



Video Recognition Gateway (VIRGO)

Documentation Version = 3.048

Publish Date = August 19, 2022

Copyright © 2022 RealNetworks, Inc. All rights reserved.

SAFR® is a trademark of RealNetworks, Inc. Patents pending.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Note: Documentation pertaining to the macOS platform is no longer being actively maintained.

Contents

1	Video Recognition Gateway (VIRGO)	3
2	VIRGO System Requirements	4
3	VIRGO Installation Guide	5
4	VIRGO in the Video Feeds Window	8
5	VIRGO Command Line Interface	13
6	Video Feeds Properties	18
7	Processing Video Files	73
8	VIRGO Tools	86
9	Factory Configuration File	87
10	Docker	92
11	GPU Support	95
12	Service Logging	98
13	Service Monitoring	100
14	VIRGO Architecture	102
15	Troubleshooting	105

1 Video Recognition Gateway (VIRGO)

Video Recognition Gateway (VIRGO) is a daemon system which runs on a Portable Operating System Interface (POSIX)-compatible system. It receives video feeds from one or more cameras and it recognizes and tracks faces in those video streams in real time. It generates tracking events and sends those events to an event server. VIRGO daemons can be managed either through the VIRGO command line interface or through the video feeds window of the Desktop Client or the Web Console.

2 VIRGO System Requirements

2.1 Linux Requirements

Video Recognition Gateway (VIRGO) requires at least the following x86_64 CPU features:

- Ivy Bridge or better CPU architecture
- SSE4
- AVX

A Linux distribution must implement at least the following components:

- LSB support
- systemd

VIRGO on Linux is able to take advantage of GPUs to accelerate video decoding, image processing, face detection, and object detection. The GPU requirements are:

- Nvidia CUDA 10.1 compatible or newer

The following additional software components must be installed to allow VIRGO to run successfully:

- libcurl4
- libgomp1
- libatomic1
- libbsd0
- libv4l-0

To install the software components listed above, execute the following commands in a shell:

```
sudo apt-get update
sudo apt-get install libcurl4 libatomic1 libgomp1 libv4l-0 libbsd0
```

2.2 macOS Requirements

- macOS 10.11 or newer is required.
- Swift 5 Runtime Support for Command Line Tools must be installed. See this Apple support article for more information.

VIRGO requires the following x86_64 CPU features:

- Ivy Bridge or better CPU architecture
- SSE 4.2
- AVX

VIRGO on macOS is able to take advantage of GPUs to accelerate video decoding, image processing, face detection, and object detection. The GPU requirements are:

- Metal version 1

3 VIRGO Installation Guide

This page describes how to install the standalone Video Recognition Gateway (VIRGO).

Note: If you installed VIRGO when you installed SAFR Platform or SAFR Desktop, there's no need to install standalone VIRGO.

3.1 System Requirements

See the VIRGO System Requirements page before you start the VIRGO installation process. Note that VIRGO depends on certain 4r party libraries which must be installed before installing VIRGO.

3.2 Download the VIRGO Installer

The macOS and Linux standalone VIRGO installers can be downloaded from the SAFR Download Portal here: <https://safr.real.com/developers>

3.3 VIRGO Installer Package

This package installs VIRGO as a system or user daemon. The system daemon installation ensures that VIRGO will be able to run independently of any logged in user and it will start running as soon as the computer is booted up. Administrator privileges are required to complete the installation. VIRGO will look for factory default settings in the `/etc/virgo-factory.conf` file. The user installation, on the other hand, links VIRGO to the user who installed it. The VIRGO daemon will only be accessible to this user and it will only run while this user is logged in. However no administrator privileges are required to install and operate VIRGO in this mode. VIRGO will look for factory default settings in the `~/virgo-factory.conf` file.

The following sections describe how to use the platform-specific version of the VIRGO installer package.

3.3.1 macOS

Installer name: `Virgo.pkg`

Follow these steps to install VIRGO on your macOS machine:

1. Download the macOS VIRGO installer from the SAFR Download Portal here: <https://safr.real.com/developers>
2. Double click it.
3. The installer will guide you through the necessary steps to complete the installation. Note that you can choose between a system and user installation by selecting the appropriate install location option.

VIRGO will be installed into the following location:

- For a user installation: `~/Library/RealNetworks`
- For a system installation: `~/Library/RealNetworks`

3.3.2 Linux

Installer name: `virgo_installer.tar.gz`

Follow these steps to install VIRGO on your Linux machine:

1. Download the Linux VIRGO installer from the SAFR Download Portal here: <https://safr.real.com/developers>
2. Decompress the package: `tar -xzf virgo_installer.tar.gz`
3. Make sure that the necessary third-party library dependencies are installed. For a list of required libraries see the VIRGO system requirements documentation.

4. Run the installer script. The installer script will by default install VIRGO as a system daemon. Although we strongly recommend that you install VIRGO as a system daemon, we do support user daemon installations. You can explicitly specify the desired type of installation by passing the `--user` or `--system` option to the script:

- `virgo_installer/install.sh --user` installs VIRGO as a user daemon.
- `virgo_installer/install.sh --system` installs VIRGO as a system daemon.

VIRGO will be installed into the following location:

- System daemon installation: `/opt/RealNetworks`
- User installation: `~/RealNetworks`

The installer script will ask you for all necessary information and guide you through the installation process.

The final VIRGO configuration information is written to a factory configuration file which is stored in the required file system location from where VIRGO is able to read it. Note that for security reasons the factory configuration file is only readable and writeable by the user who owns the VIRGO daemon. The following code block shows an example of how to install VIRGO as a system daemon:

```
> sudo virgo_installer/install.sh
```

3.4 FAQ for macOS Installations

1. I've installed VIRGO as a system daemon. How do I change the factory configuration?

Place your custom factory configuration file in the `/etc` directory and then reset the VIRGO service like this:

```
Assuming that the factory configuration file is at:  
  
/etc/virgo-factory.conf  
  
> virgo service reset
```

2. I've installed the VIRGO Package. How do I uninstall VIRGO?

For system daemon installations, execute the following command from the Terminal:

```
> sudo /Library/RealNetworks/virgo/uninstall.sh
```

For user daemon installations, execute the following command from the Terminal:

```
> ~/Library/RealNetworks/virgo/uninstall.sh
```

3. I've installed the VIRGO as a user daemon. How do I stop *virgod*?

Execute the following command from the Terminal:

```
> launchctl bootout gui/$(id -u)  
    ~/Library/LaunchAgents/com.real.virgod.plist
```

This command terminates the *virgod* daemon. Keep in mind that the VIRGO command line tool will automatically restart *virgod* when you use it again.

3.5 FAQ for Linux Installations

1. I've installed VIRGO as a system daemon. How do I change the factory configuration?

Place your custom factory configuration file in the `/etc` directory and then reset the VIRGO service like this:

```
Assuming that the factory configuration file is at:  
  
/etc/virgo-factory.conf  
  
> virgo service reset
```

2. I've installed the VIRGO Package. How do I uninstall VIRGO?

For system daemon installations, execute the following command from a shell:

```
> sudo /opt/RealNetworks/virgo/uninstall.sh
```

For user daemon installations, execute the following command from the Terminal:

```
> ~/RealNetworks/virgo/uninstall.sh
```

3. I've installed VIRGO as a user daemon. How do I stop *virgod*?

Execute the following command in a shell:

```
> systemctl stop --user com.real.virgod.service
```

This command terminates the *virgod* daemon. Keep in mind that the VIRGO command line tool will automatically restart *virgod* when you use it again.

4 VIRGO in the Video Feeds Window

One of the primary ways to manage Video Recognition Gateway (VIRGO) feeds is via the Video Feeds window of the Desktop Client or the Web Console. The sections below describe how to do so.

4.1 Create a VIRGO Feed

Newly created VIRGO feeds have the following properties:

- The feed is not affected by subsequent changes to preferences made within the Desktop Client. If you change a preference within your Desktop Client, that change is not cloned to the existing VIRGO video feed(s).
- Feeds continue running and processing their video streams regardless of whether or not the Desktop Client is running or not.
- If you shut down your machine, the video feed will try to restart itself whenever your machine is turned on again.

4.1.1 Camera Feed Analyzer Method

Windows only.

The easiest way to create a VIRGO feed is to do the following:

1. Open the Desktop Client.
2. Connect a camera to the client in the *Camera Feed Analyzer* window. (i.e. the default window)
3. Press the **Add to Video Feeds for continuous processing in the background** button as shown below highlighted by the red arrow.

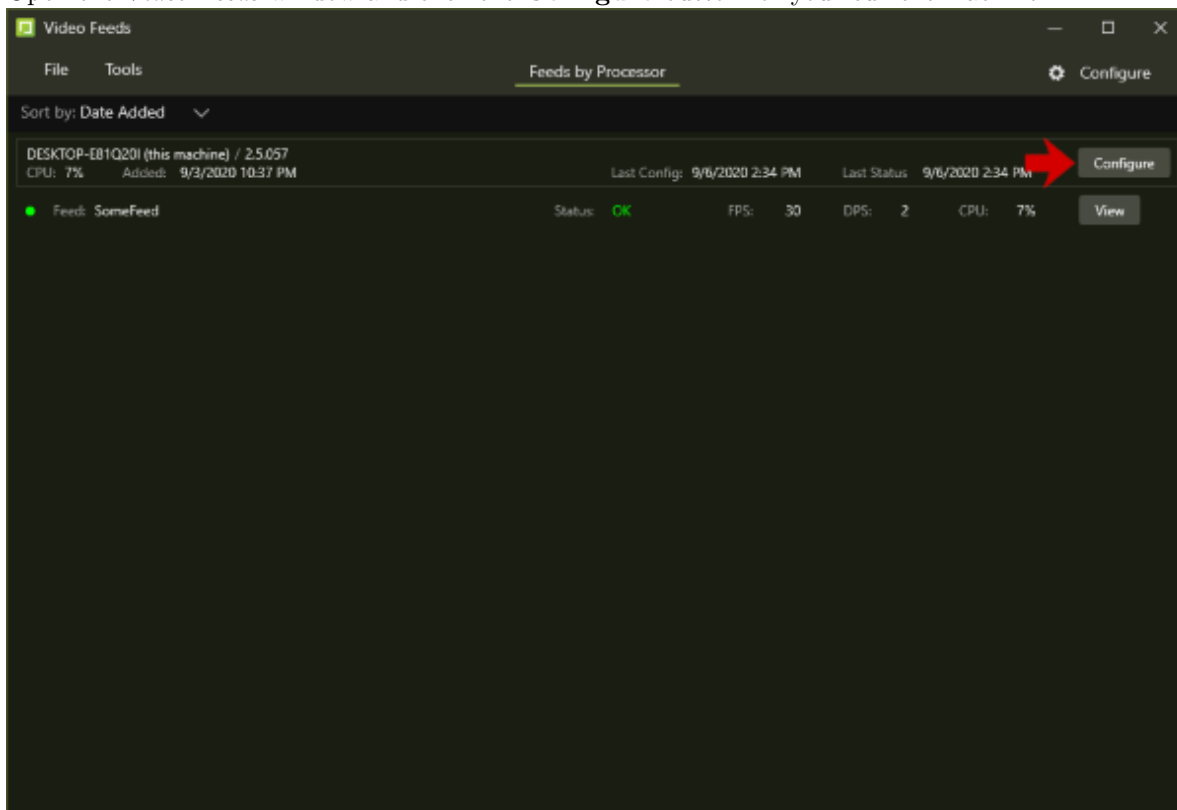


4. You'll be prompted for the following information:
 - **Feed Name:** Enter any name for the video feed you wish.
 - **Mode:** Select the operator mode from the drop-down menu that you want the VIRGO daemon to operate in. For a description of the operator modes, see here.
 - **Processor:** The machine where you want the VIRGO feed to run. The current machine is selected by default.
 - **Apply Mode Customizations from Preferences:** Enable if you want the Desktop Client preferences applied to the new VIRGO feed.

4.1.2 Manual Method

The manual way to create a VIRGO feed is as follows:

1. Connect a camera to SAFR.
2. Open the *Video Feeds* window and click the **Configure** button for your current machine.

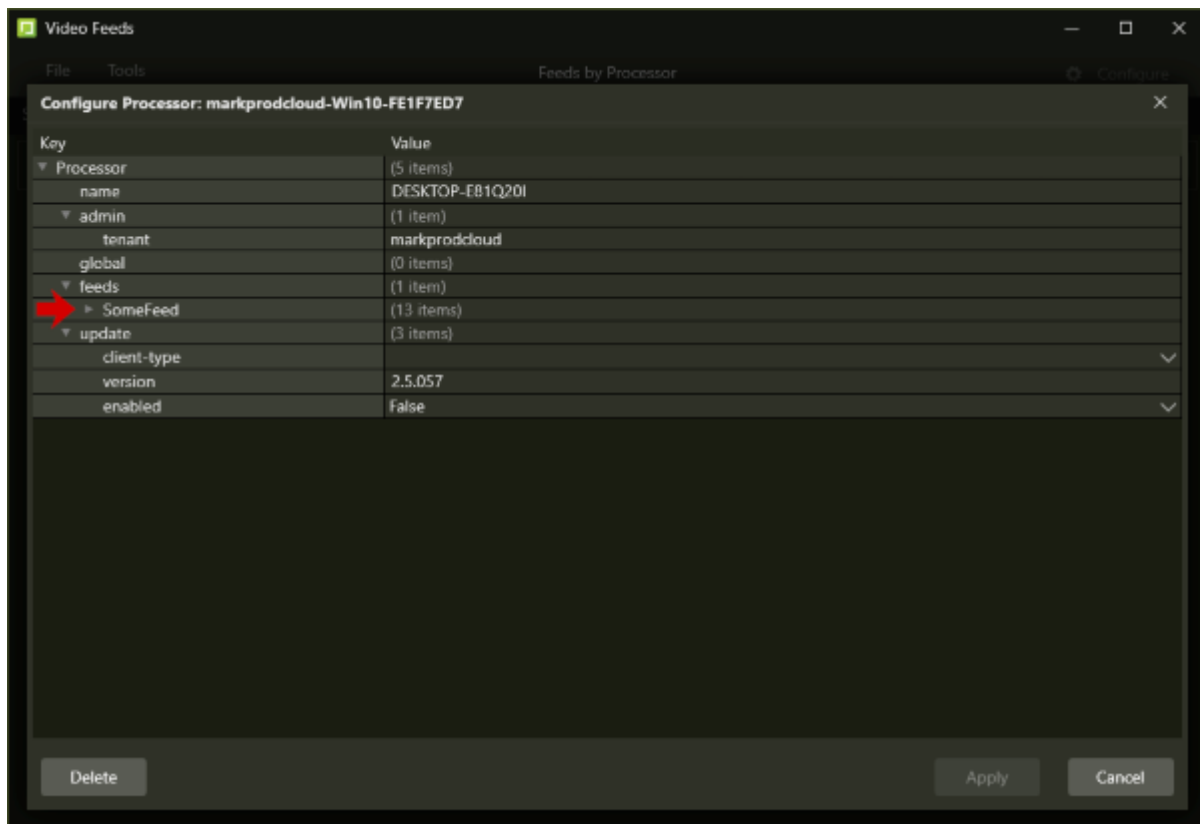


3. Hover your mouse over **Feeds**, then click the **+** button.
4. You'll be prompted for the following information:
 - **Feed Name:** Enter any name for the video feed you wish.
 - **Camera:** Select the camera feed you connected in Step #2 from the drop-down menu.
 - **Format:** Select the resolution and frame rate for the new video feed.
 - **Mode:** Select the operator mode from the drop-down menu that you want the VIRGO feed to operate in. For a description of the operator modes, see here.
 - **Apply Mode Customizations from Preferences:** Enable if you want the Desktop Client preferences applied to the new VIRGO feed.
5. Click the **Add** button.
6. Click **Apply** in the bottom right corner of the *Video Feeds* window.

Note: If you close the *Video Feeds* window without first clicking **Apply**, the video feed won't be created.

4.2 Manage VIRGO Feeds

To manage VIRGO feeds, open the *Video Feeds Window* within either the Desktop Client or the Web Console, then click the **Configure** button for your current machine. You'll see a screen similar to the following: (**Note:** You can often expose additional properties by clicking on the arrow next to entries, as shown by the arrow next to the *SomeFeed* entry below.)



- **name:** The name of the machine being managed.
- **admin:** Contains admin properties.

Property	Description
resource-type	
tenant	The tenant being managed.

- **global:** Contains global properties. See Global Properties for more information.
- **monitoring:** Monitoring properties allow you to monitor a video feed's health and send a notification email if one of the health metrics degrades to a certain level. To set up notification emails, do the following:
 1. Set up an SMTP Email Service on the Status Page of the Web Console.
 2. Enable one or more of the alarm conditions of the video feed's monitoring properties. There are 7 conditions available:
 1. **delinquent:** The video feed has stopped responding/sending status updates.
 2. **feed.error:** The video feed has encountered an error.
 3. **lowRAM:** The host machine has low RAM memory.
 4. **lowDisk:** The host machine has low hard drive storage space.
 5. **lowGPUMemory:** The host machine has low GPU memory.
 6. **lowCPU:** The host machine has low CPU processing power.
 7. **lowGPU:** The host machine has low GPU processing power.
 3. Set the subject and message properties for your enabled conditions.
 4. Set the threshold property for enabled conditions. (**Note:** The *delinquent* and *feed.error* conditions don't have threshold properties),
 5. Set the *alarm.mail.username* and *alarm.mail.password* properties to your email credentials.

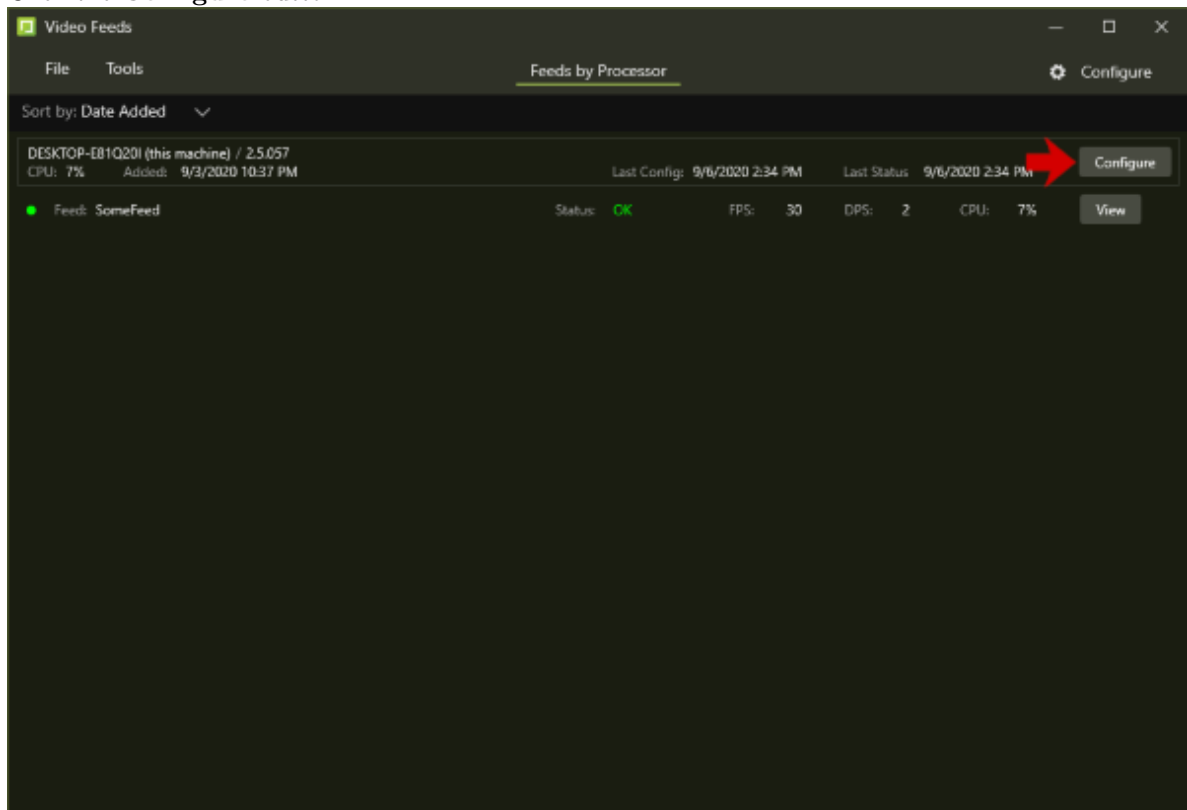
6. Set *alarm.mail.enabled* to TRUE to enable the notification emails.
See Monitoring Properties for a list of all the available monitoring properties.

- **feeds**: Specifies the feed properties. See Feeds Properties for more information.
- **update**: The update properties are not intended for public consumption at this time.

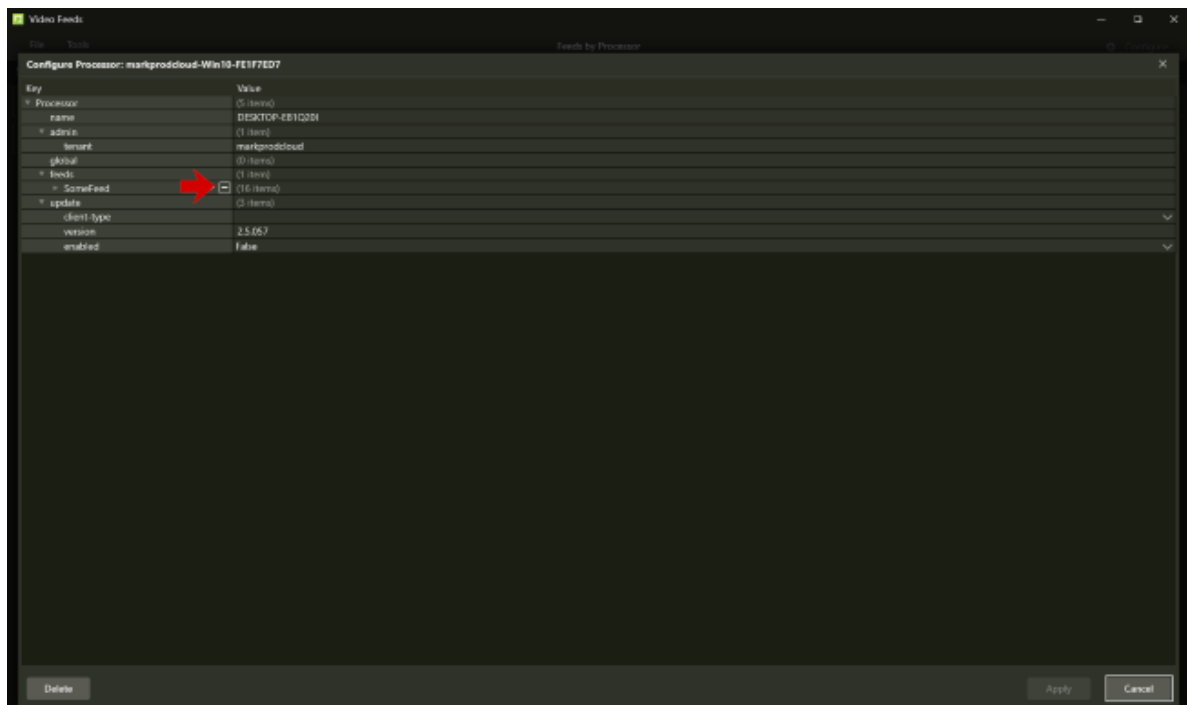
4.3 Terminate a VIRGO Feed

VIRGO feeds that haven't been terminated will continuously run in the background, and will automatically restart themselves after system shutdowns and reboots. Because each video feed consumes a significant amount of CPU resources, you'll want to terminate video feeds that are no longer of interest to you. Do the following to terminate a video feed:

1. Open the *Video Feeds Window* within either the Desktop Client or the Web Console.
2. Click the **Configure** button.



3. When you hover your mouse over name of the feed that you want to terminate, you'll see a + button and a - button. Click the - button.



4. Click the **Apply** button in the bottom right of the window. At this point, the feed will be terminated.
Note: If you click the **Cancel** button, the feed termination is undone and the feed will continue operating.

5 VIRGO Command Line Interface

The command line interface is designed based on an object - verb structure.

- Video Recognition Gateway (VIRGO) is conceptually organized into sub-systems which are represented by "objects".
- "Verbs" are commands that can be issued on an object.
- Some verbs may require additional parameters.

VIRGO currently defines the following types of objects (subsystems):

- **Service:** The VIRGO daemon itself.
- **Feed:** A video stream. (e.g. from a camera)
- **Environment:** The environment to which *virgod* connects.

The sections below describe the VIRGO command line syntax. Note that VIRGO command line options follow the standard Portable Operating System Interface (POSIX) convention. This means that many of those options come in a short (single dash prefix) and a long (double dash prefix) form.

5.1 Command Line Options

Help

```
> virgo --help
> virgo -h
<help text>
```

Shows all available VIRGO command line options.

5.1.1 Administrator

Get the current administrator configuration

```
> virgo administrator get
```

This command causes VIRGO to print the current administrator configuration. VIRGO may either be administrated by a cloud server (aka VIRGA) or it may be self-administrated. 'Virgo' is printed in the former case 'Virga' in the later.

Setting the administrator configuration

```
> virgo administrator set <name> // <name> is either 'virga' or 'virgo'
Administrator: <name>
```

This command causes VIRGO to switch to the specified administrator. Pass 'virga' if VIRGO should be administrated via the VIRGA server. Note that the environment definition must contain an admin-server-url entry in this case. Pass 'virgo' if VIRGO should be used standalone without a cloud command & control server. Standalone mode allows you to freely add, remove, and change feeds whereas the VIRGA administration mode requires that feeds are added, removed, and changed via VIRGA.

5.1.2 Service

Get information about the VIRGO service

```
> virgo service info
Version:      1.0.0
Target:       x86_64-macos
Domain:       System
Administrator: Virga
Environment:  PROD
```

```
Client ID:      <client-id>
Client Type:    <client type>
```

This command prints the following information about the installed VIRGO daemon build and its fundamental configuration.

- **Version:** The VIRGO build version.
- **Target:** Specifies for which operating system and CPU architecture the VIRGO daemon was built.
- **Domain:** Specifies whether the VIRGO daemon is running as a system-wide daemon (system) or a daemon which is only available to the currently logged in user (user). Note that user-wide VIRGO daemons will terminate when the user logs out.
- **Environment:** The environment to which the VIRGO daemon connects in order to receive commands from the command & control server.
- **Client ID:** The client ID that the VIRGO daemon sends to the command & control server to identify itself.

Get the current service status

```
> virgo service status
camera_1: ok
camera_2: ok
camera_3: inactive
```

This command tells VIRGO to print the current status of all registered feeds.

Monitor the current status of all feeds

```
> virgo service monitor
```

This command enables the service monitor. See Service Monitoring for more information.

Logging

```
> virgo service log <log specification>
```

This command displays the current service log. See Service Logging for more information.

Resetting the VIRGO daemon state

```
> virgo service reset
```

This command tells VIRGO that it should delete its current state and reinitialize it from the contents of the factory configuration file. This effectively resets the daemon back to the factory state.

Updating VIRGO

```
> virgo service update <version> <url> [--verbose]    // download an
install a new version.
> virgo service update <version> [--verbose]          // switch virgo to
a previously installed version. E.g. downgrade to an old version.
```

This command causes VIRGO to upgrade or downgrade to the specified version. <version> is the version to upgrade or downgrade to and <url> is a file or HTTP/HTTPS URL that points to VIRGO update archive. Specifying the update archive URL is only necessary if the version you are trying to switch to isn't already installed on the machine. By default VIRGO shows the current update status and progress. Specify the "-verbose" switch to cause VIRGO to show the full update log instead.

Get information about the installed VIRGO versions

```
> virgo service versions
Installed:
  1.0.0
  1.1.0
-> 1.2.0

Current:
  1.2.0
```

This command causes VIRGO to print the version numbers of all installed VIRGO packages plus the version number of the currently active and running VIRGO daemon.

5.2 Environment

A VIRGO daemon has a built-in list of supported environments. Only one of those environments can be active at a given time. The active environment determines to which VIRGA, face recognition, and event servers *virgod* and its *virgafeedd* child processes will talk.

List supported environments

```
> virgo environment list
DEV
INT2
LOCAL
PROD
```

Lists all environments supported by VIRGO.

Get the active environment

```
> virgo environment get [--verbose]
PROD
```

Returns the currently active environment. This is the environment to which *virgod* and all of its *virgofeedd* daemons connect. Additionally VIRGO will show the URLs of the individual servers in the environment if you pass the `--verbose` flag.

Set the active environment

```
> virgo environment set <environment name> [--verbose]
OK
```

Sets the environment which VIRGO and its feeds will use. Note that `<environment name>` must be one of the supported environments or one of the custom environments defined in the factory configuration file. Note that changing the environment also resets the VIRGO daemon back to the factory configuration.

By default the command prints "OK" if the switch to the new environment succeeds, while it prints an error if one or more services can not be contacted. You can pass the `--verbose` flag to get a detailed status for each service.

5.3 Cloud User

Get cloud account details

```
> virgo user get
User ID: <user id>
Password: ***
```

Prints the *User ID* and an indication whether a password was provided. Three asterisk characters indicate that VIRGO has a password on file, while an empty password line indicates that VIRGO doesn't have a password for the user on file.

Set the cloud account

```
> virgo user set
User ID: <user id>
Password: ***
```

Replaces the current cloud account's credentials with the provided *User ID* and *Password*. All currently enabled feeds are automatically restarted with the new account information.

5.4 Feeds

A single *virgod* daemon instance is capable of managing a set of video feeds. *Virgod* spawns one *virgofeedd* instance per feed and this *virgofeedd* instance is exclusively responsible for tracking its assigned feed. *Virgod* automatically respawns a *virgofeedd* instance if it dies unexpectedly.

A feed has:

- A name which is used to identify a particular feed.
- An RTSP URL which provides access to the video stream.
- Default face detection, recognition, and tracking parameters.
- Additional information to control features like lens correction.

Virgod stores the configuration information for a feed persistently. A feed can be added, removed, started, and stopped at any time. A VIRGO instance may come prepackaged with the configuration information for one or more feeds. New feeds may be added dynamically any time as long as *virgod* is running.

List feeds

```
> virgo feed list
camera_1
camera_2
```

Lists all enabled and disabled feeds that have been registered with VIRGO.

Get the configuration information for a feed

```
> virgo feed get <feed name>
{
  "active":true
  "url":"rtsp://camera.is.here/with/stream:8789"
  ...
}
```

Prints the feed configuration JSON dictionary.

Update/set the configuration information for a feed

```
> virgo feed set <feed name> <feed config file path>
```

Updates the current configuration of the feed with name <feed name>. The feed configuration file is read and the properties in the configuration file are applied to the current feed configuration stored in VIRGO. The feed configuration file is a JSON file with a single dictionary which contains the feed properties that you want to change. Note that the feed configuration file only needs to contain those properties that you want to change. See Video Feeds Properties for a list of supported feed properties.

Get the PID of a feed

```
> virgo feed get-pid <feed name>
53280
```

Prints the *PID* of the feed. -1 is printed if the feed is currently not active and thus no feed daemon is running to process the feed video stream.

Get the status of a feed

```
> virgo feed status <feed name>
ok
```

Prints the current status of a feed.

Add a new feed

```
> virgo feed add <feed name> <feed config file path>
```

Adds a new feed configuration to the persistent list of feeds. The feed name must be unique with respect to the VIRGO instance. The feed configuration is read from the supplied feed configuration file. The feed will immediately start processing if it is marked as "enabled" in the configuration file; otherwise the feed will be added to the persistent list of feeds but a separate "virgo feed start <feed name>" command will have to be issued to cause the feed to start running.

Remove an existing feed

```
> virgo feed remove <feed name>
```

VIRGO will stop the feed and then remove the feed configuration information from its persistent feed table.

Starting a feed

```
> virgo feed start <feed name>
```

VIRGO will mark the feed as active and start processing it. A video file feed starts processing from the beginning of the video while a camera feed starts processing from the current time code of the video stream. If the feed is already active and running this command instead does nothing.

Stopping a feed

```
> virgo feed stop <feed name>
```

Marks the feed as inactive and stops processing the video stream. If the feed is already marked as inactive, then this command instead does nothing.

Capturing an image from a feed

```
> virgo feed capture-image <feed name> <url or path> [--size
    <image_size>] [--max-frames <max_number_of_frames>] [--frame-delay
    <delay_in_milliseconds>]
```

Enables capturing of a single image or a series of consecutive images from the specified feed. <url or path> is a file or HTTP URL or a file system path. The URL/path is expected to point to a directory. VIRGO will create the directory if necessary and it will write all captured images to this directory. The size of the larger side of the capture image can be specified with the `--size` option. The default capture image size is 720 pixels. The maximum number of consecutive frames that should be captured can be specified with the `--max-frames` option. The default is to capture a single image. The `--frame-delay` option allows you to specify the delay between consecutive frames in milliseconds.

6 Video Feeds Properties

6.1 Global Properties

Below is a list of global processor properties.

Property	Default	Description
alarm-enabled	False	Enables alarms.
status-interval	500	Status reporting time interval in milliseconds.
avigilon	N/A	Exposes the properties related to the SAFR Avigilon integration.
genetec	N/A	Exposes the properties related to the SAFR Genetec and SAFR Genetec FaceRec integrations.
digifort	N/A	Exposes the properties related to the SAFR Digifort integration.
milestone	N/A	Exposes the properties related to the SAFR Milestone integration.
geutebrueck	N/A	Exposes the properties related to the SAFR Geutebrueck integration.
videoInsight	N/A	Exposes the properties related to the SAFR Panasonic Video Insight integration.

6.2 Monitoring Properties

Below is a list of all the monitoring properties.

Property	Default Value	Description
alarm.condition.delinquent.enabled	TRUE	Enables the delinquent alarm condition.
alarm.condition.delinquent.subject	"SAFR Feed Processor Unresponsive"	Sets the text of the subject line of the delinquent notification mail.
alarm.condition.delinquent.message	"SAFR Feed Processor %s is not responding."	Sets the text of the email body of the delinquent notification mail.
alarm.condition.feed.error.enabled	TRUE	Enables the feed.error alarm condition.
alarm.condition.feed.error.subject	"SAFR Feed Error"	Sets the text of the subject line of the feed.error notification mail.
alarm.condition.feed.error.message	"SAFR Feed Processor %s feed %s encountered an error %d: %s."	Sets the text of the email body of the feed.error notification mail.
alarm.condition.lowRAM.enabled	TRUE	Enables the lowRAM alarm condition.
alarm.condition.lowRAM.subject	"SAFR Feed Processor low on RAM"	Sets the text of the subject line of the lowRAM notification mail.

Property	Default Value	Description
alarm.condition.lowRAM.thresholdGB	0.5	The threshold, in GB, below which the lowRAM alarm condition is triggered.
alarm.condition.lowRAM.message	"SAFR Feed Processor %s RAM remaining is at %f.1GB which is below healthy threshold of %f.1GB."	Sets the text of the email body of the lowRAM notification mail.
alarm.condition.lowDisk.enabled	TRUE	Enables the lowDisk alarm condition.
alarm.condition.lowDisk.subject	"SAFR Feed Processor low on disk space"	Sets the text of the subject line of the lowDisk notification mail.
alarm.condition.lowDisk.thresholdGB	0	The threshold, in GB, below which the lowDisk alarm condition is triggered.
alarm.condition.lowDisk.message	"SAFR Feed Processor %s disk space remaining is at %f.1GB which is below healthy threshold of %f.1GB."	Sets the text of the email body of the lowDisk notification mail.
alarm.condition.lowGPUMemory.enabled	TRUE	Enables the lowGPUMemory alarm condition.
alarm.condition.lowGPUMemory.subject	"SAFR Feed Processor low on GPU memory"	Sets the text of the subject line of the lowGPUMemory notification mail.
alarm.condition.lowGPUMemory.thresholdGB	0.5	The threshold, in GB, below which the lowGPUMemory alarm condition is triggered.
alarm.condition.lowGPUMemory.message	"SAFR Feed Processor %s GPU memory remaining is at %f.1GB which is below healthy threshold of %f.1GB."	Sets the text of the email body of the lowGPUMemory notification mail.
alarm.condition.lowCPU.enabled	TRUE	Enables the lowCPU alarm condition.
alarm.condition.lowCPU.subject	"SAFR Feed Processor low on CPU"	Sets the text of the subject line of the lowCPU notification mail.
alarm.condition.lowCPU.thresholdPercent	50	The threshold, as a percentage, below which the lowCPU alarm condition is triggered.
alarm.condition.lowCPU.message	"SAFR Feed Processor %s CPU capacity remaining is at %f.1% which is below healthy threshold of %f.1%."	Sets the text of the email body of the lowCPU notification mail.
alarm.condition.lowGPU.enabled	TRUE	Enables the lowGPU alarm condition.
alarm.condition.lowGPU.subject	"SAFR Feed Processor low on GPU"	The threshold, as a percentage, below which the lowGPU alarm condition is triggered.
alarm.condition.lowGPU.thresholdPercent	50	The threshold, as a percentage, below which the lowGPU alarm condition is triggered.

Property	Default Value	Description
alarm.condition.lowGPU.message	"SAFR Feed Processor %s GPU capacity remaining is at %f.1% which is below healthy threshold of %f.1%."	Sets the text of the email body of the lowGPU notification mail.
alarm.mail.enabled	TRUE	Enables alarm notification mails.
alarm.mail.recipients	N/A	A comma-separated list of emails that specifies who should receive email notifications.
alarm.mail.username	N/A	The username for your email service.
alarm.mail.password	N/A	The password for your email service.

6.3 System Properties

Property	Default Value	Description
event.sync.enabled	False	Enables the automatic archiving of events, as described in the Web Console's Status Page and Desktop Client's System Configuration Window documentation.
identity.sync.enabled	False	Enables the automatic synchronization of identities, as described in the Web Console's Status Page and Desktop Client's System Configuration Window documentation.

6.4 Feeds Properties

Below is a list of all the supported video feed properties. **UI Setting Name** specifies the corresponding setting within the Desktop Client, and **UI Setting Location** specifies where the setting can be found in the Desktop Client.

6.4.1 Camera Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.back-channel.mobotix.cash-point	"None"	When the connected camera is a Mobotix camera, this property must be set to the configured cash point within the Mobotix app for the back-channel to work.	Cash Point	Camera Preferences menu
input.back-channel.type	"None"	When the connected camera is a Mobotix camera, you can set this property to "Mobotix MX" in order to have SAFR report STRANGER and RECOGNIZED event types to the camera. This feature is necessary if you want to make use of the Mobotix app. If the connected camera isn't a Mobotix camera, this property doesn't have any effect.	Back Channel	Camera Preferences menu
input.contrast-enhancement.enabled	FALSE	Enables contrast enhancement of the input video frame. This property is not supported by SAFR Inside.	Contrast Enhancement	Camera Preferences menu
input.contrast-enhancement.exposure-boost	0	The boost for contrast enhancement. This property is not supported by SAFR Inside.	Exposure Boost	Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.contrast-enhancement.low-light-threshold	0.02	Low-light-threshold for contrast enhancement. This property is not supported by SAFR Inside.	Low Light Threshold	Camera Preferences menu
input.lens-correction.enabled	FALSE	Enables lens correction for the camera. This property is not supported by SAFR Inside.	Lens Correction	Camera Preferences menu
input.lens-correction.k1	0	The "k1" lens correction factor. This property is not supported by SAFR Inside.	Coefficient K1	Camera Preferences menu
input.lens-correction.k2	0	The "k2" lens correction factor. This property is not supported by SAFR Inside.	Coefficient K2	Camera Preferences menu
input.mirroring.enabled	FALSE	Whether the video image should be mirrored before detection and recognition operations are executed. This property is not supported by SAFR Inside.	Enable mirroring	Camera Preferences menu
input.password	N/A	Password of the person accessing the video stream. This property is not supported by SAFR Inside.	Password	Camera Preferences menu
input.rotation.angle	0	Whether the video should be rotated before detection and recognition operations are executed. Valid values are 0, 90, 180, and 270. This property is not supported by SAFR Inside.	Rotate Image	Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.stream.rtsp.transport	"udp"	The transport protocol that should be used while accessing the RTSP video stream. Must be one of "udp", "tcp", or "udp-multicast". This property is not supported by SAFR Inside.	RTSP Transport Protocol	Camera Preferences menu
input.stream.url	N/A	The video stream URL. The URL must point to a RTSP, HTTP, or FILE stream. Note that if you want to point the input stream to a locally saved file on a Windows machine, VIRGO expects a Windows native path for this property. (e.g. C:\ProgramData\RealNetworks\SAFR\bin\foo.mp4)	Address	Camera Preferences menu
input.user-name	N/A	Username of the person accessing the video stream. This property is not supported by SAFR Inside.	User	Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.video-clock.enabled	TRUE	Enables enforcement of the video clock. Video files will be processed as fast as possible if the video clock is turned off. Streams are always processed as fast as possible so this value is ignored when input.type is set to "stream". Live streams play at the rate the data is sent, while file streams will be processed as fast as possible. When input.type is set to "file", then this value is used to determine whether the file should be processed at real time or faster than real time. This property is not supported by SAFR Inside.	Enforce timing for video frames	Camera Preferences menu
source	N/A	Source name.	Source	Camera Preferences menu
tracker.detect-direction-of-travel.person.bottom-boundary	0	The percentage of the bottom side of the camera view field to exclude from direction of travel event reporting. This property is not supported by SAFR Inside.	Bottom boundary	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.down	FALSE	Enables direction of travel detection in the downward direction. This property is not supported by SAFR Inside.	Down/Towards travel distance (the checkbox)	Direction of Travel Recognition section of the Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.detect-direction-of-travel.person.down-distance	0.1	The percentage of the camera view that a tracked person can travel in a downward direction before a direction of travel event is generated. This property doesn't have any effect if tracker.detect-direction-of-travel.person.down is set to FALSE. This property is not supported by SAFR Inside.	Down/Towards travel distance	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.left	FALSE	Enables direction of travel detection in the leftward direction. This property is not supported by SAFR Inside.	Left travel distance (the checkbox)	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.left-boundary	0	The percentage of the left side of the camera view field to exclude from direction of travel event reporting. This property is not supported by SAFR Inside.	Left boundary	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.left-distance	0.1	The percentage of the camera view that a tracked person can travel in a leftward direction before a direction of travel event is generated. This property doesn't have any effect if tracker.detect-direction-of-travel.person.left is set to FALSE. This property is not supported by SAFR Inside.	Left travel distance	Direction of Travel Recognition section of the Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.detect-direction-of-travel.person.right	FALSE	Enables direction of travel detection in the rightward direction. This property is not supported by SAFR Inside.	Right travel distance (the checkbox)	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.right-boundary	0	The percentage of the right side of the camera view field to exclude from direction of travel event reporting. This property is not supported by SAFR Inside.	Right boundary	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.right-distance	0.1	The percentage of the camera view that a tracked person can travel in a rightward direction before a direction of travel event is generated. This property doesn't have any effect if tracker.detect-direction-of-travel.person.right is set to FALSE. This property is not supported by SAFR Inside.	Right travel distance	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.top-boundary	0	The percentage of the top side of the camera view field to exclude from direction of travel event reporting. This property is not supported by SAFR Inside.	Top boundary	Direction of Travel Recognition section of the Camera Preferences menu
tracker.detect-direction-of-travel.person.up	FALSE	Enables direction of travel detection in the upward direction. This property is not supported by SAFR Inside.	Up/Away travel distance (the checkbox)	Direction of Travel Recognition section of the Camera Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.detect-direction-of-travel.person.up-distance	0.1	The percentage of the camera view that a tracked person can travel in an upward direction before a direction of travel event is generated. This property doesn't have any effect if tracker.detect-direction-of-travel.person.up is set to FALSE. This property is not supported by SAFR Inside.	Up/Away travel distance	Direction of Travel Recognition section of the Camera Preferences menu

6.4.2 Detection Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.detect-badges	FALSE	Whether detection of badges should be enabled for this feed. This property is not supported by SAFR Inside.	Enable badge detector	Badge detector section of the Detection Preferences menu
detector.detect-faces	TRUE	Whether detection of faces should be enabled for this feed.	Enable face detector	Face detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.detect-faces-input-size	"normal"	<p>Face detector input size. Applicable only when high-sensitivity detection service is used. Possible values:</p> <p>small - 320x240 or 320x180 or 240x320 or 180x320 (whichever fits better)</p> <p>normal - 640x480 or 640x360 or 480x640 or 360x640 (whichever fits better)</p> <p>large - 1280x720 or 720x1280 (whichever fits better)</p>	Input Size	Face detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.detect-faces-service	"auto"	<p>Specifies which face detection service will be used:</p> <p>standard - The standard facial detection service that SAFR uses.</p> <p>high-sensitivity - A high sensitivity facial detection service which has a lower latency and whose performance doesn't degrade when multiple faces are being analyzed simultaneously. The high sensitivity service consumes many more GPU resources than the standard service.</p> <p>auto - This value will automatically select the high sensitivity service if sufficient GPU resources are available to run it. If there are insufficient GPU resources, then the standard service is used instead.</p> <p>This property is not supported by SAFR Inside.</p>	Detection service	Face detector section of the Detection Preferences menu
detector.detect-people	FALSE	<p>Whether detection of people should be enabled for this feed. This detects any part of a person's body and not just the face. This property is not supported by SAFR Inside.</p>	Enable person detector	Person detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.detect-people-every-n-frames	1	This can be used to avoid running person detection on every frame. Since person detection requires a lot of GPU processing if the hardware is not powerful enough this value can be changed so that we only attempt to detect people every Nth frame to save processing power to keep up with realtime detection. This property is not supported by SAFR Inside.	Detect persons every	Person detector section of the Detection Preferences menu
detector.detect-people-input-size	"normal"	Valid values: normal - 416 pixel input. People detection balances speed and accuracy for best results. small - 320 pixel input. People detection will be the fastest with this input, but least accurate. large - 608 pixel input. People detection will be the slowest with this input, but most accurate. This property is not supported by SAFR Inside.	Input Size	Person detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.detect-people-model	"balanced"	Valid values: max-accuracy - Use a larger model for better accuracy, but the speed will be slower. max-speed - Use a smaller model for faster speed, but the accuracy will be lower. balanced - Use a larger model for better accuracy, but the precision will be slightly lower resulting in faster speeds than the <i>max-accuracy</i> model without sacrificing too much accuracy. This property is not supported by SAFR Inside.	Detection service	Person detector section of the Detection Preferences menu
detector.detect-rgb-liveness	FALSE	Enables the RGB liveness detector. This property is not supported by SAFR Inside.	Enable RGB liveness detector	RGB liveness detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.face-sensitivity-threshold	0	The sensitivity threshold when using the <i>High Sensitivity</i> facial detection service. The lower this value is, the more lenient the facial detection service will be when attempting to recognize a face, which can result in additional false positives. This setting is only available if you selected <i>High Sensitivity</i> for the Detection service setting above. This property is not supported by SAFR Inside.	Detection Sensitivity Threshold	Face detector section of the Detection Preferences menu
detector.final-face-selection-threshold	0.9	The final face candidate threshold that is used during face detection. This property is not supported by SAFR Inside.	Final candidate selection threshold	Face detector section of the Detection Preferences menu
detector.initial-face-selection-threshold	0.8	The initial face candidate threshold that is used during face detection. This property is not supported by SAFR Inside.	Initial candidate selection threshold	Face detector section of the Detection Preferences menu
detector.maximum-concurrent-detections	1	The maximum number of concurrent detections to allow. 0 means to automatically set this. This property is not supported by SAFR Inside.	Maximum detectors per feed	Face detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.maximum-input-resolution	720	Maximum resolution of the Input image. Bigger images are scaled down (aspect-ratio preserving) to this resolution before detection.	Reduce vertical input image size to	Face detector section of the Detection Preferences menu
detector.maximum-input-resolution-badges	4320	Maximum resolution of the Input image. Bigger images are scaled down (aspect-ratio preserving) to this resolution before detection. This property is not supported by SAFR Inside.	Reduce vertical input image size to	Badge detector section of the Detection Preferences menu
detector.middle-face-selection-threshold	0.85	The middle face candidate threshold that is used during face detection. This property is not supported by SAFR Inside.	Middle candidate selection threshold	Face detector section of the Detection Preferences menu
detector.minimum-consecutive-detections-required-person	0	This is the number of consecutive detections that are required before reporting that the person (based on object id) was actually detected and can be used to filter out false positives. This property is not supported by SAFR Inside.	Consecutive confirmation required	Person detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.minimum-required-badge-size	0	The minimum size of badges to accept from the detector. Only badges with at least this size are eligible for recognition. This property is not supported by SAFR Inside.	Minimum required badge size	Badge detector section of the Detection Preferences menu
detector.minimum-required-face-size	0	The minimum size of faces to accept from the detector. Only faces with at least this size are eligible for recognition.	Minimum required face size	Face detector section of the Detection Preferences menu
detector.minimum-required-person-to-screen-height-proportion	0	Specifies the ratio of the person to the screen height. This can be between 0 - 1 and allows for decimal precision. For example, if you don't want the person to show up unless they are greater than 25% of the image height then specify a value of 0.25. This property is not supported by SAFR Inside.	Minimum required person to screen proportion	Person detector section of the Detection Preferences menu
detector.minimum-searched-badge-size	20	The badge detector is advised to search for badges of at least this size. This value is applied while searching the image. This property is not supported by SAFR Inside.	Minimum searched badge size	Badge detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.minimum-searched-face-size	80	The face detector is advised to search for faces of at least this size. This value is applied while searching the image. This property is not supported by SAFR Inside.	Minimum searched face size	Face detector section of the Detection Preferences menu
detector.person-detection-threshold	0.4	This is the detection threshold to use when matching objects. The higher the threshold the more strict the matching will be and the higher the confidence will be that the actual object matches. This property is not supported by SAFR Inside.	Person detection threshold	Person detector section of the Detection Preferences menu
detector.person-separation-threshold	0.45	This threshold controls the object separation when the objects are overlapping. This determine how much overlap is needed before no longer detecting the object with the weaker footprint. This property is not supported by SAFR Inside.	Person separation threshold	Person detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.rgb-liveness-detection-scheme	"strict-multimodal"	<p>Specifies which RGB liveness model(s) should be used:</p> <p>texture-unimodal: Only the <i>Texture</i> model will be used.</p> <p>context-unimodal: Only the <i>Context</i> model will be used.</p> <p>strict-multimodal: Both the <i>Texture</i> and <i>Context</i> models will be used. For a subject to pass the RGB liveness test, both of the results of the models must meet or exceed the Liveness detection threshold value.</p> <p>normal-multimodal: Both the <i>Texture</i> and <i>Context</i> models will be used. For a subject to pass the RGB liveness test, the average of the results of the two models must meet or exceed the Liveness detection threshold value.</p> <p>tolerant-multimodal: Both the <i>Texture</i> and <i>Context</i> models will be used. Subjects pass the RGB liveness test when the result of either model meets or exceeds the Liveness detection threshold value.</p> <p>This property is not supported by</p>	Detection scheme	RGB liveness detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.rgb-liveness-evaluate-fake-over-n-frames	10	The number of frames over which fakeness should be evaluated. This property is not supported by SAFR Inside.	Evaluate fake over	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-evaluate-over-n-frames	10	The number of frames over which RGB liveness detection should be evaluated. This property is not supported by SAFR Inside.	Evaluate liveness over	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-fake-threshold	0.2	Specifies how difficult it will be for a subject to be verified as NOTLIVE_CONFIRMED. This property is not supported by SAFR Inside.	Fake detection threshold	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-minimum-center-pose-quality	0.2	The minimum face center pose quality for RGB liveness detection to be used. This property is not supported by SAFR Inside.	Minimum required center pose quality	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-minimum-confirmed-percent	0.85	The percentage of frames that must meet the liveness or fake threshold for the subject to be declared either LIVE-NESS_CONFIRMED or NOTLIVE_CONFIRMED. This property is not supported by SAFR Inside.	Minimum confirmations required	RGB liveness detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.rgb-liveness-minimum-face-context-percent	1.0	The minimum required extra context around faces for the <i>Context</i> model of RGB liveness detection to be used. This property is not supported by SAFR Inside.	Minimum required face context size	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-minimum-face-contrast-quality	0.45	The minimum face contrast quality for RGB liveness detection to be used. This property is not supported by SAFR Inside.	Minimum required face contrast quality	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-minimum-face-sharpness-quality	0.45	The minimum face sharpness quality for RGB liveness detection to be used. This property is not supported by SAFR Inside.	Minimum required face sharpness quality	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-minimum-face-size	150	The minimum required height and width of a face, in number of pixels, for the <i>Texture</i> model of RGB liveness detection to be used. This property is not supported by SAFR Inside.	Minimum required face size	RGB liveness detector section of the Detection Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
detector.rgb-liveness-minimum-preliminary-threshold	0.19	For multimodal detection schemes , this is the liveness threshold which the first evaluated model (the <i>Texture</i> model) must exceed before SAFR will bother evaluating the second model. If this threshold is not met, SAFR immediately returns NOTLIVE_CONFIRMED for the subject. This property is not supported by SAFR Inside.	Minimum preliminary liveness threshold	RGB liveness detector section of the Detection Preferences menu
detector.rgb-liveness-threshold	0.6	Specifies how difficult it will be for a subject to be verified as LIVE-NESS_CONFIRMED. This property is not supported by SAFR Inside.	Liveness detection threshold	RGB liveness detector section of the Detection Preferences menu
tracker.minimum-required-consecutive-badge-detections	0	The number of consecutive detections that are required before reporting that the object (based on object id) was actually detected. This property can be used to filter out false positives. This property is not supported by SAFR Inside.	Consecutive confirmations required	Badge detector section of the Detection Preferences menu

6.4.3 Tracking Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.enable-face-bounds-prediction	TRUE	Enables face bounds prediction, which predicts which direction the face is moving to maintain tracking.	Enable motion prediction	Tracking Preferences menu
tracker.enable-face-size-correlation	TRUE	Enables face correlation of tracked faces, which compares detected faces looking for a change in area.	Enable correlation of faces by size	Tracking Preferences menu
tracker.enable-high-precision	FALSE	Enables high precision tracking, which decreases event fragmentation and increases the stickiness of SAFR's tracking algorithm at the cost of computer processing power. This property should be enabled if you are experiencing duplicate or missing Direction of Travel events. See Camera Preferences for information about the Direction of Travel feature. This property is not supported by SAFR Inside.	High precision tracking	Tracking Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.failed-recognition-back-off-interval	340	After making the initial recognition attempts as quickly as possible, back up the amount specified by this setting for each subsequent recognition. This continues until the retry interval is reached.	Failed recognition back-off interval	Tracking Preferences menu
tracker.failed-recognition-retry-interval	0	The interval in which to run recognition requests if the face has not been recognized.	Retry failed recognition after every	Tracking Preferences menu
tracker.identity-relearn-interval-days	0	Updates the identity only when the currently saved identity is older than the updated identity.	Relearn face interval days	Tracking Preferences menu
tracker.identity-update-better-image	FALSE	Updates the identity when the currently saved identity is of lower quality (in all aspects) than the new image.	Update identity with better image	Tracking Preferences menu
tracker.initial-recognition-attempts	3	The number of initial recognition attempts to make on an unrecognized person as fast as possible.	Initial recognition attempts	Tracking Preferences menu
tracker.maximum-linger-frames	30	Determines for how many frames more we continue to keep a tracked face around after we have failed to detect it in the most recent frame. This makes the tracker resilient against intermittent loss of face.	Stop tracking a face after it has lingered for	Tracking Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.max-position-change-relative-to-face	115	The maximum position change, specified in percentage relative to the face size, to continue tracking.	Maximum change to continue tracking, Face position	Tracking Preferences menu
tracker.max-size-change-relative-to-face	50	The maximum size change, specified in percentage relative to the object size, to continue tracking.	Maximum change to continue tracking, Size	Tracking Preferences menu
tracker.min-failed-recognitions-to-stop-tracking-identity	3	When a face is being tracked recognitions are continually confirming the identity. The identity is also being verified if it is transferred from a person object. In these cases, if the recognition or verification fails this number of consecutive times then the identity will be reset and no longer associated with the face because we are no longer sure it is the same identity.	Minimum failed recognitions to stop tracking identity	Tracking Preferences menu
tracker.minimum-number-identical-recognitions-learn	2	The number of consecutive recognitions that need to occur before adding a new identity to the system.	Minimum recognitions to learn identity	Tracking Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.minimum-number-identical-recognitions-lock	1	The number of consecutive recognition attempts that we must run and produce the same person identity before we lock onto this identity.	Minimum recognitions to lock on to identity	Tracking Preferences menu
tracker.minimum-required-consecutive-mask-detections	1	The number of consecutive detections that are required before reporting that the masked person was detected. This property can be used to filter out false positives. This property is not supported by SAFR Inside.	Consecutive Mask Detections	Tracking Preferences menu
tracker.reconfirm-identity-in-video-on-every-key-frame	FALSE	When a key frame is encountered in a video file all the faces that are being tracked are marked as unconfirmed so that their identities are reconfirmed to make sure they are the same person. This setting only applies to video files; it can't be used with live video. This property doesn't actually appear in the VIRGO property list; possibly delete.	Reconfirm identity in video after each Key Frame	Tracking Preferences menu
tracker.reconfirmation-interval	1000	Identity reconfirmation time interval in ms.	Reconfirm identity after every	Tracking Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
tracker.stop-tracking-on-failed-re-recognition	FALSE	If recognition fails when re-recognizing a person then delete the identity that was created.	Stop tracking on failed re-recognition	Tracking Preferences menu

6.4.4 Recognition Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.3d-liveness-threshold	0.6	Windows only. Specifies the 3d liveness threshold. This property is not supported by SAFR Inside.	Liveness threshold	Detect section of the Recognition Preferences menu
recognizer.detect-3d-liveness	FALSE	Windows only. Enables 3D liveness detection on Intel RealSense D415 and D435 cameras. This property is not supported by SAFR Inside.	3D Liveness	Detect section of the Recognition Preferences menu
recognizer.detect-age	FALSE	Enables the detection of age information.	Age	Detect section of the Recognition Preferences menu
recognizer.detect-gender	FALSE	Enables the detection of gender information.	Gender	Detect section of the Recognition Preferences menu
recognizer.detect-identity	TRUE	Enables detection of an identity, which matches against the existing database of people (identities).	Identity	Detect section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.detect-mask	FALSE	When enabled, SAFR will evaluate all occluded faces to see if they're covered by a mask. If they are, then SAFR will use the mask enhanced model to attempt to recognize the face behind the mask. If the occluded face isn't covered by a mask, then the normal occluded model will be used instead.	Mask	Detect section of the Recognition Preferences menu
recognizer.detect-mask-threshold	0.5	Specifies the threshold at and above which mask detection will conclude that <i>mask=true</i> .	Mask Detection Threshold	Detect section of the Recognition Preferences menu
recognizer.detect-occlusion	FALSE	Enables occlusion detection during recognition.	Occlusion	Detect section of the Recognition Preferences menu
recognizer.detect-pose-action	FALSE	Enables pose liveness detection. This property is not supported by SAFR Inside.	Pose liveness action	Detect section of the Recognition Preferences menu
recognizer.detect-rgb-action	FALSE	Enables RGB liveness detection. Note that the RGB Liveness Detector on the Detection Preferences menu must be enabled for RGB liveness detection to work. This property is not supported by SAFR Inside.	RGB liveness action	Detect section of the Recognition Preferences menu
recognizer.detect-sentiment	FALSE	Enables the detection of sentiment information.	Sentiment	Detect section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.detect-smile-action	FALSE	Enables the smile action recognizer. This property is not supported by SAFR Inside.	Smile action	Detect section of the Recognition Preferences menu
recognizer.identity-masked-threshold-offset	0	Sets the identity threshold when detecting masks.	Masked face threshold offset	Identity recognition threshold section of the Recognition Preferences menu
recognizer.identity-proximity-threshold-allowance	0.13	A boost value that is added to the Identity Recognition Threshold.	Proximity threshold allowance	Identity recognition threshold section of the Recognition Preferences menu
recognizer.identity-recognition-threshold	0.54	Identity recognition threshold.	Identity recognition threshold	Identity recognition threshold section of the Recognition Preferences menu
recognizer.learning-enabled	FALSE	Enables the feed to learn new identities.	Identity	Detect section of the Recognition Preferences menu
recognizer.learn-occluded-faces	FALSE	Enables learning of occluded faces regardless of the maximum occlusion setting. If this is true then the server configuration will be used, which by default doesn't do any occlusion detection.	Allow learning of occluded faces	Detect section of the Recognition Preferences menu
recognizer.maximum-0.1 clip-ratio		The maximum clip ratio on either side the recognition candidate might have.	For recognition	Clipping tolerances section of the Recognition Preferences menu
recognizer.maximum-0 clip-ratio-identification		The maximum clip ratio on either side the insertion candidate might have.	For learning/strangers	Clipping tolerances section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.maximum-5 concurrent- recognitions		The maximum number of concurrent recognitions to allow. 0 means to automatically set this. This property is not supported by SAFR Inside.	Maximum recognizers per feed	Recognition Preferences menu
recognizer.maximum-0 occlusion		The maximum occlusion value that is allowed when adding new candidate images into the Person Directory. If the face is occluded with a value greater than this then the face will not be added, but if it's less than or equal to this value then it will be added.	Occlusion threshold	Detect section of the Recognition Preferences menu
recognizer.maximum-0.4 pitch- identification		The maximum pitch value used to determine if the face is looking straight ahead. The pitch value is the forward/backward movement of the face.	Max Pitch	Minimum required center pose quality section of the Recognition Preferences menu
recognizer.maximum-0.15 roll-identification		The maximum roll value used to determine if the face is looking straight ahead. The roll value is the side to side tilt movement of the face.	Max Roll	Minimum required center pose quality section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.maximum-yaw-identification-0.4		The maximum yaw value used to determine if the face is looking straight ahead. The yaw value is the side to side movement of the face.	Max Yaw	Minimum required center pose quality section of the Recognition Preferences menu
recognizer.minimum-center-pose-quality-0.05		The minimum center pose quality that a recognition candidate must have in order to allow the addition of the candidate image into the Person Directory.	For recognition	Minimum required center pose quality section of the Recognition Preferences menu
recognizer.minimum-center-pose-direct-gaze-0.70		If a face's center pose quality is above this value, the face is determined to be gazing directly at the camera, but if the center pose quality is below this value, the face is determined to be turned away. The longer the face gazes directly at the camera, the longer events' directGazeDuration is. This property is not supported by SAFR Inside.	For direct gaze detection	Minimum required center pose quality section of the Recognition Preferences menu
recognizer.minimum-center-pose-quality-identification-0.45		The minimum center pose quality that a recognition candidate must have in order to allow the addition of the candidate image into the Person Directory.	For learning/strangers	Minimum required center pose quality section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.minimum-center-pose-quality-merging	0.59	The minimum center pose quality that a recognition candidate must have in order to allow merging.	For merging	Minimum required center pose quality section of the Recognition Preferences menu
recognizer.minimum-face-contrast-quality	0.1	The minimum face contrast quality that a face image must have before recognition is attempted.	For recognition	Minimum required face contrast quality section of the Recognition Preferences menu
recognizer.minimum-face-contrast-quality-identification	0.3	The minimum face contrast quality that a recognition candidate must have in order to allow the addition of the candidate image into the Person Directory.	For learning/strangers	Minimum required face contrast quality section of the Recognition Preferences menu
recognizer.minimum-face-contrast-quality-merging	0.45	The minimum face contrast quality that a recognition candidate must have in order to allow merging.	For merging	Minimum required face contrast quality section of the Recognition Preferences menu
recognizer.minimum-face-sharpness-quality	0.1	The minimum face sharpness quality that a face image must have before recognition is attempted.	For recognition	Minimum required face sharpness quality section of the Recognition Preferences menu
recognizer.minimum-face-sharpness-quality-identification	0.3	The minimum face sharpness quality that a recognition candidate must have in order to allow the addition of the candidate image into the Person Directory.	For learning/strangers	Minimum required face sharpness quality section of the Recognition Preferences menu
recognizer.minimum-face-sharpness-quality-merging	0.45	The minimum face sharpness quality that a recognition candidate must have in order to allow merging.	For merging	Minimum required face sharpness quality section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.minimum-face-size	80	The minimum size of faces to detect. This value is applied after searching the image.	For recognition	Minimum required face size section of the Recognition Preferences menu
recognizer.minimum-face-size-identification	120	The minimum resolution that a recognition candidate image must have in order to allow the addition of the candidate image into the Person Directory.	For learning/strangers	Minimum required face size section of the Recognition Preferences menu
recognizer.minimum-face-size-merging	220	The minimum resolution a recognition candidate must have in order to allow merging.	For merging	Minimum required face size section of the Recognition Preferences menu
recognizer.pose-action-max-cpq-jump-after-discontinuity	0.15	The maximum change between samples while the pose is changing from center to profile if lingering. This property is not supported by SAFR Inside.	Max CPQ jump after tracking loss	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-max-cpq-jump-in-continuity	0.18	The maximum change between samples while the pose is changing from center to profile. This property is not supported by SAFR Inside.	Max CPQ jump in continuous tracking	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-max-profile-confidence-end	0.60	The maximum profile pose confidence to allow during the final profile pose detection phase. This property is not supported by SAFR Inside.	Min profile confidence at end	Pose liveness action configuration section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.pose-action-max-profile-pose-quality	0.26	The maximum center pose quality to use when detecting the final profile pose. This property is not supported by SAFR Inside.	Profile pose quality	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-max-profile-pose-roll	0.3	The maximum roll threshold in either direction in which the face can rotate when determining whether the face is in profile pose. This property is not supported by SAFR Inside.	Max profile pose roll	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-min-center-pose-quality	0.5	The minimum center pose quality to use when detecting the initial center pose. This property is not supported by SAFR Inside.	Center pose quality	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-min-detections-per-second	15	The minimum number of frames per second that is required during the process. This property is not supported by SAFR Inside.	Min Detections Per Second	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-min-profile-confidence-start	0.35	The minimum profile pose confidence to allow during the initial center pose detection phase. This property is not supported by SAFR Inside.	Max profile confidence at start	Detect section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.pose-action-min-profile-pose-yaw	0.81	The minimum profile pose yaw value that is required during the final profile pose detection phase. This property is not supported by SAFR Inside.	Min profile pose yaw	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-min-profile-similarity	0.86	The minimum similarity score required when verifying the final profile pose. This property is not supported by SAFR Inside.	Min profile similarity	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-min-transition-poses	2	The minimum number of required center pose samples during the transition from center to profile pose. This property is not supported by SAFR Inside.	Min transition poses	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-profile-pose-required-confirmations	1	The number of consecutive confirmations required to enter the final profile pose detection phase. This property is not supported by SAFR Inside.	Profile pose consecutive confirmations required	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.pose-action-required-confirmations	3	The number of consecutive confirmations required to enter the initial center pose detection phase. This property is not supported by SAFR Inside.	Center pose consecutive confirmations required	Pose liveness action configuration section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.pose-configuration-identification-enabled	FALSE	If this is true then pose configuration is enabled for identification. The pose configuration allows for replacing center pose quality with advanced parameters such as yaw, pitch and roll. When pose configuration is enabled, then recognizer.minimum-center-pose-quality is ignored and the following 3 properties are used instead: recognizer.maximum-yaw-identification, recognizer.maximum-pitch-identification, and recognizer.maximum-roll-identification.	Identification Enabled	Pose liveness action configuration section of the Recognition Preferences menu
recognizer.rgb-action-identity-threshold-boost	0.0	The amount to temporarily boost identity recognition attempts during RGB liveness actions. This property is not supported by SAFR Inside.	Identity recognition threshold boost	RGB liveness action configuration section of the Recognition Preferences menu
recognizer.rgb-action-min-recognitions-fake	2	The minimum number of consecutive "fake" recognitions that are required in order to consider the face fake. This property is not supported by SAFR Inside.	Consecutive recognitions for fake	RGB liveness action configuration section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.rgb-action-min-recognitions-live	2	The minimum number of consecutive "live" recognitions that are required in order to consider the face live. This property is not supported by SAFR Inside.	Consecutive recognitions for live	RGB liveness action configuration section of the Recognition Preferences menu
recognizer.smile-detect-rgb-liveness	FALSE	Enables RGB liveness detection for the smile action recognizer.	RGB liveness validation	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-rgb-action-min-recognitions-fake	2	The minimum number of consecutive "fake" recognitions that are required in order to consider the face fake.	Consecutive recognitions for fake	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-rgb-action-min-recognitions-live	2	The minimum number of consecutive "live" recognitions that are required in order to consider the face live.	Consecutive recognitions for live	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-duration	0	The amount of time that the smile should last. This property is not supported by SAFR Inside.	Smile duration	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-identity-threshold-boost	0.13	The smile threshold to boost temporarily during the smile action. This property is not supported by SAFR Inside.	Identity recognition threshold boost	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-pre-delay	100	The amount of time that there should be no smile. This property is not supported by SAFR Inside.	Pre-smile delay	Smile action configuration section of the Recognition Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
recognizer.smile-thresholds-enabled	FALSE	Enables the smile threshold values. This property is not supported by SAFR Inside.	Transition thresholds	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-threshold-neutral	-0.1	The threshold in which there is no smile. This property is not supported by SAFR Inside.	The neutral face slider below the Transition thresholds setting.	Smile action configuration section of the Recognition Preferences menu
recognizer.smile-threshold-smiling	0.7	The threshold in which there is a smile. This property is not supported by SAFR Inside.	The smiley face slider below the Transition thresholds setting.	Smile action configuration section of the Recognition Preferences menu

6.4.5 Event Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.delay	0	Delay the event reporting to the server by this amount in seconds.	Reporting delay	Report events section of the Events Preferences menu
reporter.enabled	TRUE	Enables or disables event reporting.	Report events	Report events section of the Events Preferences menu
reporter.maximum-face-image-size	240	When event images are being saved, this property specifies the maximum size of the event face images, in pixels.	Preserve Event Face Image, Max Image Size	Report events section of the Events Preferences menu
reporter.maximum-scene-image-size	320	When event scene thumbnail images are being saved, this property specifies the maximum size of the event scene thumbnail images, in pixels.	Preserve Event Scene Thumbnail Image, Max Image Size	Report events section of the Events Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.minimum-event-duration-identified	0	The minimum allowed recognized person event duration in seconds. Events shorter than this duration will not be reported.	Min Identified Event Duration	Report events section of the Events Preferences menu
reporter.minimum-event-duration-stranger	0	The minimum allowed stranger event duration in seconds. Events shorter than this duration will not be reported.	Min Stranger Event Duration	Report events section of the Events Preferences menu
reporter.minimum-event-duration-unidentified	1500	The minimum allowed unrecognizable person event duration in seconds. Events shorter than this duration will not be reported.	Min Unrecognizable Event Duration	Report events section of the Events Preferences menu
reporter.report-event-face	TRUE	Enables the inclusion of face thumbnails in event reports.	Preserve Event Face Image	Report events section of the Events Preferences menu
reporter.report-event-scene	FALSE	Enables the inclusion of scene images in event reports.	Preserve Event Scene Thumbnail Image	Report events section of the Events Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.report-secondary-events	FALSE	Reports secondary events. Secondary events are events that are associated with a primary event via the rootEventId property in the event. It is usually preferred to only report the primary events and the secondary events need to only be reported if there is more detail needed. If this is disabled then all events with a rootEventId property set to a primary event will not be reported. Only events with rootEventId not set to anything will be reported, which are the primary events.	Include Secondary Events	Report events section of the Events Preferences menu
reporter.report-speculated-events	TRUE	Reports events for speculated faces. Speculated faces are faces that aren't a 100% match, but are close.	Include Speculated Identity Events	Report events section of the Events Preferences menu
reporter.report-stranger-events	TRUE	Reports events for people that are strangers. These are people not registered by the system after running facial recognition on the face.	Include Stranger Events	Report events section of the Events Preferences menu
reporter.report-unrecognizable-events	TRUE	Reports events for people that are not recognizable.	Include Unrecognizable Events from Camera and Include Unrecognizable Events from Video	Report events section of the Events Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.stranger-events.only-if-occluded	FALSE	Specifies whether only occluded stranger events should be reported. By default stranger events are only generated if the face is not occluded, if occlusion detection is enabled, otherwise they are generated when the face meets the identification image quality metrics. If this option is set to true then stranger events will be reported only if the face is occluded.	Include Stranger Events, Only if occluded	Report events section of the Events Preferences menu
reporter.stranger-maximum-age	0	The maximum age of strangers that will trigger stranger events. If a stranger older than the specified maximum age is detected, no stranger event is generated.	Include Stranger Events, Max Age	Report events section of the Events Preferences menu
reporter.stranger-minimum-age	0	The minimum age of strangers that will trigger stranger events. If a stranger younger than the specified minimum age is detected, no stranger event is generated.	Include Stranger Events, Min Age	Report events section of the Events Preferences menu
reporter.update-images	FALSE	Updates the thumbnail images with higher quality images during the course of the event if possible.	Update in-progress event attributes, Include qualified images with updates	Report events section of the Events Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.update-in-progress-event-interval	1000	When reporter.update-in-progress-event-properties is set to TRUE, this property specifies the time interval in which to update event properties that change. When reporter.update-in-progress-event-properties is set to FALSE, this property has no effect.	Update in-progress event attributes, Update interval	Report events section of the Events Preferences menu
reporter.update-in-progress-event-properties	FALSE	If this is enabled, then any event properties that change will be updated at the specified reporter.update-in-progress-event-interval . Many properties do change periodically. (e.g. averages that are continually computed)	Update in-progress event attributes	Report events section of the Events Preferences menu

6.4.6 Miscellaneous Properties

Property	Default Value	Description	UI Setting Name	UI Setting Location
directory	N/A	Directory name.	User Directory	Account Preferences menu

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.crop-rectangle.enabled	FALSE	When this is true the defined crop rectangle is used for the camera feed. The crop rectangle is specified in a normalized coordinate system, which means the rectangle is (0, 0) x (1, 1). This property is not supported by SAFR Inside.	The Crop button	Camera Feed Analyzer
input.crop-rectangle.height	1	The normalized height value relative to the video of how big the crop rectangle size should be. This property is not supported by SAFR Inside.	The Crop button	Camera Feed Analyzer
input.crop-rectangle.left	0	The normalized left coordinate relative to the video of where the crop rectangle origin should be. This property is not supported by SAFR Inside.	The Crop button	Camera Feed Analyzer
input.crop-rectangle.top	0	The normalized top coordinate relative to the video of where the crop rectangle origin should be. This property is not supported by SAFR Inside.	The Crop button	Camera Feed Analyzer
input.crop-rectangle.width	1	The normalized width value relative to the video of how big the crop rectangle size should be. This property is not supported by SAFR Inside.	The Crop button	Camera Feed Analyzer

Property	Default Value	Description	UI Setting Name	UI Setting Location
input.loop	FALSE	Enables looping of the feed input. Only video file-based feeds support looping. Ignored when input.type is set to "stream". (e.g. when a camera feed is being processed). This property is not supported by SAFR Inside.	Loop playback	Video section of the User Interface Preferences menu
input.stream.name	N/A	A friendly name used for display purposes. This property is not supported by SAFR Inside.	Feed Name	Add Feed dialogue after pressing the Add to Video Feeds button
mode	"Enrolled and Stranger Monitoring"	Specifies which operator mode the feed is using.	The Operator Mode drop-down menu	Camera Feed Analyzer

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.events-date-timestamps-enabled	TRUE	<p>When a video file is being processed, this property is used to determine whether the events should use date timestamps relative to an initial start date or video timestamps relative to the start of the video.</p> <p>If this setting is FALSE, video timestamps will be used. In this case the timestamps are relative to the start of the video, which is typically 0. It also sets the context to "media" when posting the event so that the time will properly be interpreted as a timestamp rather than a date.</p> <p>If this setting is set to TRUE, then date timestamps will be used relative to the reporter.events-initial-date-offset. If reporter.events-initial-date-offset is null then the current system date will be used when the video processing starts. In this case, the context is set to "live" when posting events so that the time will properly be interpreted as a date.</p> <p>There are cases when videos are</p>	Actual start time	Open video file dialog

Property	Default Value	Description	UI Setting Name	UI Setting Location
reporter.events-initial-date-offset	0	When processing a video file for events this value can be used to set the initial date offset to use for the events being processed. By default video events start with current system time. This option is ignored if the reporter.events-date-timestamps-enabled option is set to FALSE because video timestamps will be used instead.	Actual start time	Open video file dialog
site	N/A	Site name, if any.	Default Site	Account Preferences menu

6.4.7 SAFR SCAN Properties

The UI for all these properties can be found in the SAFR SCAN Console.

Property	Default Value	Description	UI Setting Name	Sidetag Location	Tab Location
access.tailgating.cardId		If set, this <i>cardId</i> is sent to the connected control panel when tailgating is detected.	Tailgating Signal Card ID	Tailgating	Operation

Property	Default Value	Description	UI Setting Name	Sidetag Location	Tab Location
access.tailgating.detection-on-exit	False	If False , tailgating is only detected for people entering the facility (i.e., approaching the camera). If True , tailgating is detected for people entering or exiting the facility.	Detection direction	Tailgating	Operation
access.tailgating.distance	1.5	The maximum distance, in meters, after an authorized person where a subsequent person is considered a tailgater.	Maximum tailgating distance	Tailgating	Operation

Property	Default Value	Description	UI Setting Name	Sidetab Location	Tab Location
access.tailgating.open-signal	"disabled" disabled	Configures the door open signal. There are three possible values: disabled: The door open signal is disabled. low: By default, the door open signal will indicate that the door is closed; a charge must be sent in order to indicate that the door is open. high: By default, the door open signal will indicate that the door is open; a charge must be sent in order to indicate that the door is closed.	Door open input signal	Tailgating	Operation
access.tailgating.enabled	Enabled	Enables tailgating detection.	Tailgating detection	Tailgating	Operation
access.tailgating.facilityId	FacilityId	If set, this <i>facilityId</i> is sent to the connected control panel when tailgating is detected.	Tailgating Signal Card Facility ID	Tailgating	Operation

Property	Default Value	Description	UI Setting Name	Sidetab Location	Tab Location
access.tailgating.time	10000	The minimum amount of time, in milliseconds, where tailgating is enforced after an authorized person is granted access.	Minimum enforced tailgating time	Tailgating	Operation
display.message.tailgating	"Tailgating"	Message when tailgating is detected.	Tailgating detected message	Display	Operation
display.tailgating.duration	5000	The duration, in milliseconds, of the LED display when SAFR SCAN detects tailgating.	Tailgating detected light duration	Display	Operation
display.tailgating.max-brightness	1.0	The brightness of the LED display when SAFR SCAN detects tailgating. This value ranges from 0 to 1, inclusive.	Tailgating detected light brightness	Display	Operation
display.tailgating.rgb	"FF6D00"	RGB color code in hexadecimal of SAFR SCAN's LED display when tailgating is detected.	Tailgating detected light color	Display	Operation

6.4.8 Properties Not Appearing in the UI

Property	Default Value	Description
accelerator	"auto"	<p>The type of acceleration that a feed should use. There are three possible values:</p> <p>auto - VIRGO will automatically pick the best available acceleration type. For example, VIRGO will assign the feed to one of the available GPUs if there is still processing capacity available. Otherwise VIRGO will assign the feed to the CPU.</p> <p>cpu - The feed will exclusively run on the CPU and not use any GPU even if a GPU is available.</p> <p>gpu - The feed will exclusively run on a GPU and not use the CPU for video decoding, graphics processing, or detection. The feed will fail if no GPU is available.</p> <p>This property is not supported by SAFR Inside.</p>
accelerator.gpu-id	0	<p>The GPU identifier to use when GPU acceleration is in use. This property is only used when the accelerator property is set to <i>gpu</i> or <i>auto</i>, and a GPU is being used. If the accelerator.gpu-id property is specified it will force the specific GPU to be used, and if failure occurs SAFR will fall back to the CPU. accelerator.gpu-id is an advanced setting that should only be used in very specific cases. This property is not supported by SAFR Inside.</p>

Property	Default Value	Description
accelerator.gpu-frame-pool-size	-2	<p>Used to set the initial GPU video frame pool size.</p> <p>accelerator.gpu-frame-pool-size is an advanced setting that can be used to fine tune certain use cases. The possible values for this property are listed below.</p> <p>-2: Indicates that the platform default behavior should be used. The platform will determine what GPU frame pool size to use.</p> <p>-1: Indicates that the hardware decoder default value should be used. The only time the hardware decoder default value is not used is if it will cause a problem with the hardware accelerator implementation.</p> <p>0 - N: This value will override the automatic value and will be used unless this will cause a problem in the hardware accelerator implementation. The maximum value for N is 32. This should be changed with caution because using low values (e.g. 0, 1, 2, or 3) can cause performance issues.</p> <p>This property is not supported by SAFR Inside.</p>
capture.frame-delay	30	<p>Wall-clock time between consecutive frame captures. If this value is set to 0 then VIRGO will capture frames as fast as the native frame rate is playing the video. This property is not supported by SAFR Inside.</p>
capture.maximum-frames	3600	<p>If > 0, enables the capture of "max-frames" frames; if 0, disables capture. This property is not supported by SAFR Inside.</p>

Property	Default Value	Description
capture.overlay-level	3	<p>Specifies manner in which overlays will be provided. This property is not supported by SAFR Inside.</p> <p>0 = Won't provide overlays.</p> <p>1 = Provides overlays as separate object tracking meta-data streams.</p> <p>2 = Provides overlays rendered directly into the video frames.</p> <p>3 = Provides overlays as separate object tracking meta-data streams and rendered directly into the video frames - both or whichever is available - to the maximum available according to the declared capabilities.</p>
capture.size	480	<p>Specifies size of the smaller dimension of the image that will be sent. This property is not supported by SAFR Inside.</p>
detector.detect-vehicle	N/A	Internal use only.
detector.detect-vehicle-every-n-frames	N/A	Internal use only.
detector.detect-vehicle-input-size	N/A	Internal use only.
detector.detect-vehicle-model	N/A	Internal use only.
detector.minimum-consecutive-detections-required-vehicle	N/A	Internal use only.
detector.minimum-required-vehicle-to-screen-proportion-enabled	FALSE	Enables or disables the feed.
input.contrast-enhancement.detection-only	FALSE	<p>If true then contrast enhancement is applied to the image which is handed off to the face detector only. If false then contrast enhancement is applied to the video frame as delivered by the camera. Consequently the contrast enhancement effect is visible in the video preview if this option is off but not if it is on. This property is not supported by SAFR Inside.</p>

Property	Default Value	Description
input.stream.id	N/A	Identifier used to connect to a stream if input.stream.url is blank. This property is not supported by SAFR Inside.
input.type	"stream"	The type of feed input; either "stream" or "file".
recognizer.detect-mask-model	"precise"	Specifies the model to be used for mask detection. There are 3 possible values: Precise: This model produces the least number of false positives (i.e. detecting that a person is wearing a mask but there is no mask), but it suffers from the lowest true positive rate. (i.e. detecting masks that are actually there) Sensitive: This model produces the highest true positive rate, but it suffers from the highest number of false positives. Normal: This model produces a moderate amount of both false positives and true positives.
recognizer.mask-check-detection-edge-threshold	0.03	How far a face must be from the edge of the screen before a mask event detection is attempted. For example, if a face is 100 pixels, and recognizer.mask-check-detection-edge-threshold is set to .03 (i.e. 3%), then the face must be 3 pixels from the edge of the screen before SAFR will attempt a mask event detection. This property is not supported by SAFR Inside.

Property	Default Value	Description
recognizer.mask-check-enabled	FALSE	Enables the detection of mask event types. Mask event detection attempts can return 3 potential results: <i>mask=false</i> , <i>mask=indeterminate</i> , or <i>mask=true</i> . After the configured number of consecutive mask event detection results, the mask event state is set to the appropriate value. The mask event state can only progress from false towards true; the state never regresses back towards false. For example, once the mask event state for a viewed person becomes set to <i>mask=true</i> , then that person's mask event state won't ever regress to <i>mask=indeterminate</i> or <i>mask=false</i> . Events are generated when the mask event state is set to either <i>mask=false</i> or <i>mask=true</i> .
recognizer.mask-check-min-consecutive-mask-detections	1	Specifies the minimum number of consecutive <i>mask=true</i> mask detection results that must occur before SAFR will generate a <i>mask=true</i> event. See the recognizer.mask-check-enabled property for more information.
recognizer.mask-check-min-consecutive-no-mask-detections	2	Specifies the minimum number of consecutive <i>mask=false</i> mask detection results that must occur before SAFR will generate a <i>mask=false</i> event. See the recognizer.mask-check-enabled property for more information.
recognizer.mask-check-min-consecutive-occluded-no-mask-detections	2	Specifies the minimum number of consecutive <i>mask=indeterminate</i> mask detection results that must occur before SAFR will set the mask event state to <i>mask=indeterminate</i> . See the recognizer.mask-check-enabled property for more information.
recognizer.mask-check-min-mask-detection-size	70	The smallest face size, in pixels, upon which SAFR will attempt to detect a mask event.

Property	Default Value	Description
statistics.enabled	FALSE	Specifies whether VIRGO should record and report statistics for this feed.

6.5 Update Properties

The update properties are not intended for public consumption at this time.

Property	Default Value	Description
client-type	OS-defined client type	Internal use only.
version	N/A	Internal use only.
enabled	FALSE	Internal use only.
progress-status	N/A	Internal use only.
progress-url	N/A	Internal use only.
download-url	N/A	Internal use only.
log-enabled	FALSE	Internal use only.
progress-interval	1000	Internal use only.

7 Processing Video Files

Both the Video Feeds Window (located in the Desktop Client and the Web Console) and the Command Line Interface can be used to process video files for person detection and/or recognition. This can be useful to recognize faces for purposes of identifying people or generating events on pre-recorded videos. For example, consider the scenario where you have video footage from cameras throughout a facility and you want to determine where and when a person of interest appears in those videos. To do this, you would register the people of interest to the Person Directory (if they weren't already registered) and you could then process the videos in Enrolled Monitoring video processing mode in order to identify when and where that individual appears.

The Video Feeds Window is the best video file processing option when you only have 1 video to process. When you want to process 2 or more video files, the Command Line Interface is the better choice.

Note When you switch control from the Video Feeds Window to the Command Line Interface, any feeds that you configured in the Video Feeds Window will be stopped and will not be visible from the Command Line Interface. The feeds still exist in the Video Feeds Window, but you cannot start or manage those feeds from the Command Line Interface. You must re-create new feeds for the Command Line Interface.

7.1 Process Files with the Video Feeds Window

To process video files, the Video Feeds Window must have access to those files on the local file system. The window can work with files on an internal hard drive, an external drive attached to the machine, or a network share mounted to the file system. In this example we will use the following directories.

```
mkdir -p /files/videos
mkdir -p /files/feeds
```

- The **videos** directory will be used to store the video files that VIRGO will process.
- The **feeds** directory will be used to store feed configuration files used by VIRGO. Feed configuration files store information necessary to process video from a file or camera.
- **videos** and **feeds** can be either aliases or directories mounted to an external file system.

7.1.1 Mount Host Filesystem to VIRGO Container

In order to process files that are on a Linux machine, you must mount a file system on the host drive to the Docker container running Video Recognition Gateway (VIRGO).

You can mount a host drive onto the Docker instance by doing the following:

1. Add the following lines to the end of the file `/opt/RealNetworks/SAFR/virgo/app/docker-compose.yml`

```
vi /opt/RealNetworks/SAFR/virgo/app/docker-compose.yml

- /files:/files
```

The hyphen should be included.

The resulting file should look something like this. (DON'T USE THE EXAMPLE BELOW - Your system will have different version info.)

```
version: "3.6"
services:
  virgo:
    image: safr_virgo:1.2.22
    container_name: safr_virgo
    restart: on-failure
    pid: "host"
    volumes:
```

```
- /opt/RealNetworks/SAFR/virgo/config/./etc/virgo
- /files:/files
```

- The mount format is `<path on host>:<path on VIRGO Docker container>`
 - This is equivalent to creating an alias on the Docker container pointing to a actual directory on the host.
- `<path on VIRGO Docker container>` can be any path. It should not exist already.
- `<path on host>` should be a real directory. It can be located anywhere on the host drive.
- Using the same path for both makes it less confusing when typing filepaths because there will be no need to remember the Docker path and to append the relative host path. Just use the fully qualified host path when defining the location of a video in the VIRGO config.

2. Run the following commands to re-mount all paths on the Docker container:

```
docker-compose -f /opt/RealNetworks/SAFR/virgo/app/docker-compose.yml down
docker-compose -f /opt/RealNetworks/SAFR/virgo/app/docker-compose.yml up
-d
```

3. Check to see if the folder is mounted correctly in the Docker container by doing the following:

1. Run the following command to sign into the Docker container:

```
docker exec -it safr_virgo bash
```

2. Try to list the directory of the location mounted above:

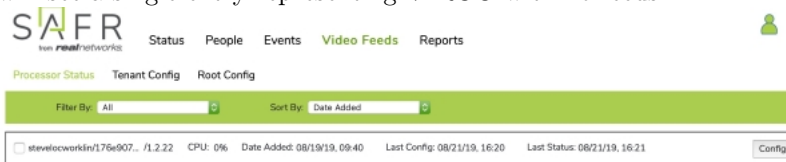
```
ls /files
```

You should see the two folders (feeds and videos) that you created listed under this location.

7.1.2 Use the Video Feeds Window

To use the Video Feeds Window, do the following:

1. Connect to the Web Console or open the Desktop Client. See [here](#) for details about the Web Console and how to connect to it.
2. Navigate to the **Video Feeds** tab and select **Processor Status**. Assuming you have no other video clients connected to the server (i.e. from additional connected VIRGO feeds or Desktop Clients), you will see a single entry representing VIRGO with no feeds.



3. Click **Config** to add feeds.
4. Enter values for the following fields:
 - **name** - User-defined name. We recommended that you only use ASCII letters and numbers (i.e. no spaces) since this is used to reference the feed on the command line.
 - **mode** - Defines the default settings for SAFR detection and recognition.
 - **input.stream.url** - Source camera URL or filename of the video to use as input. In this case we pre-pend the value with "file://" to indicate that we're processing a file. The path portion is the path to the video file relative to the Docker container, as explained above.
 - In the screenshot below, the **enabled** field still has its default value of "false". You should set it to "true" when you are ready to start processing the file.

SAFR from realnetworks Status People Events **Video Feeds** Reports

Processor Status Tenant Config Root Config

worker config Add Attributes

global Add Attributes

admin Add Attributes

tenant name stevelocworkin

add-date 1566232814629

feed list Add feed

feed 1 Delete feed Add Attributes

name EnrolledAndStrangerMonitoring

directory main

enabled false

mode Enrolled and Stranger Monitoring

input.stream.url file:///files/videos/cam4_2019-08-01-10-18-15_Org.mp4

site Site01

source Camera05

Apply Cancel

5. Before clicking **Apply**, add an additional field to specify the start time of the video. Do this by doing the following:

1. Click **Add Attributes** under the feed.
2. Search for and select the "reporter.events-initial-date-offset" property.

feed list Add feed

feed 1 Delete feed Add Attributes

name

directory

enabled

mode

input.stream.url

site

source

Add Attributes

☐ recognizer.smile-threshold-smiling

☐ recognizer.maximum-concurrent-recognitions

☐ reporter.delay

☒ reporter.events-initial-date-offset

☐ reporter.enabled

☐ reporter.minimum-event-duration-identified

☐ reporter.minimum-event-duration-unidentified

☐ reporter.report-event-face

☐ reporter.report-event-scene

☐ reporter.report-unrecognizable-events

☐ reporter.report-stranger-events

☐ reporter.report-speculated-events

6. Once added set a date value as Epoch in milliseconds. For example, for Aug 1, 2019 at 11 AM, the Epoch value is 1564682400000.

- This causes generated events to be recorded at the correct point in time rather than being recorded at the default Epoch value of "0" or January 1, 1970.
- Try the Chrome Extension 'utime' to create Epoch times. This extension allows you to type any date/time and get Epoch values as well as natural language strings such as '1 hour ago'.

Note: Be sure to set the time format to "milliseconds" or else the time will be off by many years.

7. Change the **enabled** field to "true" and click **Apply** to save the feed and start processing the video.
8. If all goes well you should see the following:

SAFR from realnetworks Status People Events **Video Feeds** Reports

Processor Status Tenant Config Root Config

Filter By: All Sort By: Date Added

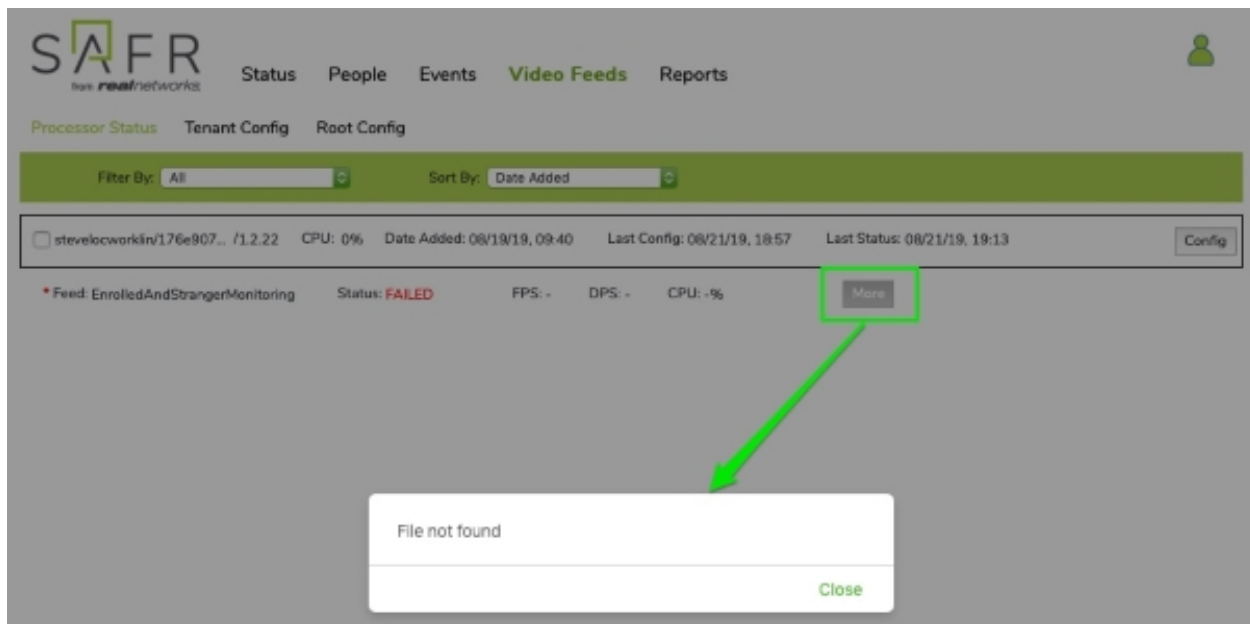
☐ stevelocworkin/176a907... /1.2.22 CPU: 40% Date Added: 08/19/19, 09:40 **Last Config: 08/21/19, 16:45** Last Status: 08/21/19, 16:45 Config

* Feed: EnrolledAndStrangerMonitoring Status: **OK** FPS: 20 DPS: 156 CPU: 40% View

9. For a few moments you may see the **Last Config** date in red which means that changes have not yet been applied to VIRGO. Once the changes have been applied, VIRGO should start processing and you should see the **Status** reported as "ok". You can click **View** to see the current frame of video being processed. This will be updated every second or so.
10. See Troubleshoot Feeds below if the video does not start processing successfully.
11. You can move to the **Events** tab to view the events being processed. You may need to change the search criteria to include August 1, 2019 for the events to show up.

7.1.3 Troubleshoot Feeds

If there is a problem, first check the **More** button on the VIRGO feed as shown below:



Additional possible troubleshooting steps:

- If you see status as "inactive", check the enabled option in the queue file. It should be "true", as in: "enabled" : true
- To start the feed, you can run the VIRGO command: `docker exec -it safr_virgo ./virgo feed start queue1`
- To avoid this problem with subsequent files, you may want to edit the feed configuration file to set enabled to "true".

If the steps above do not fix the problem, then it may help to look at the service monitor or service log. Both must be started before running VIRGO in order to capture the error output.

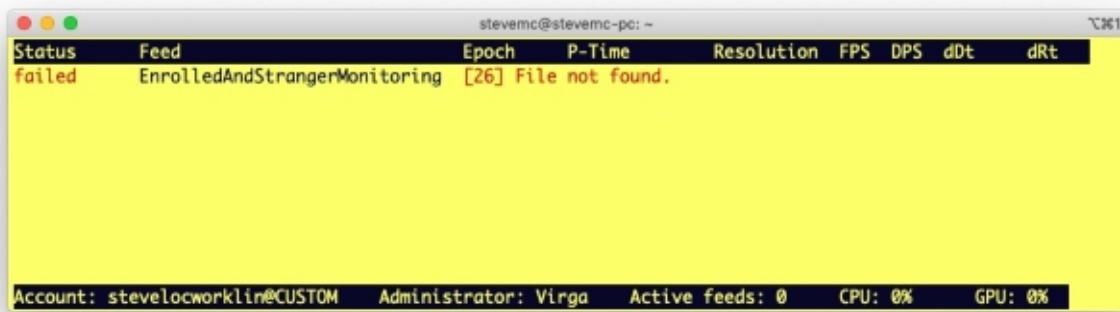
7.1.3.1 View the VIRGO Service Monitor To view the service monitor, log into SAFR Server and run the following command:

```
sudo docker exec -it safr_virgo /opt/RealNetworks/virgo/virgo service monitor
```

or

```
docker exec -it safr_virgo bash
./virgo service monitor
```

You should see the following:



7.1.3.2 View the VIRGO Service Log To view the service log, run the following command:

```
sudo docker exec -it safr_virgo /opt/RealNetworks/virgo/virgo service log
d/feed d/http-cop
```

or

```
docker exec -it safr_virgo bash
virgo service log d/feed d/http-cop
```

With the above running, restart the feed. In the Video Feeds Window you may need to do the following to successfully restart the feed:

1. Set the feed `enabled` flag to "false".
2. Save.
3. Change the `enabled` flag back to "true".

Additional Notes:

- Information will be printed to the screen. Use Unix Redirect (`>`) or "tee -a" command to write output to file.
- Additional options for VIRGO logging can be found [here](#).
- It is useful to have both the service monitor and service log open in separate windows as you start the video feed.

7.2 Process Files with the Command Line Interface

7.2.1 Create a Feed Template

To run VIRGO, you need a feed configuration file. You can use the configuration file we created above in VIRGA as a starting point. This will include most of the common configuration properties needed.

1. Export a feed configuration file from VIRGO to use as a template.
2. Get a list of VIRGO feeds.

```
> sudo docker exec -it safr_virgo ./virgo feed list

EnrolledAndStrangerMonitoring
```

- EnrolledAndStrangerMonitoring should be the only feed listed.
3. Save that configuration to a file called `template.json`.

```
> sudo docker exec -it safr_virgo ./virgo feed get
  EnrolledAndStrangerMonitoring
/files/feeds/template.json
```

- The folder /files/feeds here is relative to the VIRGO Docker container. This path is mounted to the same path (/files/feeds) on the host filesystem.

4. Ensure that the file was written.

```
> cd /files/feeds
> ls

template.json
```

5. Edit the file 'template.json' if desired. Generally it's easier to perform editing in the Video Feeds Window before exporting.

- The Video Feeds Window unnecessarily adds extra escapes to forward slashes in the JSON. This makes the string much harder to read. When using the Video Feeds Window to generate configuration files, the file path string can be simplified as follows:
 - Change this: `file:\\\\opt\\RealNetworks\\virgo\\files\\videos\\cam4Aug1_output000.mp4`
 - To this: `file:///opt/RealNetworks/virgo/files/videos/cam4Aug1_output000.mp4`

6. Create a copy of the template file and name it 'queue1.json'. You're making a copy because you'll maintain one file for each feed.

```
> cd /files/feeds
> cp template.json queue1.json
```

7. Update the feed file by setting values for site, source, start time, and filename. Use the utility script `update_virgo_feed.sh` documented below for this purpose.

```
> update_virgo_feed.sh queue1.json Site01 Camera10 1564682400000
  'file:///files/videos/test/vid00.mp4'
```

- See the script's documentation below for a description of its command line arguments.
- The file path assumes you have mounted the VIRGO Docker container to /files in the host drive as explained above.

This will set the values for the respective fields in queue1.json. View the file in an editor to confirm edits were made.

7.2.2 Create Feed

You can assign the feed configuration file created in the previous section to a feed. Before doing that, you need to first switch control from the Video Feeds Window to the Command Line Interface. As explained above, the Video Feeds Window and the Command Line Interface do not share the same configuration files. That is why you first exported a configuration file above from the Video Feeds Window. Now that you have a configuration file, you're ready to switch control to the Command Line Interface. The Command Line Interface will start out with no feeds defined. You will then create feeds from the configuration file saved above.

1. Switch control from the Video Feeds Window to the Command Line Interface

```
> sudo docker exec -it safr_virgo ./virgo administrator set virgo
```

Note: This changes control of VIRGO from the Video Feeds Window to the Command Line Interface. The Video Feeds Window will no longer be able to start or stop VIRGO feeds until control is restored

back to it. Note that all feeds added to the Video Feeds Window are not available to the Command Line Interface. You will re-create feeds as described below.

2. Create a new feed 'queue1' from the queue1.json feed configuration file created above.

```
> sudo docker exec -it safr_virgo ./virgo feed add queue1
/files/feeds/queue1.json
```

3. Confirm that the feed was added to VIRGO.

```
> sudo docker exec -it safr_virgo ./virgo feed list

queue1
```

4. VIRGO will attempt to start processing the feed right away. Run VIRGO Service Status or VIRGO Service Monitor/

```
> sudo docker exec -it safr_virgo ./virgo service status

queue1 ok
```

See the Service Monitor section above for information on how to run VIRGO Service Monitor. Initially you may see a status reported as "perolling" which means the feed is starting up.

After processing is done, the status should change from "ok" to "eos" (meaning End of Stream). If any other status is shown, see Troubleshoot Feeds above.

7.2.3 Edit Feed and Reload

To edit a feed and re-load, do the following.

1. Stop the feed with following command:

```
docker exec -it safr_virgo ./virgo feed stop queue1
```

2. Edit the file and correct problems. Re-load the file from disk using following command:

```
docker exec -it safr_virgo ./virgo feed set queue1
/files/feeds/queue1.json
```

3. Restart the feed with following command:

```
docker exec -it safr_virgo ./virgo feed start queue1
```

7.3 Batch Processing Files with the Command Line

When batch processing multiple files, it's most efficient to process two or more files in parallel. The optimal number of parallel processes is a function of processor speed and disk/network speeds and should be determined by experimentation on your hardware. VIRGO will try to process files as fast as possible and leverage multiple CPU cores and GPUs but there is generally a limit to how many CPU cores it uses. Processing more files in parallel will allow you to fully utilize the machine resources.

In general the process is as follows:

- N queues are created. Each queue will be able to process 1 file in parallel.
- 1 file is set to be processed on each queue.
- Each of the queues is started. (As long as enabled=true is set, processing will start as soon as the feed file is assigned to the queue.)
- As each queue is started, its status will be "ok".
- As each queue completes, the status will change to 'eos'.

- Any feed with status 'eos' is assigned the next file in line to be processed.
- Files continue to be added to queues until all files are processed.

7.3.1 Create Feed Queues

In order to create a feed queue, create an additional feed called `queue2`.

```
> cp template.json queue2.json
> update_virgo_feed.sh queue2.json Site06 Camera04
  'file:///files/videos/test/vid01.mp4'
> sudo docker exec -it safr_virgo ./virgo feed add queue2
  /files/feeds/queue2.json
```

You'll maintain different queue files (`queue1.json` and `queue2.json`) in order to allow `queue1` and `queue2` to be processed independently. Each queue file serves as a record of what is running on the current queue. If an error occurs on one of the queues, you can use the `get_input_stream.sh` script to identify the file that was not processed for the queue and write it to an error log file.

The 2nd feed should start automatically. If you check the VIRGO Service Monitor, you should see something like the following:

Status	Feed	Epoch	P-Time	Resolution	FPS	DPS
dDt	dRt #D					
eos	queue1	04:45:50	00:00:04.343	1920x1080	-	-
	- 816					
ok	queue2	06:28:36	00:00:04.413	1920x1080	-	-
	- 819					

7.3.2 Create Feed File List

The feed file list is a CSV file with following columns:

- **Video File Relative Path** - The path of the video file. It is relative to the base path set in the `process_files.sh` script.
- **Site** - Specifies site (building, GPS location, etc) name for the purpose of identifying location.
- **Source** - Specifies the camera name for purposes of identifying location.
- **Date** - Internal use only.
- **EpochDate** - Start time of the video file in Unix Epoch date format. This is used to ensure events are created at the proper point in the timeline.
 - This value must be in Epoch milliseconds.
 - See the Process Video Files Script section below for information on creating Epoch values using the `process_files.sh` script.
 - Epoch millisecond values can be created from human readable dates in Excel using this formula: $\text{=(D1-DATE(1970,1,1))864001000}$
 - Where "D1" is the cell containing the start date of the video.

Below is an example of the contents of the feed file list file:

```
vid001.mp4,Site05,Camera025,8/20/19 7:12 AM,1566285120000
vid002.mp4,Site03,Camera024,8/20/19 7:12 AM,1566285120000
xvid003.mp4,Site01,Camera003,8/20/19 4:48 AM,1566276480000
vid004.mp4,Site01,Camera030,8/20/19 12:00 PM,1566302400000
vid005.mp4,Site04,Camera016,8/20/19 12:00 PM,1566302400000
vid006.mp4,Site04,Camera002,8/20/19 7:12 AM,1566285120000
vid007.mp4,Site03,Camera016,8/20/19 12:00 PM,1566302400000
xvid008.mp4,Site02,Camera020,8/20/19 12:00 PM,1566302400000
vid009.mp4,Site05,Camera011,8/20/19 9:36 AM,1566293760000
```

```

vid010.mp4,Site02,Camera030,8/20/19 7:12 AM,1566285120000
vid011.mp4,Site05,Camera021,8/20/19 9:36 AM,1566293760000
vid012.mp4,Site04,Camera009,8/20/19 9:36 AM,1566293760000
vid013.mp4,Site01,Camera001,8/20/19 12:00 PM,1566302400000
vid014.mp4,Site02,Camera017,8/20/19 9:36 AM,1566293760000
vid015.mp4,Site04,Camera014,8/20/19 12:00 PM,1566302400000
vid016.mp4,Site02,Camera010,8/20/19 7:12 AM,1566285120000
vid017.mp4,Site04,Camera030,8/20/19 7:12 AM,1566285120000
vid018.mp4,Site02,Camera022,8/20/19 9:36 AM,1566293760000
vid019.mp4,Site05,Camera006,8/20/19 4:48 AM,1566276480000
vid020.mp4,Site02,Camera012,8/20/19 9:36 AM,1566293760000
vid021.mp4,Site02,Camera005,8/20/19 7:12 AM,1566285120000
xvid022.mp4,Site01,Camera019,8/20/19 4:48 AM,1566276480000
xvid023.mp4,Site02,Camera016,8/20/19 7:12 AM,1566285120000
vid024.mp4,Site04,Camera027,8/20/19 7:12 AM,1566285120000

```

7.3.3 Edit the Process Files Script

Edit the configurable parameters in the *process_files.sh* script. Below are the default values in the script.

```

## USER CONFIGURABLE PARAMETERS ##
#####
# User Directory
user_dir=main1
# Set this to the location of the feed configuraiton files.
feeds_dir=/files/feeds
# If different than above, Set to the path to feeds from inside docker
  container
docker_feeds_dir=$feeds_dir
# Set this to location of the video files
video_files_dir=/files/videos/1min_segments
#####

```

- *user_dir* is the user directory where identities and events are stored. It can be useful to edit this parameter when experimenting with different settings.
- The default values for *feeds_dir* and *docker_feeds_dir* are the same because of the most common ways that host file systems are mounted in the docker container. This may not necessarily be the case so the two attribute are separated in the script
- The script assumes *video_files* path is identical in the host as well as the docker container. Script would need to be modified if this was not the case (one part of the script checks the existence of the file from the host OS while in another place, the path to the video file is passed to VIRGO running in the docker container via the feed configuration file

7.4 Reference

This section includes reference files and example scripts useful in processing video files with the Command Line Interface.

7.4.1 VIRGO Command Line Help

Command line interface to the virgo daemon
Syntax:

'virgo' followed by one of the following:

administrator set <administrator> 'virgo' or 'virga'	sets the administrator. Either
administrator get	shows the current administrator
environment get	shows the current environment
environment set <environment>	sets the current environment
environment list	shows all supported environments
feed list	shows all known feeds
feed get <feed> [path] and optionally saves it as a feed configuration file	shows the feed configuration
feed status <feed>	shows the current feed status
feed start <feed> starts it running	marks the feed as enabled and
feed stop <feed> disabled	stops the feed and marks it as
feed remove <feed>	stops the feed and removes it
feed add <feed> <path to config> file and adds the new feed to the known feeds	reads the feed configuration
feed set <feed> <path to config> file and updates the feed with the new configuration	reads the feed configuration
feed capture-image <feed> <url or path> from the feed and stores them in the directory <url>	captures one or more images
service info	shows the service information
service status feeds	shows the current status of all
service reset state back to the factory defaults	resets the persistent virgo
service log service log information	continuously shows the current
service monitor service status and statistical information	continuously shows the current
service update <version> [<url or path>]	upgrades or downgrades virgo to
the specified version	
service versions	shows all installed versions
user get identity	shows the current user cloud
user set [user id] identity	sets the current user cloud
 <environment> is an environment name.	
<feed> is a feed name.	
<version> is a semantic version number (e.g. 1.0.3).	
 --help -h shows this help message.	
--verbose -v enables the display of more detailed information.	
 --max-frames the maximum number of frames to capture from a feed	
--frame-delay the delay between capturing frames. This is in	
milliseconds	
--size the size to which a captured image should be scaled	

7.4.2 Process Video Files Script

process_files.sh

This is the primary commands that loops through all files and adds them to VIRGO queues for processing.

```
#!/bin/bash
script_dir=$(dirname "$0")

## USER CONFIGURABLE PARAMETERS ##
#####
# Set this to the location of the feed configuraiton files.
feeds_dir=/files/feeds
# If different than above, Set to the path to feeds from inside docker
  container
docker_feeds_dir=$feeds_dir
# Set this to location of the video files
video_files_dir=/files/videos/1min_segments
# User Directory
user_dir=main1
#####

# Make sure we got correct number of args.
if [ "$#" -lt 1 ]; then
    echo "Usage: $0 <job list file>
    Where <job list file> csv file with video_file,site,source";exit;
fi

# Check if video file list exists
[ ! -f $1 ] && { echo "$1 video list file not found"; exit 99; };

### Loop thru each line in input file
#####
OLDIFS=$IFS; IFS=,
cat $1 | tr -d '\r' | while read fname site source date epoch; do
    echo "Placing file $fname for site: $site, source: $source and time:
    $epoch"
    [ ! -f "$video_files_dir/$fname" ] && {
        echo "$video_files_dir/$fname not found. Skipping" | tee -a
        error.log; continue;
    }

### Loop Feeds - Outer loop
## Get list of queues and pass this into inner loop
### Break once inner loop exits with successful assignment
#####
assigned=false; echo "processing file $fname"
while true; do

    ### Loop Feeds - Inner loop
    ### Loop thru each queue - check to see if any are ready for next job
    ###
    #####
    while read feed_line; do
        TEMP=$IFS; IFS=': ' read queue_id queue_status <<< $feed_line;
        IFS=$TEMP
```

```

if [ "$queue_status" == "failed" ]; then
    # Something went wrong. Log file being processed by this queue
    to look at later
    echo "$script_dir/get_input_stream.sh $queue_id.json" >>
        error.log
fi
if [ "$queue_status" == "ok" ] || [ "$queue_status" == "prerolling"
]; then
    # Skip and go to next feed
    echo "Skipping queue $queue_id with status $queue_status"
elif [ "$queue_status" == "eos" ] || [ "$queue_status" ==
    "inactive" ]; then
    $script_dir/update_virgo_feed.sh $feeds_dir/$queue_id.json $site
        $source $epoch "file://$video_files_dir/$fname"
    $script_dir/set_virgo_feed_attr.sh $feeds_dir/$queue_id.json
        directory $user_dir
    echo "Set file $fname on queue $queue_id" | tee -a processed.log;
    docker exec safr_virgo ./virgo feed set $queue_id
        $feeds_dir/$queue_id.json
    if [ "$queue_status" == "inactive" ]; then
        docker exec safr_virgo ./virgo feed start $queue_id
    fi
    # Skip all other feeds and go to next file
    assigned=true; break 1;
else
    # Unexpected processing status. Report in error log
    echo "Unexpected virgo feed status $queue_status" >> error.log
fi
sleep 1
done <<< "$(docker exec safr_virgo ./virgo service status)"
if $assigned; then echo "Assigned $fname to $queue_id"; break 1; fi
done
done
IFS=$OLDIFS

```

7.4.3 Utility Scripts

The following scripts are used by the `processing_files.sh` script.

update_virgo_feed.sh

```

#!/bin/bash
if [ "$#" -lt 4 ]; then
    echo "Usage: $0 <feed filename> <sitename> <sourcename> <starttime>
        '<file_path>';exit;
fi
sed -i -e "s|\"site\" *: *\"[^\"]*\"|\"site\" : \"$2\"|g" $1
sed -i -e "s|\"source\" *: *\"[^\"]*\"|\"source\" : \"$3\"|g" $1
sed -i -e "s|\"reporter.events-initial-date-offset\" *:
    *\"[^\"]*\"|\"reporter.events-initial-date-offset\" : \"$4\"|g" $1
sed -i -e "s|\"input.stream.url\" *: *\"[^\"]*\"|\"input.stream.url\" :
    \"$5\"|g" $1

```

Where:

- **<feed filename>** - Name of the file that contains feed settings.
- **<site name>** - Specifies the site (building, GPS location, etc) name for the purpose of identifying the location.
- **<sourcename>** - Specifies the camera name for the purpose of identifying the location.
- **<starttime>** - Start time/date of the video. The value given as Epoch in milliseconds. For example, for Aug 1, 2019 at 11 AM, the epoch value is 1564682400000.
 - Try the Chrome extension 'utime' to create epoch times. This extension allows you to type any date/time and get epoch values as well as natural language strings such as '1 hour ago'. Be sure to set the time format to "milliseconds" or else the time will be off by many years.
- **<file path>** - The fully qualified path to the file relative to the Docker container.

set_virgo_feed_attr.sh

```
#!/bin/bash
if [ "$#" -lt 3 ]; then
    echo "Usage: $0 <feed filename> <attr name> <attr value>";exit;
fi
sed -i -e "s|\"$2\" *: *\"[^\"]*\"|\"$2\" : \"$3\"|g" $1
```

Where:

- **<feed filename>** - Name of the file that contains feed settings.
- **<attr name>** - The JSON attribute name from the feed configuration file.
- **<attr value>** - The value to be assigned to the JSON attribute from the feed configuration file.

get_input_stream.sh

```
#!/bin/bash
if [ "$#" -lt 1 ]; then
    echo "Usage: $0 <feed filename>"
fi
grep input.stream.url $1 | sed -e 's:"input.stream.url" *\: *\"([^\"]*)\" ,*:\1:g'
```

Where:

- **<feed filename>** - Name of the file that contains feed settings.

8 VIRGO Tools

The Video Recognition Gateway (VIRGO) installation on Linux includes a couple scripts to manage VIRGO. They are located at `/opt/RealNetworks/SAFR/virgo/app`.

8.1 Back Up `virgo-factory.conf`

Before you use VIRGO tools you should back up your **virgo-factory.conf**. This configuration file can be found at `/opt/RealNetworks/SAFR/virgo/config`.

8.2 `virgo_updateip.sh`

This script uses the current local IP to update the VIRGO configuration file (**virgo-factory.conf**). To run it, do the following:

1. Go to `/opt/RealNetworks/SAFR/virgo/app`.
2. Run `virgo_updateip.sh`. The syntax to run the script is `./virgo_updateip.sh`. The following output will be shown.

```
11:22:38 CST - Collect local ip
11:22:38 CST - Local IP 10.10.51.189
11:22:38 CST - Updating configuration files
11:22:38 CST - Applying new virgo configuration
```

The IP has now been changed in **virgo-factory.conf**.

8.3 `virgo_configure.sh`

This script will reset the VIRGO service. To run it, do the following:

1. Go to `/opt/RealNetworks/SAFR/virgo/app`.
2. Run `virgo_configure.sh`. The syntax to run the script is `./virgo_configure.sh <Username> <Password>`.
 - For *<Username>*, use the *user-id* value found in the **virgo-factory.conf** file.
 - For *<Password>*, use the *user-encrypted-password* value found in the **virgo-factory.conf** file.The following output will be shown.

```
11:54:21 CST - Updating virgo factory configuration
11:54:21 CST - Username realnetworksbei13
11:54:21 CST - Password
%pHUQfSS4mk7UIrhs5au0aZ\+qshbJPGIx4rEw\ /RpCgGpUrYCiWrzc2uh8g9HsxJHf
11:54:21 CST - Cleaning out old virgo configuration
11:54:21 CST - Cloning template file
11:54:21 CST - Collect local ip
11:54:21 CST - Local IP 10-10-51-189
11:54:21 CST - Copy config to /etc/ folder
11:54:21 CST - Reset Virgo to reload configuration
11:54:21 CST - Finished
```

9 Factory Configuration File

Every Video Recognition Gateway (VIRGO) daemon on Linux ships with factory settings which define the default configuration that the daemon should use the first time it starts up. *Virgod* also reverts the current configuration back to the factory settings if it is unable to load the current configuration because of a version mismatch and it is unable to automatically convert the old configuration to the new format.

The factory settings are stored in a JSON file with the name `virgo-factory.conf`. *Virgod* looks in the following locations to find a factory configuration file:

- The home directory of the user who started *virgod*.
- The `/etc` directory.
- The VIRGO bundle directory.

Virgod loads the first factory configuration file that it finds. If it can't find any factory configuration file, it falls back to hardcoded defaults.

9.1 Factory Configuration File Format

The factory configuration file is a JSON file which is organized into (optional) sections:

```
{
  "global": {                // [optional]
    // global state
  },
  "environments": {          // [optional]
    "Foo": {
      // environment-specific URLs
    }
  },
  "feeds": {                 // [optional]
    "camera_1": {
      // feed state
    }
  }
}
```

Note: Nearly all keys in a factory configuration file are optional. Only those keys that you explicitly want to override with a custom value need to be specified. *Virgod* uses hardcoded default values for keys that are missing from a factory configuration file.

9.1.1 The Global Section

The following properties are supported in the global section:

Property	Type	Default	Description
status-interval	Int?	5000	Status reporting time interval in ms.
environment	String?	PROD	The name of the environment which should be used by <i>virgod</i> . See the "Environments Section" below for a list of pre-defined environment names.

Property	Type	Default	Description
machine-id-prefix	String?	empty string	The machine ID prefix. The default machine ID prefix is the empty string.
machine-id	String?	OS defined machine ID	The machine ID. The default machine ID is derived from the OS provided machine ID. The concatenation of the machine-id-prefix and the machine-id is sent to the cloud in the X-CLIENT-ID header.
client-type	String?	OS defined client type	The client type. This value is sent to the cloud in the X-CLIENT-TYPE header.
user-id	String		The user ID for the cloud account.
user-password	String		The password for the cloud account. Note that the password is stored in clear text. Use <i>user-encrypted-password</i> whenever possible instead.
user-encrypted-password	String		The encrypted password for the cloud account.
administrator	String?	cloud	Specifies whether VIRGO should be administrated by VIRGA or whether it should be self-administrated. A self-administrated VIRGO allows you to manage feeds via the VIRGO command line tool.

Property	Type	Default	Description
visible-accelerator-ids	[Int]?		Allows you to specify which GPUs/accelerators VIRGO is allowed to use for video decoding and detection tasks. Only the accelerators listed in this array will be used by VIRGO; all others will be ignored. The value is an array of accelerator IDs. VIRGO will use all available accelerators if this property is not set.

Note that feeds which are assigned to a specific accelerator ID will fail with an error at startup if that accelerator is not in the set of visible accelerator IDs.

9.1.2 The Environments Section

The environments section defines the available cloud environments. Each environment has a name and a set of URLs that point to the hosts in the cloud that provide the required services. An environment may override one of the pre-defined environments. The environment name is used to identify the environment and to switch among environments with the `virgo environment set` command.

The following properties are supported in the environments section:

Property	Type	Default	Description
covi-server-url	URL	none	The face recognition service.
rncv-server-url	URL?	none	The face detection service.
event-server-url	URL	none	The detection and recognition event recording service.
object-server-url	URL	none	The service which stores objects such as images and logs.
admin-server-url	URL	none	The VIRGO administration service.

The following table lists the pre-defined environments:

Name	Alternative name
SAFR Local	LOCAL
SAFR Developer Cloud	DEV
SAFR Partner Cloud	INT2
SAFR Cloud	PROD

You can use the alternative environment name in place of the full environment name.

9.1.3 The Client ID

VIRGO computes the client ID by concatenating the machine-id-prefix and the machine-id properties.

9.2 Example Configuration Files

The following subsections show some typical factory configuration files.

9.2.1 Using VIRGO with a VIRGA Server

This is an example of a configuration file which configures VIRGO to run as a slave to a Video Recognition Gateway Admin (VIRGA) server. VIRGO will continuously report its status to the VIRGA server and the VIRGA server is responsible for pushing state changes to VIRGO.

```
{
  "global" : {
    "environment": "PROD",
    "machine-id-prefix": "foo",
    "user-id": <user ID>,
    "user-password": <password>
  }
}
```

9.2.2 Using VIRGO Standalone

This is an example of a configuration file which configures VIRGO to run as a standalone daemon which does not connect to a VIRGA server. VIRGO starts processing the declared feeds as soon as it starts up. Note that you still have to provide a user ID and a password to allow VIRGO to use the (cloud-based) face recognition and event recording service.

```
{
  "global":{
    "environment": "PROD",
    "machine-id": "argusrn",
    "user-id": <user ID>,
    "user-password": <password>,
    "administrator":"self"
  },
  "feeds":{
    "camera_1":{
      "directory":"test",
      "input.type":"stream",
      "input.stream.url":"file://<absolute path to a movie file>",
      "recognizer.learning-enabled":true,
      "enabled":true
    }
  }
}
```

9.2.3 Defining Custom Environments

This is an example of a configuration file which defines two custom cloud environments. Note that the first custom environment has a new unique name that is separate from any of the pre-defined environments.

The second custom environment, on the other hand, overrides the pre-defined environment name `PROD`. Consequently `VIRGO` will use the URLs of the custom environment if the `PROD` environment is selected. This allows you to replace the built-in definition of the pre-defined environment.

```
{
  "global": {
    "environment": "Test"
  },
  "environments": {
    "Test": {
      "covi-server-url": "https://covi.test.real.com",
      "event-server-url": "https://event.test.real.com",
      "object-server-url": "https://object.test.real.com",
      "admin-server-url": "https://admin.test.real.com"
    },
    "PROD": {
      "covi-server-url": "https://covi.sim.real.com",
      "event-server-url": "https://event.sim.real.com",
      "object-server-url": "https://object.sim.real.com",
      "admin-server-url": "https://admin.sim.real.com"
    }
  }
}
```

10 Docker

The Video Recognition Gateway (VIRGO) application runs as a Docker Container alongside all the other native services as part of the SAFR Linux Platform.

10.1 Initial Configuration

The VIRGO container starts for the first time with no factory configuration file. It will remain in this state until a new configuration has been generated and activated.

10.2 Configuration

The factory configuration file is generated when the following configuration script is called by CoVi during licensing (kickoff):

```
/opt/RealNetworks/SAFR/virgo/app/virgo/app/virgo_configure.sh
```

The script requires both a *username* and a hashed *password* to be passed in.

NOTE: If either of these are missing the script will not generate the configuration.

The script requires a template file `/opt/RealNetworks/SAFR/virgo/app/virgo/config/virgo-factory.template` in order to generate a new configuration.

Once executed the script will generate a working configuration and will store it in the following file.

NOTE: The existing configuration will be overwritten!

```
/opt/RealNetworks/SAFR/virgo/app/virgo/config/virgo-factory.conf
```

After the configuration is generated the VIRGO container will be restarted to activate the newly generated configuration.

10.3 Service Status

There are two ways to confirm that VIRGO is running and to confirm how long it has been operational.

1. Use the **check** utility located in `/opt/RealNetworks/SAFR/bin`
2. Use the Docker command to show active running containers:
 - **# sudo docker ps**

CONTAINER	ID	IMAGE	COMMAND	CREATED
	STATUS	PORTS	NAMES	
cf2a2dd33875		safr_virgo:1.1.38	"/bin/sh -c \$VIRGO_AÃq"	18 hours ago
	Up 18 hours		safr_virgo	

If there is no output check the following:

- Run the same command again with the `-a` switch to determine if the container is stopped or restarting (i.e. failing).
- Verify there is a valid `virgo-factory.conf` file located in `/opt/RealNetworks/SAFR/virgo/configs/`
 - Correct user name and IP address. (Same as host IP.)
 - Password is not readable so hard to validate

10.4 Execution

VIRGO container will remain operational both after a failure has occurred as well as if the OS is restarted.

The container is automatically started by the SAFR Platform Installer and stopped by the SAFR Platform Uninstaller.

10.5 Logging

Execute the following command to provide logging output.

```
sudo docker exec -it safr_virgo /opt/RealNetworks/virgo/virgo service log
<log options>
```

NOTE: Refer to the VIRGO Logging documentation for more information on logging options.

10.6 Service Monitor

Live view

```
sudo docker exec -it safr_virgo /opt/RealNetworks/virgo/virgo service
monitor
```

Active Feeds to CSV

```
sudo docker exec -i safr_virgo /opt/RealNetworks/virgo/virgo service
monitor > {CSV File} --active-only
```

NOTE: The stats are added to the CSV file every second so the usable data can be large depending on the number of active feeds.

10.7 Add Volume Mount to Existing Container

1. Update the compose file to add the additional volume instructions.

- The format is <local folder>:<docker folder>
- The <docker folder> will be created if not already existing.

Example: (Your folder names might be different.)

```
version: "3.6"
services:
  virgo:
    image: safr_virgo:1.2.12
    container_name: safr_virgo
    restart: on-failure
    pid: "host"
    volumes:
      - /opt/RealNetworks/SAFR/virgo/config:/etc/virgo
      - /opt/RealNetworks/SAFR/virgo/files:/opt/RealNetworks/virgo/files
```

2. Create the local folder to mount into the container.

```
# mkdir -p /opt/RealNetworks/SAFR/virgo/files
```

3. Stop and delete the container.

```
# docker-compose -f /opt/RealNetworks/SAFR/virgo/app/docker-compose.yml
down
```

4. Create a container instance with new volume mount.

```
# docker-compose -f /opt/RealNetworks/SAFR/virgo/app/docker-compose.yml
up -d
```

5. Create a test file in local mount point.

```
# touch /opt/RealNetworks/SAFR/virgo/files/testfile
```

6. Check that the test file exists inside the container's mount location.

```
# docker exec -it safr_virgo ls -l /opt/RealNetworks/virgo/files
```

11 GPU Support

Starting with version 1.1.16, Video Recognition Gateway (VIRGO) supports acceleration of video decoding, graphics processing, and face detection functions via one or more GPUs. VIRGO automatically detects the presence of a compatible graphics card and will use it. On systems without a GPU, VIRGO falls back to doing everything on the CPU.

Only Nvidia Compute Unified Device Architecture (CUDA) GPUs are currently supported.

11.1 Linux GPU Requirements and Prerequisites

NVIDIA drivers version 418.67 or greater are required. The CUDA toolkit is not required. In addition, you need to install some prerequisites as described below.

11.1.1 Install Prerequisites

1. Install dependencies.
 - For Ubuntu: Run `DEBIAN_FRONTEND=noninteractive apt-get update -y && apt-get install -y gcc make`
 - For Centos: Run `yum install -y gcc make kernel-devel`
 - For Amazon: Run `yum install -y gcc make "kernel-devel-uname-r == $(uname -r)"`
2. Download the most recent NVIDIA Linux drivers from <https://www.nvidia.com/object/unix.html>.
 - Example: `curl -LO http://us.download.nvidia.com/tesla/418.67/NVIDIA-Linux-x86_64-418.67.run`
3. Stop x-windows, if running:
 - For Ubuntu: Run `service lightdm stop`
4. Run driver installer:
 - Run `sudo bash NVIDIA-Linux-x86_64-418.67.run --silent`
5. Verify that your installation was successful:
 - Run `nvidia-smi`

```
[root@ip-10-232-103-191 ~]# sudo bash NVIDIA-Linux-x86_64-418.67.run --silent
Verifying archive integrity... OK
Uncompressing NVIDIA Accelerated Graphics Driver for Linux-x86_64 418.67.....

.....

.....

.....

WARNING: nvidia-installer was forced to guess the X library path '/usr/lib64' and X
module path '/usr/lib64/xorg/modules'; these paths were not queryable from
the system. If X fails to find the NVIDIA X driver module, please install
the 'pkg-config' utility and the X.Org SDK/development package for your
distribution and reinstall the driver.

[root@ip-10-232-103-191 ~]# nvidia-smi
Wed Jul 3 07:52:34 2019

+-----+
| NVIDIA-SMI 418.67      Driver Version: 418.67      CUDA Version: 10.1      |
+-----+-----+-----+
| GPU   Name                Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
+-----+-----+-----+
|    0  Tesla V100-SXM2...    Off      | 00000000:00:1E:0 Off |                    0 |
| N/A   39C    P0      39W / 300W | 0MiB / 16130MiB |      0%      Default |
+-----+-----+-----+

+-----+
| Processes:                                     GPU Memory |
|  GPU       PID    Type    Process name                     Usage      |
+-----+-----+-----+
| No running processes found                  |
+-----+
```

6. If your installation was unsuccessful, view the log:
 - Run `less /var/log/nvidia-installer.log`

11.2 Windows GPU Requirements

NVIDIA drivers version 418.67 or greater are required. The CUDA toolkit is not required.

11.3 Enable a Feed to Run on a GPU

There's nothing you need to do to make this happen; VIRGO automatically detects the presence of a suitable GPU and assigns a feed to it. A feed will automatically fall back to the CPU if there's a problem with the GPU or all GPU resources have been exhausted.

VIRGO also takes advantage of multiple GPUs installed in the system. It automatically distributes feeds across all available GPUs. This enables you to easily scale up a system which allows you to run more feeds on a single VIRGO host.

VIRGO returns comprehensive statistical information about a feed. This includes information about which GPU a feed is running on as well as how much of its processing power it's using per second.

11.3.1 Manual Feed Assignment

Sometimes more control over which feed is assigned to the CPU vs a GPU is desired. VIRGO allows you to individually specify for each feed whether it should exclusively run on a GPU or the CPU. This allows you to

maximize the use of all available GPUs and the CPU by assigning some feeds exclusively to the GPU and some exclusively to the CPU. The following table shows the available feed accelerator configurations:

VIRGO Feed Property	Property Value	Description
accelerator	auto	VIRGO will automatically pick the best available acceleration type. For example, VIRGO will assign the feed to one of the available GPUs if there is still processing capacity available. Otherwise VIRGO will assign the feed to the CPU.
accelerator	cpu	The feed will exclusively run on the CPU and not use any GPU even if a GPU is available.
accelerator	gpu	The feed will exclusively run on a GPU and not use the CPU for video decoding, graphics processing, or detection. The feed will fail if no GPU is available.

12 Service Logging

The Video Recognition Gateway (VIRGO) command line tool has a simple logger built in. You enable logging by executing the following command in a shell:

```
> virgo service log <log specification>
```

where the log specification is a space-separated list of log predicates. A log predicate looks like this:

```
level/tag  
level/tag[feedName]
```

The first variant sets the log level for the package *tag* to *level* on a global basis. Consequently this log predicate applies to the VIRGO daemon and all feeds it spawns. The second variant allows you to apply the log predicate to a single feed with the name *feedName*. If you specify both a global- and a feed-specific log level for a tag then the level with higher priority is applied.

Note: The VIRGO daemon does not keep a log history. Log information is only generated and retained while you are actively running a `virgo service log` command.

Examples:

Enable DEBUG-level logging for the ‘tracking’ package in all feeds:

```
> virgo service log D/tracking
```

Enable DEBUG-level logging for the ‘capture’ and the ‘cop-http’ packages in all feeds:

```
> virgo service log D/capture D/cop-http
```

Enable DEBUG-level logging for the ‘tracking’ package in the feed ‘foo’: (This does not change the current log configuration for any other feed.)

```
> virgo service log D/tracking[foo]
```

The following log levels are supported:

Level	Description
V	Verbose
D	Debug
I	Info
W	Warn
E	Error
O	Off

The order in terms of verbosity, from least to most verbose is OFF, ERROR, WARN, INFO, DEBUG, and VERBOSE.

The following log packages are supported:

Package	Supports feed name?	Description
detection	yes	Object detector related messages
recognition	yes	Face recognizer related messages
tracking	yes	Object tracker messages
capture	yes	Image capture related messages
events	yes	Event reporting related messages
pose-liveness	yes	Pose Liveness Action Recognizer related messages

Package	Supports feed name?	Description
feed	yes	Feed life cycle related messages
cop-http	no	COP over HTTP related messages
config	no	<i>Virgod</i> configuration management related messages
updates	no	<i>Virgod</i> update initiation mechanism related messages

13 Service Monitoring

The Video Recognition Gateway (VIRGO) command line tool has a service monitoring user interface built in. Execute the following command in a shell window to activate continuous monitoring:

```
> virgo service monitor
```

After executing this command, VIRGO clears the terminal window and presents the following live screen:

Status	Feed	PID	Epoch	P-Time	Resolution	FPS	DPS	
	dDt	dRt	#D	#D-Badge	#D-Face	#D-Skip	#R	#R-Face
	#R-Err	#R-Skip	#Evt	%CPU	GPU#	GPU	GPU-Name	
ok		camera_1	14536	12/06/17	00:24:13.450	1280x720	120	8ms
	250ms	120	18	10	0	0	8	0
	0		1240	1%	0	VF	GTX 1060	
ok		camera_2	67289	13:07:12	80:10:00.000	1920x1080	29.97	8ms
	250ms	1920	1400	0	0	0	1000	50
	0		10	4%	1	VF	GTX 1050	1
inactive		camera 3						

Note that the screen is live, which means that VIRGO continuously updates it every second. You can quit monitoring by pressing the ‘q’ key or by pressing Ctrl-C. Also please keep in mind that VIRGO only shows as many columns as fit on the screen. If you do not see all columns then this means that your terminal window is not wide enough. Make the window wider to see all of the columns.

The service monitor UI allows you to scroll up and down when there are more feeds than fit vertically in the terminal window. Use the cursor up key to scroll up and the cursor down key to scroll down.

The following table explains what the various columns in the monitoring output mean:

Column Name	Description
Status	The feed status. This is one of ok, inactive, eos, error, or failure.
Feed	The feed name.
PID	The PID of the feed daemon if the daemon is running
Epoch	The time when the feed processed the first frame in the video stream.
P-Time	The amount of time that the feed has spent on processing the video stream. This is in terms of milliseconds.
Resolution	The width and height of a video frame in pixels
FPS	The frames per second of the input video.
DPS	The number of detections per second.
dDt	The latency of a single detection operation in milliseconds.
dRt	The latency of a single recognition operation in milliseconds.
#D	The number of detection operations that have been triggered.
#D-Badge	The number of badges that have been detected.
#D-Face	The number of faces that have been detected.
#D-Skip	The number of detection operations that have been skipped due to detector overcommitment. This means that no detector was available for a video frame because all detectors were busy at that time.

Column Name	Description
#R	The number of face recognition or reconfirmation operations that have been triggered.
#R-Face	The number of successful face recognition or reconfirmation operations that have been run.
#R-Err	The number of face recognition or reconfirmation operations that have failed for some reason.
#R-Skip	The number of recognition operations that have been skipped due to recognizer overcommitment. This means that no recognizer was available for a face image because all recognizers were busy at that time.
#Evt	The number of events that have been reported.
%CPU	How CPU is used by the feed. Note that this number is in the range 0% to CPU_COUNT * 100%.
GPU#	The GPU ID. Every GPU in the system is assigned a unique ID. This entry is blank if the feed does not use a GPU.
GPU	A string which indicates which modules in the feed are using the GPU: V -> video decoder F -> face detector B -> badge detector O -> object detector
GPU-Name	An empty/non-existing string indicates that the feed is not using the GPU at all. The name of the GPU. Note that the name is not unique because a system may be equipped with more than one GPU of the same model and make. This entry is blank if the feed does not use a GPU.

13.1 Creating CSV Files

You can create a CSV file with all the information from the live service monitor screen by invoking the service monitor like this:

```
> virgo service monitor > my.csv
```

This command tells VIRGO that it should write the service monitor information into a CSV file instead of showing it on the screen. VIRGO will continue to write feed statistics once per second to the CSV file until you stop it by pressing Control-C in your terminal window.

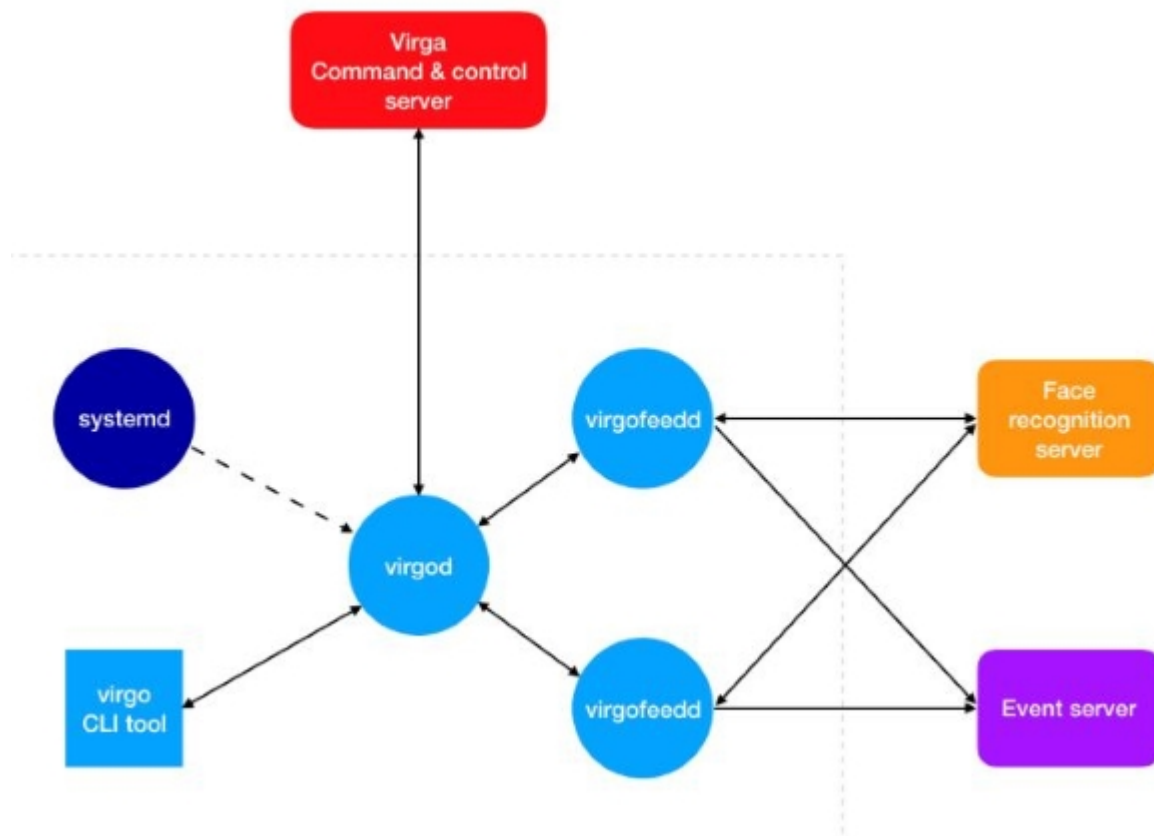
VIRGO writes one line per feed to the CSV file and it repeats this process every second. It even includes inactive feeds by default. If you only want to include active feeds in the CSV file then pass the "--active-only" command line switch to VIRGO.

14 VIRGO Architecture

A single Video Recognition Gateway (VIRGO) installation consists of the following components:

- **virgod**: The VIRGO control daemon. One such daemon is spawned and maintained per VIRGO hardware.
- **virgofeedd**: A *virgod* child process which handles a single video feed.
- **virgo**: The locally available VIRGO command line tool which acts as a Command Line Interface (CLI)-based user interface to the VIRGO daemon.

This diagram shows how those components fit together:



virgod:

- Spawned by the operating system systemd/launchd service. The daemon is automatically restarted by the OS if the hardware power cycles or *virgod* terminates for some unexpected reason.
- Runs as its own user. The VIRGO user is limited to read/write access to the "virgo" home directory.
- The VIRGO user home directory contains just the ~/Library directory which is the place where libFoundation (used in the implementation of VIRGO) stores the daemon settings.
- Is responsible for spawning the per-video-feed child processes: *virgofeedd*.
- *virgod* monitors each *virgofeedd* child process that it has spawned and it automatically restarts a *virgofeedd* if it unexpectedly terminates for some reason. (e.g. it ran out of memory)
- Is responsible for carrying out all the necessary steps for an update to the VIRGO daemon system.
- Is the only process on the machine which talks to the VIRGA command & control server.
- Carries out any command sent by VIRGO to *virgod*.
- Regularly informs the Video Recognition Gateway Administrator (VIRGA) command and control server about the current status of *virgod*.

virgofeedd:

- Spawned by *virgod*.
- Runs as the same user as *virgod*.
- Receives a video stream. Detects and recognizes faces in that video stream, generates events, and reports them to the event server.
- Receives commands from *virgod*.

virgoupdaterd:

- Spawned by *virgod* after it has received an update request.
- Runs as the same user as *virgod*.
- Downloads the update archive, extracts it, installs the update bundle, and saves the current persistent *virgod* state.
- Restarts *virgod*. (*virgod* takes care of data migration.)
- Monitors *virgod* after restart and rolls back to the previous *virgod* version if the new *virgod* fails to startup or fails to check back in with a commit message in less than a couple seconds.
- After the update has finished, the updater exits.

virgo:

- Implements the local (CLI-based) user interface to *virgod*.
- Offers commands to show the current status, selects the cloud environment, gets a screen capture from a feed, etc.

14.1 VIRGO File System Layout

VIRGO ships as a bundle which supports multiple versions of the VIRGO daemon. The VIRGO bundle directory contains a "versions" directory which in turn contains one sub-directory per installed VIRGO version. The name of a version sub-directory is the semantic version number of the VIRGO installation. The "versions" directory also contains a symlink named "current". This symlink points to the version sub-directory which is currently active.

The version sub-directory stores all necessary executable, library, and data files for the VIRGO version.

VIRGO bundle layout:

```
virgo/
  versions/
    1.0.0/
      virgo
      virgod
      virgofeedd
      virgoupdaterd
      lib/
        <shared libraries>
      model/
        <tensor flow model files>
      virgo-factory.config
      current -> ./1.0.0
      virgo -> ./versions/current/virgo
```

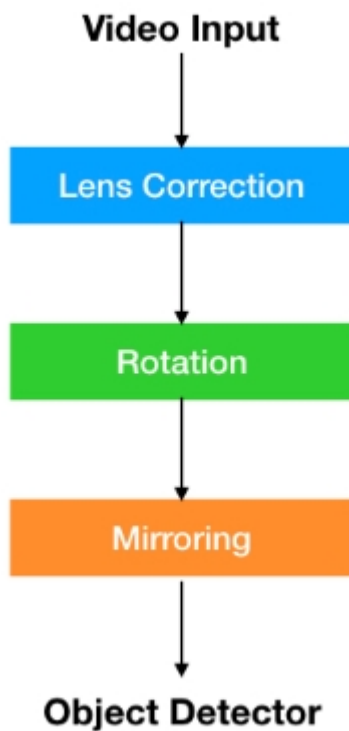
14.2 VIRGO Feeds

A single *virgod* instance manages a set of feeds. Each feed represents a video stream from a camera, a file, or some other video source. Each feed is associated with a set of configuration information which is stored persistently by VIRGO. The configuration information for the feeds can be managed through the VIRGO command line tool or through the video feeds window of the Desktop Client or the Web Console.

Each feed has a name which is unique among the set of feeds of a single *virgod* instance. These names are used as a simple and convenient way to refer to a feed and its configuration. Each feed is managed by a separate *virgofeedd* instance which is started and monitored by *virgod*. *Virgod* will automatically restart a *virgofeedd* instance if it dies for some unexpected reason.

A feed may be enabled or disabled. Only enabled feeds are associated with a *virgofeedd* instance. The enabled state of a feed may be changed through the VIRGO command line tool by issuing a **feed start** or a **feed stop** command. A feed may also be enabled or disabled in the video feeds windows by changing the **enabled** setting. This allows the system to reclaim resources like memory and network bandwidth if a feed is temporarily not needed. Feeds which are no longer needed at all should be removed altogether.

A feed has an input which connects the feed to a video stream. The two types of input currently supported are "stream" and "file". A stream input is specified by a URL which may point to a publicly accessible RTSP or HTTP video stream. Each video frame from the input is first sent through a video post-processing pipeline before it is fed into the object detector and recognizer sub-systems:



First a lens correction algorithm is applied to an incoming video frame. This step removes distortions that may be introduced by the optical system of a camera. After that the image will be rotated to compensate for any undesired rotation that may have been introduced by the physical orientation of the camera. Finally the image may be mirrored to ensure that a camera that is facing a user will produce an image that aligns with what a user expects to see.

15 Troubleshooting

15.1 Linux

15.1.1 Which Linux distributions are supported?

- Ubuntu 16.04(.5+) is known to work and has seen extensive testing.
- Ubuntu 18.04(.2+) appears to work but has not seen extensive testing.
- All other Linux distribution may or may not work; they have not seen any testing.

15.1.2 I just want to do a quick experiment with Video Recognition Gateway (VIRGO). Do I really have to do a full installation?

Actually no. If you just want to run VIRGO temporarily (e.g. to do testing) then there is no need to do a full installation. Do this instead:

1. Create a **virgo-factory.conf** file in your home directory which contains the necessary account, environment, and feed information.
2. Open a shell window and run `virgo/versions/current/virgod -l` in it.
3. Open a second shell window and use it to control VIRGO from there. For example, type `virgo/virgo service monitor` to see the current status of VIRGO.

Once you're done with your work you should terminate VIRGO by typing Control-C in the shell window in which you started *virgod*.

Here is a small example `virgo-factory.conf` file:

```
{
  "global":{
    "environment": "PROD",
    "machine-id-prefix": "vRGo-Rea18L-X-",
    "user-id": "<Your SAFR cloud account ID here>",
    "user-password": "<Your SAFR cloud account password here>",
    "remote-control-enabled":false
  },

  "feeds":{
    "Axis Q6128-E": {
      "directory":"testy",
      "input.type": "stream",
      "input.stream.url":"rtsp://user:password@101.102.103.104/axis-media/media.
    },
  }
}
```

Note that this quick & dirty way of running VIRGO is not suitable for a production system.

For example, VIRGO will stop running as soon as you log out of the system and the VIRGO factory configuration file is not secured which means that passwords (SAFR cloud account, camera IP passwords, etc) may be exposed to 3rd parties.

15.1.3 I've installed VIRGO but all my feeds die with an "Unexpected termination" error. What is wrong?

Your Linux installation is most likely missing a required APT package/library. Please make sure that you follow the installation instructions for Linux precisely. See this page for the list of required APT packages.

To find out which library is exactly missing, invoke the VIRGO feed daemon directly like this:

```
> virgo/versions/current/virgofeedd
```

This will cause the operating system to print the name of the missing library (.so file). Note that this command will print an error message about a missing/broken pipe if no library is missing. This later error is expected but any complaint about a missing dependency/library is not expected and points to a problem you need to fix.

If you see the following, it means that all dependencies are satisfied:

```
> virgo/versions/current/virgofeedd
```

```
Fatal error: 'try!' expression unexpectedly raised an error:
  virgofeedd.DTPErrror.io(message: "Bad file descriptor (9)": file
  /var/lib/jenkins/workspace/ubuntu_16_04_virgo_trunk_daily/build/virgo-build-x86_64-
  line 31
```

If, on the other hand, you see the following, it means that a library is missing:

```
> virgo/versions/current/virgofeedd
```

```
virgo_installer/virgo/versions/current/virgofeedd: error while loading
shared libraries: libcuda.so.1: cannot open shared object file: No
such file or directory
```

15.1.4 I've connected a camera to VIRGO and it is perpetually stuck in prerolling mode with the error `Codec parameters not found` . What's going on?

Some cameras have buggy firmware which fail to generate a correct H264 PPS packet if the RTSP transport protocol is set to UDP. Note that VIRGO connects to RTSP cameras via UDP by default because UDP requires less networking resources and has lower latency compared to TCP.

However in this case and to fix this problem you need to tell VIRGO to connect to the camera using TCP instead. Do this by adding the following property to the feed dictionary for the camera:

```
"input.stream.rtsp.transport": "tcp"
```

15.1.5 I've just installed VIRGO, changed some things in the `virgo-factory.conf` file, and now *virgod* seems to crash all the time?!

Most likely there's a syntax error in the *virgo-factory.conf* file now. For example, you may have forgotten to add a comma at the end of a property. You can run *virgod* like this to see the actual error message:

```
> virgo/versions/current/virgod -l
```

```
Factory config error:
dataCorrupted(Swift.DecodingError.Context(codingPath: [],
debugDescription: "The given data was not valid JSON.",
underlyingError: Optional(Error Domain=NSCocoaErrorDomain Code=3840
"Badly formed object around character 54."
UserInfo={NSDebugDescription=Badly formed object around character
54.})))
```

You can also check the *virgod* exit code. It will be 78 (POSIX EX_CONFIG) if there is a syntax error in the factory configuration file.

Note that this kind of error can not be captured by the VIRGO logging system because it happens at the very startup of *virgod* and before the logging system has been initialized.

15.2 macOS

15.2.1 VIRGO crashes when I try to use it

You are most likely trying to run VIRGO on a system which does not have the Swift 5 runtime libraries installed. VIRGO depends on those libraries and Apple started shipping them with macOS beginning with version 14.4.4. If you are running an older OS and are not able to upgrade to a recent version of macOS then you should download the Swift 5 runtime libraries from Apple. See this support article for instructions on how to do this.

15.3 Docker

15.3.1 Feed reports "No Recogniser Available" after feed is added.

This type of error is normally produced when the Face Service is too busy to accept additional requests for recognition.

It can also be generated when the VIRGO configuration is incorrect and as such the requests are not getting sent to CoVi and thus time out.