Yayın Vlc

VLC İle Yayın Yapma

Benim tercihim kolaylık açısında ffmpeg olurdu ama görev olarak vlc kullanılması istendiği için vlc kullanımını öğrenmem gerekordiyu. Bu aşamada yapay zekadan ve kendi sitelerinde açıklamalardan çok yardım aldım.

```
 Öncelikle bir konuya açıklık getirmeliyim cvlc benim sunucumda bir
 türlü çalışmadı bu yüzden direk vlc kullandım.


 Ayrıca her şeyi ben yazamam bu yüzden de yardım almam gerekiyordu.
 Bu linkte bir çok açıklama mevcuttur:
 https://chatgpt.com/share/68e1244d-ec60-8013-acb8-0b3c6d141ad5
Benim yapacağım özet geçmek olacak.
```

Öncelikle genel olarak local ağda bağlantı yaptığımız için iki cihazın da aynı ağda bağlı olduğundan emin olmalıyız.

VLC Komut Satırı Örnekleri (Sunucu ve İstemci tarafı)

Aşağıda yaygın kullanılan, H.264 kodlaması içeren örnekler. Bu komutları sisteminize özel parametrelere göre uyarlamalısınız.

> Not: VLC'nin komut satırı yardımcı modülleri (örneğin `--sout`, `transcode`, `standard`, `rtp`, `sdp`, vs.) VideoLAN'ın streaming wiki sayfalarında ayrıntılı şekilde anlatılıyor (VLC command line help).

2.1 Sunucu tarafı – Dosyadan yayın

Örnek: `video.mp4` dosyasını LAN içerisinde RTSP olarak yayınlama:

```
vlc -vvv /path/to/video.mp4 \   --sout
'#transcode{vcodec=h264,vb=2000,scale=1,acodec=none}:rtp{sdp=rtsp://:
8554/stream}'
```

Açıklamalar:

- `-vvv` — verbose çıktı, hata / debug bilgisi verir.
- `transcode{vcodec=h264,vb=2000,scale=1}` — video codec'i H.264, bit hızı 2000 kb/s, ölçek %100 (orijinal çözünürlük). Gerekirse `scale`, `fps`, `width` / `height` parametreleri ekleyebilirsin.
- `acodec=none` — ses kısmı yok (ya da ses istemiyorsan devre dışı bırak).
- `rtp{sdp=rtsp://:8554/stream}` — RTP ile paketle, SDP (Session Description Protocol) manifestini `rtsp://<sunucuIP>:8554/stream` URI'si üzerinden sun.

İstemci bu URI ile bağlanabilir:

```
vlc rtsp://192.168.1.100:8554/stream
```

İstemci tarafı komutu

İstemcide gecikmeyi optimize etmek için tampon (caching / network-caching) ayarı önemli. Örneğin:

```
vlc -vvv --network-caching=150 rtsp://192.168.1.100:8554/stream
```

Burada `--network-caching=150` milisaniye değeridir. Çok düşük olursa takılmalar olabilir, çok yüksek olursa gecikme artar.

Alternatif olarak (TCP interleaved üzerinden), istemci tarafında:

```
vlc -vvv --rtsp-tunnel --network-caching=150
rtsp://192.168.1.100:8554/stream
```

Ya da VLC arayüzünde "Input / Codecs → RTP/RTSP → Use RTP over RTSP (TCP)" gibi seçenek aktif edilir.

2.4 HTTP (alternatif) yayın örneği

Eğer HTTP üzerinden yayın yapmak istersen (örneğin HTTP Progressive / TS):

```
vlc -vvv /path/to/video.mp4 \   --sout
'#transcode{vcodec=h264,vb=1500,scale=1}:std{access=http{mux=ts,dst=:
```

```
8080/},mux=ts}'
```

⚡ 1. Minimum Gecikme (Low-Latency) Optimizasyonu

Gecikme (latency) üç noktadan gelir:

1. **Kodlama (encoding)** süresi,
2. **Ağ tamponlaması (network buffering)**,
3. **İstemci tarafı video tamponu (playback buffer)**.

Bu üçüne ayrı ayrı müdahale edebilirsin.

1.1. VLC yayın tarafında gecikme azaltma

Komut satırında şu ayarları uygula:

`--sout-mux-caching=50 --network-caching=50 --live-caching=50 --sout-rtp-caching=50`

Bu parametreler VLC'nin kullandığı iç buffer'ların (milisaniye cinsinden) süresini azaltır.
Örneğin `--live-caching=50` → canlı kaynak (kamera) için 50 ms tampon.
Varsayılan değerler genelde 1000–3000 ms civarındadır (yani 1–3 saniye gecikme yaratır).

Ama dikkat: çok düşürürsen takılma başlayabilir.
Tavsiye edilen aralık: 100–200 ms.

Örnek:

```
"C:\Program Files\VideoLAN\VLC\vlc.exe" dshow:// :dshow-vdev="Integrated Camera" ^ --intf=dummy --vout=dummy ^ --sout="#transcode{vcodec=h264,vb=1500,scale=1}:rtp{sdp=rtsp://:8554/live}" ^ --sout-mux-caching=100 --network-caching=100 --live-caching=100`
```

🎬 1. İstemci (Viewer) tarafında otomatik yeniden bağlanma

VLC’nin doğrudan “otomatik reconnect” parametresi yok, ama bunu şu yollarla çözebilirsin:

Yöntem 1: Döngü (loop) mantığı ile yeniden bağlanma

Bir `.bat` veya `.sh` dosyası hazırlarsın; VLC kapandığında (yani bağlantı koptuğunda) yeniden başlar:

Windows için (`reconnect.bat`):

```
@echo off
:loop
"C:\Program Files\VideoLAN\VLC\vlc.exe"
rtsp://192.168.1.10:8554/live.sdp --network-caching=100
timeout /t 3
goto loop
```

Bu script, VLC kapanırsa 3 saniye bekler ve tekrar açar.
Kopmalar 2–3 saniyelik gecikmeyle telafi edilir.

Yöntem 2: VLC’nin `--repeat` veya `--loop` parametresi

Eğer VLC, RTSP bağlantısı koptuğunda dosya gibi “bitti” sinyali gönderiyorsa bu işe yarar:

```
vlc rtsp://192.168.1.10:8554/live.sdp --repeat --network-caching=100
```

Ama dikkat: Bazı RTSP sunucuları kopmayı “bitme” olarak algılamaz, bu durumda VLC donuk kalabilir. O yüzden bu yöntem her zaman işe yaramaz.

📡 2. Sunucu (Streaming) tarafında sürekli yayın

Senin tarafında (yayın yapan taraf) VLC'nin yayını kesmeden devam etmesi için şunlara dikkat et:

RTSP yayını için

```
vlc dshow:// :dshow-vdev="Integrated Camera" ^
--sout "#rtp{sdp=rtsp://:8554/live.sdp}" ^
--sout-keep
```

Buradaki `--sout-keep`, "bağlantı koptuğunda yayın pipeline'ını kapatma, devam et" anlamına gelir.
Bu sayede istemci yeniden bağlandığında aynı yayını kaldığı yerden alır.

HTTP yayını için

HTTP tabanlı yayınlarda da benzer mantık geçerli:

```
vlc dshow:// :dshow-vdev="Integrated Camera" ^
--sout
"#transcode{vcodec=h264,acodec=mp4a}:http{mux=ffmpeg{mux=flv},dst=:80
80/live}" ^
--sout-keep
```

⚙️ 3. Gecikme ve yeniden bağlanma optimizasyonu

Bu ayarları da eklersen daha kararlı olur:

```
--rtsp-tcp                  # UDP yerine TCP kullanarak daha sağlam
bağlantı
--network-caching=100       # 100ms buffer (gecikmeyi azaltır)
--sout-rtp-caching=50       # RTP buffer süresi
--file-caching=100          # İstemci tarafında da düşük buffer
```

🎯 Özetle

Durum	Çözüm	Açıklama
İstemci koparsa	`.bat` loop veya `--repeat`	Yeniden bağlanır
Sunucu yayını kopturmaz	`--sout-keep`	Pipeline'ı açık tutar
Bağlantı daha sağlam olsun	`--rtsp-tcp`	UDP yerine TCP kullanır
Düşük gecikme	`--network-caching=100`	100ms buffer