# Tutorial Final — Stack NVR com **Podman Compose** (Rocky Linux 10)

**Usuário:** `jmsalles` (rootless) · **Serviços:** Frigate + go2rtc + Mosquitto · **Firewall:** template `--add-port`

## 1) Pré-requisitos (rodar como **root**, uma vez)

```
# (opcional) SELinux fora do caminho
sudo setenforce 0 || true
sudo sed -i 's/^SELINUX=.*/SELINUX=disabled/' /etc/selinux/config

# pacotes base
sudo dnf -y install podman fuse-overlayfs slirp4netns shadow-utils firewalld
pipx

# firewalld ativo
sudo systemctl enable --now firewalld

# rootless user-namespace OK (setuid + faixas únicas de subuid/subgid)
sudo chmod 4755 /usr/bin/newuidmap /usr/bin/newgidmap
sudo sed -i '/^jmsalles:/d' /etc/subuid
sudo sed -i '/^jmsalles:/d' /etc/subgid
echo 'jmsalles:100000:65536' | sudo tee -a /etc/subuid
echo 'jmsalles:100000:65536' | sudo tee -a /etc/subgid

# permitir systemd de usuário iniciar no boot (sem login)
sudo loginctl enable-linger jmsalles
```

## 2) Instalar **podman-compose** (como **jmsalles**)

```
pipx install podman-compose
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc
source ~/.bashrc

podman-compose --version
podman --version
```

## 3) Pastas e **configs** (como **jmsalles**)

```
mkdir -p ~/frigate/{config,media}
mkdir -p ~/frigate/config/mosquitto/{config,data,log}
```

### 3.1 `~/frigate/config/go2rtc.yml` *(ajuste user/pass/IP da sua câmera)*

```
api:    { listen: ":1984" }
rtsp:   { listen: ":8554" }
webrtc: { listen: ":8555" }

streams:
  garagem_main: "rtsp://user:pass@192.168.31.201:554/Streaming/Channels/101"
  garagem_sub:  "rtsp://user:pass@192.168.31.201:554/Streaming/Channels/102"
```

### 3.2 `~/frigate/config/mosquitto/config/mosquitto.conf`

```
listener 1883 0.0.0.0
allow_anonymous true
persistence true
persistence_location /mosquitto/data/
log_timestamp true
log_type error
log_type warning
log_type notice
log_type information
```

### 3.3 `~/frigate/config/frigate.yml` *(1 câmera: detect no substream; gravação no main)*

```
mqtt:
  host: 127.0.0.1
  port: 1883

ffmpeg:
  # hwaccel_args: preset-vaapi   # ative depois que VAAPI (iHD) estiver OK
  output_args:
    record: preset-record
    detect: preset-detect

go2rtc:
  streams:
    garagem_main: rtsp://127.0.0.1:8554/garagem_main
    garagem_sub:  rtsp://127.0.0.1:8554/garagem_sub

detectors:
  cpu1: { type: cpu }

cameras:
  garagem:
    detect: { width: 640, height: 360, fps: 10 }
    ffmpeg:
      inputs:
```

```
        - { path: rtsp://127.0.0.1:8554/garagem_sub,  roles: [detect] }
        - { path: rtsp://127.0.0.1:8554/garagem_main, roles: [record, audio] }
    record:
      enabled: true
      retain: { days: 7, mode: motion }
```

---

## 4) **Compose** do stack (`~/frigate/compose.yml`)

```
version: "3.9"

services:
  go2rtc:
    image: ghcr.io/alexxit/go2rtc:latest
    container_name: go2rtc
    volumes:
      - ./config/go2rtc.yml:/config/go2rtc.yaml:Z
    ports:
      - "1984:1984"
      - "8554:8554"
      - "8555:8555/tcp"
      - "8555:8555/udp"
    restart: unless-stopped

  mqtt:
    image: docker.io/eclipse-mosquitto:2
    container_name: mqtt
    command: mosquitto -c /mosquitto/config/mosquitto.conf
    volumes:
      - ./config/mosquitto/config:/mosquitto/config:Z
      - ./config/mosquitto/data:/mosquitto/data:Z
      - ./config/mosquitto/log:/mosquitto/log:Z
    ports:
      - "1883:1883"
    restart: unless-stopped

  frigate:
    image: ghcr.io/blakeblackshear/frigate:stable
    container_name: frigate
    shm_size: "256m"
    volumes:
      - ./config:/config:Z
      - ./media:/media:Z
    devices:
      - "/dev/dri:/dev/dri"
    ports:
      - "5000:5000"
    depends_on:
      - mqtt
      - go2rtc
    restart: unless-stopped
```

**Subir o stack:**

```
cd ~/frigate
podman-compose -f compose.yml up -d
podman ps
```

**URLs:**

- Frigate → `http://SEU_IP:5000/`
- go2rtc → `http://SEU_IP:1984/`

---

## 5) **Firewall** (template `--add-port`)

```
sudo firewall-cmd --permanent --add-port=5000/tcp
sudo firewall-cmd --permanent --add-port=1984/tcp
sudo firewall-cmd --permanent --add-port=8554/tcp
sudo firewall-cmd --permanent --add-port=8555/tcp
sudo firewall-cmd --permanent --add-port=8555/udp
sudo firewall-cmd --permanent --add-port=1883/tcp
sudo firewall-cmd --reload

# conferência
sudo firewall-cmd --list-ports
```

**Testes (de outra máquina na LAN):**

```
curl -s http://SEU_IP:1984/api
curl -s http://SEU_IP:5000/ | head
```

---

## 6) **Autostart** com systemd (usuário) chamando o compose

### 6.1 Wrappers (garantem ambiente/`PATH` corretos)

```
mkdir -p ~/bin
cat > ~/bin/frigate-stack-up.sh <<'SH'
#!/usr/bin/env bash
set -euo pipefail
export XDG_RUNTIME_DIR="/run/user/$(id -u)"
export PATH="/usr/bin:/usr/local/bin:/bin:/usr/sbin:/usr/local/sbin:$PATH"
cd "$HOME/frigate"
exec /usr/bin/podman-compose -f compose.yml up -d
SH
```

```
cat > ~/bin/frigate-stack-down.sh <<'SH'
#!/usr/bin/env bash
set -euo pipefail
export XDG_RUNTIME_DIR="/run/user/$(id -u)"
export PATH="/usr/bin:/usr/local/bin:/bin:/usr/sbin:/usr/local/sbin:$PATH"
cd "$HOME/frigate"
exec /usr/bin/podman-compose -f compose.yml down
SH

chmod +x ~/bin/frigate-stack-*.sh
```

### 6.2 Unit do usuário

```
mkdir -p ~/.config/systemd/user
cat > ~/.config/systemd/user/frigate-stack.service <<'UNIT'
[Unit]
Description=Frigate/go2rtc/MQTT via podman-compose (rootless)
Wants=network-online.target
After=network-online.target default.target

[Service]
Type=oneshot
ExecStart=%h/bin/frigate-stack-up.sh
ExecStop=%h/bin/frigate-stack-down.sh
RemainAfterExit=yes
TimeoutStartSec=0

[Install]
WantedBy=default.target
UNIT

systemctl --user daemon-reload
systemctl --user enable --now frigate-stack.service
```

**Status/Logs:**

```
systemctl --user status frigate-stack.service --no-pager
journalctl --user -xeu frigate-stack.service --no-pager
```

> Boot sem login já garantido por: `sudo loginctl enable-linger jmsalles`

---

## 7) (Opcional) Habilitar **VAAPI (Intel iHD)** depois

1. Instale **intel-media-driver (iHD)** no host e valide `iHD_drv_video.so`.
2. Ative no `~/frigate/config/frigate.yml`:

```
ffmpeg:
  hwaccel_args: preset-vaapi
```

3. Aplique:

```
podman-compose -f ~/frigate/compose.yml up -d
podman exec -it frigate ffmpeg -hide_banner -hwaccels | grep -i vaapi ||
true
```

---

## 8) Troubleshooting rápido

```
# listeners e portas
ss -ltnp | egrep '1984|5000|8554|8555|1883' || true
ss -lunp | egrep '8555' || true
sudo firewall-cmd --list-ports

# logs dos serviços
podman logs --tail=80 go2rtc
podman logs --tail=80 mqtt
podman logs --tail=150 frigate

# streams registradas no go2rtc
curl -s http://SEU_IP:1984/api/streams
```

**Notas:**

- Mensagem do go2rtc sobre `bind IPv6 :8555` é inofensiva; se quiser forçar IPv4:

  ```
  # em go2rtc.yml
  webrtc:
    listen: ":8555"
    ip: "SEU_IP"
  ```

  Depois: `podman-compose up -d`.

- RTSP timeout no Frigate indica URL/credenciais/rota da câmera incorretos; teste no go2rtc primeiro.

---

**Pronto.** Tutorial final no seu modelo: compose declarativo, firewall `--add-port`, autostart via systemd de usuário, 1 câmera funcional e caminho para VAAPI.