Video Stream Exporter - Prometheus 指标 API 文档

概述

Video Stream Exporter 通过 Prometheus 格式暴露视频流监控指标，供监控后台系统调用。所有指标通过 HTTP 端点 `/metrics` 提供，支持 Prometheus 标准查询语言（PromQL）。

访问端点

- **指标端点**: `http://<listen_addr>/metrics`
- **默认地址**: `http://localhost:8080/metrics`
- **格式**: Prometheus Text Format
- **更新频率**: 每次请求时实时更新

指标类型

所有指标均为 **Gauge** 类型，表示当前时刻的状态值，可以上升或下降。

通用标签（Labels）

所有指标都包含以下 4 个标签，用于标识和过滤流：

标签名	说明	示例值
`project`	项目名称	`"project1"`
`id`	流标识符（桌台ID）	`"D001"`
`name`	流名称	`"stream-01"`
`url`	流地址	`"https://example.com/live/stream.flv"`

指标分类

1. 流状态指标

`video_stream_up`

功能: 指示流是否在线（可连接）

标签: `project`, `id`, `name`, `url`

值范围:

- `1`: 流在线（可连接）
- `0`: 流离线（无法连接）

示例:

```
video_stream_up{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 1
```

使用场景:

- 监控流是否在线
- 计算在线率：`sum(video_stream_up) / count(video_stream_up) * 100`
- 告警：`video_stream_up == 0`

`video_stream_healthy`

功能: 指示流的健康状态

标签: `project`, `id`, `name`, `url`

值范围:

- `1`: 健康（流在线且无连续失败）
- `0`: 不健康（流离线或存在连续失败）

示例:

```
video_stream_healthy{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 1
```

使用场景:

- 监控流健康状态
- 计算健康率：`sum(video_stream_healthy) / count(video_stream_up) * 100`
- 告警：`video_stream_healthy == 0`

`video_stream_playable`

功能: 指示流是否可播放（满足最低播放要求）

标签: `project`, `id`, `name`, `url`

值范围:

- `1`: 可播放（满足最低关键帧要求）
- `0`: 不可播放（关键帧不足）

示例:

```
video_stream_playable{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 1
```

使用场景:

- 监控流可播放性
- 计算可播放率：`sum(video_stream_playable) / count(video_stream_up) * 100`
- 告警：`video_stream_playable == 0`

---

2. 数据包统计指标

`video_stream_total_packets`

功能: 采样周期内接收到的总数据包数

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 包数

示例:

```
video_stream_total_packets{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 1250
```

使用场景:

- 监控数据包接收情况
- 分析流数据量
- 检测数据包丢失

`video_stream_video_packets`

功能: 采样周期内接收到的视频数据包数

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 包数

示例:

```
video_stream_video_packets{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 850
```

使用场景:

- 监控视频数据包接收情况
- 分析视频流数据量
- 计算视频包占比：`video_stream_video_packets / video_stream_total_packets`

`video_stream_audio_packets`

功能: 采样周期内接收到的音频数据包数

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 包数

示例:

```
video_stream_audio_packets{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 400
```

使用场景:

- 监控音频数据包接收情况
- 分析音频流数据量
- 计算音频包占比：`video_stream_audio_packets / video_stream_total_packets`

`video_stream_keyframes`

功能: 采样周期内接收到的关键帧（I帧）数量

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 帧数

示例:

```
video_stream_keyframes{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 15
```

使用场景:

- 监控关键帧数量
- 判断流是否可播放（需要至少 2 个关键帧）
- 分析 GOP 结构

---

3. 码率指标

`video_stream_bitrate_bps`

功能: 当前流的实时码率

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（浮点数）

单位: 比特每秒（bps）

示例:

```
video_stream_bitrate_bps{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 753000.0
```

使用场景:

- 监控实时码率
- 检测码率异常（过高或过低）
- 告警：`video_stream_bitrate_bps < 500000`（低于 500Kbps）
- 转换为 Kbps：`video_stream_bitrate_bps / 1000`

`video_stream_avg_bitrate_bps`

功能: 采样周期内的平均码率

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（浮点数）

单位: 比特每秒（bps）

示例:

```
video_stream_avg_bitrate_bps{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 720000.0
```

使用场景:

- 监控平均码率
- 分析码率稳定性
- 与实时码率对比，评估码率波动

---

4. 质量指标

video_stream_framerate

功能: 流的帧率

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（浮点数）

单位: 帧每秒（fps）

示例:

```
video_stream_framerate{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 30.0
```

使用场景:

- 监控帧率
- 检测帧率异常（过低）
- 告警：`video_stream_framerate < 15`（低于 15fps）

---

video_stream_response_ms

功能: FLV HTTP 请求的响应时间

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 毫秒（ms）

示例:

```
video_stream_response_ms{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 150
```

使用场景:

- 监控连接响应时间
- 检测网络延迟
- 告警：`video_stream_response_ms > 2000`（超过 2 秒）

---

video_stream_gop_size

功能: GOP（Group of Pictures）大小，即两个关键帧之间的帧数

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 帧数

示例:

```
video_stream_gop_size{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 30
```

使用场景:

- 监控 GOP 结构
- 分析编码参数
- 评估流切换延迟（GOP 越大，切换延迟越高）

`video_stream_quality_score`

功能: 流质量评分（综合码率、帧率、关键帧等因素）

标签: `project`, `id`, `name`, `url`

值范围:

- `2`: 良好（good）
- `1`: 一般（fair）
- `0`: 较差（poor）

示例:

```
video_stream_quality_score{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 2
```

使用场景:

- 监控流质量等级
- 计算质量分布：`count by (quality_score) (video_stream_quality_score)`
- 告警：`video_stream_quality_score < 1`（质量低于一般）

`video_stream_stability_score`

功能: 码率稳定性评分

标签: `project`, `id`, `name`, `url`

值范围:

- `2`: 稳定（stable）

- `1`: 中等（moderate）
- `0`: 不稳定（unstable）

示例:

```
video_stream_stability_score{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 2
```

使用场景:

- 监控码率稳定性
- 分析码率波动情况
- 告警：`video_stream_stability_score < 1`（稳定性低于中等）

---

5. 网络指标

`video_stream_rtt_ms`

功能: 往返时延（Round-Trip Time）

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 毫秒（ms）

示例:

```
video_stream_rtt_ms{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 45
```

使用场景:

- 监控网络延迟
- 检测网络质量
- 告警：`video_stream_rtt_ms > 200`（RTT 超过 200ms）

`video_stream_packet_loss_ratio`

功能: 数据包丢失率

标签: `project`, `id`, `name`, `url`

值范围: `0.0 - 1.0`（浮点数）

- `0.0`: 无丢包
- `1.0`: 100% 丢包

单位: 比率（ratio）

示例:

```
video_stream_packet_loss_ratio{project="project1",id="D001",name="stream-01",url="https://example.com/live/stream.flv"} 0.02
```

使用场景:

- 监控网络丢包情况
- 检测网络质量
- 告警：`video_stream_packet_loss_ratio > 0.05`（丢包率超过 5%）
- 转换为百分比：`video_stream_packet_loss_ratio * 100`

`video_stream_network_jitter_ms`

功能: 网络抖动（数据包到达时间的变化）

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 毫秒（ms）

示例:

```
video_stream_network_jitter_ms{project="project1",id="D001",name="stream-01",url="https://example.com/live/stream.flv"} 12
```

使用场景:

- 监控网络抖动
- 检测网络稳定性
- 告警：`video_stream_network_jitter_ms > 50`（抖动超过 50ms）

`video_stream_reconnect_count`

功能: 当前检查周期内的重连次数

标签: `project`, `id`, `name`, `url`

值范围: `>= 0`（整数）

单位: 次数

示例:

```
video_stream_reconnect_count{project="project1",id="D001",name="stream-
01",url="https://example.com/live/stream.flv"} 0
```

使用场景:

- 监控连接稳定性
- 检测频繁重连问题
- 告警：`video_stream_reconnect_count > 0`（存在重连）
- 注意：此指标为 Gauge 类型，表示当前周期的重连次数，不是累计值

API 调用示例

1. 获取所有指标

```
curl http://localhost:8080/metrics
```

2. 使用 PromQL 查询

查询所有在线流

```
video_stream_up == 1
```

查询特定项目的流

```
video_stream_up{project="project1"}
```

查询特定桌台的流

```
video_stream_up{project="project1", id="D001"}
```

计算在线率

```
sum(video_stream_up) / count(video_stream_up) * 100
```

查询码率低于阈值的流

```
video_stream_bitrate_bps < 500000
```

查询响应时间超过阈值的流

```
video_stream_response_ms > 2000
```

按项目统计在线流数量

```
sum(video_stream_up) by (project)
```

查询质量评分分布

```
count by (quality_score) (video_stream_quality_score)
```

集成 Prometheus

Prometheus 配置示例

```yaml
scrape_configs:
  - job_name: 'video-exporter'
    scrape_interval: 30s
    static_configs:
      - targets: ['localhost:8080']
```

Grafana 查询示例

在线流数量

```
sum(video_stream_up{project=~"$project", id=~"$id", name=~"$name"})
```

平均码率（Kbps）

```
avg(video_stream_bitrate_bps{project=~"$project", id=~"$id",
name=~"$name"}) / 1000
```

平均帧率

```
avg(video_stream_framerate{project=~"$project", id=~"$id", name=~"$name"})
```

平均 RTT

```
avg(video_stream_rtt_ms{project=~"$project", id=~"$id", name=~"$name"})
```

平均丢包率（百分比）

```
avg(video_stream_packet_loss_ratio{project=~"$project", id=~"$id",
name=~"$name"}) * 100
```

告警规则示例

Prometheus AlertManager 配置

```yaml
groups:
  - name: video_stream_alerts
    rules:
      # 流离线告警
      - alert: StreamDown
        expr: video_stream_up == 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "流离线: {{ $labels.project }}/{{ $labels.id }}/{{
$labels.name }}"

      # 码率过低告警
      - alert: LowBitrate
        expr: video_stream_bitrate_bps < 500000
        for: 2m
        labels:
          severity: warning
        annotations:
          summary: "码率过低: {{ $labels.name }} ({{ $value }} bps)"

      # 响应时间过长告警
      - alert: HighResponseTime
        expr: video_stream_response_ms > 2000
        for: 2m
```

```
        labels:
          severity: warning
        annotations:
          summary: "响应时间过长: {{ $labels.name }} ({{ $value }} ms)"

      # 丢包率过高告警
      - alert: HighPacketLoss
        expr: video_stream_packet_loss_ratio > 0.05
        for: 2m
        labels:
          severity: warning
        annotations:
          summary: "丢包率过高: {{ $labels.name }} ({{ $value |
humanizePercentage }})"

      # 质量评分过低告警
      - alert: PoorQuality
        expr: video_stream_quality_score < 1
        for: 3m
        labels:
          severity: warning
        annotations:
          summary: "流质量较差: {{ $labels.name }}"
```

指标更新机制

更新频率

- **检查间隔**: 由配置项 `check_interval` 控制（默认 600 秒）
- **采样时长**: 由配置项 `sample_duration` 控制（默认 10 秒）
- **指标更新**: 每次访问 `/metrics` 端点时实时更新

指标生命周期

- 指标在流检查完成后立即更新
- 当流从配置中移除时，指标不再更新
- Prometheus 会根据 `scrape_interval` 定期抓取指标
- 指标在 Prometheus 中的保留时间由 Prometheus 配置决定

注意事项

1. **指标类型**: 所有指标均为 Gauge 类型，表示当前状态值
2. **重连次数**: `video_stream_reconnect_count` 是 Gauge 类型，表示当前周期的重连次数，不是累计值
3. **零值处理**: 当指标值为 0 时，Prometheus 可能不暴露该指标（Counter 行为），但 Gauge 类型会正常暴露
4. **标签匹配**: 查询时可以使用正则表达式匹配标签值，如 `project=~"project.*"`
5. **性能考虑**: 每次请求 `/metrics` 都会触发指标更新，建议合理设置 Prometheus 的 `scrape_interval`

版本信息

- **文档版本**: 1.0
- **最后更新**: 2025-01-XX
- **指标版本**: 与 Video Stream Exporter 版本同步

相关文档

- 项目结构文档
- 部署文档
- 故障排查文档