas:

```
Enter password for new role: (come up with and enter Watcher super admin password)
```

Type the password again for confirmation:

```
Enter it again: (re-enter Watcher super admin password)
```

Create the database vsaas_production with the created user vsaas as the owner:

```
sudo -u postgres -i createdb -O vsaas -e -E UTF8 -T template0 vsaas_production
```

The system's response if the database was created successfully:

```
CREATE DATABASE vsaas_production OWNER vsaas ENCODING 'UTF8' TEMPLATE template0;
```

In the Flussonic's web UI (http://flussonic:8080/admin), go to IP cameras and specify the path to the database in the Database path box.

Important. Replace VSAAS_PASSWORD with the real password of the vsaas user that you created in previous steps.

For cluster only: Add the mode cluster option in the configuration file /etc/flussonic/flussonic.conf:

```
vsaas {
database postgresql://vsaas:vsaas_password@localhost/vsaas_production;
endpoint enabled;
mode cluster;
}
```

After editing the file restart Flussonic Media Server with the command:

```
service flussonic restart
```

Finally, go to the the Flussonic UI in the browser.

Now if you go to the main page of the Flussonic UI http://FLUSSONIC:8080, the Watcher's web interface will open instead of Flussonic's.

To return to the Flussonic Media Server UI, go to http://FLUSSONIC:8080/admin.

Now it's time to create the Watcher main administrator.

Creating the Watcher administrator

Go to http://FLUSSONIC:8080 and the administrator control panel opens (we also call it the Watcher web UI). On the very first launch, the system will ask you to enter the login and come up with the password to create the first Watcher's administrator account.

The installation for single-server mode is now complete.

If you want to create a cluster, you will need to prepare streamers and set up Watcher to work as part of a cluster (see next steps).

(For cluster mode only) Installing Flussonic Media Server on streamers

Flussonic Media Server is installed on all streaming servers.

Run the command:

```
curl -sSf https://flussonic.com/raw/install.sh | sh
```

Now start Flussonic Media Server:

```
/etc/init.d/flussonic start
```

Learn more about installation of Flussonic Media Server in the Flussonic documentation:

- Quick start with Flussonic Media Server — briefly describes how to install Flussonic and start using it.
- Installing Flussonic Media Server — detailed comments about the installation, system requirements, and more.

On each streamer set up HTTPS and specify the cluster key in Flussonic's settings – see Create a cluster below.

(For cluster mode only) Creating a cluster (multiple server mode)

Creating a cluster means to add streamers (streaming servers) in the settings of Flussonic Watcher. It is necessary to add at least one streamer on which IP cameras are added to configuration. This will allow you to start taking video from cameras in cluster mode.

The UI page Settings > Streamers is essential for the cluster.

Pre-conditions

- For each streamer, install Flussonic Media Server on a separate server, which will act as a streamer. In other words, besides the server with Flussonic Watcher, you must have at least one other server with a public IP address and Flussonic Media Server installed.
- After you install Flussonic on a streamer, immediately change the administrator's login and password on each streamer.

Configure HTTPS on each streamer. It is enough to set the port for HTTPS, and Flussonic will use self-signed SSL certificates.

Set identical date and time on the managing server and on each streamer.

In each streamer settings (on the managing server), specify cluster_key (the key must be the same as the cluster key for Flussonic Watcher). Learn more.

Configure the DNS zone for the managing server.

For cluster only: For Watcher to work correctly in a cluster, you need to add the A record in the DNS zone settings for your domain, where you specify the host name. This hostname must also be registered in the operating system on the server with Flussonic Watcher (the managing server). This is necessary for streamers to access the managing server.

To check that the hostname resolves, run on the managing server the command hostname — it must return the correct hostname specified in the DNS settings, for example, example.com.

When the server intended to stream video from cameras is ready for work, you need to add it in the settings of Flussonic Watcher.

Adding streamers to Watcher

Log in to Flussonic Watcher as the administrator.

Go to Settings > Streamers and click the "+" icon to add a streamer:

Hostname – the domain name of the streaming server. Example: streamer2.example.com
Cluster key – the cluster key used in the cluster (the cluster_key option in the configuration file). If the streamer’s and Watcher's cluster keys are identical, there is no need to fill this field.
DVR path – the path to the archive. This field is required for the archive to be recorded. For example: /dvr. Multiple paths are possible (separate the paths with spaces): /dvr1 /dvr2.

Hostname – the domain name of the streaming server. Example: streamer2.example.com

Cluster key – the cluster key used in the cluster (the cluster_key option in the configuration file). If the streamer’s and Watcher's cluster keys are identical, there is no need to fill this field.

DVR path – the path to the archive. This field is required for the archive to be recorded. For example: /dvr. Multiple paths are possible (separate the paths with spaces): /dvr1 /dvr2.

When you have added a number of streamers, you must select the default one. Click the streamer in the list and then click Default. All new cameras will be added to the default stream server.

Important!

In the UI section Streamers you don’t need to add the host where Flussonic Watcher itself is deployed. On all servers in a cluster identical date and time must be set.

For each streamer, you can enable the automatic use of redundant (backup) servers for streams ingest in case this streamer fails (see Failover).

Updating Flussonic Watcher

To update Watcher:

```
apt-get update
apt-get -y install flussonic-watcher
/etc/init.d/flussonic restart
```

During its update, Watcher automatically migrates the database to work with the new version. In rare cases it might be necessary to migrate the database manually. Watcher will show the message about that in the UI.

We strongly recommend that you back up your database every day and before you update Watcher.

The Watcher database

This section provides the instructions on how to maintain the database used by Watcher.

Migrate to PostgreSQL (necessary starting from version 19.03)

Update the database structure manually (migth be necessary in some cases, Watcher will inform you about it)

Migration from SQLite to PostgreSQL

Back up these files:

```
/etc/flussonic/flussonic.conf

/opt/flussonic/priv/vsaas.db
```

Install the latest version of Flussonic Watcher (to do this, it is enough to update Flussonic Media Server). Run two commands:

```
apt update
```

and

```
apt install flussonic
```

Learn more about the update process

Back up all the data by using our migration tool:

```
cd /opt/flussonic/apps/vsaas

/opt/flussonic/bin/python -m manage backup create
```

The tool creates a file like this:
/opt/flussonic/apps/vsaas/backups/20190215201434-b62d21842ab7-WatcherBackup.gz

Install PostgreSQL

Edit the database line in the Flussonic configuration file flussonic.conf:

```
database postgresql://vsaas:vsaas@localhost/vsaas_production;
```

You can edit text files with the text editor nano.

Reload the Flussonic service:

```
/etc/init.d/flussonic restart
```

Restore data from the file that was created by the migration tool:

```
cd /opt/flussonic/apps/vsaas
```

```
/opt/flussonic/bin/python -m manage backup restore -d 20190215201434
```

Installing and configuring PostgreSQL

To install PostgreSQL, with root access in the console execute the command:

```
apt install postgresql
```

Create a PostgreSQL user and a database. Type two commands in the console one by one. First, create the user vsaas:

```
sudo -u postgres createuser -P vsaas
```

The system will prompt you to enter the password that will be used for the user vsaas:

Enter password for new role: (come up with and enter Watcher super admin password)

Enter the password one more time:

Enter it again: (re-enter Watcher super admin password)

Create the database vsaas_production with the created user vsaas as the owner:

```
sudo -u postgres createdb -O vsaas -e -E UTF8 -T template0 vsaas_production
```

System's response if the database was created successfully: CREATE DATABASE vsaas_production OWNER vsaas ENCODING 'UTF8' TEMPLATE template0

Go to the IP cameras in the administrative interface to Flussonic Media Server (see Watcher UI and Flussonic UI below) and specify the path to the database in the Database path box.

Inportant. Replace VSAAS_PASSWORD with the real password of the vsaas user that you created in previous steps.

Updating the database structure

Some Flussonic Watcher updates include structural changes to the database. In this case, you will see the following message:

The script /opt/flussonic/contrib/watcher_db_migrate.sh adds changes to the database structure.

Run this script on the server manually.

Here is a log of the successful execution of the script:

```
# /opt/flussonic/contrib/watcher_db_migrate.sh
INFO  [alembic.runtime.migration] Context impl PostgresqlImpl.
INFO  [alembic.runtime.migration] Will assume non-transactional DDL.
INFO  [alembic.runtime.migration] Running upgrade dfd74e510414 -
> 1822b8f25e20, agent:model, agent:camera
INFO  [alembic.runtime.migration] Running upgrade 1822b8f25e20 -> 1a71a9477bbb, streamer: cluster_key
INFO  [alembic.runtime.migration] Running upgrade 1a71a9477bbb -
> 7a3ab2550cab, streamer_fkey cascade
Restarting Watcher
```

After the Flussonic Watcher restarts, you will be able to use the Watcher web interface.

If while running watcher_db_migrate.sh you encounter an error similar to this one

```
alembic.util.exc.CommandError: Can't locate revision identified by 'ebdce5515b6d
```

then install the previously used version (that you used before Watcher update or downgrade) and run:

```
cd /opt/flussonic/apps/vsaas
/opt/flussonic/bin/python -m manage db downgrade
```

Important! Backup your database before running the watcher_db_migrate.sh script.

The database location is stored in the configuration file /etc/flussonic/flussonic.conf.

```
vsaas {
  database postgresql://vsaas:VSAAS_PASSWORD@localhost/vsaas_production;
}
```

Create the directory where the backup copy will be stored (/PATH/TO/BACKUP/ in the example below).

Create a backup copy using the pg_dump utility:

```
pg_dump -h localhost -U vsaas -d vsaas_production > /PATH/TO/BACKUP/vsaas.sql
```

System's response: Password:

Enter the password VSAAS_PASSWORD, that you specified for the user vsaas.

Settings

The Watcher's ddministrator can edit the following settings in the Watcher web interface.

Operating mode – a multi-server cluster mode or a plugin mode that works on one server.

Streamers - the link to the page for streamers management. It becomes active in cluster mode.

Archive path — the path to the DVR archive storage. It can be a Swift storage, available in a standalone mode, with a single server launch. In cluster mode, the DVR path must be configured for each streamer in the streamer settings on the managing server.

API key — a token used for mobile access. You need to use it in your Account on the Flussonic website to activate access to mobile applications.

Guest access – allow unregistered users to log in.

Registration allowed — turn on the self-registration option.

Homepage — select what users will see upon login: map or dashboard.

Guest Homepage — select what guests will see: map or dashboard.

Dashboard — select what items to show on a dashboard: camera or groups.

External Authentication – External authorization. Specify http address or address of RADIUS server.

Show map in navigation — show or hide the map from the side menu.

Map Center — geographic coordinates for map centering.

Map Language – map language.

Map zoom — set the default map zoom.

Map Provider — select the map provider: Google, Openstreetmaps or OpenStreetMaps Offline.

Cameras serial numbers management – control camera serial IDs in Flussonic Watcher. It is necessary when you use Agent.

Custom header text — see details in the section «Interface customization»

Custom page title — see details in the section «Interface customization»

Custom logo — see details in the section «Interface customization»

Custom color theme — see details in the section «Interface customization»

Reset password

A user can use the

Forgot password

option. A password recovery link will be automatically sent via email. Read how to configure an SMTP server.

An administrator can change the user password with the reset_password utility.

Usage:

```
reset_password
```

Example:

```
/opt/flussonic/contrib/reset_password email.address@example.com pwd123xyz
```

You will see this message if everything went OK:

```
Changing password for email.address@example.com
```

Failover

A failover cluster is a group of servers that work together to maintain the overall service stability and exclude any possible downtime of any part of the system. If one of the servers fails, another cluster server takes over its workload. This process is called

failover

.

In Watcher cluster, in case of a streamer server failure, the camera traffic to this server will automatically redistribute between other cluster servers, also called donors. The video archive becomes unavailable on the streamer server that had a failure. The new video archive will be temporarily recorded on the donor server.

When the connection to the primary server that had an issue is renewed, the traffic is automatically redirected back. Provided that the storage on the primary server wasn't damaged during the failover, the access to the main video archive will be re-established. However, the temporary video archive will be deleted from the donor server.

Turn on the failover option for each streamer server individually in the Streamer section and change mode in /etc/flussonic/flussonic.conf file on Watcher server:

```
vsaas {
  mode cluster+failover=30;
  ...
}
```

Where

30 — how often (in seconds) Watcher checks connection to streamers.

To apply new settings, restart Watcher:

```
/etc/init.d/flussonic restart
```

Motion Detection Events processing

Motion Detection Events processing

Flussonic Watcher can receive events by email, for this purpose it has a special SMTP server.

It receives emails with motion events from cameras and add corresponding marks in the archive recordings in the places when motion was detected.

You can configure motion detection in two steps:

1) Enable receiving emails in Watcher.

2) Configuring a camera to send emails.

How motion recording works:

Flussonic Watcher maintains a permanent recording with the specified recording depth. When an e-mail arrives with traffic, Flussonic Watcher marks the time interval in the database to reflect the tags in the archive player.

The most important! The record is protected from deletion.

The duration of a protected recording is determined by two timestamps, the first of which is calculated as the current time minus 10 seconds, and the second timestamp is the current time plus 30 seconds.

You just need to set the depth of the archive: for example, 6 hours. Then enable the reception of events.

You will receive 6 hours of continuous archive + sensor entries will be stored as long as there is free space on the disk.

Recording new events will delete the old ones.

Calculating the necessary disk space based on the bit rate of the cameras and the frequency of movements, you can save up to 50-90% of the disk space compared to the normal recording without events.

Enabling event capture in Flussonic Watcher

Add the lines with the camera_alarm plugin into /etc/flussonic/flussonic.conf:

```
plugin camera_alarm {
  catch motion;
  listen smtp://0.0.0.0:1025;
}
```

The catch parameter specifies the word that Flussonic Watcher will search for in the subject of the message. Most cameras send letters with the default subject like this: "Camera 123 Motion Detected at 14:21 27-10-2017".

If your camera sends a letter with a different subject or has the possibilty to specify your own subject, then you configure catch as you like.

The listen parameter specifies the interface and port for the built-in SMTP server. You can set login and password for SMTP:

```
listen smtp://username:password@0.0.0.0:1025;
```

Then restart Flussonic Media Server.

The link to the event list will appear on the page of cameras control.

Configure a camera

The next step requires that you set email notifications for Flussonic Media Server in the camera interface.

Important! Specify a sender and recipient as a a full camera names (camera name and ID). The full name can be found in the Watcher UI.

Example: cam1-abcdefg@example.com, where cam1 is a cam's name in Watcher and abcdefg is an ID in Watcher.

The motion marks will appear in the camera archive interface.

License Plate Detection Events

Flussonic can detect license plates and recognize Russian car numbers on the video transmitted by an IP camera (including special transport like Emergency and Fire cars). This functionality is known as ANPR (automatic number plate recognition).

Flussonic does the following:

- Creates events of license plate detection.

  Video comes from IP cameras to a streamer (in cluster mode) or to the managing server (in single mode), where the number recognition takes place.
- Provides the Watcher UI for viewing plate detection events.

  In the new Watcher UI, starting from version 18.11, you can view registered events and watch the recorded video of each event.
- Provides the API for integration with external services.

To start detecting car numbers:

- Prepare hardware and software for the Flussonic server that will carry out number recognition.
- Turn on and configure the car number recognition. To configure the feature, use the web interface or the configuration file, but please remember that some options can be set only through the file.

On this page:

Installing the ANPR module

To start using number recognition, install Flussonic Media Server and Watcher first (if you didn't have them installed). The ANPR module can work both in cluster and single mode of Flussonic and Watcher installation.

Number recognition takes splace on a streaming server in cluster mode or on a managing server in single mode. To this server, you need to connect cameras that transmit video from the place where you want to detect car numbers. This server must have at least one high-performance videocard GPU NVIDIA with at least 6 Gb video memory.

System requirements for Flussonic's ANPR module

- OS: Ubuntu 16.04 x64
- GPU: Nvidia (Pascal) minimum 6 GB VRAM (for a certain project we could tell more exactly how much memory is required).
- CPU: 4+ cores
- RAM: 8+ GB

- Flussonic Media Server (on streamers in a cluster)
- Flussonic Media Server + Watcher (single server)

Attention. In a cluster installation, the number recognition module works on a streamer server, where Flussonic Media Server must be installed first. If you use a single server, install Flussonic Media Server and Watcher before you install the recognition system.

Installation

To install the number recognition module, add the official Nvidia repository to the system and then install the flussonic-vision package from the Flussonic's repository:

```
wget http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-
ubuntu1604_9.2.148-1_amd64.deb
dpkg -i cuda-repo-ubuntu1604_9.2.148-1_amd64.deb
apt-key adv --fetch-
keys http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub
apt update
apt install flussonic-vision
```

Setting up ANPR in the configuration file

Note. You can do the same through the Watcher UI – the settings will be saved to the config file automatically. But you will still need to check the GPU number in the file and edit it, if necessary.

- Open the file /etc/flussonic/flussonic.conf.
- Add the line plugin vision;, which enables the recognition system.
- Add the vision directive to a stream's settings, and specify the GPU number:

```
stream cam1 {
 url rtsp://192.168.0.11:554/h264;
 vision gpu=0;
}

stream cam2 {
 url rtsp://192.168.0.12:554/h264;
 vision gpu=1;
}

plugin vision;
```

- gpu (required) – GPU number. You can get it by using the nvidia-smi tool.

By default, the recognition system searches for car numbers over all the field of view of a camera.

Setting up ANPR in the UI

Before you configure cameras in the UI, make sure you enable the recognition system. Add the line plugin vision; to the configuration file /etc/flussonic/flussonic.conf.

To turn on plate detection and number recognition for a camera:

- In the Watcher UI, go to Cameras. Find the camera in the list and open its settings by clicking the icon in the upper right corner of the player.

Select the ANPR check box and click Save.

Now Flussonic will recognize car numbers that appear in the frame of this camera, and mark the time when the car entered and left the scene.
Flussonic modifies the stream settings in the configuration file /etc/flussonic/flussonic.conf.

You may need to edit manually the GPU number (see the previous section about setting up ANPR in the configuration file).

Viewing ANPR events in Watcher

Flussonic creates events of two types:

enter – a car number appeared in the field of view of a camera

leave – a car number left the field of view.

To see detected car numbers for a camera:

In the Watcher UI, go to Notifications. The list of all events opens.

To find certain events, use filtering and search capabilities on the right:

In Source, select Plate detector.

In From and To, select the date and time of beginning and end of the period when events were detected.

In Search, type the car number.
To delete the specified search criteria, click Clear Filter.

In Source, select Plate detector.

In From and To, select the date and time of beginning and end of the period when events were detected.

In Search, type the car number.

To delete the specified search criteria, click Clear Filter.

The list of events is filtered as you enter search parameters.

To view the recording of an event, click in the line with this event. The player will appear on the bottom right to play the recording. To enlarge the player, just double-click it.

The API of the ANPR module

After you correctly set up the recognition module, you can get events data in the JSON format via the Watcher API.

The detailed API reference is availabe at https://flussonic.github.io/watcher-docs/api.html.

Below is an example of an API request and a response:

```
root@ubuntu:~# curl localhost/vsaas/api/v2/events?type=activity -H 'x-vsaas-api-key: dfb21d1f-3e00-44a2-a706-36d99f9e9d73'
 {
  "start_at": 1538645882872,
  "type": "activity",
  "id": "7ecb0a13-414c-462f-a206-3c5d047baad4",
  "ext_data": null,
  "object_id": "A123AA 77",
  "end_at": null,
  "source": "plate_detector",
  "camera_id": "cam0-00",
  "source_id": "0",
  "object_class": "leave",
  "event_data": null
 }
```

start_at - the start time of an event

id - a unique ID of the record

object_id - car number

camera_id - camera name in Watcher

object_class - the event, which can be enter or leave (a car entered or left the camera's field of view).

Auto-login

Flussonic Watcher allows its users to login by a special URL (auto-login link), without entering a password. This may be necessary if you want to simplify the access to Flussonic Watcher for your users or prevent the transfer of passwords to third parties.

The auto-login link is issued to an authorized client.

In order to generate a URL for auto-login, you must first request a token by using the link /vsaas/api/generate-autologin-token. After that, the user can be authorized via POST to /vsaas/autologin.

Follow the steps:

1) First, generate an administrator's token. To do so, you'll need the KEY API. The KEY API is shown in the Flussonic Watcher administrative interface (admin panel, Watcher UI) on the Settings page.

```
sign = md5(salt + ":" + utc + ":" + api_key)
admin_token = salt + ":" + utc + ":" + sign
```

Where:

salt — any random string;

utc — the current UTC time in seconds;

api_key — API key from the settings page of the Flussonic Watcher admin panel.

The generated token has a finite lifetime and is bound to the specified time in UTC. For example, if

salt = "20a666"

utc = "1487258700"

api_key = "HELLO"

then the final resulting admin_token will be: "20a666: 1487258700: 4b60f36de708e5b3472155db2fea990a"

2) So, the admin token is ready, now you need to get a token for the autologin of a particular user. To do this, you'll need to make a POST request in JSON-format:

```
curl --header "X-Vsaas-Api-Key: ADMIN_TOKEN" --header "Content-Type: application/json" --request POST --data '{"login": LOGIN, "valid_till": VALID_TILL, "lifetime": LIFETIME}' "http://watcher.com/vsaas/api/generate-autologin-token"
```

ADMIN_TOKEN — the token generated in step 1. This token is transmitted in the HTTP headerX-Vsaas-Api-Key.

LOGIN — login (the same as an email) of the user to whom you want to give access. Line. Required.

VALID_TILL — the UTC time in seconds until this token is valid for autologin. Integer. Optional parameter.

LIFETIME — the duration of the session opened through the autologin, in seconds. Integer. Optional parameter.

The response will be in the JSON format:

```
{
   "autologin_token": "demo:1487258314:f8b1:b4bdaac58cbe94638e5b14a3728b8e6d633f3c6e",
   "success": true
}
```

You'll need this autologin_token.

3) The autologin_token received in step 2 is used for the POST request to Flussonic Watcher. For example, this way:

At the click on the submit button, the user will be logged in automatically into the Flussonic Watcher web interface.

User Interface

A user can log into Flussonic Watcher using an email and password after adding to the system.

New users can access cameras that were assigned to them by an administrator. Cameras may be public, available after authorization, and private.

A user interface has two screens: the first one shows cameras on a map and the second one features a list of cameras.

The following menus are available:

- Dashboard.
- Map.
- Favorite.
- Cameras.
- Users and Groups. (For Administrator)
- Settings. (For Administrator)

Dashboard

Dashboard — is a primary webpage for users to access cameras.

The following access filters are available:

- Any.
- Public.
- Authorized.
- Private.

Camera display modes:

- Medium Thumbnails.
- Large Thumbnails.
- List.

You can also toggle Compact and Hide inactive cameras.

The list below shows automatically updated screenshots from all cameras. In order to watch a certain camera or show its video archive, click on the respective image or Play button.

You can also click on the little arrow button on the top right corner of each block to access the video archive. To open camera in full screen, click on a camera name:

Mosaic shows up to 8 selected cameras on a single screen. To use a multi-camera view, create a client mosaic in the Cameras section.

Map

The map shows cameras that have coordinates configured.

Favorites

A user can mark a camera with a star. After that, it'll show up in the «Favorite» section.

Cameras

Here you can perform all operations with cameras:

- Add a camera.
- Camera search via ONVIF.
- Import Flussonic Cameras.

Also you can do the following:

- Export a list of files in CSV format.
- View camera serial numbers list.
- Create a Mosaic.

More information on adding cameras: «Camera management» section.

Users and Groups

In this menu, you can add and modify Flussonic Watcher users. For more information see «User Management».

Use groups to simplify user and camera navigation by making logical grouping (for example, by floors, departments, regions, neighborhoods, etc.), as well as simplify the user right management to provide camera visibility. See the «Groups» section for more information.

Settings

This menu is used to configure the Flussonic Watcher. More in the section «Settings».

Mobile Applications

Watcher offers mobile applications for real time access to a video surveillance system.

It provides:

- Watching live video from IP cameras with the ultra low latency
- Viewing the archive with no limits on its depth
- The access control based on a fingerprint or a PIN code
- TLS encryption of video streams
- Push notifications about events
- Downloading video screenshots.

Mobile applications need to know the address of your Watcher system to get video from it. Get your Operator ID that stores the URL of your Watcher system beforehand.

By default, the application tries to reach the Flussonic server, so a mobile app can't authenticate without a personal Operator ID.

To get your Operator ID, log in to your account, click on the installation key on the main portal screen and specify the public Watcher URL in the following format: http://APIKEY@watcher-host. The host address should be accessible from the Internet. It's highly recommended to use a domain name, not just IP address.

To start using the application, download it from the Apple Store or Google Play, and then authenticate using your Watcher login, password and the Operator ID that you received from manage.erlyvideo.org.

APIKEY — an authentication key that you can obtain in the Watcher settings UI.

Watcher-Hostname — a publicly visible path to Watcher.

Важно! Make sure that your server is visible from the Internet and that it has a permanent white IP address.

If your server is using NAT or a Firewall, contact our support and we will help with all necessary configuring on the terms of extended support.

Interface customization

The Watcher Cluster version provides all necessary tools for interface customization. Here is what you can do from the

Settings

menu:

- Custom header text. Add arbitrary text to the title of the interface.
- Custom page title. Change the title.
- Custom Logo. File requirements: .png image, height 50 px. A white or gray-colored logo with a transparent background will work the best.
- Login page custom logo.
- Custom color theme. You can select from three demo templates or fine-tune CSS. Save your custom CSS rule-sets to the file /vsaas/static/css/user.css. The file user.css.example can be used as an example.

Also you can customize the email password recovery template. Read more in the article Email customization.

Email customization

Setting the SMTP server

Configure your mailbox settings to make password recovery email reach your users. Add the mail server parameters to the configuration file.

For example, use these parameters:

Email: email.address@example.com;
Password: xyz123;
Mail server address: smtp.example.com;
Port: 465;
Connection security: SSL

Config file:

```
vsaas {
  database postgresql://vsaas:PASSWORD@localhost/vsaas_production;
  smtp_server smtp.example.com:465;
  smtp_login email.address:xyz123;
  email_from "Flussonic Watcher ";
  smtp_opts ssl;
}
```

Changing the template of forgot password emails

You can change the template for password recovery emails. Follow these steps:

- Go to /opt/flussonic/apps/vsaas/watcher/templates.
- Use the password_reset_request.email and password_changed.email text files as an examples.
- Add the custom_ file prefix to your custom template versions: custom_password_reset_request.email and custom_password_changed.email.
- If you want to use custom hypertext templates, add the .html extension to these files as follows: custom_password_reset_request.email.html and custom_password_changed.email.html. Save these files in the same directory.

The template consists of two parts, the header and the body. You can also add a subject to the header.

In addition, you can use variables {{data.base_url}} and {{data.token}} in the message body.

Text template example:

```
custom_password_reset_request.email:
---
subject: "ABC surveillance password reset"
---

Thank you for using the forgot password option. Follow this link to reset your password to the ABC surveillanc
e system:
```

{{data.base_url}}/vsaas/forgot-password/{{data.token}}

HTML template example:

custom_password_reset_request.email.html:

```
---
subject: "ABC surveillance password reset"
---
```

Thank you for using the forgot password option.to the ABC surveillance system:
Reset your password

Watcher API

You can integrate the Watcher video surveillance system into your system and create custom mobile applications using the flexible API.

Flussonic Watcher API let you import or export users and cameras individually or in bulk. API has the advanced integration options with client billing. It allows you manage the availability of cameras, private archives and billing, change passwords, check camera status and solve other tasks that are available in the GUI.

Also, the API let you configure authentication via RADIUS-server or use an authorization backend.

User Import API   Camera Import API   integration with billing   Change password   Backend for user authorization   RADIUS authorization

User import API

You can import users and settings from third-party systems, databases or spreadsheets using API. It requires API authorization. To do that, include the APIKEY that can be found in the Watcher settings in the

x-vsaas-api-key

HTTP header.

To import a list of users, create a CSV file and send it to:

http(s)://YOUR_WATCHER_URL/vsaas/api/import/users

CSV file fields:

email — the main user ID for authentication that is also used for password recovery.

password — plain text password.

is_active — 1: active user; 0: blocked user.

is_admin — 1: administrator user; 0: regular user.

groups — a list of groups to which the user belongs. Delimiter is a semicolon ';'.

note — a comment to this user.

Command-line import:

```
curl --data-binary @mydata.csv -H 'Content-type:text/plain; charset=utf-8' -H 'x-vsaas-api-
key: API_KEY_ВАШЕГО ВОТЧЕРА' http://WATCHER-HOSTNAME/vsaas/api/import/users
```

mydata.csv example:

```
email,password,is_active,is_admin,note,groups
ivanov@domain.tld,CergitMig,1,0,user1,grp1;grp2
petrov@domain.tld,LajQuolOy,0,1,user2,grp2
```

One string example:

```
echo -
e "email,password,is_active,is_admin,note,groups\nivanov@domain.tld,CergitMig,1,0,user1,grp1;grp2\npetro
v@domain.tld,LajQuolOy,0,1,user2,grp2" | curl http://127.0.0.1:8080/vsaas/api/import/users?type=csv --data-
binary @-  -H 'Content-type:text/plain; charset=utf-8' -H 'x-vsaas-api-key: 3a7d9386-6c3a-440d-a75d-
e6b3fdc3368e'
```

Response:

```
{"success": true}
```

Camera Import API

dd {   font-style: italic; /* cursive <em>/   }   dt {   margin-top: 1em; /</em> top margin */   }

A POST request is used to import the cameras to the following URL:
http(s)://YOUR_WATCHER_URL/vsaas/api/interchange/usercameras

Console:

```
curl http://127.0.0.1:8080/vsaas/api/interchange/usercameras --data-binary @mydata.csv -H 'Content-type:text/csv' -H 'x-vsaas-api-key: '
```

mydata.csv example:

```
stream_url,substream_url,thumbnails,onvif_url,onvif_profile,ptz,dvr_depth,dvr_path,enabled,access,title
rtsp://127.0.0.1:554,,,http://127.0.0.1:8899,000,0,3,storage,1,private,office_cam1
rtsp://127.0.0.2:554,,,http://127.0.0.2:8899,000,0,3,storage,1,private,office_cam2
```

One string example:

```
echo -e "stream_url,substream_url,thumbnails,onvif_url,onvif_profile,ptz,dvr_depth,dvr_path,enabled,access,title\nrtsp://127.0.0.1:554,,,http://127.0.0.1:8899,000,0,3,storage,1,private,office_cam1\nrtsp://127.0.0.2:554,,,http://127.0.0.2:8899,000,0,3,storage,1,private,office_cam2" | curl http://127.0.0.1:8080/vsaas/api/interchange/usercameras?type=csv --data-binary @-  -H 'Content-type:text/csv' -H 'x-vsaas-api-key: 3a7d9386-6c3a-440d-a75d-e6b3fdc3368e'
```

Response:

```
{"zu": 0, "cameras": [{"name": "office_cam2-689f1b1548", "created": true}, {"name": "office_cam1-c0ce3faa10", "created": true}], "users": [], "success": true, "zc": 2}}
```

CSV or JSON fields:

title: camera name.

name: stream name. Default is 'title' with a random suffix.

static: 1: constantly working stream, 0: on-demand stream.

stream_url: main stream RTSP URL.

substream_url: secondary stream RTSP URL (for multi-bitrate). Turned off by default.

thumbnails (string): camera snapshot URL. Send 0 instead of the URL to reset thumbnails. If you don't know the camera snapshot URL, use 1. Watcher will turn it on automatically.
Note: we recommend that you use a direct URL. Otherwise, the server load will increase at 10%. If you specify the URL, a direct communication with the camera is established, saving your server processing power.

onvif_url: camera URL to access via the onvif protocol. By default, it's set to 'no.'
Onvif_profile — UNFIV profile.

onvif_profile: ONFIV profile

ptz: (0 or 1) — turn PTZ off/on (if camera supports).

access: camera access type (private / public / authorized). Public will be accessible to all users, private — to a camera owner only. Private is a default value.

owner: camera owner login.

enabled: (0 or 1) turns camera on / off.

dvr_path: path to save the archive. Default is no archive.

dvr_depth: (integer in days) — number of days to store the archive. 0 — disables the archive.

coordinates:geographical coordinates.

postal_address: camera’s postal address.

comment: camera comment.

agent_model (string): camera model.

agent_serial (string): camera serial number.

agent_id (string): camera agent unique number.

agent_key (string): special field used for Watcher authorization.

agent_pin (string): special field used for Watcher authorization.

Client billing integration

Camera provisioning automation (i.e., billing redirect) requires a firmware update from Flussonic Agent. Flussonic Agent is an optional module within the Watcher software system that automates the camera activation process through the client billing. This module establishes a secure tunnel between cameras and Watcher that allows connecting through NAT, and also it improves the video transport stability. Firmware modification can be customized for each client's camera model.

Learn how Agent works

When the very first connection is established between the camera and Agent, a request is sent to the provisioning server (i.e., activation server). It redirects Agent to the client's server that has Watcher up and running.

After sending a request to the activation server, Agent generates a CSV and exports it to a client billing system. The export URL should be predefined.

The list of CSV attributes, sent from the Flussonic activation server to the client billing:

agent_model (string) - camera model.

agent_serial (string) - camera's serial number.

agent_id (string) - Agent UID on the camera.

agent_key (string) - special field used for Watcher authorization.

agent_pin (string) - special field used for Watcher authorization.

stream_url (string) - primary stream's RTSP-URL.

substream_url (string) - secondary stream's RTSP-URL.

thumbnails (string) - URL snapshots from the camera.

onvif_url (string) - URL at which the camera can be accessed via onvif protocol.

onvif_profile (string) -service field.

ptz (0 or 1) - turns PTZ on / off.

The activation server sends information to the client billing to create new objects for each camera. It will append new cameras to others currently registered in the billing system. Another camera registration system can also be used instead of the standard billing. The agent_serial parameter is used for camera registration.

After the registration, other attributes can be added per camera object on the client billing side.

Client-side billing CSV Attributes:

owner (string) — login of the connected account.

dvr_depth (integer) - number of days to store video archives for the camera. 0 — disables the archive.

enabled (0 or 1) - turns camera on or off.

access (private / public) - camera visibility. It can be either accessible by the camera owner or my all users.

Note: These attributes are not transmitted.

The user should develop the mechanism of billing plans and other logic that will affect access to the file and settings, and to a certain video archive's depth.

After you apply these settings (owner, dvr_depth, enabled, access), the data on all cameras and users will be automatically pushed to a CSV file here by the following:

http(s)://YOUR-WATCHER-URL/vsaas/api/interchange/usercameras?x-vsaas-api-key=(obtain the key in the Watcher settings)

User information also should be exported along with the camera information. User CSV Attributes: http(s)://YOUR-WATCHER-URL/vsaas/api/import/users

login (string) — the user's Watcher login.

is_active (0 limits watcher access) — cameras work, but the user can not enter Watcher.

dvr_allowed (0 limits video archive access) — cameras write to the archives, but users cannot have access to them.

It's possible to add cameras with users before activation. The process is described below:

1) Add a camera to the client billing (or another registration system) as an item with the attribute: agent_serial (serial number on a camera box). The following attributes should be set on the camera purchasing step:

owner — client login

dvr_depth — archive depth

enabled — camera status

access — access type

2) Then, when a technician turns on the camera in the client's home, the camera sends a request to the activation server, receives necessary activation attributes, then pushes the gathered information to the CSV file in the client billing. It will append new items based on the agent_serial attribute for previously added cameras.

3) When you have the camera information, it will be updated automatically.

Backend for user authorization

How it works:

An operator needs to realize the HTTP request handler, that has the logic to authenticate users.

The operator enters the path to the Flussonic Watcher authentication backend (Settings — Authentication backend).

A user logs into Flussonic Watcher using login/password.

Watcher transfers this data to the backend operator in a request payload.

Backend checks the incoming data.

If the user can be authenticated, the system returns the 200 HTTP code.

If the authentication data is incorrect, it returns the 403 HTTP code.

If the user is not found, the system gives 404.

Along with the login permissions, the backend can return the list of groups in the JSON body.

Flussonic Watcher checks the information about the groups in the database.

If the permission is granted, the password is passed as ancoded hash string and the group belongings are checked.

In case of decline, the user password is reset.

If the authentication backed was out of reach or send the response less than 2 seconds. The user check if being done in the Watcher’s database.

```
import falcon, json

class AuthResource:
  def on_get(self, req, resp):
    print "GET %r\n%r" % (req.uri, req.params)
    login = req.params.get('login', None)
    password = req.params.get('password', None)
    if not login or not password:
      print 'incorrect request login: %r, pass: %r' % (login, password)
      resp.status = falcon.HTTP_400
      return

    if login == 'user0':
      if password == 'letmein':
        return
      resp.status = falcon.HTTP_403
      return

    if login == 'user1':
      if password == 'letmein':
        resp.body = json.dumps(dict(groups=['a', 'b']))
        return
      resp.status = falcon.HTTP_403
      return
    resp.status = falcon.HTTP_404

app = falcon.API()
ad = AuthResource()

app.add_route('/auth', ad)
```

Examples

A user can get through with groups A and B:

```
curl -vvv http://localhost:8001/auth\?login\=user1\&password\=letmein

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user1&password=letmein HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
```

A user can get through without groups:

```
curl -vvv http://localhost:8001/auth\?login\=user0\&password\=letmein

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user0&password=letmein HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
```

A user can’t get through:

```
curl -vvv http://localhost:8001/auth\?login\=user0\&password\=wrong

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user0&password=wrong HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
```

A user is not found:

```
curl -vvv http://localhost:8001/auth\?login\=user10\&password\=wrong

*   Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 8001 (#0)
> GET /auth?login=user10&password=wrong HTTP/1.1
> Host: localhost:8001
> User-Agent: curl/7.47.0
> Accept: */*
>
```

RADIUS authentication

RADIUS server can be used to authenticate Watcher users. It is especially useful, if you have a large number of users.

The setting can be enabled via admin interface:

The address 'radius://ldap.example.com:1812/secret' consists of 3 parts: host, port and secret. Change it according to your RADIUS server settings. Now, when a user try to login, Watcher redirects to the server via RADIUS protocol. Watcher sends User-Name and User-Password in the Access-Request query. The RADIUS server responds by giving the group list for this user. An attribute Filter-Id(11) is used here. Each group is stored in a separate attribute.

- Watcher redirects to RADIUS on every user log in.
- If the RADIUS answers Access-Accept, Watcher logs user in and saves the HEX password and group belongings to the database.
- If the RADIUS answers Access-Reject, the user becomes locked in the database.
- If the RADIUS did not answer, Watcher searches a user in the database.

It is necessary to bear in mind that RADIUS should know about all users, including administrators. The administrator user attribute can not be transferred to the RADIUS response and it can be assigned through Watcher only.

API for mobile applications

POST: /vsaas/api/login

Example:

```
`curl -H 'Content-Type: application/json' -
d '{"email": "email@example.com", "password": "passwwd"}' http://localhost:8080/vsaas/api/login`
```

Parameters:

HTTP request payload

```
{"email": , "password": }
```

Reply: JSON

```
{
    success: boolean,
    session:
}
```

Adding a camera to Favorites

POST: /vsaas/my/fav/cameras

Parameters:

HTTP request payload

```
{"id": } // camera id
```

Reply: JSON

```
{"success": true}
```

Removing a camera from Favorites

DELETE: /vsaas/my/fav/cameras/{cam_id}

Parameters:

cam_id camera id

Reply: JSON

```
{"success": true}
```

All the available cameras

GET: /vsaas/api/my/cameras

Parameters:

HTTP request headers

x-vsaas-session: <session_token>

A list of favourites cameras

GET: /vsaas/api/my/cameras?filter=favorite

Parameters:

HTTP request headers

x-vsaas-session: <session_token>

Reply: JSON

```
{
   "id": string
   "title": string
   "access": string // [“private”, “public”, “authorized”]
   "comment": string
   "postal_address": string
   "coordinates": string // "55.7512 37.6184"
   "streamer_url": string // "http://demo-watcher.flussonic.com"
   "dvr_depth": int // the depth of the archive (per days)
   "permissions": { // the user rights status
      "dvr": boolean // archive status
      "ptz": boolean // PTZ control
      "view": boolean // live-view
   }
   "ptz": boolean // PTZ control status
   "server": string // demo-watcher.flussonic.com
   "static": boolean // stream type: static or on-demand
   "status": { // information about the stream condition
      "source_error": string
      "alive": boolean
   }
   "thumbnails": boolean // the camera's snapshots status
   "urls": { // the links of any protocols receiving the stream
      "hls": string
      "hds": string
      "jpg": string
      "rtmp": string
      "rtsp": string
      "media_info": string // URL information of the stream
      "recording_status": string // URL archive status
   }
}
```

Camera management

After the module is installed, you will see no camera items, even if they are present on Flussonic Media Server:

To add a camera to the system, click New camera :

Camera settings:

Title. Must be filled with English letters and numbers, as it will be used in URLs.

URL. The camera URL. If you have the camera URL, you can import the camera from the stream server (Flussonic) or use auto-search. For example: rtsp: //mycam.local/stream0

Substreams. Additional address. For example: rtsp: //mycam.local/stream1; rtsp: //mycam.local/stream2

Substreams. Additional address. For example: rtsp: //mycam.local/stream1; rtsp: //mycam.local/stream2

Thumbnails URL. Address the removal of screenshots from the camera.

Location. Location of the camera. Shows the camera placement on the map. You can change its placement. We recommend that you place all camera objects to their actual locations on the map to help users finding them.

Address. The postal address of the camera.

Owner. Choice of camera owner. Owners are configured in the [Users and Groups] menu (manage-users);

Access. The camera availability: to all users, only to registered users or to specific users:

Public (public) - available to all users of Flussonic Watcher, regardless of the settings of groups and users. When setting the value of public source appears in the list of cameras for all users of your server. Also, without restrictions on access, the camera can be published on a web page with permanent access without authorization.
For customers (authorized) — also available to all users, but you need to authorize on your Flussonic Watcher.
Private — is available to the user only when adding access to the camera in the Users and Groups section .

Public (public) - available to all users of Flussonic Watcher, regardless of the settings of groups and users. When setting the value of public source appears in the list of cameras for all users of your server. Also, without restrictions on access, the camera can be published on a web page with permanent access without authorization.

For customers (authorized) — also available to all users, but you need to authorize on your Flussonic Watcher.

Private — is available to the user only when adding access to the camera in the Users and Groups section .

Server. Selecting the Flussonic Media Server.

DVR. Recording depth. In other words, it’s a number of days after recording when the video archive is available. To enable this option, configure the repository in the System settings or Streamer settings.

Options.

Turn the camera on and off.
Constantly working camera or on request.
Image available for preview or not.
Restrict access to the camera archive.

Turn the camera on and off.

Constantly working camera or on request.

Image available for preview or not.

Restrict access to the camera archive.

Comment. Text comment. Auxiliary text that describes camera positioning and other information.

Press Save. After that, the camera will appear on the management page and in the dashboard:

Creating a client mosaic

Mosaic is the ability to show up to 16 selected cameras on a single screen.

Creating a mosaic

To create a mosaic, log into Flussonic Watcher as an administrator. Then do the following:

1) Go to the «Cameras» section.

2) Press «Mosaic».

3) Click the «Add» button.

4) In the window opened:

— Enter the mosaic name.

— Mark the «enabled» checkbox.

— Select the number of cameras.

— Transfer the selected cameras into the Internet/Ethernet. Use filters if there are lots of cameras.

5) Click «Save».

Edit or delete mosaic in the «Cameras» > «Mosaic» section.

All created mosaics are available in «Dashboard» > «Mosaic».

User Management

After the module installation, you will see one administrator user on the user management page:

Add a user

To add a new user, you must enter:

- Login
- Password
- Internal gray IP for auto login
- Email

Checkboxes note:

- Active. Check to have the account active.
- Administrator. Mark if the user needs administrative rights.
- DRV access. Mark if the user needs access to the archive of records.

You can also leave a note for the user.

Adding cameras

To add a camera user, click on the tag «0 shared cameras», then «Add» and in the opened list select those cameras that are available to the user.

Click «Delete» if you want to close the user's access to the camera.

By tag «0 groups» you can add a group of cameras to the user. More in the article Groups.

After adding cameras, they will be available to the user in the dashboard.

Groups

Use groups to simplify user and camera navigation by making logical grouping (for example, by floors, departments, regions, neighborhoods, etc.), as well as simplify the user right management to provide camera visibility. To do that, create a group with a few cameras and then add users to it. A user will have access to all private cameras of the group in a few clicks.

To create camera and user groups, go to the

Users and Groups

>

Panel

>

Add

. Next, enter the group name and description (if necessary), and click «Save».

Now, in the group list, you will see the new group.

By clicking on the «0 cameras» button, you can add cameras from the list, and if you click on the «0 users» button, you can add users to the group.

You can perform the same task if you click on the group name > «Cameras» > «Add».

You can turn on the file archive visibility (DVR) for each user in the list, as well as use an option to control the camera PTZ (certain cameras may not support it).

Important! If you use groups, it’s necessary to change the camera status to Private. It’ll give camera access to a limited number of users. A Public status makes a camera visible to all users, regardless of belonging to any group.

Sharing cameras

With the camera sharing tool in Flussonic Watcher, an administrator can give private camera access to certain users without creating a group.

To do this, go to the «Users and Groups» and click «Add» next to a selected user where it says «0 shared cameras».

Select cameras you want to add from the list and click on the «Add» button. After that, the selected user will have access to these private cameras without being their owner.

You can perform the same operation via the user card. Click on «0 cameras» at the top and use the option to add one or more cameras that will be shared with this user.

Camera sharing is available for administrators only, and there is no such an option for users yet.

Serial Numbers

If you purchased Watcher with camera agents, you need to create a user in the Users and Groups section to set a user auto-snap for this camera.

Then go to the Camera > Serial Numbers and add the serial number from the camera box (this camera will go to the end user).

Then, select the depth for the archive, assign this serial number to the end user, and provide login / password to the user.

When you turn the camera on for the first time, it will go through the activation procedure, provisioning to your server, and binding it to a certain end user using the serial number. The DVR setting that determines the archive depth will be applied as well.

From the user’s perspective, the process will require these steps:

- Turn on the camera.
- Connect the camera to the Internet.
- Log in to Watcher using the login/password.

After these steps, the camera is up and running, and the user can see the video stream from the web-application.

Import Cameras and Search

Import cameras from Flussonic Media Server

On a Flussonic Watcher launch, cameras will be available on the network. You can use either Onvif search or import streams from Flussonic Media Server.

You can import a Flussonic Watcher camera that was previously added to Flussonic Media Server.

Camera Search

Flussonic Watcher uses the WS-Discovery mechanism to search Onvif-compatible cameras. It can detect Ubiquity, Samsung and other cameras.

Some cameras require login and password for authorization. If you cannot find a camera, enter your login and password on the camera and search again.

Many cameras has two or more H264 streams, so you can add all of them as show on the video:

After you selected the camera profile, the correct value will appear in the RTSP URL. Enter a name that will be used for this camera.

See how to set the DVR path in the settings.