Referencia de Sapsan

Servidor de medios de streaming

Flussonic

© Flussonic 2026

Table of contents

1. Sapsan 3
   1.1 Funcionalidades principales 3
   1.2 Inicio rápido 3
   1.3 Navegador de la documentación 3
2. Arquitectura técnica 4
   2.1 Componentes 4
   2.2 Flujos de datos 4
   2.3 Direccionamiento del contenido 4
   2.4 Ciclo de vida de la configuración 5
   2.5 Pila tecnológica 5
3. Instalación 6
   3.1 Despliegue de la solución 6
   3.2 Licenciamiento 6
   3.3 Configuración inicial 6
4. Licenciamiento 7
   4.1 Disponibilidad de funciones 7
   4.2 Licenciamiento en línea 7
   4.3 Llave USB de licencia para redes aisladas 7
   4.4 Archivos de licencia en línea 7
   4.5 Estados 7
   4.6 Qué viene después 8
5. Configuración de Sapsan 9
   5.1 Estructura de la configuración 9
   5.2 Validación 9
   5.3 Recarga en caliente 10
6. Referencia de funciones 12
   6.1 Referencia de funciones 13
   6.2 Medios 20
   6.3 Gestión 240
   6.4 Clúster 257
   6.5 Dispositivos 268
   6.6 Gateway 273
   6.7 ACS 274
   6.8 Funciones adicionales 275

© Flussonic 2026

1. Sapsan

Flussonic Sapsan (en adelante, Sapsan) es un servidor de medios de streaming. Captura vídeo en vivo por protocolos estándar de la industria, lo procesa (transcodificación, ensamblado multibitrate, capturas), lo graba en su propio archivo DVR en disco y lo entrega a los espectadores tanto en vivo como a partir del archivo.

Propósito: la captura, el procesamiento, la grabación y la entrega de vídeo en vivo en proyectos de streaming por internet, IPTV y videovigilancia. El listado detallado de funciones está en la página características funcionales; el diseño del sistema, en la página arquitectura técnica.

Sapsan y central se entregan juntos como parte de una solución de nivel superior. Sapsan se configura mediante la API de central; los ejemplos de parámetros YAML se encuentran en Configuración.

1.1 Funcionalidades principales

- **Captura**: RTSP (cámaras IP), MPEG-TS (UDP y HTTP), SRT, RTMP, un canal a partir de un archivo, fuente sintética de prueba.
- **Publicación**: RTMP publish, SRT publish, WebRTC WHIP.
- **Reserva de la captura**: varias fuentes por stream con conmutación automática (LSI) y un archivo de reserva.
- **Transcodificación**: preparación de la escalera de calidades multibitrate (MBR) basada en FFmpeg — h264/hevc/av1, aac/opus.
- **Reproducción**: HLS y LL-HLS (fMP4), DASH, MPEG-TS por HTTP (incluido CBR), SRT, RTSP, WebRTC WHEP.
- **Retransmisión**: push de un stream por RTMP, SRT y UDP (multicast/unicast).
- **DVR**: archivo append-only en varios discos, reproducción del archivo, rewind, vistas previas de fotogramas, reproducción sin interrupciones de archivos de otros servidores (remotes).
- **Capturas**: vistas previas JPEG de los streams en vivo y del archivo.
- **Autorización**: tokens de reproducción y publicación, backends de autorización externos.
- **ONVIF**: descubrimiento de cámaras en la red, eventos de movimiento y detección.
- **Gestión**: Admin API, gestión externa de la configuración mediante `managed_by`, recarga en caliente por SIGHUP, compatibilidad con las URL y la API de Flussonic.
- **Pasarela Peeklio (rproxy)**: acceso a cámaras y dispositivos detrás de NAT mediante agentes.
- **Observabilidad**: logs estructurados, métricas de Prometheus, trazas de OpenTelemetry, contadores de calidad de la captura.

1.2 Inicio rápido

- Instalar y ejecutar Sapsan
- Configurar su primer stream
- Reproducir el stream por HLS

1.3 Navegador de la documentación

Consulte la referencia de funciones para encontrar parámetros y comportamientos por dominio y sección de la matriz tarifaria.

- **Medios**
- **Gestión**
- **Clúster**
- **Dispositivos**
- **Gateway**
- **Analítica de vídeo**

© Flussonic 2026

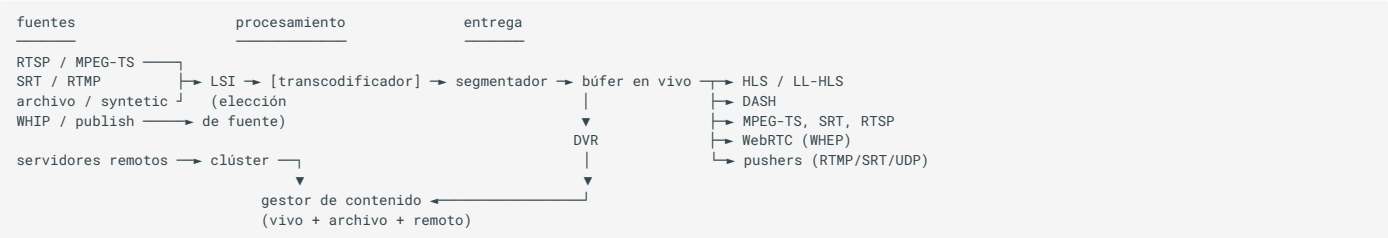
2. Arquitectura técnica

Flussonic Sapsan es una aplicación de servidor escrita en Rust. Todos los subsistemas se ejecutan dentro de un único proceso `sapsan` como tareas del runtime asíncrono tokio e interactúan por canales de mensajes tipados — sin estado mutable compartido entre subsistemas.

2.1 Componentes

Componente	Función
Gestor	carga y validación de la configuración, arranque de los subsistemas, vinculación de los listeners de red, recarga en caliente, Admin API
Listeners de red	HTTP (HLS, DASH, MPEG-TS, WHIP/WHEP, API), RTSP, RTMP, WebRTC (UDP), puertos SRT por stream
Gestor de streams	ciclo de vida de los streams: entradas y conmutación de fuentes (LSI), publicación, pushers, capturas
Transcodificador	decodificación y codificación (FFmpeg): H264/HEVC/AV1, AAC/Opus, la escalera de calidades multibitrate
Segmentador y búfer en vivo	troceado del stream en fragmentos fMP4, el búfer circular del borde en vivo
Gestor de contenido	la superficie unificada de contenido: reúne el búfer en vivo, el archivo local y los archivos remotos; todos los protocolos de entrega leen de ella
DVR	almacenamiento del archivo en disco: blobs append-only por hora, un catálogo sobre una base de datos clave-valor embebida, un indexador en segundo plano (reconstrucción, relleno de metadatos, limpieza)
Clúster	lectura de archivos de servidores Sapsan remotos por el streaming-api interno
Sesiones y autorización	registro de sesiones de espectadores y publicadores, tokens, backends de autorización externos
Pasarela Peeklio	un proxy inverso para dispositivos detrás de NAT: registro de agentes, túneles
Telemetría	logs estructurados, métricas de Prometheus, trazas de OpenTelemetry

2.2 Flujos de datos



- **Camino de escritura**: la entrada activa del stream (la elegida por LSI) entrega los fotogramas; si hace falta, pasan por el transcodificador; el segmentador ensambla fragmentos fMP4, que entran en el búfer en vivo y, en paralelo, se añaden al DVR.
- **Camino de lectura**: un módulo de protocolo pide los datos al gestor de contenido, que elige la fuente de forma transparente — el búfer en vivo, el archivo local o un servidor remoto. Los reproductores nunca saben de dónde llegaron físicamente los datos.
- **Aislamiento de lectura y escritura en el DVR**: la escritura de cada stream se serializa en su propia tarea; las lecturas la esquivan y van directo a los blobs a través de una caché compartida — los lectores no compiten con el escritor.

2.3 Direccionamiento del contenido

La unidad de almacenamiento y entrega es el fragmento, direccionado por tiempo UTC absoluto (DTS + timescale). El mismo nombre de fragmento es válido en el búfer en vivo, en el archivo y en todos los servidores del clúster. Esta propiedad sostiene la unión sin interrupciones del vivo con el

© Flussonic 2026

archivo y la fusión de archivos entre servidores. El contrato de URL (rutas, URI de las playlists, referencias a fragmentos) está definido en un único módulo streaming-api, común al servidor y a los clientes del clúster.

2.4 Ciclo de vida de la configuración

1. La configuración se lee de `sapsan.yaml` y se valida como un todo; una configuración no válida no se aplica.
2. Al recibir SIGHUP, la configuración se relee: los streams, la autorización y el DVR se reconfiguran al vuelo; los listeners solo se revinculan cuando cambia un puerto.

2.5 Pila tecnológica

Capa	Tecnología
Lenguaje	Rust
Runtime asíncrono	tokio (E/S de red, tareas), rayon (paralelismo de CPU)
Asignador	jemalloc
Transcodificación	FFmpeg
Catálogo del DVR	base de datos clave-valor embebida (LSM)
Observabilidad	OpenTelemetry (OTLP), Prometheus

© Flussonic 2026

3. Instalación

Sapsan se entrega como componente de soluciones de nivel superior de Flussonic. Para instalarlo, siga la documentación de la solución elegida: Catena, Mcaster, Agora o Watcher, en inglés. Sapsan y central forman un sistema inseparable y se entregan juntos: Sapsan procesa los medios y central gestiona su configuración.

3.1 Despliegue de la solución

Instale Sapsan y central siguiendo las instrucciones de la solución elegida. Estas instrucciones especifican los componentes suministrados, los requisitos del entorno y el procedimiento de inicio.

Pase los discos de archivo como volúmenes montados: el punto de montaje de cada disco o el volumen montado entero — por ejemplo, `-v /storage:/storage` cuando los discos están montados en `/storage/d1` y `/storage/d2`. Un directorio del host sin un disco montado y un volumen normal de Docker no pasan la comprobación de montaje, y la escritura del archivo en ellos se rechaza. Después de montar un disco en el host, reinicie el contenedor: un contenedor en marcha no ve el nuevo montaje.

3.2 Licenciamiento

Elija el licenciamiento según el entorno: licenciamiento en línea con acceso a internet o una llave USB de licencia para una red aislada. Consulte Licenciamiento.

3.3 Configuración inicial

Tras el despliegue, configure Sapsan mediante la API de central. Los ejemplos YAML de este manual describen los parámetros de Sapsan; estos fragmentos se pueden enviar mediante la API de central. Consulte Configuración para saber cómo usar los ejemplos.

Las funciones y sus parámetros se enumeran en la referencia de funciones.

© Flussonic 2026

4. Licenciamiento

Sapsan admite licenciamiento en línea y licenciamiento con una llave USB para redes aisladas.

4.1 Disponibilidad de funciones

La licencia puede determinar qué funciones y características están disponibles. Por tanto, esta documentación puede abarcar más capacidades de las que están disponibles en una entrega concreta al cliente.

4.2 Licenciamiento en línea

El licenciamiento en línea utiliza un archivo de licencia emitido en el momento de la compra. Puede descargar la clave de licencia desde el área de clientes en my.flussonic.com.

La verificación utiliza los servidores de licenciamiento de Flussonic, por lo que este modo requiere acceso a internet.

4.3 Llave USB de licencia para redes aisladas

Una red aislada sin acceso a internet utiliza una llave USB física de licencia. La llave se suministra para la solución elegida; las instrucciones de conexión y activación se incluyen en la documentación de la entrega.

4.4 Archivos de licencia en línea

En el modo de licenciamiento en línea, el archivo de licencia se almacena junto a la configuración. Sapsan crea los demás archivos en el directorio de estado durante la activación.

Archivo	Ubicación	Contenido
`license.txt`	`/etc/sapsan/` (junto a `sapsan.yaml`)	la clave de licencia emitida en la compra
`server.id`	`/var/lib/sapsan/`	identificador único del servidor (UUID)
`activation-<version>.json`	`/var/lib/sapsan/`	activación para una versión concreta del producto
`keys/`	`/var/lib/sapsan/`	directorio con el material de claves que genera Sapsan

Las variables de entorno controlan dónde busca Sapsan los archivos:

- Directorio de estado — `SAPSAN_STATE_DIR`; en su defecto, el directorio padre de `SERVER_ID_PATH`; en su defecto, el directorio de trabajo actual. El paquete deb y la imagen de Docker fijan `SERVER_ID_PATH=/var/lib/sapsan/server.id`, de modo que el directorio de estado es `/var/lib/sapsan`.
- Archivo de licencia — `LICENSE_PATH`; en su defecto, `{dirname(CONFIG_PATH)}/license.txt`; en su defecto, `{state_dir}/license.txt`. Con `CONFIG_PATH=/etc/sapsan/sapsan.yaml` esto es `/etc/sapsan/license.txt`.

Consulte Instalación para elegir el modo de licenciamiento durante el despliegue.

4.5 Estados

Estado	Cuándo	Qué funciona
`Licensed`	la licencia está verificada	todo
`Restricted`	la licencia falta, ha caducado o no superó la verificación	la interfaz de administración sigue disponible, el streaming queda limitado

© Flussonic 2026

El estado actual de la licencia lo entrega la API: `GET /streamer/api-v4/license/status`.

4.6 Qué viene después

- Admin API

© Flussonic 2026

5. Configuración de Sapsan

Sapsan y central forman un sistema inseparable y se entregan juntos. Sapsan se configura mediante la API de central.

Este manual de Sapsan puede presentar los parámetros de las funciones como fragmentos YAML. Estos fragmentos se pueden enviar mediante la API de central. Describen la configuración de Sapsan; consulte la documentación de la solución elegida para conocer el endpoint y el formato de las solicitudes a la API de central.

A continuación se describen la estructura de la configuración y las reglas para aplicarla.

Consulte la unión de streamers para añadir una máquina al clúster y conocer las ventanas de unión y la autenticación.

5.1 Estructura de la configuración

```
listeners:        # puertos en los que escucha el servidor
  http:
    - port: 80
  rtsp:
    - port: 554
  rtmp:
    - port: 1935
  webrtc:
    - port: 5005

streams:          # lista de streams
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://admin:password@10.0.0.5/stream0
    dvr: {}

dvr:              # almacenamiento compartido del archivo
  root: /storage
  disks:
  - path: d1

auth: {}            # autorización de reproducción y publicación
rproxy: {}          # pasarela Peeklio
api_auth:           # usuario/contraseña del Admin API
  login: admin
  password: secret
```

Sección	Qué configura	Detalles
`listeners`	puertos HTTP, RTSP, RTMP y WebRTC del servidor	esta página
`streams`	streams: fuentes, transcodificador, DVR, push, capturas de pantalla	captura
`dvr`	almacenamiento en disco del archivo	grabación DVR
`auth`	tokens y backends de autorización	autorización
`rproxy`	pasarela Peeklio para agentes	pasarela Peeklio
`api_auth`	acceso al Admin API	Admin API

5.2 Validación

Sapsan valida la configuración al cargarla y se niega a arrancar con una que no sea válida:

- los campos desconocidos son un error (protección contra erratas);
- cada entrada de un stream debe contener exactamente un protocolo;
- los puertos no pueden repetirse entre listeners y streams;
- si un stream tiene `dvr` habilitado, debe existir la sección global `dvr`.

© Flussonic 2026

5.3 Recarga en caliente

Los cambios de configuración se envían mediante la API de central. A nivel de proceso, Sapsan admite la recarga de `sapsan.yaml` con `SIGHUP` sin reiniciarse:

- los streams, la autorización y el DVR se reconfiguran sobre la marcha;
- los listeners solo se vuelven a enlazar cuando cambia un puerto;
- la pasarela rproxy conserva las conexiones de los agentes entre recargas (pero cambiar `streampoint_key` / `endpoint_auth_url` exige reiniciar el proceso).

© Flussonic 2026

© Flussonic 2026

6. Referencia de funciones

© Flussonic 2026

6.1 Referencia de funciones

© Flussonic 2026

- **Media**

© Flussonic 2026

- **Live**
  - **Inputs**
  - **Streams**
- **Ingest**
  - **RTSP**
  - **RTMP**
  - **SRT**
  - **RIST**
  - **HLS**
  - **DASH**
  - **TSHTTP**
  - **MPEGTS**
  - **DVB TS**
  - **WEBRTC**
  - **File**
  - **SDI**
  - **SDI 12g**
  - **NDI**
  - **ASI**
  - **ST2110**
  - **H323**
  - **Shoutcast**
  - **HDMI**
  - **DVB**
  - **M4f**
  - **M4s**
  - **MBR**
  - **Syntetic**
- **Publish**
  - **SRT**
  - **SRT Streamkey**
  - **RTMP**
  - **RTMP Streamkey**
  - **WHIP**
- **Mux**
  - **MPTS**
  - **Simulcrypt**
  - **PSI**
  - **PCR**
  - **Stuffing**
  - **EIT**
  - **Data**

© Flussonic 2026

- **Probe**
  - **Tr101290**
  - **SRT**
  - **RTSP**
  - **Content**
  - **Loudness**
  - **SDI**
  - **Counters**
  - **Stream Validator**
- **Playout**
  - **Playlist**
- **Source**
  - **Failover**
  - **Slate**
  - **Quality**
  - **Seamless**
  - **Realtime Aligner**
- **Process**
  - **OCR Subtitles**
  - **Speech To Text**
  - **Thumbnails Http**
  - **Thumbnails Keyframe**
- **Transcode**
  - **Video**
  - **VOD**
  - **Audio**
  - **MBR**
  - **CBR**
  - **HW**
  - **Deinterlace**
  - **Interlace**
  - **Logo**
  - **SAR**
- **Ads**
  - **Scte35**
  - **SSAI**
  - **Bypass**

© Flussonic 2026

- DVR
  - Record
  - Depth
  - Replication
  - Object Storage
  - Cache
  - Catchup
  - Export
  - Timelapse
  - Remote
- Play
  - Sessions
  - Ports
  - HLS
  - LLHLS
  - DASH
  - MSS
  - MPEGTS
  - MP4
  - RTSP
  - RTSP DVR
  - RTMP
  - SRT
  - WEBRTC
  - Moq
  - DVR
  - Startover
  - VOD
  - NDI
- Push
  - RTMP
  - SRT
  - RIST
  - UDP TS
  - UDP Standby
  - SDI
  - NDI
  - ASI
  - ST2110
  - QAM
  - Supervisor

© Flussonic 2026

- Access
  - Auth Backend
  - Token
  - Geo
  - DRM
  - Cas
- Episodes
  - Management
  - Ipcam
  - External
  - DVR

- Control
  - Plane
    - API
    - Telemetry
  - Observability
    - Logs
    - Traces
    - Metrics
    - Prometheus
    - Incidents
  - Audit
    - Log
    - Integrity
    - Retention
    - Export
  - Access
    - Roles
    - Directory
    - Tokens
    - Policy

- Cluster
  - Control
    - Management
    - Layouter
    - Nodes
  - Delivery
    - Zones

© Flussonic 2026

- **Device**
  - **ONVIF**
    - **Discovery**
    - **Events**
    - **Control**
    - **Compliance**
  - **Agent**
    - **Runtime**
    - **Firmware**
- **Gateway**
  - **Tunnel**
    - **Agents**
- **ACS**
  - **Access**
    - **Passes**
    - **Gates**
  - **Report**
    - **Passages**

6.1.1 Funciones fuera de la matriz tarifaria

- **Subtítulos ocultos**
- **Validación**
- **Recarga en caliente**
- **Compatibilidad con Flussonic**

© Flussonic 2026

6.2 Medios

6.2.1 Streams en directo (media.live)

Live stream

`media.live.streams`

Un **live stream** en Sapsan es un stream con nombre que reúne la captura de medios, su procesamiento y su entrega a los consumidores. Los medios llegan y se procesan a medida que aparecen nuevos fotogramas. El stream tiene una configuración y un estado actual. Bajo un mismo nombre puede haber:

- **Vídeo** — varias calidades.
- **Audio** — pistas en distintos idiomas.
- **Subtítulos** — pistas de texto.

Todos siguen siendo pistas del mismo stream.

En un servidor individual, el live stream gestiona el pipeline multimedia en ejecución. En central, el mismo stream se representa como un objeto del clúster: sus funciones pueden ejecutarse en varios servidores a la vez.

STREAM EN UN SERVIDOR

Una instancia local del stream es un **gestor del pipeline**. Conecta los componentes de procesamiento, les proporciona la configuración y gestiona su ciclo de vida. La composición del pipeline depende de la configuración y del papel de este servidor en el servicio del stream.

Componente	Función
Entrada	Recibe medios de una fuente o acepta una publicación. Gestiona la conexión, las reconexiones y la selección de la fuente de reserva.
Transcodificador	Decodifica y codifica los medios según la configuración del stream, generando los códecs y la escalera de calidades necesarios.
Procesamiento	Realiza el procesamiento configurado: extracción de subtítulos, generación de miniaturas y procesamiento de metadatos y eventos. Tareas como el reconocimiento de voz pueden trabajar sobre un archivo ya grabado.
Segmentador	Compone segmentos y segmentos parciales, y mantiene una ventana en directo para la entrega por segmentos.
Grabación DVR	Envía los segmentos preparados al subsistema de archivo y supervisa el estado de la grabación.
Pushes	Envían medios a los destinos configurados. Cada destino tiene su propio estado de conexión y reconexión.
Retransmisores y suscriptores	Reciben fotogramas o segmentos del stream para entregarlos a espectadores y otros servidores mediante los protocolos correspondientes.

Los datos se distribuyen por varias ramas: el mismo resultado del procesamiento puede enviarse simultáneamente al archivo, a los pushes y a la entrega a espectadores. Cada espectador adicional no requiere una captura y transcodificación independientes. La entrega por segmentos lee los segmentos preparados, mientras que los protocolos de transmisión continua reciben un flujo de fotogramas o paquetes de transporte.

El gestor acepta los cambios de configuración, actualiza los componentes y recopila su estado. La reconexión de la entrada, la recuperación de un push y el reinicio del transcodificador forman parte del ciclo de vida del pipeline local. Cambiar parámetros o recuperar un componente puede interrumpir la parte correspondiente del procesamiento.

© Flussonic 2026

Más información sobre los componentes:

- **Entradas y LSI**.
- **Reserva de fuentes**.
- **Transcodificación**.
- **Grabación del archivo**.
- **Gestión de pushes**.

STREAM DEL CLÚSTER EN CENTRAL

Central almacena la configuración general del stream: qué fuente capturar y qué funciones realizar. El operador la modifica mediante la API de central. El layouter determina qué streamers servirán el stream y con qué roles. Central genera y entrega a cada servidor la configuración de su parte del trabajo.

Así, un único stream del clúster tiene varias representaciones locales. El servidor de captura ejecuta el pipeline de captura y procesamiento. En el streamer DVR, el mismo stream recibe medios preparados para grabarlos. En un edge, atiende la entrega a espectadores y utiliza una caché cuando es necesario. Cada servidor realiza las funciones que le asigna su configuración.

Nivel	Qué describe
Stream del clúster	La configuración general y el nombre del stream, que se mantienen cuando cambia su ubicación.
Asignación	Qué streamer desempeña un rol concreto: captura, grabación del archivo o entrega.
Instancia local	El pipeline que se ejecuta en el servidor con la configuración que ha recibido.

La distribución permite utilizar varios servidores a la vez para distintas funciones de un mismo stream. Por ejemplo, la captura y la transcodificación pueden ejecutarse en un servidor con el hardware adecuado, el archivo puede grabarse en almacenamiento dedicado y la entrega puede atenderse desde edges cercanos a los espectadores. Los roles disponibles y sus reglas de asignación se describen en la ubicación de streams.

EJEMPLO DE DISTRIBUCIÓN

El stream `news` tiene activadas la transcodificación y la grabación del archivo. Utiliza los siguientes servidores:

- **Servidor A — captura:** recibe la fuente y genera la escalera de calidades.
- **Servidor B — grabación:** recibe los medios preparados de A y graba el archivo.
- **Servidores C y D — entrega:** reciben los medios en directo de A, leen el archivo de B y atienden a los espectadores. Con la caché del archivo activada, las solicitudes repetidas pueden atenderse localmente.

El operador gestiona un único stream, `news`. En los servidores se ejecutan distintas partes de su procesamiento y entrega. Añadir un edge no requiere crear otro stream gestionado por el operador con un nombre diferente.

CAMBIOS Y FALLOS

Cuando cambia la ubicación, central entrega configuraciones actualizadas a los streamers afectados. El nombre y la configuración general del stream del clúster se mantienen; pueden cambiar el conjunto de instancias locales y las funciones que realizan.

La recuperación local y la reasignación en el clúster resuelven tareas diferentes. El propio streamer reconecta su entrada o cambia a una fuente de reserva configurada. El layouter decide trasladar el trabajo a otro servidor según el estado del clúster y las reglas de ubicación.

Si se pierde la conexión con central, un streamer que ya está en ejecución sigue utilizando la última configuración aplicada. La entrega de nuevos ajustes y la reasignación requieren un plano de control operativo. Los estados de los componentes y servidores individuales deben considerarse en conjunto: que un edge entregue los medios correctamente no confirma por sí solo que la grabación del archivo continúe en el streamer DVR.

© Flussonic 2026

Fuentes de live stream y LSI

`media.live.inputs`

LSI (Live Source Input) gestiona las entradas de un live stream. Captura los medios de la fuente seleccionada, supervisa su estado y gestiona la recuperación cuando surgen problemas. Si el stream tiene entradas de reserva, LSI evalúa si son adecuadas y elige desde dónde continuar la captura.

Para emitir las 24 horas no basta con abrir una dirección de entrada una sola vez. Una fuente puede mantener la conexión, pero dejar de enviar vídeo. La ruta principal puede recuperarse solo durante unos segundos. La reserva puede no estar disponible justo cuando se necesita. LSI observa las fuentes durante toda la vida del stream y toma decisiones a partir de los datos multimedia.

La gestión de fuentes está integrada en el pipeline de Sapsan:

- **Supervisión de la captura** detecta la ausencia de fotogramas y los problemas de pistas individuales.
- **Evaluación de calidad** permite detectar daños en el flujo antes de que la captura se detenga por completo.
- **Comprobación de reservas** proporciona información sobre su funcionamiento antes de que falle la entrada principal.
- **Conmutación coordinada** prepara una nueva fuente y envía sus medios a las siguientes etapas del pipeline.
- **Observabilidad** muestra la entrada seleccionada y los motivos de las decisiones, y permite medir el tiempo de funcionamiento con entradas de reserva.

QUÉ SIGNIFICA QUE UNA FUENTE FUNCIONA

Una conexión de red establecida correctamente no significa todavía que la fuente esté lista para emitir. LSI tiene en cuenta la llegada de datos multimedia y la existencia de un punto desde el que iniciar su procesamiento. Para una entrada de vídeo basada en fotogramas, ese punto es un fotograma clave; para una fuente solo de audio, un fotograma de audio. Cuando la captura conserva fragmentos existentes, se utiliza el límite de un fragmento.

El estado de una fuente depende de las observaciones disponibles para su protocolo y de las reglas configuradas:

Observación	Qué permite detectar
Hora del último fotograma	La fuente ha dejado de enviar medios, aunque la conexión siga abierta.
Llegada de vídeo y audio	La desaparición de un componente del flujo mientras el otro sigue llegando.
Errores de continuidad MPEG-TS	Daños en el flujo de transporte, por ejemplo por pérdida de paquetes.
Tasa de bits	Una disminución sostenida por debajo del umbral configurado o la superación de este.
Configuración de códecs	Llegan fotogramas, pero faltan datos necesarios para decodificarlos.
Pérdidas en el búfer de recepción y descarte de pistas desincronizadas	Problemas de procesamiento del flujo de entrada en el servidor receptor.

LSI distingue entre **fallo** y **degradación de calidad**. Ante un fallo, no se puede continuar la captura desde esa entrada: por ejemplo, ha vencido el tiempo de espera por ausencia de fotogramas o el tiempo de espera de vídeo configurado. La entrada se cierra y comienza la búsqueda de una alternativa.

Con una degradación de calidad, los medios siguen llegando. Por ejemplo, desaparece el audio o el número de errores de continuidad supera el umbral permitido. LSI busca una fuente más adecuada mientras mantiene la captura actual. Si no hay una reserva en buen estado, sigue utilizando la fuente degradada: la parte de la emisión que aún se conserva sigue disponible.

Este control evalúa el estado técnico de los medios. Los fotogramas que llegan a tiempo con una imagen fija no implican por sí solos que haya vencido el tiempo de espera de vídeo. Las reglas de degradación se describen en el control de calidad de las fuentes.

© Flussonic 2026

CÓMO ELIGE LSI UNA FUENTE

Varias entradas en `inputs` describen fuentes alternativas de un mismo stream. Durante el funcionamiento normal, los medios de la entrada seleccionada se envían a las siguientes etapas del pipeline. De forma predeterminada, la prioridad sigue el orden de la lista; `source_options.priority` permite cambiarla explícitamente. Un número menor indica una prioridad mayor.

Situación	Acción de LSI
Inicio o fallo de la fuente actual	Se conecta a una candidata disponible, teniendo en cuenta su estado y los retrasos entre reintentos. Inicia la captura en un fotograma de inicio adecuado o en el límite de un fragmento.
La fuente actual está degradada y una reserva se ha comprobado recientemente y está en buen estado	Abre la reserva y conmuta cuando está lista. No requiere otra ventana completa de observación.
La fuente actual está degradada y no hay una confirmación reciente del buen estado de la reserva	Comprueba una candidata adecuada mientras mantiene la captura actual.
Se ha recuperado una fuente de mayor prioridad	La comprueba durante una ventana limpia y solo entonces devuelve la captura a esa fuente.
La fuente principal funciona y corresponde comprobar una reserva	Abre temporalmente la reserva, evalúa su calidad y la cierra, conservando el resultado de la comprobación.

Una **ventana limpia** es un intervalo sin infracciones de las reglas de calidad. Su duración se establece con `return_alive_secs`, con un valor predeterminado de 15 segundos. Para volver a una fuente de mayor prioridad, no basta con establecer la conexión: LSI debe observar una captura estable. Esto protege la emisión frente a conmutaciones repetidas a una fuente que se recupera solo brevemente.

La conmutación de emergencia tras el fallo de la entrada actual no espera esa ventana completa: la captura se reanuda en cuanto la candidata está lista. Si aún hay que abrir la reserva, el tiempo de recuperación incluye el establecimiento de la conexión y la espera de un punto de inicio adecuado.

Las conexiones fallidas se reintentan con un retraso creciente. Por tanto, una dirección no disponible no provoca un ciclo continuo de intentos inmediatos. Los parámetros detallados se describen en la redundancia de fuentes.

CÓMO SE COMPRUEBA LA RESERVA

LSI puede comprobar una reserva mientras la fuente principal sigue funcionando. La comprobación recibe datos multimedia reales y evalúa su calidad. El resultado incluye el estado de la fuente y la hora de observación; está disponible a través de la API junto con la información de las pistas.

La comprobación periódica se configura con `recheck_secondary_inputs_interval`: el valor predeterminado es 900 segundos, y `0` la desactiva. Es un intervalo de planificación de comprobaciones en segundo plano, no una garantía de abrir cada reserva exactamente una vez cada 900 segundos. La actividad actual de LSI y los retrasos de reconexión afectan al momento de la comprobación.

Una comprobación en segundo plano correcta termina con el cierre de la entrada de reserva. Por tanto, el estado `working` de una fuente no seleccionada indica que funcionaba correctamente **en la última observación**. La antigüedad del resultado se evalúa mediante `checked_at`. Esto no implica una captura paralela permanente de todas las reservas.

El método de comprobación depende de la entrada. Una fuente a la que Sapsan se conecta puede abrirse temporalmente. En una publicación, el emisor inicia la conexión: LSI observa la publicación entrante, pero no puede abrirla en nombre del emisor. La ventana limpia de retorno descrita se aplica a las entradas que pueden comprobarse; una publicación entrante aceptada puede seleccionarse ya en su primer fotograma de inicio adecuado.

CONMUTACIÓN Y CONTINUIDAD DE LA EMISIÓN

En una conmutación preparada, LSI mantiene la captura actual hasta que la sustituta está lista. Por ejemplo, el stream utiliza una entrada de reserva y la principal se ha recuperado:

1. LSI abre la entrada principal; los consumidores siguen recibiendo los medios de la reserva.
2. La principal supera la ventana limpia.
3. En un fotograma clave adecuado o en el límite de un fragmento, LSI selecciona la entrada principal.
4. El pipeline recibe una notificación del cambio de fuente y los nuevos parámetros multimedia antes de los fotogramas de la nueva entrada.

© Flussonic 2026

Así, la espera de la comprobación de la fuente principal no se convierte en una pausa de emisión. La conmutación se realiza dentro del mismo stream con nombre; el operador no necesita detenerlo y crearlo de nuevo.

El procesamiento del tiempo también tiene en cuenta el cambio de fuente. Para las entradas que necesitan alinearse con la hora del servidor, el componente correspondiente del pipeline establece de nuevo esa alineación. Los protocolos con una escala temporal de calendario fiable y los modos que conservan fragmentos siguen sus propias reglas de conservación de marcas de tiempo.

Estos mecanismos permiten organizar una conmutación sin interrupciones con fuentes preparadas y compatibles. El resultado para el espectador depende de toda la cadena:

- **Compatibilidad multimedia.** Alinee los códecs y la composición de las pistas. Compruebe también los parámetros de vídeo y audio: un cambio de formato puede requerir la reconfiguración del decodificador.
- **Sincronización del contenido.** Dos fuentes del mismo canal de televisión pueden tener retrasos distintos. Sin alinear el contenido, la conmutación puede repetir u omitir una parte del programa.
- **Disponibilidad del punto de inicio.** Un GOP largo aumenta la espera de un fotograma clave. Las entradas con fragmentos existentes conmutan en el límite de un fragmento.
- **Tipo de fallo.** Si la fuente actual desaparece de repente, la conmutación por sí sola no puede recuperar los fotogramas ya perdidos. El tiempo de detección y la disponibilidad de la reserva influyen en la pausa.
- **Trayecto de salida.** Pruebe la conmutación con los protocolos de entrega y reproductores reales y, si hay transcodificación, con la configuración de transcodificador elegida.

LSI no comprueba la equivalencia completa de los parámetros de codificación antes de cada conmutación. Tener dos direcciones en `inputs` no garantiza por sí solo una transición imperceptible. Al diseñar el sistema, prepare las fuentes y compruebe la reproducción final. Si ninguna entrada está disponible, la función independiente de cortinilla permite reproducir un archivo preparado.

EJEMPLO DE CONFIGURACIÓN

Sapsan recibe la configuración a través de la API de central. El siguiente fragmento YAML define dos fuentes UDP alternativas. Los ajustes de `lsi` establecen la política general del stream, mientras que `source_options` precisa los parámetros compatibles de una entrada individual.

```
streams:
  - name: tv1
    inputs:
      - udp:
          host: 239.0.0.1
          port: 5000
        source_options:
          priority: 1
      - udp:
          host: 239.0.0.2
          port: 5000
        source_options:
          priority: 2
    lsi:
      source_timeout: 5
      video_timeout: 3
      audio_timeout: 3
      cc_errors_limit: 10
      return_alive_secs: 15
      recheck_secondary_inputs_interval: 900
      reconnect_initial_delay: 1500
      reconnect_delay_step: 5000
      reconnect_max_delay: 60000
```

En este ejemplo, no recibir ningún fotograma durante 5 segundos o ningún vídeo durante 3 segundos constituye un fallo de la entrada. La ausencia de audio durante 3 segundos constituye una degradación de calidad. El límite de errores de continuidad es 10 por ventana de evaluación; superarlo degrada el estado de la fuente. Se requiere una ventana limpia de 15 segundos para comprobar el retorno a la principal.

Los tiempos de espera e intervalos de observación del ejemplo se indican en **segundos**. Los parámetros `reconnect_*` se indican en **milisegundos**: retraso inicial de 1500 ms, incremento de 5000 ms y límite de 60000 ms. Estos parámetros de reconexión se establecen en `lsi`.

Los valores de tiempo de espera son ejemplos. Ajústelos según el comportamiento de la fuente y la interrupción de emisión aceptable. De forma predeterminada, `source_timeout` es de 60 segundos y los tiempos de espera independientes de vídeo y audio están desactivados.

© Flussonic 2026

QUÉ SUPERVISAR

La API de central permite consultar el estado de las entradas del stream:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://central:8080/central/api-v4/streams/stats/tv1 \
  | jq '{name, inputs: [.inputs[] | {
      url, current, status,
      problems: (.problems // []),
      checked_at, error, failure_cause
    }]}'
```

El campo `current` identifica la entrada seleccionada, mientras que `status` describe su estado. Son características independientes: la entrada actual puede estar `degraded` y una reserva, `working`. Durante una búsqueda de emergencia, `current` todavía no confirma que se haya reanudado la captura: tenga también en cuenta `status` y la reproducción real.

- **`unchecked`** — todavía no hay un resultado de observación.
- **`starting`** — la entrada está abierta, pero aún no ha llegado un punto de inicio adecuado.
- **`working`** — la fuente funciona correctamente según las observaciones disponibles.
- **`degraded`** — llegan medios con infracciones de las reglas de calidad.
- **`dead`** — la fuente no está disponible o se ha declarado fallida.

`problems` informa de las infracciones detectadas, mientras que `error` y `failure_cause` ayudan a diagnosticar los errores. `checked_at` contiene la hora de observación en UTC, en formato RFC 3339; puede no estar presente en una fuente que aún no se haya comprobado. Solicitar estadísticas no inicia por sí solo una comprobación de la reserva.

Las siguientes señales son útiles para la supervisión operativa:

- **Degradación de la entrada actual** — `current: true` junto con `degraded` o un fallo.
- **Pérdida de la reserva** — una entrada de reserva ha fallado o lleva mucho tiempo sin una confirmación reciente de buen estado, aunque la entrega actual siga funcionando.
- **Conmutaciones frecuentes** — aumenta `input_switches`; compruebe la estabilidad de las fuentes y los ajustes de retorno.
- **Funcionamiento prolongado con la reserva** — aumenta `num_sec_on_secondary_input`; la recuperación de la principal requiere atención.
- **Ausencia de medios** — aumenta `num_sec_no_data`.

Los contadores detallados de `.input` están disponibles solicitando las estadísticas de un stream al **streamer que realiza la captura**: `/streamer/api-v4/streams/stats/tv1`. El estado compacto de central no sustituye esta sección. En la medición del tiempo en las entradas se proporciona un ejemplo de consulta y las reglas de cálculo de intervalos.

`input_switches` cuenta las transiciones entre distintas entradas configuradas; reconectarse a la misma entrada no constituye por sí solo una conmutación. Los contadores de tiempo distinguen la primera entrada de la lista de las demás, aunque se hayan cambiado las prioridades. Son valores acumulados: para evaluar un periodo, compare observaciones sucesivas y tenga en cuenta que las estadísticas se reinician al volver a arrancar.

CÓMO VERIFICAR LA REDUNDANCIA ANTES DE LA PUESTA EN SERVICIO

1. Confirme que se ha seleccionado la entrada principal esperada. Obtenga un resultado reciente de la comprobación de la reserva y compare `inputs[].media_info`: los parámetros de las entradas deben ajustarse al diseño del sistema.
2. Interrumpa la fuente principal de forma controlada. Compruebe el cambio de `current` y el aumento de `input_switches` mientras observa la reproducción real.
3. Restablezca la fuente principal. En las entradas a las que Sapsan se conecta, confirme que la reserva sigue funcionando durante la comprobación y que el retorno se produce después de la ventana limpia.
4. Pruebe por separado la degradación de calidad sin interrumpir la conexión, por ejemplo la pérdida de la pista de audio con `audio_timeout` activado.
5. Pruebe el fallo de la reserva mientras la principal funciona: la supervisión debe detectar la pérdida de redundancia antes de que falle la entrada principal.

© Flussonic 2026

QUÉ APORTA LSI A UN SERVICIO DE EMISIÓN

FFmpeg proporciona herramientas de procesamiento multimedia y puede restablecer conexiones; por ejemplo, HTTP dispone de opciones de reconexión. Al construir un servicio en torno a un comando de FFmpeg, la gestión del ciclo de vida de varias fuentes requiere una solución independiente.

En Sapsan, LSI reúne este trabajo. El operador define las fuentes y la política; el streamer supervisa la calidad y prepara las conmutaciones, mientras que la API proporciona el estado para la supervisión. Así, la redundancia se convierte en una parte gestionable de la emisión continua: se puede comprobar una ruta de reserva con antelación, explicar por qué cambió la fuente y confirmar que la principal se ha recuperado de forma estable.

© Flussonic 2026

6.2.2 Captura (media.ingest)

RTSP

`media.ingest.rtsp`

DESCRIPCIÓN

Sapsan toma el video de cámaras IP y de otras fuentes RTSP en modo cliente (pull).

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://admin:password@10.0.0.5:554/stream0
        peer_timeout_ms: 5000
```

PARÁMETROS

Parámetro	Descripción
`url`	dirección de la cámara; el usuario y la contraseña van en la URL
`peer_timeout_ms`	tiempo de espera de inactividad de la fuente, tras el cual la entrada se considera caída
`via_agent`	captura a través de un agente Peeklio (`agent://ID`) cuando la cámara está tras un NAT — vea la pasarela Peeklio

AUTENTICACIÓN

Se soportan Basic y Digest (con repliegue automático a Basic cuando la cámara rechaza Digest). Las credenciales se toman de la URL y no viajan en claro hacia la cámara cuando se usa Digest.

OPCIONES DE TRANSPORTE

RTP/AVP/TCP (interleaved): los medios viajan dentro de la conexión TCP RTSP.

CÓDECS ADMITIDOS

El SDP de la fuente identifica el códec, pero la captura también depende del empaquetado de los fotogramas en RTP. Los siguientes formatos disponen de procesamiento específico de la carga útil:

Códec	Nombre en SDP / RTP	Condiciones de captura
H.264	`H264`	Reconstrucción de NAL individuales, paquetes de agregación STAP-A y fragmentos FU-A
HEVC / H.265	`H265` , `HEVC`	Reconstrucción de NAL individuales y fragmentos FU; esta entrada no desempaqueta los paquetes de agregación AP
AV1	`AV1`	Reconstrucción de unidades temporales a partir de paquetes RTP; la configuración llega en la cabecera de secuencia del stream
AAC	`MPEG4-GENERIC`	Empaquetado AAC-hbr con cabeceras AU; parámetros del códec en `config` del SDP
AAC en LATM	`MP4A-LATM`	`cpresent=0` , con `StreamMuxConfig` en `config` del SDP; el empaquetado LATM se elimina durante la captura
G.711 A-law	`PCMA`	Incluido el tipo de carga útil RTP estático 8
G.711 μ-law	`PCMU`	Incluido el tipo de carga útil RTP estático 0

Aquí, `MPEG4-GENERIC` y `MP4A-LATM` designan formas de transportar AAC por RTP, no dos códecs de audio adicionales. Por eso no se incluye «MPEG-4 Audio» como entrada independiente junto a AAC. Los formatos se especifican en RFC 3640 y RFC 6416.

© Flussonic 2026

Las demás pistas reconocidas usan un modo limitado: las cargas útiles se concatenan hasta el bit marker de RTP, sin analizar las cabeceras RTP específicas del códec. Esto se aplica a los siguientes nombres SDP:

- **Opus:** `opus`.
- **JPEG / MJPEG:** `JPEG`.
- **VP9:** `VP9`.
- **PCM lineal:** `L16`.
- **MPEG Audio:** `MPA` y `MP3`.
- **Vídeo sin comprimir:** `raw`.

La compatibilidad depende del empaquetado de la fuente y del uso del bit marker. Reconocer un nombre SDP no garantiza la captura correcta de la paquetización RTP estándar de estos formatos. La entrada RTSP no reconoce VP8, por lo que no figura entre los códecs admitidos.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

El cambio de `url` o `via_agent` reinicia la fuente; el cambio de solo `peer_timeout_ms` se aplica en caliente.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Rtsp

Campo	Tipo
`bytes`	`counter`
`reads`	`counter`
`frames`	`counter`
`errors`	`counter`
`warnings`	`counter`
`broken_interleaved`	`counter`
`desync`	`counter`
`memory_bytes`	`gauge`

© Flussonic 2026

RtpChannel

Campo	Tipo
channel_id	label
rtp_packets	counter
rtcp_packets	counter
bytes	counter
frames	counter
pt_reject_count	counter
pt_reject_bytes	counter
ts_goes_backwards	counter
ts_jump_forward	counter
ts_stuck	counter
errors_dts_stuck	counter
sr_ts_stuck	counter
ts_stuck_restarts	counter
resync_sr	counter
marker_packets_count	counter
no_marker_mode_flag	gauge
errors_broken_payload	counter
errors_lost_packets	counter
errors_dropped_packets	counter
errors_late_packets	counter
errors_incomplete_frames	counter
nack_count	counter
onvif_buffer_overflow	counter
onvif_seq_gap	counter
onvif_marker_mismatch	counter
onvif_resync_dropped	counter

OTROS DETALLES

Verificación

Abra `http://server/streaming/v/cam1/index.m3u8` en un reproductor o solicite una captura de pantalla en `http://server/streaming/live-preview-jpeg/cam1`.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

• Diagnóstico RTSP

© Flussonic 2026

RTMP

`media.ingest.rtmp`

DESCRIPCIÓN

Sapsan puede tomar un stream RTMP de otro servidor en modo cliente (pull). Para aceptar publicaciones RTMP entrantes (push hacia Sapsan), vea publicación.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
  - name: relay
    inputs:
    - rtmp:
        url: rtmp://origin.example.com/live/streamkey
        peer_timeout_ms: 5000
```

PARÁMETROS

Parámetro	Descripción
`url`	dirección del stream RTMP remoto.
`peer_timeout_ms`	espera sin datos.

AUTENTICACIÓN

Indique las credenciales en la URL: `rtmp://user:password@origin.example.com/live/streamkey`. Se admite autenticación Adobe/FMS `authmod=adobe`.

OPCIONES DE TRANSPORTE

RTMP sobre TCP. Se admiten streams solo de audio.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC (Enhanced RTMP, `hvc1`)**
- **AV1 (Enhanced RTMP, `av01`)**
- **AAC**
- **MP3**

RTMP transporta mensajes FLV; Enhanced RTMP identifica los códecs de vídeo mediante FourCC. La lista describe la recepción de frames comprimidos y su envío al procesamiento. AV1 requiere Enhanced RTMP con `av01`; seleccionar AV1 en el emisor sin este empaquetado no basta. MP3 se acepta como audio FLV, por separado de AAC.

COMPORTAMIENTO ANTE ERRORES

Los códigos de rechazo del servidor se distinguen y son visibles en el estado de la entrada: stream no encontrado, acceso denegado, stream detenido, error de protocolo. También se distinguen una conexión rota y un servidor silencioso (`peer_timeout_ms`). Ante cualquiera de estas causas, el LSI conmuta a una entrada de reserva.

APLICACIÓN DE CAMBIOS

El cambio de `url` reinicia la fuente; el cambio de solo `peer_timeout_ms` se aplica en caliente.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

© Flussonic 2026

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Rtmp

Campo	Tipo
`bytes`	`counter`
`outgoing_bytes`	`counter`
`frames`	`counter`
`rtmp_script_data`	`counter`
`memory_bytes`	`gauge`

OTROS DETALLES

Los metadatos `onMetaData` acompañan al stream. Los metadatos opcionales dañados no detienen la captura de vídeo y audio.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

SRT

`media.ingest.srt`

DESCRIPCIÓN

Sapsan se conecta a un servidor SRT remoto y captura MPEG-TS en modo caller. La entrada abre un puerto UDP local efímero, independiente de `listeners` y de los puertos de publicación. `host` y `port` identifican el servidor remoto. Véase Publicación SRT para conexiones entrantes.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - srt:
      host: origin.example.com
      port: 9000
      stream_id: '#!::r=studio,m=request'
      latency_ms: 120
      peer_timeout_ms: 5000
```

PARÁMETROS

Parámetro	Descripción
`host`	IP o DNS del servidor remoto
`port`	su puerto UDP.
`stream_id`	`stream_id` se envía al conectar.
`passphrase`	contraseña de cifrado acordada.
`latency_ms`	latencia de recepción y presupuesto de recuperación.
`handshake_timeout_ms`	tiempo de conexión
`peer_timeout_ms`	espera sin datos.

AUTENTICACIÓN

`passphrase` debe coincidir con el emisor. El servidor remoto define el formato de `stream_id`, incluido el token de acceso si lo requiere.

OPCIONES DE TRANSPORTE

- **UDP** — conexión cliente al listener SRT remoto.
- **ACK/ACKACK** — confirmación de entrega.
- **NAK/ARQ** — solicitudes de retransmisión y recuperación.
- **Búfer de latencia** — ordenación de paquetes.
- **Too-late drop** — descarte de paquetes tardíos.
- **Keepalive** — supervisión de la conexión.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**

© Flussonic 2026

- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

El cambio de dirección, puerto, `passphrase` o `stream_id` reinicia la fuente; el cambio de solo los tiempos de espera se aplica en caliente.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

© Flussonic 2026

Srt

Campo	Tipo
incoming_bytes	counter
outgoing_bytes	counter
incoming_packets	counter
outgoing_packets	counter
incoming_lost_packets	counter
outgoing_lost_packets	counter
incoming_retransmitted_packets	counter
outgoing_retransmitted_packets	counter
incoming_recovered_packets	counter
missing_packets	counter
incoming_ack_packets	counter
outgoing_ack_packets	counter
incoming_ackack_packets	counter
outgoing_ackack_packets	counter
incoming_drop_request	counter
incoming_drop_requests	counter
outgoing_drop_requests	counter
incoming_nak_packets	counter
outgoing_nak_packets	counter
outgoing_drop_too_late_packets	counter
incoming_drop_late_seq	counter
incoming_drop_late_time	counter
incoming_drop_far_time	counter
incoming_drop_too_late	counter
incoming_drop_buffer_full	counter
loss_tracking_overflows	counter
keepalive_timeouts	counter
receive_buffer_size	gauge
send_buffer_size	gauge
rtt_us	gauge
rtt_var_us	gauge
source_clock_drift_us	gauge
memory_bytes	gauge

© Flussonic 2026

Mpegts

Campo	Tipo
`bytes`	`counter`
`sync_bytes`	`counter`
`packets`	`counter`
`frames`	`counter`
`errors`	`counter`
`errors_ts_pat`	`counter`
`errors_ts_service_lost`	`counter`
`broken`	`counter`
`memory_bytes`	`gauge`

© Flussonic 2026

MpegtsPid

© Flussonic 2026

Campo	Tipo
pid	label
pnr	label
packets	counter
frames	counter
empty_packets	counter
errors_adaptation_broken	counter
errors_ts_scrambled	counter
errors_ts_pmt	counter
errors_ts_cc	counter
errors_ts_tei	counter
errors_ts_psi_checksum	counter
errors_pid_lost	counter
broken_pes_count	counter
broken_pes_sum	counter
time_corrections	counter
corrected_backward_pts	counter
discarded_buffer_count	counter
discarded_buffer_sum	counter
fillers_count	counter
fillers_sum	counter
unpaired_fields	counter
warning_invalid_sibling_frames	counter
padding_pes_count	counter
padding_pes_sum	counter
orphan_pes_count	counter
dts_goes_backwards	counter
dts_jump_forward	counter
too_large_dts_jump	counter
pcr_resync	counter
pcr_jitter_ok_0	counter
pcr_jitter_ok_500ns	counter
pcr_jitter_fail_500us	counter
pcr_jitter_fail	counter
hrd_buffer_min	gauge

© Flussonic 2026

Campo	Tipo
`hrd_buffer_max`	`gauge`
`hrd_buffer_underrun`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`

OTROS DETALLES

Además del vídeo y el audio, MPEG-TS transporta:

- **Teletexto DVB**.
- **Metadatos KLV**.
- **Marcadores SCTE-35**.
- **Descriptores de idioma y subtítulos DVB** de la PMT.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo
- Diagnóstico SRT

© Flussonic 2026

RIST

`media.ingest.rist`

DESCRIPCIÓN

Capture MPEG-TS por RIST con recuperación de pérdidas mediante retransmisión. Se admiten Simple Profile y Main Profile.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - rist:
      host: 0.0.0.0
      port: 5000
      profile: simple
      buffer_ms: 1000
```

PARÁMETROS

Parámetro	Descripción
`host`	dirección local o grupo multicast
`port`	puerto par de medios Simple Profile.
`buffer_ms`, `reorder_ms`, `max_retries`	`buffer_ms` (1000), `reorder_ms` (70), `max_retries` (7) controlan recuperación.
`peer_timeout_ms`	espera de la fuente.
`multicast_interface`	interfaz del grupo.
`profile`	`simple` o `main`.
`tunnel_port`, `tunnel_role`, `encryption_passphrase`, `allow_insecure_encryption`	Main usa `tunnel_port`, `tunnel_role` (`client` / `server`, entrada `server` por defecto), `encryption_passphrase`, `allow_insecure_encryption` (`false` por defecto).

AUTENTICACIÓN

Main Profile admite PSK compartida mediante `encryption_passphrase`. Simple Profile no proporciona cifrado.

OPCIONES DE TRANSPORTE

- **Simple Profile** — RTP en puerto par P, RTCP en P+1.
- **Main Profile** — túnel GRE en `tunnel_port`.
- **ARQ** — retransmisión dentro del presupuesto del búfer.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**

© Flussonic 2026

- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

© Flussonic 2026

Rist

© Flussonic 2026

Campo	Tipo
`incoming_packets`	`counter`
`incoming_lost_packets`	`counter`
`incoming_recovered_packets`	`counter`
`incoming_unrecovered_packets`	`counter`
`incoming_duplicate_packets`	`counter`
`incoming_retransmitted_packets`	`counter`
`incoming_reordered_packets`	`counter`
`incoming_late_packets`	`counter`
`incoming_invalid_datagrams`	`counter`
`incoming_foreign_ssrc_packets`	`counter`
`incoming_foreign_profile_packets`	`counter`
`incoming_undecryptable_packets`	`counter`
`incoming_encryption_mismatch_packets`	`counter`
`source_restarts`	`counter`
`outgoing_nak_packets`	`counter`
`outgoing_packets`	`counter`
`outgoing_bytes`	`counter`
`outgoing_retransmitted_packets`	`counter`
`incoming_nak_packets`	`counter`
`retransmits_denied_cache`	`counter`
`retransmits_denied_rate`	`counter`
`cache_evicted_by_ceiling`	`counter`
`tunnel_keepalives_sent`	`counter`
`tunnel_keepalives_received`	`counter`
`tunnel_discarded_unknown_version`	`counter`
`tunnel_discarded_bad_encapsulation`	`counter`
`tunnel_discarded_unroutable`	`counter`
`tunnel_discarded_queue_full`	`counter`
`tunnel_discarded_foreign_profile`	`counter`
`rtt_us`	`gauge`
`tunnel_up`	`gauge`
`tunnel_silence_ms`	`gauge`
`tunnel_streams`	`gauge`
`tunnel_insecure_iv_allowed`	`gauge`

© Flussonic 2026

Campo	Tipo
`memory_bytes`	`gauge`

Mpegts

Campo	Tipo
`bytes`	`counter`
`sync_bytes`	`counter`
`packets`	`counter`
`frames`	`counter`
`errors`	`counter`
`errors_ts_pat`	`counter`
`errors_ts_service_lost`	`counter`
`broken`	`counter`
`memory_bytes`	`gauge`

© Flussonic 2026

MpegtsPid

© Flussonic 2026

Campo	Tipo
pid	label
pnr	label
packets	counter
frames	counter
empty_packets	counter
errors_adaptation_broken	counter
errors_ts_scrambled	counter
errors_ts_pmt	counter
errors_ts_cc	counter
errors_ts_tei	counter
errors_ts_psi_checksum	counter
errors_pid_lost	counter
broken_pes_count	counter
broken_pes_sum	counter
time_corrections	counter
corrected_backward_pts	counter
discarded_buffer_count	counter
discarded_buffer_sum	counter
fillers_count	counter
fillers_sum	counter
unpaired_fields	counter
warning_invalid_sibling_frames	counter
padding_pes_count	counter
padding_pes_sum	counter
orphan_pes_count	counter
dts_goes_backwards	counter
dts_jump_forward	counter
too_large_dts_jump	counter
pcr_resync	counter
pcr_jitter_ok_0	counter
pcr_jitter_ok_500ns	counter
pcr_jitter_fail_500us	counter
pcr_jitter_fail	counter
hrd_buffer_min	gauge

© Flussonic 2026

Campo	Tipo
`hrd_buffer_max`	`gauge`
`hrd_buffer_underrun`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`

OTROS DETALLES

Además del vídeo y el audio, MPEG-TS transporta:

- **Teletexto DVB**.
- **Metadatos KLV**.
- **Marcadores SCTE-35**.
- **Descriptores de idioma y subtítulos DVB** de la PMT.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

HLS

`media.ingest.hls`

DESCRIPCIÓN

Sapsan sigue una playlist HLS ajena en modo cliente (pull): sondea la playlist, descarga los segmentos y entrega fotogramas al pipeline igual que RTSP, SRT o MPEG-TS. Así entrega el stream casi todo agregador, toda CDN ajena y todo socio al que no le han permitido abrir UDP o SRT.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
  - name: chan1
    inputs:
    - hls:
        url: https://cdn.example.com/live/index.m3u8
```

PARÁMETROS

Parámetro	Por defecto	Descripción
`url`	—	dirección de la playlist exactamente tal como la escribió el operador
`headers`	ninguno	cabeceras añadidas a cada petición a la fuente: playlist, segmentos, claves
`variant_mode`	`single`	`single` — una variante del master, `ladder` — toda la escalera
`variant_bandwidth`	ninguno	con qué bitrate casar la variante (el más cercano por debajo); sin él gana el mayor `BANDWIDTH`
`skew_threshold_ms`	`500`	el desfase a partir del cual la escalera se declara desincronizada
`skew_policy`	`report`	qué hacer ante un veredicto `skewed`: `report` — continuar y mostrarlo, `fallback_to_single` — retroceder a una sola variante
`connect_timeout_ms`	`5000`	tiempo de espera del establecimiento de la conexión
`read_timeout_ms`	`15000`	tiempo de espera del cuerpo: playlist, segmento, clave
`max_body_bytes`	64 MiB	límite de tamaño de una única respuesta
`peer_timeout_ms`	global	cuánto esperar fotogramas antes de dar la fuente por perdida

AUTENTICACIÓN

Dos formas, y se pueden combinar:

- **cabeceras** — `headers`, por ejemplo `Authorization: Bearer <token>`; van con la playlist, las variantes, los segmentos init, los segmentos y las claves del mismo origin, y nunca siguen una redirección a un origin ajeno;
- **credenciales en la propia URL** — autenticación Basic corriente hecha por el cliente HTTP.

Los valores de las cabeceras jamás llegan al log: solo se imprimen sus nombres. La dirección de la entrada se escribe entera en los logs y la API de gestión la devuelve tal cual: qué parte de esa dirección es el secreto lo sabe solo el operador, y enmascarar a ciegas o dejaría el token en la query o recortaría justo aquello por lo que la entrada se reconoce en el log.

OPCIONES DE TRANSPORTE

HTTP o HTTPS; listas/manifiestos y segmentos usan solicitudes independientes.

CÓDECS ADMITIDOS

Los códecs admitidos dependen del contenedor de los segmentos. MPEG-TS y fMP4 son contenedores; HLS obtiene playlists y segmentos.

© Flussonic 2026

Segmentos MPEG-TS:

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

Segmentos fMP4:

- **H.264**
- **HEVC / H.265**
- **AV1**
- **MPEG-2 Video**
- **JPEG**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **Opus**
- **G.711 A-law / μ-law**

Son los códecs que la entrada extrae de fMP4, no una lista de capacidades de decodificación. H.264 usa `avc1` / `avc3`, HEVC usa `hvc1` / `hev1` y AV1 usa `av01` con `av1C`. MPEG-2 Video requiere `mp4v` con configuración MPEG-2; esto no implica compatibilidad con MPEG-4 Visual. JPEG es una pista de imágenes. MPEG Audio se identifica mediante `esds` o una sample entry MP3; `mp4a` por sí sola no significa AAC. Opus usa `Opus` / `dOps` y G.711 usa `alaw` / `ulaw`. Las declaraciones del contenedor deben coincidir con los datos; declarar un códec en el MPD no sustituye un segmento de inicialización válido.

Esta entrada no acepta packed audio (ADTS independiente o ID3+AAC sin MPEG-TS/fMP4).

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Solo un cambio de `url`, `headers` o de los ajustes HTTP (`connect_timeout_ms`, `read_timeout_ms`, `max_body_bytes`) reinicia la entrada: una fuente nueva tiene su propia numeración de pistas, su propio init y su propia posición en la playlist. Todo lo demás — modo de variante, umbral y política de desfase, `peer_timeout_ms` — se aplica en caliente, sin interrumpir la emisión.

© Flussonic 2026

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Hls

Campo	Tipo
`segments`	`counter`
`bytes`	`counter`
`frames`	`counter`
`playlist_reloads`	`counter`
`discontinuities`	`counter`
`segments_skipped`	`counter`
`segments_gone`	`counter`
`segments_cancelled`	`counter`
`segments_gap`	`counter`
`errors_playlist`	`counter`
`errors_segment_network`	`counter`
`errors_segment_status`	`counter`
`errors_segment_decode`	`counter`
`errors_decrypt`	`counter`
`fragment_gaps`	`counter`
`fragment_overlaps`	`counter`
`fragment_seam_error_ms`	`counter`
`edge_distance_ms`	`counter`
`playlist_updated_at`	`gauge`
`frames_emitted_at`	`gauge`
`memory_bytes`	`gauge`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`

OTROS DETALLES

Regla de inicio

Todo lo que ya había en la playlist en el momento de conectar es historia. La captura empieza con el primer segmento que aparece después de la conexión, y en una reconexión se comporta igual.

© Flussonic 2026

No existe a propósito un ajuste de profundidad de inicio: de lo contrario, cada caída de conexión haría que el stream vaciase por el pipeline, en segundos, la ventana acumulada, en lugar de emitir.

Escalera de calidades y desfase de representaciones

Con `variant_mode: ladder` se capturan todas las variantes del master. La de mayor `BANDWIDTH` pasa a ser la referencia, y las demás se comparan con ella segmento a segmento, por número de secuencia de media. El veredicto queda expuesto en la estadística de la entrada, campo `hls_ladder`:

- **`synced`** — las representaciones van juntas, conmutar entre ellas es seguro;
- **`skewed`** — el desfase superó el umbral o la estructura está rota: la conmutación dará tirones o desincronizará;
- **`unmeasurable`** — no hay con qué comparar, ningún par de segmentos comparte número.

El desfase se mide de dos maneras a la vez. El **declarado** es la diferencia de `EXT-X-PROGRAM-DATE-TIME` en el marcado; el **real** es la diferencia del tiempo de media de los primeros fotogramas. Si los dos no coinciden, lo enfermo es el marcado y no el stream, y quien lo arregla es el dueño de la fuente.

> **Note**
>
> Sapsan no lleva las variantes de la escalera a una escala de tiempos común, y no repara en absoluto el contenido defectuoso: una reparación silenciosa no hace conmutable la escalera, solo esconde el defecto. Por eso el desfase se muestra como número y decide el operador.

La política `fallback_to_single` baja la captura a una sola variante ante un veredicto `skewed` — para algunas fuentes es el único modo viable. El retroceso dura hasta que la entrada se reinicia.

Cifrado

Se admiten `AES-128` (el segmento entero) y `SAMPLE-AES` (por fotograma, para H.264 y AAC). La clave la recoge el mismo cliente y con las mismas cabeceras que todo lo demás, y se guarda en caché por dirección, de modo que la rotación de claves en medio de una playlist funciona sola: una clave nueva tiene otra dirección.

Un servidor de claves que calla es un error de entrada con la dirección de la clave y el código de respuesta; el cuerpo del segmento no se descarga en absoluto, pues no habría con qué descifrarlo.

CENC/DRM (Widevine, PlayReady, FairPlay) no está admitido y se rechaza por el nombre de `KEYFORMAT`.

© Flussonic 2026

Qué se admite

Posibilidad	Estado
Playlist de media: `EXT-X-MEDIA-SEQUENCE`, `EXTINF`, `EXT-X-ENDLIST`	sí
Playlist master: selección de variante, representaciones `EXT-X-MEDIA`	sí
Segmentos fMP4 (`EXT-X-MAP`) y MPEG-TS	sí
`EXT-X-BYTERANGE`	sí
`EXT-X-DISCONTINUITY`, recreación de la playlist, retraso respecto a la ventana	sí
`EXT-X-GAP`	sí, se salta sin error
`EXT-X-PROGRAM-DATE-TIME` — anclaje del tiempo de media a UTC	sí
Cifrado `AES-128` y `SAMPLE-AES` (H.264, AAC)	sí
Escalera: captura de todas las variantes, medición del desfase	sí
LL-HLS: `EXT-X-PART`, blocking reload	no, las partes se ignoran
CENC/DRM	no
Audio «empaquetado»: ADTS puro o ID3+AAC en lugar de un contenedor	no, el formato se nombra en el error
Subtítulos TTML y segmentos WebVTT	no, la pista se reconoce y se salta
Captura MPEG-DASH	no

Comprobación

Abra `http://server/streaming/v/chan1/index.m3u8` en un reproductor o pida una captura en `http://server/streaming/live-preview-jpeg/chan1`.

Más allá de «ya fluyen fotogramas», mire tres cosas:

- **cuánto se retrasa la salida respecto al borde de la playlist** — sin eso, la queja «el stream va con retraso» es indistinguible de «la fuente va con retraso», y las dos se arreglan en sitios distintos;
- **los segmentos saltados** y su desglose: lavados fuera de la ventana, cancelados en vuelo, declarados con `EXT-X-GAP`. Lo último no es un error de la fuente, sino su declaración honesta;
- **la costura entre fragmentos vecinos** — un defecto silencioso: todos los segmentos se descargaron, sin errores de análisis, y sin embargo el medio de dentro no se une, y el archivo conserva el agujero.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

DASH

`media.ingest.dash`

DESCRIPCIÓN

Sapsan captura MPEG-DASH en vivo de un servidor remoto, actualizando el MPD y descargando nuevos segmentos fMP4. Admite una representación o una escalera de calidades.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - dash:
      url: https://origin.example.com/studio/manifest.mpd
      representation_mode: single
```

PARÁMETROS

Parámetro	Descripción
`url`	URL del manifiesto MPD.
`headers`	Cabeceras HTTP de las solicitudes a la fuente.
`representation_bandwidth`	Bitrate objetivo en bit/s: selecciona la representación más cercana por debajo; sin valor, la de mayor `bandwidth`.
`representation_mode`	`single` — una representación (por defecto); `ladder` — la escalera completa de calidades.
`skew_threshold_ms`	Desfase permitido entre calidades, en milisegundos; por defecto 500.
`skew_policy`	`report` — continuar la captura e informar del desfase (por defecto); `fallback_to_single` — cambiar a una representación.
`connect_timeout_ms`	Tiempo de conexión HTTP, en ms; por defecto 5000.
`read_timeout_ms`	Tiempo de recepción del cuerpo HTTP, en ms; por defecto 15000.
`max_body_bytes`	Tamaño máximo del cuerpo, en bytes; por defecto 67 108 864 (64 MiB).
`peer_timeout_ms`	Tiempo sin fotogramas, en ms; utiliza el control común del estado de la fuente.
`clear_keys`	Mapa KID → clave CENC en hexadecimal; KID también admite notación UUID.

AUTENTICACIÓN

`headers` proporciona cabeceras de autorización HTTP. `clear_keys` proporciona pares KID/clave para CENC. Una fuente cifrada sin clave se rechaza.

OPCIONES DE TRANSPORTE

HTTP o HTTPS; listas/manifiestos y segmentos usan solicitudes independientes.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **MPEG-2 Video**
- **JPEG**
- **AAC**
- **AC-3**

© Flussonic 2026

- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **Opus**
- **G.711 A-law / μ-law**

Son los códecs que la entrada extrae de fMP4, no una lista de capacidades de decodificación. H.264 usa `avc1` / `avc3`, HEVC usa `hvc1` / `hev1` y AV1 usa `av01` con `av1C`. MPEG-2 Video requiere `mp4v` con configuración MPEG-2; esto no implica compatibilidad con MPEG-4 Visual. JPEG es una pista de imágenes. MPEG Audio se identifica mediante `esds` o una sample entry MP3; `mp4a` por sí sola no significa AAC. Opus usa `Opus` / `dOps` y G.711 usa `alaw` / `ulaw`. Las declaraciones del contenedor deben coincidir con los datos; declarar un códec en el MPD no sustituye un segmento de inicialización válido.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

© Flussonic 2026

Dash

Campo	Tipo
`segments`	`counter`
`bytes`	`counter`
`frames`	`counter`
`manifest_reloads`	`counter`
`discontinuities`	`counter`
`segments_skipped`	`counter`
`segments_gone`	`counter`
`segments_cancelled`	`counter`
`segments_gap`	`counter`
`errors_manifest`	`counter`
`errors_segment_network`	`counter`
`errors_segment_status`	`counter`
`errors_segment_decode`	`counter`
`errors_decrypt`	`counter`
`fragment_gaps`	`counter`
`fragment_overlaps`	`counter`
`fragment_seam_error_ms`	`counter`
`media_time_drifts`	`counter`
`edge_distance_ms`	`counter`
`manifest_updated_at`	`gauge`
`frames_emitted_at`	`gauge`
`memory_bytes`	`gauge`

OTROS DETALLES

La captura comienza en el borde en vivo. Se admiten MPD dinámicos y fMP4 con `SegmentTemplate`. No se aceptan MPD estáticos, `SegmentList` ni `SegmentBase`. Se controla la alineación de la escalera; la entrada no solicita licencias DRM a servidores Widevine/PlayReady/FairPlay.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

TSHTTP

`media.ingest.tshttp`

DESCRIPCIÓN

Sapsan se conecta al servidor HTTP y recibe MPEG-TS continuo del cuerpo de la respuesta. Véase MPEG-TS para UDP y multicast.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - tshttp:
      url: https://origin.example.com/studio/mpegts
      peer_timeout_ms: 5000
```

PARÁMETROS

Parámetro	Descripción
`url`	dirección de la fuente HTTP
`peer_timeout_ms`	espera sin datos.

OPCIONES DE TRANSPORTE

HTTP o HTTPS, incluida transferencia chunked. Esta entrada no requiere un puerto UDP de escucha.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Cambiar `url` reinicia la entrada; cambiar solo `peer_timeout_ms` se aplica sin desconexión.

© Flussonic 2026

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Mpegts

Campo	Tipo
`bytes`	`counter`
`sync_bytes`	`counter`
`packets`	`counter`
`frames`	`counter`
`errors`	`counter`
`errors_ts_pat`	`counter`
`errors_ts_service_lost`	`counter`
`broken`	`counter`
`memory_bytes`	`gauge`

© Flussonic 2026

MpegtsPid

© Flussonic 2026

Campo	Tipo
pid	label
pnr	label
packets	counter
frames	counter
empty_packets	counter
errors_adaptation_broken	counter
errors_ts_scrambled	counter
errors_ts_pmt	counter
errors_ts_cc	counter
errors_ts_tei	counter
errors_ts_psi_checksum	counter
errors_pid_lost	counter
broken_pes_count	counter
broken_pes_sum	counter
time_corrections	counter
corrected_backward_pts	counter
discarded_buffer_count	counter
discarded_buffer_sum	counter
fillers_count	counter
fillers_sum	counter
unpaired_fields	counter
warning_invalid_sibling_frames	counter
padding_pes_count	counter
padding_pes_sum	counter
orphan_pes_count	counter
dts_goes_backwards	counter
dts_jump_forward	counter
too_large_dts_jump	counter
pcr_resync	counter
pcr_jitter_ok_0	counter
pcr_jitter_ok_500ns	counter
pcr_jitter_fail_500us	counter
pcr_jitter_fail	counter
hrd_buffer_min	gauge

© Flussonic 2026

Campo	Tipo
`hrd_buffer_max`	`gauge`
`hrd_buffer_underrun`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`

OTROS DETALLES

Además del vídeo y el audio, MPEG-TS transporta:

- **Teletexto DVB**.
- **Metadatos KLV**.
- **Marcadores SCTE-35**.
- **Descriptores de idioma y subtítulos DVB** de la PMT.

QUÉ SIGUE

- Alineación con el tiempo real
- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

MPEG-TS

`media.ingest.mpegts`

DESCRIPCIÓN

Capture MPEG-TS por UDP unicast o multicast. Seleccione un programa MPTS con `program`. Véase TSHTTP para MPEG-TS por HTTP.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - udp:
      host: 239.0.0.1
      port: 5000
      program: 101
      peer_timeout_ms: 5000
```

PARÁMETROS

Campo	Descripción
`host`, `port`	dirección y puerto de recepción; use el grupo para multicast.
`program`	programa; si se omite, se toma el primero con medios.
`source`	emisor SSM IPv4.
`interface`	interfaz multicast.
`recv_buffer_bytes`	búfer (4 MiB por defecto).
`peer_timeout_ms`	espera sin datos.

OPCIONES DE TRANSPORTE

- **UDP unicast**.
- **UDP multicast IPv4/IPv6** — suscripción IGMP/MLD.
- **SSM IPv4** — filtro del emisor con `source`.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

© Flussonic 2026

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Cambiar dirección o programa reconecta la entrada. `recv_buffer_bytes` actualiza el socket UDP compartido en vivo; prevalece el último valor explícito de los streams que lo usan. `peer_timeout_ms` cambia sin desconectar.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Mpegts

Campo	Tipo
`bytes`	`counter`
`sync_bytes`	`counter`
`packets`	`counter`
`frames`	`counter`
`errors`	`counter`
`errors_ts_pat`	`counter`
`errors_ts_service_lost`	`counter`
`broken`	`counter`
`low_recv_buffer_secs`	`counter`
`recv_buffer_granted_bytes`	`gauge`
`recv_buffer_requested_bytes`	`gauge`
`recv_buffer_required_bytes`	`gauge`
`recv_buffer_drops`	`counter`
`memory_bytes`	`gauge`

© Flussonic 2026

MpegtsPid

© Flussonic 2026

Campo	Tipo
`pid`	`label`
`pnr`	`label`
`packets`	`counter`
`frames`	`counter`
`empty_packets`	`counter`
`errors_adaptation_broken`	`counter`
`errors_ts_scrambled`	`counter`
`errors_ts_pmt`	`counter`
`errors_ts_cc`	`counter`
`errors_ts_tei`	`counter`
`errors_ts_psi_checksum`	`counter`
`errors_pid_lost`	`counter`
`broken_pes_count`	`counter`
`broken_pes_sum`	`counter`
`time_corrections`	`counter`
`corrected_backward_pts`	`counter`
`discarded_buffer_count`	`counter`
`discarded_buffer_sum`	`counter`
`fillers_count`	`counter`
`fillers_sum`	`counter`
`unpaired_fields`	`counter`
`warning_invalid_sibling_frames`	`counter`
`padding_pes_count`	`counter`
`padding_pes_sum`	`counter`
`orphan_pes_count`	`counter`
`dts_goes_backwards`	`counter`
`dts_jump_forward`	`counter`
`too_large_dts_jump`	`counter`
`pcr_resync`	`counter`
`pcr_jitter_ok_0`	`counter`
`pcr_jitter_ok_500ns`	`counter`
`pcr_jitter_fail_500us`	`counter`
`pcr_jitter_fail`	`counter`
`hrd_buffer_min`	`gauge`

© Flussonic 2026

Campo	Tipo
`hrd_buffer_max`	`gauge`
`hrd_buffer_underrun`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`

OTROS DETALLES

Además del vídeo y el audio, MPEG-TS transporta:

- **Teletexto DVB**.
- **Metadatos KLV**.
- **Marcadores SCTE-35**.
- **Descriptores de idioma y subtítulos DVB** de la PMT.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

WEBRTC

`media.ingest.webrtc`

DESCRIPCIÓN

`media.ingest.webrtc` describe captura WebRTC de una fuente remota iniciada por el servidor. La publicación entrante mediante WHIP es la función media.publish.whip. La API actual de Sapsan no expone una entrada cliente WebRTC; `publish` no configura este modo de captura.

CONFIGURACIÓN

El esquema actual de Sapsan no contiene una entrada cliente WebRTC; no hay una configuración YAML definida.

PARÁMETROS

La API no publica campos de dirección remota ni negociación para esta entrada.

AUTENTICACIÓN

No se definen parámetros de autenticación para entrada cliente WebRTC. `publish_tokens` controla publicación entrante, no acceso a la fuente remota.

OPCIONES DE TRANSPORTE

WebRTC utiliza negociación y transporte de medios protegido. WHIP y WHEP del servidor cubren publicación y reproducción; no configuran una entrada cliente.

CÓDECS ADMITIDOS

Sapsan no define una lista de códecs de captura para esta entrada.

COMPORTAMIENTO ANTE ERRORES

Los fallos de sesión WHIP corresponden a publicación. La API actual no tiene un estado de entrada cliente WebRTC.

APLICACIÓN DE CAMBIOS

No se define una operación de la API de central para modificar entrada cliente WebRTC.

CONTADORES

La implementación actual no tiene contadores de entrada cliente WebRTC. Las estadísticas de publicación WHIP no describen esta entrada.

OTROS DETALLES

Un reproductor WebRTC o receptor WHIP no implica una entrada cliente WebRTC.

QUÉ SIGUE

- Referencia de funciones
- Publicación WHIP

© Flussonic 2026

FILE

`media.ingest.file`

DESCRIPCIÓN

La entrada `file` convierte un archivo MP4 o Matroska local en un stream en vivo pleno: el archivo se reproduce en bucle. Es una herramienta de producción para canales de cortinilla: un canal técnico de «volvemos enseguida», un canal de promoción y un respaldo para cuando caen las fuentes.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
  - name: promo
    inputs:
    - file:
        path: /storage/promo.mp4
    dvr: {}
```

PARÁMETROS

Parámetro	Descripción
`path`	ruta del archivo MP4 o Matroska en el streamer. El proceso Sapsan debe poder leerlo.

OPCIONES DE TRANSPORTE

Sistema de archivos local, contenedor MP4 o Matroska. La reproducción es cíclica; no requiere conexión de red.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG (MP4)**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (MP4: Layer II / III; Matroska: MP3)**
- **Opus**

MP4 y Matroska son contenedores. El lector Matroska procesa los archivos `.mkv` y `.webm`; las demás rutas usan el lector MP4. Matroska admite vídeo H.264, HEVC y AV1; JPEG corresponde a MP4. En MP4, AV1 requiere `av01` / `av1C`, Opus requiere `Opus` / `dOps` y MPEG Audio se identifica mediante `esds` o una sample entry MP3. La sample entry `mp4a` por sí sola no significa AAC. Leer frames comprimidos del archivo no los transcodifica automáticamente.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

© Flussonic 2026

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

FilePlay

Campo	Tipo
`bytes`	`counter`
`frames`	`counter`
`memory_bytes`	`gauge`

OTROS DETALLES

Cómo preparar el archivo con ffmpeg

Para que un archivo MBR se reproduzca sin costuras, todas las calidades deben estar perfectamente alineadas: timestamps idénticos y keyframes en los mismos sitios. Se logra con una sola pasada de ffmpeg con el filtro `split`:

```
ffmpeg -y -i source.mkv \
  -filter_complex "\
[0:v:0]split=3[vhi][vme][vlo]; \
[vhi]scale=1920:1080:force_original_aspect_ratio=decrease,pad=1920:1080:(ow-iw)/2:(oh-ih)/2,setsar=1,fps=25,setpts=PTS-STARTPTS[v1]; \
[vme]scale=1280:720:force_original_aspect_ratio=decrease,pad=1280:720:(ow-iw)/2:(oh-ih)/2,setsar=1,fps=25,setpts=PTS-STARTPTS[v2]; \
[vlo]scale=960:540:force_original_aspect_ratio=decrease,pad=960:540:(ow-iw)/2:(oh-ih)/2,setsar=1,fps=25,setpts=PTS-STARTPTS[v3]" \
  -map "[v1]" -c:v:0 libx264 -b:v:0 6000k -g:v:0 100 -keyint_min:v:0 100 \
    -sc_threshold:v:0 0 -x264-params:v:0 "scenecut=0:open_gop=0" \
  -map "[v2]" -c:v:1 libx264 -b:v:1 3000k -g:v:1 100 -keyint_min:v:1 100 \
    -sc_threshold:v:1 0 -x264-params:v:1 "scenecut=0:open_gop=0" \
  -map "[v3]" -c:v:2 libx264 -b:v:2 1200k -g:v:2 100 -keyint_min:v:2 100 \
    -sc_threshold:v:2 0 -x264-params:v:2 "scenecut=0:open_gop=0" \
  -map 0:a:0 -c:a:0 aac -b:a:0 192k -ac 2 -ar 48000 \
  -metadata:s:v:0 title="1080p" -metadata:s:v:1 title="720p" -metadata:s:v:2 title="540p" \
  -metadata:s:a:0 language=eng \
  -movflags +faststart \
  promo.mp4
```

Lo que aquí importa:

- **una sola pasada, una sola fuente,** `split` — todas las calidades obtienen timestamps idénticos;
- `setpts=PTS-STARTPTS` — la línea de tiempo de cada calidad empieza en cero;
- el mismo `fps` en todas las calidades;
- **GOP rígido**: `-g` = `-keyint_min`, `-sc_threshold 0` y `scenecut=0:open_gop=0` — keyframes estrictamente cada N fotogramas y en los mismos sitios; el codificador no puede insertarlos en los cambios de escena;
- audio AAC a 48 kHz estéreo;
- `-movflags +faststart` — el índice al principio del archivo;
- `-metadata:s:v title=...` / `language=...` — las etiquetas de calidad y los idiomas de las pistas acaban en las playlists.

Un ejemplo completo y funcional con varias pistas de audio está en el repositorio de sapsan: `storage/king.sh`.

Comprobación

Abra `http://server/streaming/v/promo/index.m3u8` — en la playlist master deben estar todas las calidades, y la conmutación entre ellas no debe dar tirones.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

SDI

`media.ingest.sdi`

DESCRIPCIÓN

Sapsan recibe vídeo sin comprimir y audio embebido de tarjetas SDI. Familias admitidas:

- **DekTec**
- **DeckLink**
- **AJA**
- **Streamlabs**
- **Softlab**
- **Magewell**
- **Acasis**

12G-SDI es una ampliación independiente. La salida SDI y la captura ASI tienen páginas separadas.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - sdi:
      device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
      pixel_format: v210
```

PARÁMETROS

Campo	Finalidad
`device`	Ruta del dispositivo de captura V4L2; obligatoria
`standard`	Estándar opcional, como `hd1080i50` o `uhd2160p50`. Se detecta a partir de la señal; indíquelo si el controlador no puede detectarlo antes de iniciar la captura
`pixel_format`	8 bits, predeterminado; `v210`: 10 bits
`audio`	Lista de pistas de audio; por defecto, estéreo de los canales 1–2
`ad_marks`	Recibir marcadores SCTE-104 de VANC; activado por defecto
`timecode`	Usar código de tiempo ATC; activado por defecto
`peer_timeout_ms`	Tiempo de espera de fotogramas; 30 000 ms por defecto

OPCIONES DE TRANSPORTE

La captura requiere un controlador con V4L2 y acceso al dispositivo para Sapsan. Utilice una ruta estable `/dev/v4l/by-path/…`.

CÓDECS ADMITIDOS

- **UYVY 4:2:2 (8-bit)**
- **v210 4:2:2 (10-bit)**
- **PCM**
- **AC-3 (SMPTE ST 337)**
- **E-AC-3 (SMPTE ST 337)**

© Flussonic 2026

UYVY y v210 son formatos de píxeles sin comprimir, y PCM es audio sin comprimir; no son códecs de vídeo H.264/HEVC. Configure transcodificación para obtener una salida comprimida. AC-3/E-AC-3 se extraen de un par de canales de audio solo con `sample_type: smpte337`; SMPTE ST 337 describe el empaquetado, no un códec de audio independiente.

COMPORTAMIENTO ANTE ERRORES

Cuando desaparece la señal, la entrada espera su recuperación durante `peer_timeout_ms`; al agotarse el plazo, la causa del fallo es `no_signal`. Un cambio de estándar de vídeo reinicia la captura con los nuevos parámetros. Si el dispositivo está ocupado, compruebe la dirección del puerto y los procesos que lo utilizan.

La entrada utiliza los mecanismos comunes de reintento y conmutación a reserva. Si el formato del dispositivo es incompatible, compruebe que esté instalado un controlador compatible con la tarjeta.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

© Flussonic 2026

Sdi

Campo	Tipo
`line_state`	`gauge`
`producer_restarts`	`counter`
`producer_restarts_wrong_standard`	`counter`
`producer_restarts_silence`	`counter`
`datagrams`	`counter`
`boundary_loss`	`counter`
`crc_errors`	`counter`
`unsupported_codec`	`counter`
`vanc_unknown`	`counter`
`vanc_cdp`	`counter`
`vanc_cdp_pairs`	`counter`
`vanc_cdp_bad_identifier`	`counter`
`vanc_cdp_bad_checksum`	`counter`
`vanc_cdp_malformed`	`counter`
`vanc_cdp_without_data`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`
`timecode_missing`	`counter`
`timecode_rejected`	`counter`
`timecode_untrusted`	`counter`
`timecode_jumps`	`counter`

OTROS DETALLES

Pistas de audio

La posición en `audio` define el número de pista de audio del stream. `channels` contiene los números de canales de audio SDI, del 1 al 16, en el orden deseado; `lang` es el idioma según ISO 639-2. Un canal no puede pertenecer a dos pistas. Una lista vacía selecciona estéreo del primer par.

`sample_type: pcm` selecciona audio sin comprimir. Si un canal transporta datos en lugar de PCM, se sustituye por silencio en la pista PCM.

AC-3 y E-AC-3 mediante SMPTE 337

Para audio comprimido en un par de canales SDI, configure `sample_type: smpte337` y exactamente dos números de canal. La entrada extrae AC-3 o E-AC-3 sin decodificarlos a PCM. Los tipos no admitidos, incluido Dolby E, y los modos ST 337 de 20/24 bits se contabilizan en `sdi_unsupported_codec_total`.

```
audio:
  - channels: [1, 2]
    lang: rus
  - channels: [3, 4]
    sample_type: smpte337
    lang: rus
```


© Flussonic 2026

Datos auxiliares VANC

- **OP-47 y teletexto VBI** alimentan la pista de teletexto.
- **SCTE-104** se convierte en marcadores de la pista común `scte35`; consulte SCTE-35 para la entrega posterior.
- **CDP (CEA-608/708)** se transporta junto al fotograma a la codificación H.264/HEVC y se publica en SEI. El modo `closed_captions` del stream controla los subtítulos integrados y las pistas WebVTT; consulte Closed captions.
- **Código de tiempo ATC** se utiliza cuando `timecode` está activado.
- **Pares DID/SDID desconocidos** se contabilizan.

Captura y salida son funciones independientes. La salida requiere un puerto de salida; ASI tiene una página separada.

QUÉ SIGUE

- Supervisión de la fuente SDI
- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

12G-SDI

`media.ingest.sdi_12g`

DESCRIPCIÓN

`media.ingest.sdi_12g` amplía la captura SDI básica. Incluye código para cargas especialmente elevadas, próximas a los límites del hardware moderno. El servidor considera NUMA y el diseñador debe considerar el ancho de banda real del bus de memoria.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - sdi:
      device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
      standard: uhd2160p50
      pixel_format: v210
```

PARÁMETROS

Parámetro	Descripción
—	Los campos coinciden con SDI básico.
`standard`	Indique `standard` explícitamente en puertos 12G de DTA-2178 para iniciar la captura.

OPCIONES DE TRANSPORTE

La captura requiere un controlador con V4L2 y acceso al dispositivo para Sapsan. Utilice una ruta estable `/dev/v4l/by-path/…`.

CÓDECS ADMITIDOS

- **UYVY 4:2:2 (8-bit)**
- **v210 4:2:2 (10-bit)**
- **PCM**
- **AC-3 (SMPTE ST 337)**
- **E-AC-3 (SMPTE ST 337)**

UYVY y v210 son formatos de píxeles sin comprimir, y PCM es audio sin comprimir; no son códecs de vídeo H.264/HEVC. Configure transcodificación para obtener una salida comprimida. AC-3/E-AC-3 se extraen de un par de canales de audio solo con `sample_type: smpte337`; SMPTE ST 337 describe el empaquetado, no un códec de audio independiente.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

© Flussonic 2026

Sdi

Campo	Tipo
line_state	gauge
producer_restarts	counter
producer_restarts_wrong_standard	counter
producer_restarts_silence	counter
datagrams	counter
boundary_loss	counter
crc_errors	counter
unsupported_codec	counter
vanc_unknown	counter
vanc_cdp	counter
vanc_cdp_pairs	counter
vanc_cdp_bad_identifier	counter
vanc_cdp_bad_checksum	counter
vanc_cdp_malformed	counter
vanc_cdp_without_data	counter
ad_marks	counter
errors_ad_marks_dropped	counter
warning_ad_mark_descriptors_dropped	counter
timecode_missing	counter
timecode_rejected	counter
timecode_untrusted	counter
timecode_jumps	counter

OTROS DETALLES

Los fotogramas sin comprimir cargan PCIe y memoria independientemente del bitrate de salida. Dimensione las entradas según la capacidad real del bus, la resolución y la frecuencia.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

ASI

`media.ingest.asi`

DESCRIPCIÓN

ASI transporta MPEG-TS por un conector físico, mientras que SDI transporta vídeo sin comprimir. Para recibir una señal ASI en Sapsan, utilice una pasarela externa de ASI a red y configure una entrada MPEG-TS de red.

CONFIGURACIÓN

Configure la pasarela para enviar MPEG-TS al streamer receptor. Configure la entrada UDP correspondiente en Sapsan:

```
streams:
- name: asi_gateway
  inputs:
  - udp:
      host: 0.0.0.0
      port: 5000
```

PARÁMETROS

Los valores `host` y `port` corresponden al receptor de red. La tarjeta, el conector y el modo ASI se configuran en la pasarela.

OPCIONES DE TRANSPORTE

La pasarela recibe ASI físico y envía MPEG-TS por UDP. Sapsan recibe el stream MPEG-TS de red.

CÓDECS ADMITIDOS

Consulte la lista de códecs MPEG-TS. La pasarela debe conservar el flujo de transporte y sus tablas de servicio.

COMPORTAMIENTO ANTE ERRORES

Compruebe la señal del conector ASI, el destino de la pasarela y la entrega de red. Diagnostique el stream receptor con los contadores de entrada MPEG-TS.

APLICACIÓN DE CAMBIOS

Aplique la configuración de la entrada de red mediante la API de central y compruebe el estado de la entrada y la llegada de fotogramas.

CONTADORES

Use los contadores de entrada MPEG-TS de red. El inventario de tarjetas describe el hardware conectado al servidor.

OTROS DETALLES

ASI depende del modo del firmware y de la dirección del puerto físico. Un conector BNC por sí solo no identifica el tipo de señal.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

NDI

`media.ingest.ndi`

DESCRIPCIÓN

Sapsan recibe vídeo y audio de fuentes NDI en la red local. Conéctese al puerto de medios o seleccione el nombre exacto del catálogo del emisor. Con `decode: false`, omite la descompresión SpeedHQ y transmite el vídeo comprimido por el sistema.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - ndi:
      url: ndi://192.0.2.10:5961
      transport: tcp
      decode: false
```

PARÁMETROS

Campo	Descripción
`url`	`ndi://host[:port]`, sin ruta ni credenciales.
`source`	nombre exacto del catálogo; omítalo para conexión directa.
`transport`	transporte elegido.
`multicast_interface`	dirección IPv4 local para multicast.
`decode`	descomprimir SpeedHQ (`true` por defecto); `false` conserva fotogramas comprimidos.
`peer_timeout_ms`	tiempo de espera sin datos.

OPCIONES DE TRANSPORTE

- **TCP** — `tcp`, predeterminado.
- **UDP unicast** — `udp_unicast`.
- **UDP multicast** — `udp_multicast`.
- **RUDP/QUIC** — `rudp`.

Los metadatos de control usan TCP. Seleccionar UDP no activa la vuelta automática a medios TCP.

CÓDECS ADMITIDOS

- **SpeedHQ SHQ0**
- **SpeedHQ SHQ2**
- **SpeedHQ SHQ4**
- **SpeedHQ SHQ7**
- **SpeedHQ SHQ9**
- **PCM s16**
- **Float32 audio**

SpeedHQ es vídeo NDI comprimido. Con `decode: false`, la entrada pasa esos frames sin descomprimirlos; con `decode: true`, los decodifica en planos YUV de 8 bits sin comprimir. PCM s16 y Float32 son formatos de muestras de audio sin comprimir. Admitir NDI no implica admitir H.264/HEVC de NDI HX.

© Flussonic 2026

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

`bytes` cuenta bytes entrantes; `video_frames` y `audio_frames`, fotogramas recibidos. `frames_dropped` cuenta pérdidas de ensamblado o cola. `packets_lost` cuenta datagramas UDP perdidos; `packets_repeated`, repetidos; `fec_packets_recovered`, recuperados por FEC. `quic_stream_retransmits` y `quic_stream_retransmit_bytes` describen retransmisión QUIC local saliente, no pérdida de vídeo entrante.

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Ndi

Campo	Tipo
`memory_bytes`	`gauge`
`bytes`	`counter`
`video_frames`	`counter`
`audio_frames`	`counter`
`metadata_frames`	`counter`
`errors_video_decoder`	`counter`
`errors_audio_decoder`	`counter`
`frames_dropped`	`counter`
`datagrams`	`counter`
`packets_lost`	`counter`
`packets_repeated`	`counter`
`repeated_bytes`	`counter`
`fec_packets_recovered`	`counter`
`fec_bytes`	`counter`
`quic_packets_lost`	`counter`
`quic_stream_retransmits`	`counter`
`quic_stream_retransmit_bytes`	`counter`
`quic_receive_queue_bytes`	`gauge`

OTROS DETALLES

Con `decode: false`, el siguiente consumidor debe aceptar SpeedHQ comprimido. Añada transcodificación para salida HLS. `decode: true` convierte el vídeo en fotogramas sin comprimir y aumenta carga de CPU y memoria.

© Flussonic 2026

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

ST 2110

`media.ingest.st2110`

DESCRIPCIÓN

Sapsan recibe señales de estudio ST 2110 con pistas separadas de vídeo sin comprimir, audio y datos auxiliares. Dos fuentes de configuración actúan simultáneamente: central coloca el stream en un streamer y define su entrada; un controlador NMOS externo configura direcciones multicast y formatos de recepción.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - st2110:
      network: studio
      receivers:
        video:
          enabled: true
        audio:
          enabled: true
        data:
          enabled: true
```

PARÁMETROS

Campo	Descripción
`network`	etiqueta de la red de medios configurada en la interfaz del streamer.
`receivers`	pistas receptoras; `enabled` activa la recepción.
`reduce_to_8bit`	reducir vídeo de 10 a 8 bits.
`recv_buffer_bytes`	búfer del socket; `queue_max_bytes` : límite de cola; `audio_frame_ms` : duración de fotograma de audio. El modo manual usa `host` , `port` , `source` y `format` ; `sdp` aporta la descripción de transporte.

AUTENTICACIÓN

El streamer sirve IS-05 directamente. Limite el acceso del controlador NMOS a la red de gestión; NMOS no sustituye la autorización de la API de central.

OPCIONES DE TRANSPORTE

- **ST 2110-20** — vídeo por RTP/UDP multicast.
- **ST 2110-30** — audio.
- **ST 2110-40** — datos auxiliares.

PTP sincroniza la línea temporal. SSM especifica la dirección del emisor.

CÓDECS ADMITIDOS

- **YCbCr 4:2:2 (8-bit)**
- **YCbCr 4:2:2 (10-bit)**
- **PCM L16**
- **PCM L24**

YCbCr describe píxeles sin comprimir; L16/L24 son muestras PCM sin comprimir. El vídeo usa ST 2110-20 y el audio ST 2110-30. La recepción de AM824/AES3 (ST 2110-31) no es un formato de entrada admitido. SMPTE ST 291 para ANC empaqueta datos auxiliares; no es un códec de vídeo.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

© Flussonic 2026

APLICACIÓN DE CAMBIOS

Los cambios de central y los comandos NMOS coexisten: central controla la ubicación y composición de la entrada; el controlador, la conexión de pistas. El streamer aplica la activación IS-05 a la dirección del receptor. Cambiar dirección o formato reconecta el receptor afectado.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

St2110

Campo	Tipo
`declared_tracks`	`gauge`
`receiving_tracks`	`gauge`
`diverged_clock_tracks`	`gauge`
`clock_drift_ms`	`gauge`
`xdp_reopens`	`counter`

© Flussonic 2026

RtpChannel

Campo	Tipo
channel_id	label
rtp_packets	counter
rtcp_packets	counter
bytes	counter
frames	counter
pt_reject_count	counter
pt_reject_bytes	counter
ts_goes_backwards	counter
ts_jump_forward	counter
ts_stuck	counter
errors_dts_stuck	counter
sr_ts_stuck	counter
ts_stuck_restarts	counter
resync_sr	counter
marker_packets_count	counter
no_marker_mode_flag	gauge
errors_broken_payload	counter
errors_lost_packets	counter
errors_dropped_packets	counter
errors_late_packets	counter
errors_incomplete_frames	counter
nack_count	counter
onvif_buffer_overflow	counter
onvif_seq_gap	counter
onvif_marker_mismatch	counter
onvif_resync_dropped	counter

OTROS DETALLES

NMOS IS-04 anuncia receptores e IS-05 los conecta. Sin controlador, las direcciones y formatos pueden configurarse manualmente mediante la API de central. El vídeo sin comprimir requiere ancho de banda de red y memoria suficiente; añada transcodificación para HLS y otras salidas comprimidas.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

H.323

`media.ingest.h323`

DESCRIPCIÓN

Esta función está disponible **solo en Flussonic Media Server**. Sapsan no proporciona entrada H.323. FMS recibe dispositivos de videoconferencia iniciando una llamada saliente. Consulte el manual FMS.

CONFIGURACIÓN

Esta configuración corresponde a Flussonic Media Server:

```
stream polycom1 {
  input h323://192.168.100.150 vb=2000k id="Flussonic";
}
```

PARÁMETROS

Parámetro	Descripción
`h323://…`	dirección del dispositivo
`vb`	bitrate de codificación en FMS
`id`	nombre mostrado por el dispositivo remoto.

OPCIONES DE TRANSPORTE

FMS inicia una llamada H.323 saliente al dispositivo de videoconferencia.

CÓDECS ADMITIDOS

Los códecs de entrada dependen de FMS y del dispositivo remoto. En este escenario, FMS produce audio AAC. Sapsan no tiene entrada H.323.

COMPORTAMIENTO ANTE ERRORES

Diagnostique la llamada en FMS; Sapsan no crea esta entrada.

APLICACIÓN DE CAMBIOS

Modifique la configuración FMS. Estos parámetros no se envían a Sapsan por la API de central.

CONTADORES

Sapsan no tiene contadores H.323. Las estadísticas de captura pertenecen a FMS.

OTROS DETALLES

La función está en la tarifa compartida; su presencia en esta referencia no significa que esté implementada en Sapsan.

QUÉ SIGUE

- Manual FMS H.323
- Referencia de funciones

© Flussonic 2026

HDMI

`media.ingest.hdmi`

DESCRIPCIÓN

Capture vídeo sin comprimir y audio embebido por HDMI con tarjetas **DeckLink** y la serie **HWS** (AVMatrix, Acasis). Las resoluciones y frecuencias dependen del modelo y de la señal de entrada.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - hdmi:
      device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
```

PARÁMETROS

Parámetro	Función
`device`	Ruta obligatoria del dispositivo HDMI V4L2; se recomienda un enlace estable `/dev/v4l/by-path/…`
`standard`	Estándar de señal opcional; si se omite, se detecta en la entrada
`pixel_format`	8 bits por defecto; `v210` — 10 bits. HDMI RGB utiliza el ARGB negociado por el controlador
`audio`	Lista de pistas de audio; una lista vacía selecciona una pista PCM estéreo del primer par de canales
`audio[].channels`	Números de los canales de audio de entrada desde uno; el orden define la disposición de canales de la pista
`audio[].sample_type`	`pcm` por defecto; `smpte337` extrae audio comprimido AC-3/E-AC-3 de un par de canales
`audio[].lang`	Idioma opcional de la pista
`peer_timeout_ms`	Tiempo de espera de fotogramas en milisegundos; por defecto 30 000

Utilice la sección de entrada independiente `hdmi`. Los campos SDI `ad_marks` y `timecode` no están disponibles: HDMI no transporta VANC SCTE-104 ni código de tiempo ATC.

Ejemplo de una pista PCM estéreo explícita; su número corresponde a su posición en la lista `audio`, desde uno:

```
hdmi:
  device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
  audio:
  - channels: [1, 2]
    sample_type: pcm
    lang: eng
```

OPCIONES DE TRANSPORTE

La captura requiere un controlador de tarjeta HDMI con interfaz V4L2 y acceso al dispositivo para el proceso Sapsan. Al abrirlo se comprueba que el controlador lo identifique como HDMI; se rechazan los dispositivos SDI. Utilice un enlace estable `/dev/v4l/by-path/…`.

CÓDECS ADMITIDOS

- **UYVY 4:2:2 (8-bit)**
- **v210 4:2:2 (10-bit)**
- **ARGB (HDMI RGB)**
- **PCM**
- **AC-3 (SMPTE ST 337)**
- **E-AC-3 (SMPTE ST 337)**

© Flussonic 2026

UYVY, v210 y ARGB son formatos de píxeles sin comprimir, y PCM es audio sin comprimir; no son códecs de vídeo H.264/HEVC. Configure transcodificación para obtener una salida comprimida. AC-3/E-AC-3 se extraen de un par de canales de audio solo con `sample_type: smpte337`; SMPTE ST 337 describe el empaquetado, no un códec de audio independiente.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

Las estadísticas de entrada utilizan la categoría `hdmi`. Los contadores de hardware se comparten con SDI, pero el procesamiento VANC y ATC está desactivado: sus contadores no representan actividad HDMI.

HDMI

Campo	Tipo
`line_state`	`gauge`
`producer_restarts`	`counter`
`producer_restarts_wrong_standard`	`counter`
`producer_restarts_silence`	`counter`
`datagrams`	`counter`
`boundary_loss`	`counter`
`crc_errors`	`counter`
`unsupported_codec`	`counter`
`vanc_unknown`	`counter`
`vanc_cdp`	`counter`
`vanc_cdp_pairs`	`counter`
`ad_marks`	`counter`
`errors_ad_marks_dropped`	`counter`
`warning_ad_mark_descriptors_dropped`	`counter`
`timecode_missing`	`counter`
`timecode_rejected`	`counter`
`timecode_untrusted`	`counter`
`timecode_jumps`	`counter`

© Flussonic 2026

OTROS DETALLES

Los fotogramas sin comprimir cargan PCIe y memoria independientemente del bitrate de salida. Dimensione las entradas según la capacidad real del bus, la resolución y la frecuencia.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

M4F / M4S / M4F2

`media.ingest.m4f`

DESCRIPCIÓN

Se admiten los protocolos anteriores M4F/M4S y el nuevo M4F2 para capturar de otros servidores Flussonic. M4F2 maneja pistas de forma independiente, transfiriendo las necesarias y reduciendo el tráfico entre servidores. La entrega conserva la línea temporal y la descripción de pistas.

CONFIGURACIÓN

Configure el stream mediante la API de central. Fragmento de configuración:

```
streams:
- name: source1
  inputs:
  - m4f2:
      url: m4f2://origin.example.com/studio?token=ACCESS_TOKEN
```

PARÁMETROS

Parámetro	Descripción
`m4f.url`, `m4s.url`, `m4f2.url`	`m4f.url`, `m4s.url`, `m4f2.url` identifican la versión de la fuente.
`cluster_key`	M4F/M4S anteriores usan `cluster_key` para la clave compartida.
`via_agent`	M4S y M4F2 admiten `via_agent` para conectarse mediante `agent://ID`.

AUTENTICACIÓN

M4F/M4S envían cabeceras `cluster-auth-*` con la clave compartida. M4F2 propaga los parámetros de consulta, incluido `token`, a las solicitudes.

OPCIONES DE TRANSPORTE

- **M4F** — captura de segmentos anterior.
- **M4S** — captura de fotogramas anterior.
- **M4F2** — catálogo y solicitudes independientes de fragmentos/partes de pistas por HTTP; `m4f2s://` usa HTTPS.

CÓDECS ADMITIDOS

M4F/M4S heredados:

- **H.264**
- **HEVC / H.265**
- **AV1**
- **VP9**
- **MPEG-2 Video**
- **MJPEG / JPEG**
- **JPEG 2000**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **Opus**
- **G.711 A-law / μ-law**
- **PCM**

© Flussonic 2026

La lista enumera los frames de audio y vídeo que el analizador M4F/M4S heredado extrae mediante los nombres de códec de las pistas. M4F es un contenedor propio, no otro nombre de fMP4 ni un códec. La fuente debe declarar correctamente el códec y su configuración; recibir frames no implica compatibilidad de decodificación ni que el protocolo de salida admita todos esos códecs.

M4F2: las descripciones de pistas se leen de segmentos de inicialización fMP4 y los frames de fragmentos y partes separados. Se aplica el conjunto de códecs fMP4. Esta ruta no admite VP9, JPEG 2000, MJPEG ni PCM sin comprimir de la lista M4F/M4S heredada. Conservar la línea temporal y los límites de fragmentos de la fuente no amplía las sample entries admitidas.

COMPORTAMIENTO ANTE ERRORES

El estado de la entrada y la causa del fallo están en las estadísticas del stream. Ante desconexión o ausencia de fotogramas se aplican los mecanismos de reintento y conmutación a reserva.

APLICACIÓN DE CAMBIOS

Aplique los cambios mediante la API de central. Cambiar la dirección o el formato vuelve a abrir la captura y puede interrumpirla durante la conexión. Compruebe después el estado y la llegada de fotogramas.

CONTADORES

Las tablas indican los nombres exactos de los campos de estadísticas del protocolo, incluidos contadores de eventos e indicadores del valor actual. Los campos nulos pueden omitirse. API y Prometheus exponen proyecciones distintas; un único resultado de Admin API no incluye necesariamente toda la lista. El detalle depende de la licencia de contadores.

Totales comunes de entrada: `bytes`, `frames`, `errors`, `warnings`. Agregan datos del protocolo; no deben sumarse otra vez a los contadores del protocolo.

M4f2

Campo	Tipo
`version`	`gauge`
`bytes`	`counter`
`frames`	`counter`
`parts`	`counter`
`fragments`	`counter`
`fragments_from_parts`	`counter`
`part_sets_broken`	`counter`

OTROS DETALLES

M4F2 no exige enviar todas las calidades de vídeo a cada consumidor. Las solicitudes independientes permiten obtener solo las calidades y pistas de audio necesarias, sin transcodificar.

QUÉ SIGUE

- Reserva de fuentes
- Transcodificación
- Grabación en archivo

© Flussonic 2026

Captura multibitrate

La entrada `mbr` reúne varias calidades de una señal en un stream. Conserva todas las pistas de la primera fuente, incluido el audio, y toma solo vídeo de las demás. Descarta el audio de las fuentes adicionales.

CONFIGURACIÓN

```
streams:
  - name: cam1
    inputs:
    - mbr:
        inputs:
        - rtsp:
            url: rtsp://admin:password@10.0.0.5/stream0
        - rtsp:
            url: rtsp://admin:password@10.0.0.5/stream1
```

Parámetro	Descripción
`mbr.inputs`	Lista de entradas secundarias. Cada elemento tiene la estructura de una entrada normal. Coloque primero la fuente con el audio necesario.

TRANSPORTE Y CÓDECS

MBR no tiene transporte propio: cada entrada define su conexión. Los códecs admitidos dependen de esas entradas y del protocolo de reproducción elegido. MBR une las pistas sin recodificarlas.

ALINEACIÓN Y LL-HLS

Para el cambio adaptativo, las calidades deben representar el mismo instante de la señal: los timestamps y los límites de GOP, con sus fotogramas clave correspondientes, deben estar perfectamente alineados. El orden de las fuentes determina la composición de la entrada; no garantiza el orden de las calidades en la playlist.

Las cámaras normalmente no ofrecen esa alineación. Sapsan usa mecanismos complejos de normalización temporal y segmentación para compensar las incompatibilidades entre los streams multibitrate de cámaras y los protocolos modernos, incluido LL-HLS. Esto no crea fotogramas clave ausentes ni garantiza reparar cualquier GOP incompatible. Pruebe los cambios de calidad y la latencia con la señal real.

ERRORES Y CAMBIOS DE CONFIGURACIÓN

Si una rama no se abre, falla toda la entrada MBR y se cierran las ramas ya abiertas. Si una rama termina o devuelve un error durante la ejecución, se cierra toda la entrada MBR y la causa pasa al mecanismo de entradas LSI. LSI gestiona la reconexión y la selección de la fuente de reserva; MBR no tiene un ciclo de reconexión propio. `source_open_timeout` limita la apertura de toda la escalera de calidades.

Cambiar `mbr.inputs` requiere reiniciar la entrada y abrir de nuevo todas las ramas. Aplicar la misma configuración no reinicia MBR.

ALTERNATIVA: TRANSCODIFICACIÓN

Con una sola fuente, o para crear calidades alineadas de nuevo, use el transcodificador.

QUÉ SIGUE

- Reproducción HLS
- Grabación del archivo
- Reserva de fuentes

© Flussonic 2026

Fuente sintética

La entrada `syntetic` es un generador de señal de prueba integrado: barras SMPTE con un reloj corriendo y un tono. No necesita datos externos y sirve para comprobar la instalación, depurar y hacer pruebas de carga.

CONFIGURACIÓN

El generador emite fotogramas sin comprimir (YUV y PCM en bruto). Es deliberado: la señal en bruto se puede enviar sin pérdidas directamente a SDI. Para servir el stream por protocolos de red (y obtener las variantes de códec y calidad que necesita), combine el generador con un transcodificador:

```
streams:
  - name: syn
    inputs:
    - syntetic: {}
    transcoder:
      output:
      - source: !content video
        codec: !set h264
      - source: !content audio
        codec: !set aac
    segments: 6
```

Sin una sección `transcoder` el stream lleva fotogramas en bruto: un reproductor corriente no puede reproducirlo, pero justamente ese es el stream necesario para la salida a SDI.

Parámetros del generador (todos opcionales, `syntetic: {}` da una señal de demostración):

```
    - syntetic:
        video: !video
          rate:
            numerator: 30000     # se admiten frecuencias de fotograma fraccionarias (NTSC)
            denominator: 1001
        audio: !audio
          sample_rate: 48000
          channels: 2
```

TRANSPORTE Y CÓDECS

Es un generador local sin transporte de red. Produce YUV y PCM. Para HLS/DASH y otras entregas de red, codifique a los códecs admitidos por cada protocolo, como en el ejemplo anterior.

PARÁMETROS

Parámetro	Descripción
`video`	Generador de vídeo `!video`. Si solo se define audio, no se añade vídeo.
`audio`	Generador de audio `!audio`. Si solo se define vídeo, no se añade audio.
`video.rate`	Frecuencia: `numerator` / `denominator`. Un `!video {}` explícito usa 30 fps de forma predeterminada.
`video.size`	Tamaño predeterminado: `{width: 1920, height: 1080}`.
`audio.sample_rate`	Frecuencia de muestreo PCM, 48000 Hz de forma predeterminada.
`audio.channels`	Número de canales, 2 de forma predeterminada; limitado a 1–2.
`audio.bitrate`	El campo existe en la configuración, pero el generador PCM no lo usa; defina el bitrate de audio comprimido en el transcodificador.

Si faltan tanto video como audio, se activa la señal de demostración completa: 1920×1080 a 25 fps y PCM estéreo de 48000 Hz. Es distinto de la frecuencia predeterminada de `video: !video {}` explícito.

APLICACIÓN DE CAMBIOS

Los ajustes se actualizan durante la ejecución sin reabrir la entrada. Después del cambio, compruebe las pistas y sus propiedades en las estadísticas del stream.

© Flussonic 2026

TIMECODE EN LA IMAGEN

El PTS de cada fotograma se dibuja en la banda de luminancia de la imagen y puede volverse a leer a máquina. Eso permite medir la latencia de extremo a extremo y encontrar fotogramas perdidos a partir de la propia imagen — práctico a la hora de depurar el transcodificador y los protocolos.

SIGUIENTES PASOS

- Transcodificación
- Canal a partir de un archivo

© Flussonic 2026

6.2.3 Publicación

Publicación SRT

`media.publish.srt`

La fuente se conecta a Sapsan y envía el stream. Es publicación; la captura cliente desde un servidor remoto pertenece a `media.ingest`.

El stream receptor se selecciona por su puerto SRT. Configure `passphrase` para autenticar la fuente y cifrar la conexión. Véase Autorización de publicación SRT.

CONFIGURACIÓN

```
streams:
- name: studio
  srt_publish:
    port: 9000
    passphrase: verysecretpass
    latency_ms: 120
```

URL del emisor: `srt://server:9000?passphrase=verysecretpass`. El puerto pertenece a un stream. `srt_publish` implica una entrada `publish`; no es necesario declararla. `latency_ms` establece latencia de recepción, 120 ms por defecto. El puerto selecciona el stream; `streamid` no lo hace.

AUTENTICACIÓN

La `passphrase` del stream (10–79 bytes) controla acceso y cifrado. `auth.publish_tokens` y `streamid` no autorizan publicación SRT. Sin `passphrase`, el puerto acepta conexiones sin cifrar.

TRANSPORTE

MPEG-TS sobre SRT/UDP. `srt_publish.port` acepta conexiones del emisor.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC / H.265**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC (ADTS)**
- **AC-3**
- **E-AC-3**
- **MPEG Audio (Layer II / III)**
- **SMPTE ST 302M**

MPEG-TS es el contenedor; SRT, RIST, UDP y HTTP son transportes de entrega. La lista describe la extracción de frames de MPEG-TS, no su decodificación ni la compatibilidad del reproductor.

AV1 requiere `stream_type 0x06` y el descriptor de registro `AV01` en la PMT; cada PES debe contener un temporal unit completo. No se admite AV1 en MPEG-TS sin esta señalización. AAC se analiza como ADTS; reconocer `stream_type 0x11` no implica compatibilidad con LATM. Opus se puede identificar en la PMT, pero el analizador PES actual no emite sus frames.

COMPORTAMIENTO ANTE ERRORES

Un stream acepta una publicación activa a la vez. Al terminar la conexión, la fuente deja de estar disponible y se aplican las reglas de reserva. El emisor inicia la reconexión.

Si una fuente operativa precede a publish, el codificador RTMP o SRT se acepta como reserva. Tras la recuperación de la fuente principal y su ventana limpia, permanece conectado como reserva. Se rechaza un segundo codificador aunque el primero esté en reserva o aún no envíe cuadros

© Flussonic 2026

utilizables. El cambio al vídeo espera el siguiente cuadro clave; solo audio espera un cuadro de audio utilizable. Se mantienen los tiempos de espera de fuente y transporte.

```
streams:
- name: studio
  srt_publish:
    port: 9000
    passphrase: verysecretpass
  inputs:
  - rtsp:
      url: rtsp://camera/main
  - publish: {}
```

QUÉ SIGUE

- SRT / MPEG-TS
- Grabación en archivo
- Reproducción HLS
- Referencia de publicación

© Flussonic 2026

Autorización de publicación SRT

`media.publish.srt_streamkey`

El stream receptor se selecciona por su puerto SRT. Configure `passphrase` para autenticar la fuente y cifrar la conexión.

OBJETIVO

Limite la publicación a los codificadores que conocen la contraseña del stream receptor. En un clúster, conecte el codificador a la dirección y al puerto SRT del streamer que recibe esa publicación.

DIFERENCIA RESPECTO A LA PUBLICACIÓN BÁSICA

La sección `srt_publish` configura el puerto receptor y su contraseña. El valor `streamid` no selecciona un stream ni autoriza la publicación. Un token HTTP de reproducción no es una contraseña SRT.

CONFIGURACIÓN

Defina el puerto y la contraseña en el stream receptor:

```
streams:
- name: studio
  srt_publish:
    port: 9000
    passphrase: verysecretpass
    latency_ms: 120
```

La contraseña debe tener entre 10 y 79 bytes. Configure el emisor con `srt://server:9000?passphrase=verysecretpass`. Cada puerto receptor pertenece a un solo stream.

SIGUIENTES PASOS

- Publicación SRT
- SRT / MPEG-TS

TRANSPORTE

MPEG-TS sobre SRT/UDP.

CÓDECS COMPATIBLES

La lista de códecs coincide con la de publicación SRT.

© Flussonic 2026

Publicación RTMP

`media.publish.rtmp`

La fuente se conecta a Sapsan y envía el stream. Es publicación; la captura cliente desde un servidor remoto pertenece a `media.ingest`.

CONFIGURACIÓN

En la consola, abra el stream, añada una entrada **Publish** explícita en la posición deseada y configure la clave en su tarjeta. Introduzca la clave manualmente o pulse **Regenerar clave**: al menos 5 letras latinas, dígitos o `._-`. Se guarda en `inputs[index].publish.rtmp_streamkey`; configurar la clave no añade ni reordena entradas. No hay límite superior de longitud. La regeneración crea una clave aleatoria de 10 caracteres.

El campo oculta la clave. Use el botón de copia para configurar el codificador; el ojo permite mostrar temporalmente el valor.

```
listeners:
  rtmp:
  - port: 1935
streams:
- name: studio
  inputs:
  - publish:
      rtmp_streamkey: Abc012._-Z
```

En OBS, use el servidor `rtmp://server:1935` y la clave `Abc012._-Z`. En la API central, envíe `PATCH /central/api-v4/streams/configs/studio` con `{"inputs":[{"publish":{"rtmp_streamkey":"Abc012._-Z"}}]}`. El array `inputs` se sustituye entero: incluya las demás entradas en orden, con todos sus ajustes y secretos.

Si una fuente operativa precede a publish, el codificador RTMP o SRT se acepta como reserva. Tras la recuperación de la fuente principal y su ventana limpia, permanece conectado como reserva. Se rechaza un segundo codificador aunque el primero esté en reserva o aún no envíe cuadros utilizables. El cambio al vídeo espera el siguiente cuadro clave; solo audio espera un cuadro de audio utilizable. Se mantienen los tiempos de espera de fuente y transporte.

AUTENTICACIÓN

La clave autoriza la publicación RTMP en la única entrada publish explícita. Sin clave, incluso con `publish: {}`, RTMP se rechaza; el nombre del stream no sustituye al secreto. Central comprueba atómicamente la unicidad en la configuración actual de todo el clúster, incluidos los streams desactivados. No se guarda historial de claves. La validación rechaza una segunda entrada publish y publish dentro de MBR.

Tras aplicar el cambio o borrado de la clave, el servidor desconecta la publicación RTMP actual y rechaza la clave anterior. Borrar con `rtmp_streamkey: null` conserva la entrada publish. Cambiar la clave RTMP conserva la publicación SRT; otros cambios conservan la conexión RTMP si la clave no cambia.

TRANSPORTE

RTMP sobre TCP. El emisor se conecta al listener RTMP compartido del servidor.

CÓDECS ADMITIDOS

- **H.264**
- **HEVC (Enhanced RTMP, `hvc1`)**
- **AV1 (Enhanced RTMP, `av01`)**
- **AAC**
- **MP3**

RTMP transporta mensajes FLV; Enhanced RTMP identifica los códecs de vídeo mediante FourCC. La lista describe la recepción de frames comprimidos y su envío al procesamiento. AV1 requiere Enhanced RTMP con `av01`; seleccionar AV1 en el emisor sin este empaquetado no basta. MP3 se acepta como audio FLV, por separado de AAC.

COMPORTAMIENTO ANTE ERRORES

Un stream acepta una publicación activa a la vez. Al terminar la conexión, la fuente deja de estar disponible y se aplican las reglas de reserva. El emisor inicia la reconexión.

© Flussonic 2026

QUÉ SIGUE

- Grabación en archivo
- Reproducción HLS
- Referencia de publicación

© Flussonic 2026

Publicación RTMP mediante una clave

`media.publish.rtmp_streamkey`

La clave selecciona un stream y autoriza la publicación RTMP entrante. Sin `rtmp_streamkey` se rechaza la publicación, incluso si el stream declara una entrada `publish`.

FINALIDAD

El codificador necesita la dirección del servidor y la clave; no envía el nombre del stream en el comando de publicación. Conéctese al servidor configurado para recibir este stream. El enrutamiento automático de publicaciones entre servidores no se admite.

DIFERENCIA RESPECTO A LA PUBLICACIÓN BÁSICA

La clave autoriza y selecciona el stream para la publicación RTMP. No cambia el transporte ni los códecs.

CONFIGURACIÓN

En la consola, abra el stream, añada una entrada **Publish** explícita en la posición deseada y configure la clave en su tarjeta. Introduzca la clave manualmente o pulse **Regenerar clave**: al menos 5 letras latinas, dígitos o `._-`. Se guarda en `inputs[index].publish.rtmp_streamkey`; configurar la clave no añade ni reordena entradas. No hay límite superior de longitud. La regeneración crea una clave aleatoria de 10 caracteres.

El campo oculta la clave. Use el botón de copia para configurar el codificador; el ojo permite mostrar temporalmente el valor.

```
listeners:
  rtmp:
  - port: 1935
streams:
- name: studio
  inputs:
  - publish:
      rtmp_streamkey: Abc012._-Z
```

En OBS, use el servidor `rtmp://server:1935` y la clave `Abc012._-Z`. En la API central, envíe `PATCH /central/api-v4/streams/configs/studio` con `{"inputs":[{"publish":{"rtmp_streamkey":"Abc012._-Z"}}]}`. El array `inputs` se sustituye entero: incluya las demás entradas en orden, con todos sus ajustes y secretos.

La clave autoriza la publicación RTMP en la única entrada publish explícita. Sin clave, incluso con `publish: {}`, RTMP se rechaza; el nombre del stream no sustituye al secreto. Central comprueba atómicamente la unicidad en la configuración actual de todo el clúster, incluidos los streams desactivados. No se guarda historial de claves. La validación rechaza una segunda entrada publish y publish dentro de MBR.

Tras aplicar el cambio o borrado de la clave, el servidor desconecta la publicación RTMP actual y rechaza la clave anterior. Borrar con `rtmp_streamkey: null` conserva la entrada publish. Cambiar la clave RTMP conserva la publicación SRT; otros cambios conservan la conexión RTMP si la clave no cambia.

Si una fuente operativa precede a publish, el codificador RTMP o SRT se acepta como reserva. Tras la recuperación de la fuente principal y su ventana limpia, permanece conectado como reserva. Se rechaza un segundo codificador aunque el primero esté en reserva o aún no envíe cuadros utilizables. El cambio al vídeo espera el siguiente cuadro clave; solo audio espera un cuadro de audio utilizable. Se mantienen los tiempos de espera de fuente y transporte.

QUÉ SIGUE

- Referencia de publicación

TRANSPORTE

RTMP sobre TCP. La clave se envía como nombre de publicación; no cifra el transporte.

CÓDECS COMPATIBLES

Los mismos que en la publicación RTMP.

© Flussonic 2026

Publicación WHIP

`media.publish.whip`

El emisor se conecta a Sapsan por WHIP y publica en una entrada `publish`. Los fotogramas recibidos pasan al procesamiento multimedia común; la grabación del archivo y otros protocolos de reproducción se configuran por separado.

CONFIGURACIÓN

Configure estas opciones mediante central. Ejemplo de config del streamer:

```
listeners:
  http:
  - port: 8080
auth:
  publish_tokens:
  - PUBLISH_TOKEN
streams:
- name: studio
  inputs:
  - publish: {}
```

`listeners.webrtc` define puertos UDP compartidos por WHIP y WHEP. Si falta o está vacío, el sistema operativo selecciona un puerto libre. `port: 0` también activa la selección automática. Los candidatos ICE del SDP anuncian el puerto asignado; no se reserva un puerto por sesión. Volver a aplicar la misma lista conserva el puerto. Un reinicio puede cambiarlo.

Parámetro	Valor	Comportamiento
`listeners.webrtc`	Ausente o `[]`	Un puerto UDP automático
`listeners.webrtc[].port`	Obligatorio por entrada, `0` – `65535`	`0` selecciona el puerto automáticamente; otro valor lo fija

```
listeners:
  webrtc:
  - port: 5005
  - port: 5006
```

Para reglas fijas del firewall o reenvío detrás de NAT, configure puertos explícitos y permita el acceso de los clientes. El proxy HTTP gestiona el SDP; los medios UDP deben llegar al propio streamer. La dirección ICE se obtiene del enrutamiento IPv4 local; `listeners.webrtc` no ofrece ajustes de dirección ICE pública ni TURN. La selección automática del puerto no configura NAT.

URL DE PUBLICACIÓN

URL directa del streamer:

```
https://streamer.example.com/streaming/whip/studio
```

AUTENTICACIÓN

Envíe `Authorization: Bearer PUBLISH_TOKEN` o `?token=PUBLISH_TOKEN`. Configure el acceso con `auth.publish_tokens` o un backend de autorización. Los permisos de publicación y reproducción se configuran por separado.

TRANSPORTE

HTTP(S) gestiona el SDP y el control de la sesión. Los medios utilizan UDP con ICE y DTLS-SRTP/SRTCP. Use HTTPS en el navegador; el acceso HTTP y el acceso al puerto UDP son requisitos independientes.

Si la página del navegador y el endpoint WHIP/WHEP usan distintos orígenes, añada en el proxy HTTPS la cabecera de respuesta `Access-Control-Expose-Headers: Location` para que el cliente pueda leer la URL de sesión. No es necesario si la página y la API comparten origen.

CÓDECS COMPATIBLES

- **H.264**
- **AV1**
- **Opus**

© Flussonic 2026

El vídeo utiliza H.264 o AV1; el audio, Opus. H.264 tiene prioridad en la negociación; WHEP también depende de los códecs de origen y del reproductor. WebRTC no transcodifica: convierta previamente otros códecs de audio a Opus. HEVC y VP8/VP9 no están disponibles en este procesamiento. WHEP requiere una pista de vídeo compatible; el audio es opcional.

API HTTP

Método y ruta	Resultado
`POST /streaming/whip/{stream}`	Oferta SDP (`application/sdp`); devuelve `201 Created` , respuesta SDP y URL de sesión en `Location`
`GET /streaming/whip/{stream}`	`204 No Content` ; no devuelve vídeo ni una página de reproductor
`OPTIONS /streaming/whip/{stream}`	CORS y tipos de contenido aceptados
`DELETE /streaming/whip/session/{id}`	Cierra la sesión; devuelve `204 No Content`
`PATCH /streaming/whip/session/{id}`	Un fragmento SDP válido no vacío recibe `501 Not Implemented`

La oferta debe incluir todos los candidatos ICE del cliente; no se admite añadirlos después mediante PATCH. Guarde `Location` para cerrar la sesión. El servidor responde `recvonly` para WHIP y `sendonly` para WHEP.

GESTIÓN DE ERRORES

Código	Causa
`400`	SDP incorrecto u oferta no negociable
`403`	Acceso denegado; WHEP también rechaza contenido cifrado
`404`	El stream no está activo; WHEP también requiere información de pistas disponible
`409`	WHIP: falta la entrada `publish` o ya existe un emisor activo
`415`	Content-Type incorrecto; WHEP también requiere vídeo compatible
`503`	No se pudo registrar la ruta multimedia; un grupo de espectadores WHEP también puede estar temporalmente no disponible

Un stream acepta un emisor WHIP activo. El emisor vuelve a conectarse tras una desconexión; el respaldo de entradas determina el comportamiento posterior.

LIMITACIONES

La oferta y la respuesta iniciales contienen todos los candidatos ICE. No se admiten Trickle ICE, reinicio ICE, contraoferta WHEP ni alternativa TCP. Cambiar la lista de puertos UDP reconstruye el transporte: vuelva a conectar los clientes. Una lista sin cambios conserva los sockets.

SIGUIENTES PASOS

- Reproducción WebRTC (WHEP)
- Publicar en Agora
- Grabación del archivo
- Reproducción HLS
- Referencia de publicación

© Flussonic 2026

6.2.4 Calidad de la señal (media.probe)

Supervisión de la fuente SDI

`media.probe.sdi`

Sapsan supervisa la fuente SDI durante la captura: comprueba el estado de la señal, la secuencia de fotogramas y el código de tiempo, y contabiliza los errores notificados por el controlador de la tarjeta. Esto permite distinguir los problemas de la fuente y del cable, los cambios de estándar de vídeo y las pérdidas entre la tarjeta y el pipeline multimedia.

La captura SDI describe la conexión y la captura. Este artículo se centra en el diagnóstico de una entrada ya configurada. El análisis se realiza en el pipeline activo; no requiere un proceso independiente que abra el mismo puerto de la tarjeta.

QUÉ SE COMPRUEBA EN CADA FOTOGRAMA

El controlador entrega a Sapsan un fotograma de vídeo y los datos asociados. La entrada los contrasta con el estado de captura:

- **Estado de la línea** — si hay señal y si corresponde al formato de captura. Se comprueba el estado actual, por lo que la pérdida de una notificación de recuperación de la señal no debería dejar la entrada en su estado anterior.
- **Secuencia de fotogramas** — los saltos en los números de secuencia de los búferes de la tarjeta indican fotogramas perdidos antes de que Sapsan los reciba. El salto también se refleja en las escalas temporales del vídeo y el audio.
- **CRC de las líneas SDI** — se acumulan los errores de los metadatos del fotograma si la tarjeta y el controlador ofrecen esta comprobación.
- **Código de tiempo ATC** — con `timecode` activado, se comprueba que la hora del día sea válida y que su avance concuerde con la secuencia de fotogramas.
- **Datos auxiliares** — se contabilizan los paquetes VANC reconocidos y los errores de procesamiento de los datos admitidos. Las posibilidades de extracción se describen en la captura SDI.

Recibir búferes de la tarjeta no demuestra por sí solo la presencia de una señal útil. Tras desconectar el cable, el controlador puede seguir entregando fotogramas con una marca de error. Sapsan tiene en cuenta esa marca para que la entrada no parezca correcta solo porque aumenta el número de fotogramas.

ESTADO DE LA LÍNEA Y RECUPERACIÓN

La entrada V4L2 actual informa de tres estados:

- **`ok`** — el fotograma recibido no tiene una marca de error de línea. No es una evaluación del contenido de la imagen o del audio.
- **`no_signal`** — el controlador informa de que no hay señal.
- **`wrong_standard`** — hay señal, pero el fotograma está marcado como erróneo respecto al formato de captura actual.

Para tolerar cambios breves, la entrada espera 500 ms de estado de error continuo antes de dejar de enviar fotogramas al pipeline multimedia. Si `wrong_standard` persiste, la captura se vuelve a abrir a los 2 segundos. Un evento de cambio de señal del controlador puede provocar un reinicio anterior si el estándar se detecta automáticamente.

Si no llegan fotogramas, la captura se reinicia según `peer_timeout_ms`. Con la detección automática del formato, la entrada muestra `no_signal` mientras espera la señal; si esta no aparece dentro de ese tiempo de espera, el intento de apertura falla. Con un `standard` explícito, se omite esta fase de espera. El valor predeterminado de `peer_timeout_ms` es 30 000 ms. El mecanismo general de LSI gestiona los reintentos y el paso a la reserva; su tiempo de conmutación no equivale a la espera de 500 ms.

Los cambios de estándar y los reinicios de captura se contabilizan por separado de los fallos de la fuente. Un aumento de reinicios después de conmutar la señal en un router requiere una interpretación distinta de los reinicios por ausencia de fotogramas.

COMPROBACIÓN DE LA ESCALA TEMPORAL Y DEL CÓDIGO DE TIEMPO

La escala temporal del vídeo se construye a partir de la secuencia de fotogramas y de la frecuencia fraccionaria exacta del formato. La escala del audio tiene en cuenta el número de muestras de audio. Con `timecode` activado, la entrada analiza ATC y puede vincular ambas escalas a la hora de la fuente. La fecha se obtiene del reloj del servidor, ya que ATC contiene la hora del día.

© Flussonic 2026

Ejemplo de configuración mediante la API de central:

```
streams:
  - name: studio
    inputs:
      - sdi:
          device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
          timecode: true
          peer_timeout_ms: 30000
```

`timecode` está activado de forma predeterminada. Con `false`, la captura sigue funcionando, pero ATC no controla la escala temporal y no se realizan estas comprobaciones del código de tiempo.

La vinculación requiere un código de tiempo válido:

- **Hora del día** — la diferencia respecto al reloj del servidor no debe superar los 30 segundos. El reloj del servidor debe estar sincronizado; puede rechazarse un contador desde el inicio del material o un código de tiempo de otra zona horaria.
- **Frecuencia fraccionaria** — 29,97 y 59,94 requieren drop-frame; el código de tiempo a 23,98 no se utiliza para esta vinculación.
- **Secuencia** — el código de tiempo debe avanzar de forma coherente con los fotogramas. Una desviación aislada no cambia la escala; un nuevo desfase estable durante aproximadamente un segundo de fotogramas provoca una nueva vinculación.

Antes de la primera vinculación correcta se utiliza una escala sin ATC, que después se alinea con el reloj del servidor. Tras la vinculación, la ausencia del código de tiempo o el fallo de su comprobación conservan el desfase anterior. Una nueva vinculación notifica al pipeline multimedia una discontinuidad de la escala para que el empaquetado posterior tenga en cuenta el cambio de tiempo.

Contador del código de tiempo	Significado
`timecode_missing`	Fotogramas sin ATC después de una vinculación correcta. Antes de la vinculación, la ausencia permanente de ATC no aumenta este contador.
`timecode_rejected`	Código de tiempo no válido o incoherente: valor dañado, modo de frecuencia inadecuado, código detenido o desviación inestable.
`timecode_untrusted`	Código de tiempo fuera de la ventana de confianza respecto al reloj del servidor.
`timecode_jumps`	Nuevas vinculaciones a un desfase estable.

ESTADÍSTICAS MEDIANTE LA API

Solicite las estadísticas detalladas al streamer que captura SDI:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/streams/stats/studio \
  | jq '{name, inputs: [(.inputs // [])[] | select(.protocol == "sdi")], input}'
```

`inputs` es la lista de fuentes. Una fuente SDI ofrece la siguiente información:

Campo	Cómo interpretarlo
`current`, `status`, `failure_cause`, `error`	Si la fuente está seleccionada, su estado general y la causa del fallo.
`sdi_line_state`	Estado de la línea como cadena; el valor `ok` puede omitirse en JSON.
`sdi_producer_restarts_total`	Reinicios de captura acumulados. El nombre histórico `producer` no implica un proceso de captura independiente.
`sdi_boundary_loss_total`	Pérdidas detectadas por saltos en los números de secuencia de los búferes de la tarjeta.
`sdi_capture`	Formato de captura aceptado por el controlador, antes de la transcodificación.

`sdi_capture` contiene las dimensiones del fotograma, el FourCC numérico `pixel_format`, los bytes por línea en `stride` y la frecuencia de fotogramas completos `fps_num / fps_den`. El campo opcional `video_bytes_per_second` estima el flujo mínimo de datos de vídeo DMA en bytes por segundo, sin audio ni sobrecarga PCIe.

© Flussonic 2026

En la sección `input` del stream, `sdi_datagrams_total` cuenta los fotogramas recibidos del dispositivo. Pese a su nombre, no es el número de datagramas de red. Los campos con valor cero pueden omitirse. La ausencia de una sección de estadísticas no debe interpretarse como una medición completada sin errores.

Las estadísticas compactas de central incluyen el estado de las fuentes, pero no sustituyen la respuesta detallada del streamer. CRC y todos los contadores de código de tiempo no se presentan en la API como un único objeto `input.sdi`; utilice la exportación de métricas para consultarlos.

MÉTRICAS Y MEDICIONES TEMPORALES

La exportación de métricas del streamer está disponible en `/streamer/api-v4/live-metrics` con autenticación. La disponibilidad y el nivel de detalle dependen de la licencia de contadores.

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/live-metrics \
  | rg '^stream_input_sdi_.*stream="studio"'
```

Serie de Prometheus	Significado
`stream_input_sdi_line_state`	Estado actual: `0` — `ok`, `1` — `no_signal`, `2` — `wrong_standard`.
`stream_input_sdi_datagrams_total`	Fotogramas recibidos del dispositivo.
`stream_input_sdi_boundary_loss_total`	Fotogramas perdidos según los números de secuencia de los búferes.
`stream_input_sdi_crc_errors_total`	Errores CRC de líneas notificados por la tarjeta.
`stream_input_sdi_producer_restarts_total`	Todos los reinicios de captura.
`stream_input_sdi_producer_restarts_wrong_standard_total`	Reinicios por el estándar o un evento de cambio de señal.
`stream_input_sdi_producer_restarts_silence_total`	Reinicios por ausencia de fotogramas del dispositivo.
`stream_input_sdi_timecode_missing_total`	ATC ausente después de la vinculación.
`stream_input_sdi_timecode_rejected_total`	ATC rechazado.
`stream_input_sdi_timecode_untrusted_total`	ATC fuera de la ventana de confianza del reloj.
`stream_input_sdi_timecode_jumps_total`	Nuevas vinculaciones de la escala mediante ATC.

El vocabulario común de estados de línea también incluye `late`, `duplicate` e `invalid`; la entrada V4L2 actual no los detecta como tres defectos independientes de la señal. Para algunos contadores, la serie solo aparece después de un evento, por lo que su ausencia no confirma que la tarjeta admita la comprobación.

Además, el monitor general de la escala temporal compara la hora de llegada de los medios con DTS antes de la alineación de las marcas de tiempo. Las estadísticas del stream contienen `source_timing_offset_ms` y `source_timing_jitter_ms`, además de los límites `source_timing_offset_min_ms` / `source_timing_offset_max_ms`. `source_timing_offset_ms` muestra el desfase actual; el jitter y los límites del desfase corresponden a la ventana de un minuto completada. `source_timing_input` y `source_timing_window_started_at` indican la fuente y el inicio de la ventana.

Estas mediciones ayudan a detectar cambios de retraso e irregularidades en la llegada de los medios al pipeline de software. No miden el jitter eléctrico de SDI. Comprobar cada fotograma tampoco significa que la API exporte el historial de todas las marcas de tiempo: expone el estado y los resultados agregados.

© Flussonic 2026

CÓMO INTERPRETAR LOS RESULTADOS

Observación	Qué comprobar
`no_signal`	La fuente, el cable, el enrutamiento y la entrada seleccionada de la tarjeta.
Aumentan los reinicios por estándar	Las conmutaciones de fuente, el formato detectado automáticamente y el `standard` explícito.
Aumentan los reinicios por ausencia de fotogramas	El controlador y el funcionamiento de la captura; contraste los eventos con los logs del servidor.
Aumenta `boundary_loss`	Las pérdidas por números de secuencia de los búferes: la carga del servidor y la capacidad del pipeline para recibir a tiempo los fotogramas de la tarjeta. El contador por sí solo no identifica la causa.
Aumenta `crc_errors`	La integridad de la señal de entrada según los datos de la tarjeta. Cero no confirma la ausencia de errores si la tarjeta no proporciona CRC.
Aumenta `timecode_untrusted`	La sincronización del reloj del servidor y la finalidad del código de tiempo de la fuente.
Aumentan `timecode_rejected` o `timecode_jumps`	La configuración y estabilidad del generador ATC y las conmutaciones de fuentes.

Compare el incremento de los contadores durante un intervalo, no solo el valor acumulado. Contraste el estado de la línea con la llegada real de fotogramas: cero pérdidas en una entrada detenida no significa que la señal sea correcta.

El diagnóstico depende de los datos que proporciona el controlador. Estos indicadores no miden el jitter eléctrico de SDI, la amplitud ni el diagrama de ojo. Para esas mediciones se necesita un analizador de señal física. La supervisión de ATC tampoco comprueba si la imagen está congelada o si el audio está en silencio.

Para la captura de alta velocidad, tenga en cuenta por separado los requisitos de servidor y memoria de 12G-SDI. Evalúe el formato de captura y las pérdidas en la tarjeta antes de transcodificar: reducir la tasa de bits de salida no reduce el volumen de los fotogramas de entrada sin comprimir.

© Flussonic 2026

TR 101 290

`media.probe.tr101290`

Las tablas relacionan los identificadores de ETSI TR 101 290 V1.3.1, §5.2 con los contadores de Sapsan. Los contadores MPEG-TS ampliados no implican que se implementen todas las comprobaciones del estándar. Los campos relacionados se indican explícitamente y no equivalen a los indicadores ETSI.

PRIORIDAD 1

ID	ETSI	Contadores Sapsan	Cobertura
1.1	`TS_sync_loss`	—	No hay un contador específico.
1.2	`Sync_byte_error`	—	No hay un contador específico.
1.3	`PAT_error`	`errors_ts_pat`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
1.3.a	`PAT_error_2`	`errors_ts_pat`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
1.4	`Continuity_count_error`	`errors_ts_cc`	Infracciones del continuity counter por PID.
1.5	`PMT_error`	`errors_ts_pmt`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
1.5.a	`PMT_error_2`	`errors_ts_pmt`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
1.6	`PID_error`	`errors_pid_lost`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.

© Flussonic 2026

PRIORIDAD 2

ID	ETSI	Contadores Sapsan	Cobertura
2.1	`Transport_error`	`errors_ts_tei`	Paquetes con TEI por PID.
2.2	`CRC_error`	`errors_ts_psi_checksum`	Errores CRC de las tablas PSI procesadas.
2.3	`PCR_error`	`pcr_resync`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
2.3a	`PCR_repetition_error`	—	No hay un contador específico.
2.3b	`PCR_discontinuity_indicator_error`	`pcr_resync`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
2.4	`PCR_accuracy_error`	`pcr_jitter_fail_500us`, `pcr_jitter_fail`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
2.5	`PTS_error`	—	No hay un contador específico.
2.6	`CAT_error`	—	No hay un contador específico.

© Flussonic 2026

PRIORIDAD 3

ID	ETSI	Contadores Sapsan	Cobertura
3.1	`NIT_error`	—	No hay un contador específico.
3.1.a	`NIT_actual_error`	—	No hay un contador específico.
3.1.b	`NIT_other_error`	—	No hay un contador específico.
3.2	`SI_repetition_error`	—	No hay un contador específico.
3.3	`Buffer_error`	`hrd_buffer_min`, `hrd_buffer_max`, `hrd_buffer_underrun`	Diagnóstico relacionado; no es una comprobación completa de TR 101 290.
3.4	`Unreferenced_PID`	—	No hay un contador específico.
3.4.a	`Unreferenced_PID`	—	No hay un contador específico.
3.5	`SDT_error`	—	No hay un contador específico.
3.5.a	`SDT_actual_error`	—	No hay un contador específico.
3.5.b	`SDT_other_error`	—	No hay un contador específico.
3.6	`EIT_error`	—	No hay un contador específico.
3.6.a	`EIT_actual_error`	—	No hay un contador específico.
3.6.b	`EIT_other_error`	—	No hay un contador específico.
3.6.c	`EIT_PF_error`	—	No hay un contador específico.
3.7	`RST_error`	—	No hay un contador específico.
3.8	`TDT_error`	—	No hay un contador específico.
3.9	`Empty_buffer_error`	—	No hay un contador específico.
3.10	`Data_delay_error`	—	No hay un contador específico.

PCR / HRD

Histograma PCR por PID: `pcr_jitter_ok_0` es cero; `pcr_jitter_ok_500ns` hasta 500 ns; `pcr_jitter_fail_500us` más de 500 ns hasta 500 µs; `pcr_jitter_fail` más de 500 µs. `pcr_resync` cuenta resincronizaciones del reloj, no todas las condiciones de 2.3b. HRD muestra los valores mínimo y máximo en bits y vaciados del búfer modelo; no modela todos los búferes T-STD de 3.3.

© Flussonic 2026

CONTADORES FUERA DE TR 101 290

Contadores	Significado
`errors_ts_scrambled`	Paquetes cifrados; no comprueba la presencia de CAT (2.6).
`errors_adaptation_broken`, `broken_pes_count`, `broken_pes_sum`	Adaptation fields dañados, número y tamaño de paquetes PES dañados.
`errors_ts_service_lost`	Desaparición del servicio seleccionado de la señalización del transporte.
`empty_packets`, `packets`, `frames`, `bytes`, `sync_bytes`	Recuento de paquetes, frames y bytes, incluidos paquetes vacíos y bytes de búsqueda de sincronización. No son los indicadores 1.1/1.2.
`dts_goes_backwards`, `dts_jump_forward`, `too_large_dts_jump`, `time_corrections`, `corrected_backward_pts`	Infracciones y correcciones de la línea temporal; no comprueban la repetición de PTS (2.5).
`discarded_buffer_count`, `discarded_buffer_sum`, `fillers_count`, `fillers_sum`	Número y tamaño de búferes descartados y rellenos insertados.
`unpaired_fields`, `warning_invalid_sibling_frames`, `padding_pes_count`, `padding_pes_sum`, `orphan_pes_count`	Campos sin pareja, frames asociados no válidos, PES de relleno y PES huérfanos.
`recv_buffer_granted_bytes`, `recv_buffer_requested_bytes`, `recv_buffer_required_bytes`, `recv_buffer_drops`, `low_recv_buffer_secs`	Tamaños del búfer del socket UDP, descartes del kernel y segundos con margen de búfer insuficiente.

© Flussonic 2026

USO Y LÍMITES

Compare los incrementos de errores por PID en intervalos iguales. Un contador a cero no valida una comprobación sin campo específico. El nivel RF y BER requieren mediciones del hardware receptor. Las series detalladas dependen de la licencia; consulte monitorización.

© Flussonic 2026

SRT

`media.probe.srt`

© Flussonic 2026

CONTADORES

Las tablas muestran los nombres de contadores ampliados. El acceso a las series detalladas depende de la licencia; consulte monitorización.

© Flussonic 2026

Contador	Sig
`incoming_bytes`, `outgoing_bytes`, `incoming_packets`, `outgoing_packets`	By po inc rec Sa ou en
`incoming_lost_packets`, `outgoing_lost_packets`, `missing_packets`	Pé de dir pa fal
`incoming_retransmitted_packets`, `outgoing_retransmitted_packets`, `incoming_recovered_packets`	To ret po po pa rec pé ret pu du tar
`incoming_ack_packets`, `outgoing_ack_packets`, `incoming_ackack_packets`, `outgoing_ackack_packets`, `incoming_nak_packets`, `outgoing_nak_packets`	Pa co AC po NA so ret em en de nu pa rec
`incoming_drop_request`, `incoming_drop_requests`, `outgoing_drop_requests`	Pa de pe em me pe de am dir
`outgoing_drop_too_late_packets`, `incoming_drop_late_seq`, `incoming_drop_late_time`, `incoming_drop_far_time`	De en rec se tie vál
`incoming_drop_too_late`, `incoming_drop_buffer_full`, `loss_tracking_overflows`	Pé om

© Flussonic 2026

Contador	Sig
	ver de des bú des del de
`keepalive_timeouts`	Tir ke co
`receive_buffer_size`, `send_buffer_size`	Oc de rec en val est co ac
`rtt_us`, `rtt_var_us`, `source_clock_drift_us`	RT RT ac rel de co mi
`memory_bytes`	Me tra by

MPEG-TS / TR 101 290

SRT transporta MPEG-TS: se incluyen todos los contadores MPEG-TS, incluidas las comprobaciones TR 101 290. Las pérdidas recuperadas por SRT pueden no generar errores CC; el aumento de pérdidas no recuperadas junto con `errors_ts_cc` indica pérdida de medios tras fallar la recuperación.

SIGUIENTES PASOS

- SRT ingest
- SRT publish
- Counters

© Flussonic 2026

RTSP

`media.probe.rtsp`

CONTADORES

Las tablas muestran los nombres de contadores ampliados. El acceso a las series detalladas depende de la licencia; consulte monitorización.

Los contadores RTP se registran por separado para cada canal de vídeo o audio.

Contador	Significado
`channel_id`, `content`	Identificador del canal RTP y tipo de contenido.
`rtp_packets`, `rtcp_packets`, `bytes`, `frames`	Paquetes RTP/RTCP recibidos, bytes y frames ensamblados.
`pt_reject_count`, `pt_reject_bytes`	Paquetes y bytes rechazados por payload type inesperado.
`ts_goes_backwards`, `ts_jump_forward`, `ts_stuck`	Timestamps RTP que retroceden, saltan hacia delante o quedan fijos.
`errors_dts_stuck`, `sr_ts_stuck`, `ts_stuck_restarts`, `resync_sr`	DTS fijo, tiempo fijo de Sender Report, reinicios por timestamps fijos y resincronizaciones con SR.
`marker_packets_count`, `no_marker_mode_flag`	Paquetes con marker e indicador del modo sin marker.
`errors_broken_payload`, `errors_incomplete_frames`	Payloads dañados y frames que no se pudieron ensamblar por completo.
`errors_lost_packets`, `errors_dropped_packets`, `errors_late_packets`	Las pérdidas de red, descartes locales y paquetes tardíos se cuentan por separado.
`nack_count`	Solicitudes NACK de retransmisión enviadas.
`onvif_buffer_overflow`, `onvif_seq_gap`, `onvif_marker_mismatch`, `onvif_resync_dropped`	Desbordamientos del búfer ONVIF, huecos de secuencia, discrepancias de marker y descartes por resincronización.

Los totales de la conexión RTSP también incluyen:

Contadores	Significado
`bytes`, `reads`, `frames`	Lecturas del socket, bytes recibidos y frames emitidos
`errors`, `warnings`, `broken_interleaved`, `desync`	Errores, advertencias, bloques interleaved dañados y desincronización del parser
`memory_bytes`	Memoria de captura en bytes

SIGUIENTES PASOS

- RTSP ingest
- Counters

© Flussonic 2026

Contadores detallados

`media.probe.counters`

CONTADORES EXTENDIDOS

La opción de contadores extendidos abre el detalle causal y de protocolo en todas partes a la vez: en las secciones `input` y `dvr.write` del Admin API, en las series del servidor Prometheus integrado y en el scrape de Prometheus. A la telemetría de Retroview estos contadores van siempre, independientemente de la opción — por eso el análisis de causas está disponible en Retroview incluso sin ella.

Causas de los errores de entrada

El `errors` agregado responde «en la entrada hay un problema»; el desglose causal responde «cuál exactamente». Las causas vienen en dos familias:

- **defectos de medios de un stream vivo** — `lost_packets`, `broken_payload`, `desync`, `ts_pat`: los datos llegan, pero dañados;
- **fallos de la fuente** — `connection_closed`, `connection_refused`, `timeout`, `http_request_error`, `not_found`, `denied`, `decode_error`, `protocol_error`, `io_error`, `other`: la fuente murió y la causa de muerte está clasificada.

La clasificación sale de la categoría tipada del error y no del texto del log, y cubre también los fallos al abrir una fuente. Cada fallo entra además en el `errors` agregado; los fallos repetidos en los reintentos se cuentan cada vez — la frecuencia de eventos es exactamente la señal de que «la fuente sigue muerta». Las paradas normales — reconfiguración, fin del stream, desalojo por una fuente de mayor prioridad — no se cuentan como errores.

La serie del desglose

En el servidor Prometheus integrado el desglose viaja como una serie con la etiqueta de causa — `errors_detail_total{name, cause}`; en central las series llevan también `node`. Las causas sin eventos no crean series.

```
# errores del stream en 5 minutos, desglosados por causa
sum by (cause) (increase(errors_detail_total{name="tv1"}[5m]))
```

MPEG-TS por cada PID

Por cada PID del flujo de transporte: `errors_ts_cc` (violaciones del continuity counter — el indicador principal de pérdidas de transporte), `errors_ts_tei`, `errors_ts_scrambled` (no se retiró el cifrado — problemas de CAM/llaves), `errors_ts_psi_checksum`, PES rotos, buckets de jitter del PCR, vaciamientos del búfer del decodificador (HRD).

Por qué: este es material de monitorización de nivel broadcast. Una alerta sobre `rate(errors_ts_cc) > 0` por cada PID caza la degradación del transporte antes de que la vea el espectador; el jitter del PCR y el HRD muestran si el stream sobrevivirá a un decodificador de hardware.

SRT

RTT y su variación, pérdidas y retransmisiones en ambas direcciones, descartes por llegar tarde, llenado de los búferes, tiempos de espera de keepalive.

Por qué: ajustar `latency` al canal real y diagnosticar «quién pierde — nosotros o el lado remoto». Las retransmisiones crecen con el RTT estable — pérdidas en el camino; el RTT salta — el canal está congestionado.

RTP/RTSP por canal

Paquetes perdidos, saltos y bloqueos de los timestamps, NACK, desbordamientos de búfer — por cada canal RTP (video/audio) de una cámara.

Por qué: diagnóstico del parque de cámaras — qué cámara está fallando, antes de que lleguen las quejas por artefactos.

Captura SDI

Los contadores de captura V4L2 utilizan el prefijo `stream_input_sdi_`. El nivel básico incluye:

- **Estado de la línea** — `sdi_line_state`.
- **Reinicios de captura** — `sdi_producer_restarts_total`.
- **Fotogramas perdidos desde la tarjeta** — `sdi_boundary_loss_total`, según los huecos en la secuencia de búferes.

© Flussonic 2026

Los contadores ampliados añaden:

- **Causas de reinicio** — cambio de estándar o ausencia de fotogramas.
- **Tipos y modos ST 337 no admitidos** — `sdi_unsupported_codec_total`.
- **Datos VANC desconocidos** — `sdi_vanc_unknown_total`.
- **Datos CDP** — número de paquetes y pares `cc_data`.
- **Código de tiempo** — valores aceptados, rechazados y ausentes, y cambios de la línea temporal.

Los nombres con `producer` se conservan por compatibilidad y corresponden a la captura V4L2. `sdi_boundary_loss_total` indica pérdidas según los números de búfer; un salto PTS por sí solo no demuestra la pérdida de un fotograma. Consulte Captura SDI.

Rendimiento de escritura del DVR

`segments_written_slow`, `segments_written_delayed`, `segments_written_collapsed`, `segments_discontinuity`.

Por qué: una señal temprana de «el disco no da abasto». La proporción de slow/delayed crece con el mismo tráfico — el almacenamiento se está degradando; `discontinuity` significa huecos que luego verá en `recorded_hours`.

CONTADORES DE CALIDAD

Lo que hay disponible para alertar:

- **HLS MPEG-TS para decodificadores**: segmentos `ts/mux/` incompletos y ensamblajes fuera de la caché. Véase Entrega de HLS MPEG-TS.
- **Captura MPEG-TS**: por cada PID — errores CC, TEI, paquetes cifrados, errores CRC de PSI, PES rotos, estado del búfer del decodificador (HRD).
- **SRT**: RTT, pérdidas y retransmisiones en ambas direcciones, tamaños de búfer, tiempos de espera de keepalive.
- **Fuentes (LSI)**: tiempo en la entrada principal/reserva, tiempo sin datos, reintentos, validez de las reservas, divergencia de parámetros entre las entradas.
- **Sesiones**: abiertas, rechazadas, autorizadas, denegaciones de reautorización.
- **DVR**: espacio en los discos, contadores de operaciones, número de blobs, bytes ocupados por el archivo, progreso de la reconstrucción del catálogo.
- **Pushes**: por cada push — estado con la causa del fallo, velocidad de envío, bytes, errores, reconexiones, retransmisiones SRT/RIST. Véase Estadísticas de pushes.
- **Agentes de Peeklio**: estado de las conexiones, bytes, errores.

ENTREGA DE HLS MPEG-TS PARA DECODIFICADORES

La entrega mixta `ts/mux/` ensambla cada segmento a partir del vídeo y el audio, y en el borde en vivo el audio puede no llegar a tiempo. Dos contadores del stream muestran lo que el espectador recibió de verdad:

Serie de Prometheus	Qué cuenta	Cómo leerla
`stream_mpegts_hls_incomplete_segments_total`	segmentos de la ventana en vivo entregados sin parte del audio: el audio se retrasó, desapareció en la entrada, está declarado pero no llega, o el inicio de su ventana ya salió de la ventana en vivo	crecimiento constante — el audio de la entrada va por detrás del vídeo o no existe; una unidad por segmento de variante, así que un stream con tres vídeos da +3 por un trozo de audio retrasado
`stream_mpegts_hls_uncached_segments_total`	segmentos ensamblados fuera de la caché: el conjunto de pistas no está entre las variantes del maestro en vivo (una ventana de archivo sobre datos en vivo, un conjunto anterior al cambio de la descripción del stream, un subconjunto que el cliente pide por su cuenta)	crecimiento constante — los decodificadores reproducen lo que la caché no guarda, y cada ensamblaje cuesta procesador

Ambas series son de nivel básico: en el servidor Prometheus integrado las tienen todos, en el scrape del nodo — con la opción de contadores extendidos. En la Admin API llegan como tasas en la sección `mpegts_hls` de las estadísticas del stream (`incomplete_segments_rate`,

© Flussonic 2026

`uncached_segments_rate`); la sección está presente desde que el stream tuvo un segmento así. A la telemetría de Retroview llega el incremento por ventana.

```
# streams cuyos decodificadores reciben segmentos sin audio
rate(stream_mpegts_hls_incomplete_segments_total[5m]) > 0
```

El contador no distingue «el audio llegó tarde» de «no hay audio en la entrada»: mire la entrada del stream.

© Flussonic 2026

Validador de streams

`media.probe.stream_validator`

La validación de streams es un mecanismo que permite determinar si un stream HLS o MPEG-DASH es adecuado para su procesamiento mediante la comprobación de su integridad y de la corrección de su empaquetado. Comprueba las playlists o los manifiestos y los segmentos de medios asociados, sin decodificar el stream ni validar la conformidad a nivel de códec.

El paquete `stream-validator` proporciona acceso a este mecanismo mediante la línea de comandos. Funciona independientemente del servidor y no requiere licencia. Sus mensajes están en inglés.

INSTALACIÓN

Se admiten Ubuntu 22.04 o posterior y Debian 12 o posterior en amd64 o arm64. Ejecute los comandos como root; deben estar instalados `curl` y los certificados CA:

```
curl -fsSL https://apt.flussonic.com/repo/master/dev.key -o /etc/apt/trusted.gpg.d/flussonic-dev.gpg
echo 'deb http://apt.flussonic.com/branch/sapsan/master repo/' > /etc/apt/sources.list.d/stream-validator.list
apt-get update
apt-get install stream-validator
```

El paquete instala ambos comandos y la guía en inglés en `/usr/share/doc/stream-validator/README.md`.

DOCKER

La imagen `flussonic/stream-validator` admite amd64 y arm64. El punto de entrada predeterminado ejecuta la validación HLS. Para DASH, seleccione `dash_validator` con `--entrypoint`.

```
docker run --rm flussonic/stream-validator:latest \
  http://server:5080/streaming/v/tv1/index.m3u8 --capture 30
```

```
docker run --rm --entrypoint dash_validator flussonic/stream-validator:latest \
  http://server:5080/streaming/v/tv1/Manifest.mpd --capture 30
```

En los ejemplos, `server` es el nombre de host o la dirección IP de su streamer. Dentro de un contenedor, `localhost` se refiere al propio contenedor. En Docker Desktop, use `host.docker.internal` para acceder a un servicio del host.

HLS

La validación HLS captura una emisión HLS en directo, actualiza las playlists y descarga segmentos de inicialización y medios de forma predeterminada. Comprueba la latencia LL-HLS, la disponibilidad de partes y el intervalo de partes independientes relacionado con el GOP. La estimación basada en PART-HOLD-BACK no mide la latencia de la cámara a la pantalla.

Ejecución

```
hls_validator --capture 30 --target-latency 2 http://server:5080/streaming/v/tv1/index.m3u8
```

Opciones

Opción	Función	Predeterminado
URL, `-u`, `--url`	URL de HLS; también se acepta como argumento posicional.	—
`-t`, `--capture`	Duración de la captura en directo.	20
`--request-timeout`	Tiempo de espera de cada solicitud HTTP.	10
`--target-latency`	Objetivo de latencia para el diagnóstico.	2
`--preload-timeout`	Plazo de sondeo de la parte PRELOAD-HINT.	2.5
`--playlists-only`	Comprueba playlists sin descargar medios ni sondear partes/preload en tiempo real.	desactivado
`-h`, `--help`	Muestra la ayuda.	—

`--timeout` sigue siendo un alias de `--capture`. El comando interno anterior se llamaba `validator`.

© Flussonic 2026

Códigos de diagnóstico

- `LL-DISABLED` — la baja latencia está desactivada.
- `TARGETDURATION-INFLATED` — TARGETDURATION está sobredimensionado.
- `PART-TARGET-LARGE` — PART-TARGET es demasiado grande.
- `SPARSE-INDEPENDENT-PARTS` — hay pocas partes independientes debido a un GOP largo.

MPEG-DASH

La validación DASH comprueba el MPD y descarga segmentos de inicialización y medios. Detecta desalineaciones de pistas ABR, errores temporales y deriva del borde en directo entre audio y vídeo.

Modos

- **auto** — Automático: determina el modo según `MPD@type`.
- **--static** — Comprueba una vez una presentación VOD estática.
- **--live** — Actualiza un MPD dinámico durante la captura y comprueba huecos y solapamientos entre actualizaciones de SegmentTimeline.

Ejecución

```
dash_validator --live --capture 30 http://server:5080/streaming/v/tv1/Manifest.mpd
```

Opciones

Opción	Función	Predeterminado
`-u`, `--url`	URL del MPD; también se acepta como primer argumento posicional.	—
`-t`, `--capture`	Duración de la captura en directo; no se aplica al modo estático.	20
`-p`, `--poll`	Intervalo de consulta del MPD en directo; se limita al rango interno permitido.	minimumUpdatePeriod o 2 s
`--request-timeout`	Tiempo de espera HTTP en todos los modos de validación y durante la detección de modo.	10
`--static`, `--live`	Fuerza el modo en lugar de detectarlo automáticamente.	automático
`-h`, `--help`	Muestra la ayuda.	—

`--timeout` es un alias de `--capture`, no un tiempo de espera HTTP. Al migrar desde la utilidad interna `dash-validate`, sustituya su opción HTTP `--timeout` por `--request-timeout`.

RESULTADO

El informe muestra errores y advertencias con su ubicación. Códigos de salida:

- `0` — validación correcta, incluso con solo advertencias.
- `1` — errores de validación o inicio.
- `2` — argumentos CLI incorrectos.

Las duraciones se expresan en segundos y admiten números fraccionarios positivos.

QUÉ SIGUE

Para configurar el servidor, consulte:

- **Reproducción HLS**
- **LL-HLS**
- **Reproducción DASH**

© Flussonic 2026

6.2.5 Playout (media.playout)

Playlist

`media.playout.playlist`

Una lista del servidor crea una emisión lineal con archivos o streams en directo. En FMS, la fuente es `playlist:///path/playlist.txt`; las líneas contienen rutas VOD o nombres de streams.

`#EXTINF:60` limita un elemento a 60 segundos; `#EXT-X-UTC` define el inicio en Unix UTC y `#EXT-X-PROGRAM-DATE-TIME` en ISO 8601. La lista se relee después de cada archivo. `#EXT-X-MEDIA-SEQUENCE` permite continuar con el número siguiente al último reproducido.

© Flussonic 2026

6.2.6 Gestión de fuentes (media.source)

Vinculación con la hora real

`media.source.realtime_aligner`

Para un live stream, no basta con conocer el orden de los fotogramas y la velocidad de reproducción. El streamer necesita saber a qué fecha y hora corresponde cada fotograma: así puede grabarlo en DVR y encontrar después el momento necesario por fecha y hora.

Las fuentes ofrecen distintas bases para esta decisión. Algunas solo tienen una escala relativa; otras disponen de relojes propios que no siempre son fiables. La red puede entregar en ráfagas un flujo uniforme en tiempo multimedia. Sapsan vincula estas escalas con la hora real, conservando los intervalos entre fotogramas, y revisa la vinculación cuando cambia el comportamiento de la fuente.

TIEMPO MULTIMEDIA, RELOJ DE LA FUENTE Y HORA DE LLEGADA

Un fotograma tiene marcas de tiempo: DTS indica el momento de decodificación y PTS el de presentación. Determinan su posición en la escala multimedia, pero no siempre contienen una fecha y hora del día.

Fuente de tiempo	Qué sabe el streamer	Tarea
DTS/PTS relativos, por ejemplo en MPEG-TS	Orden de los fotogramas e intervalos entre ellos	Situar esta escala en el calendario del streamer.
Marcas de calendario, por ejemplo tras vincular RTP mediante un RTCP Sender Report	Hora absoluta según el reloj del emisor	Determinar si ese reloj es fiable.
Hora real de llegada	Cuándo estuvo disponible el fotograma para el pipeline del servidor	Tener en cuenta los retrasos de red y la entrega en ráfagas sin distorsionar la escala multimedia.

Esta información puede aparecer en distintos momentos dentro de un mismo stream. Por ejemplo, una entrada RTSP recibe primero fotogramas con marcas RTP relativas y después obtiene de RTCP la correspondencia con la hora absoluta. Reconectar la cámara o cambiar de fuente puede volver a modificar la base de la escala.

MPEG-TS SIN FECHA Y HORA

Las marcas de tiempo MPEG-TS permiten reproducir vídeo y audio al ritmo correcto, pero no indican que un fotograma corresponde, por ejemplo, a las 14:00:00 de hoy. El contador puede comenzar en un valor arbitrario y dar la vuelta al alcanzar el límite de su rango.

Sapsan elige un desfase entre la escala multimedia y el reloj del streamer. Después lo aplica a DTS y PTS, conservando las distancias entre marcas:

```
calendar_time = source_time + offset
```

Por ejemplo, un fotograma con DTS relativo de 42,000 s se vincula a las 14:00:00. El siguiente fotograma, con DTS de 42,040 s, recibe la hora 14:00:00,040. Su tiempo no se sustituye por el momento de la siguiente lectura del socket.

Así, el archivo obtiene una escala de calendario basada en la llegada del stream a Sapsan. Para una fuente sin marcas absolutas, es una vinculación del servidor: no permite recuperar la hora original de captura ni el retraso hasta el streamer, que se desconocen. Con el búfer de entrada activado, la vinculación se realiza después de que este entregue los fotogramas, por lo que su retraso también influye en la posición de la escala.

RTSP Y WEBRTC: LA HORA ABSOLUTA NECESITA VALIDACIÓN

Las entradas basadas en RTP, incluidas RTSP y WebRTC, pueden obtener la hora absoluta mediante RTCP Sender Report. Pero una marca de calendario no garantiza un reloj correcto: la cámara puede haber perdido su configuración horaria, funcionar sin sincronización o indicar una hora muy alejada en el pasado o el futuro.

En el trayecto normal de alineación, Sapsan distingue las marcas de calendario de las relativas. Se considera de calendario una escala de tiempo Unix no anterior al año 2000. Si una marca de este tipo difiere del reloj del streamer en no más de **60 segundos**, se conserva: el intervalo de entrega no se convierte en un motivo para reasignar la hora de la fuente.

© Flussonic 2026

Si la diferencia supera la ventana de confianza, la marca se trata como una escala que requiere vinculación al servidor. La comprobación es simétrica para el pasado y el futuro. Esto evita grabar la emisión actual con un calendario de cámara claramente incorrecto.

La ventana de confianza no garantiza una sincronización precisa: los errores de reloj dentro de ella pueden mantenerse. Sigue siendo necesario sincronizar la hora de la fuente y del servidor para que la grabación del archivo corresponda con precisión al evento real.

La decisión también se aplica a pistas individuales. Una pista con una hora de calendario fiable no recibe de nuevo el desfase elegido para una escala relativa: de lo contrario, el servidor podría sumar la época Unix a una marca que ya es absoluta.

ENTREGA EN RÁFAGAS: HLS → HTTP MPEG-TS

Un servidor intermedio puede recibir varios segundos de HLS, analizar el segmento y enviar su contenido por HTTP MPEG-TS casi en una sola ráfaga. Después hay una pausa hasta el siguiente segmento. El tiempo multimedia avanza de forma uniforme dentro de cada ráfaga, mientras que la hora de llegada de los fotogramas varía bruscamente.

Si se asignara a cada fotograma la hora de su llegada, varios segundos de vídeo se comprimirían en una breve ráfaga de red y la pausa entre segmentos se convertiría en un hueco en la escala. Cambiar continuamente el desfase con cada fluctuación produciría discontinuidades y nuevas vinculaciones repetidas.

Sapsan procesa esta entrada por etapas:

1. El búfer de entrada acumula fotogramas y los entrega al ritmo de DTS. La temporización del flujo determina los intervalos de entrega, mientras la cola absorbe las ráfagas de red.
2. El alineador vincula la escala multimedia resultante con la fecha y hora. Mantiene el desfase elegido mientras la diferencia siga siendo aceptable.
3. Los componentes posteriores reciben marcas vinculadas para transcodificar, empaquetar y grabar.

En una entrada `tshttp`, la profundidad predeterminada del búfer es de **30 000 ms**. La configuración se envía mediante la API de central:

```
streams:
  - name: tv1
    inputs:
      - tshttp:
          url: http://origin.example/live/tv1.ts
        source_options:
          buffer_ms: 30000
```

`buffer_ms` se establece en `source_options` de la entrada concreta. Es una profundidad objetivo en tiempo multimedia, no un número de bytes. Un `0` explícito desactiva el búfer; un valor positivo también lo activa para otros tipos de entrada. En los demás tipos está desactivado de forma predeterminada. Aumentar la profundidad permite cubrir pausas más largas a costa de latencia y memoria.

El búfer no puede compensar una parada indefinida de la fuente. Cuando se agotan los fotogramas acumulados, la entrega se detiene. Si se acumula demasiado material, puede descartarse contenido antiguo para reducir el retraso respecto al directo; estos eventos están disponibles en las estadísticas.

CUÁNDO CAMBIA LA VINCULACIÓN

Para una escala que requiere un desfase del servidor, el alineador comprueba la diferencia entre el reloj actual y el DTS ya desplazado. Si el valor absoluto de la diferencia se hace **mayor de 5 segundos**, se elige un nuevo desfase. Esto permite reaccionar a reinicios de marcas, diferencias de ritmo acumuladas y cambios de escala tras una reconexión.

La decisión se basa en una pista principal de audio o vídeo. El vídeo tiene prioridad sobre el audio. Las demás pistas con tiempo relativo reciben el desfase común: sus diferencias no deben hacer que reconstruyan por turnos la escala de todo el stream. Si la pista principal no llega durante más de 5 segundos, otra pista de audio o vídeo que siga llegando puede asumir ese papel.

El desfase se aplica tanto a DTS como a PTS. El alineador conserva su posición relativa; no corrige una desincronización de audio y vídeo ya existente en la fuente.

Una nueva vinculación envía al pipeline multimedia un evento de cambio de escala. El empaquetado posterior puede cerrar los fragmentos actuales e iniciar otros. La nueva vinculación puede desplazar el tiempo hacia delante o hacia atrás: el mecanismo restablece la correspondencia con la hora real, sin prometer ocultar cualquier salto de la fuente sin discontinuidad.

Las fuentes con una escala de calendario ya preparada utilizan un modo independiente de conservación del tiempo. Las entradas directas HLS y DASH pasan sin esta corrección; su retraso normal de varios segmentos no provoca nuevas vinculaciones constantes. Por eso, la captura directa de

© Flussonic 2026

HLS y la captura de HTTP MPEG-TS generado a partir de HLS por un servidor intermedio requieren comportamientos distintos. La conservación del tiempo también se aplica a los trayectos internos que transportan fotogramas ya vinculados, para no volver a modificar la escala en cada etapa.

La alineación está integrada en el pipeline en directo. Los umbrales de 5 y 60 segundos describen la implementación actual; la configuración del stream no ofrece parámetros YAML independientes para ellos.

QUÉ SUPERVISAR

Solicite estadísticas al streamer que captura la fuente:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/streams/stats/tv1 \
  | jq '{name, dts_source, source_timing_offset_ms,
         source_timing_offset_min_ms, source_timing_offset_max_ms,
         source_timing_jitter_ms, source_timing_input,
         source_timing_window_started_at, input}'
```

El campo `dts_source` indica la base de la escala:

- **`realtime`** — el tiempo se mantiene mediante un desfase respecto al reloj del streamer.
- **`input`** — se conservan las marcas de la fuente: han superado la validación de hora de calendario o la entrada utiliza un modo de conservación de la escala.

El campo puede no aparecer antes de recibir fotogramas y después de un silencio prolongado. En la exportación de métricas, `stream_dts_source` utiliza `1` para `realtime` y `2` para `input`.

`source_timing_offset_ms` muestra el desfase actual medido por el monitor temporal. Los límites `source_timing_offset_min_ms` / `source_timing_offset_max_ms` y `source_timing_jitter_ms` corresponden a la ventana de un minuto completada; `source_timing_input` y `source_timing_window_started_at` indican la fuente y el inicio de la ventana. Para una escala relativa, el monitor elige su propia referencia, por lo que estos valores no miden el retraso desconocido entre la cámara y el streamer.

El monitor está antes del alineador, pero después del búfer de entrada. Con el búfer activado, sus medidas ya reflejan una entrega suavizada. Para evaluar la irregularidad original, compárelas con el estado del búfer y sus vaciados, no solo con una gráfica de jitter.

La sección `input` proporciona indicadores del búfer:

Campo	Significado
`buffer_target_ms`	Profundidad objetivo en milisegundos de tiempo multimedia.
`buffer_depth_ms`	Profundidad actual.
`buffer_underruns`	Número acumulado de episodios de agotamiento del búfer.
`buffer_skips`	Número acumulado de reducciones de la cola por desbordamiento.
`buffer_skipped_ms`	Tiempo multimedia total descartado, en milisegundos.

Los campos con valor cero pueden omitirse en la respuesta. Compare el incremento de los contadores durante un intervalo; la ausencia de toda la sección de estadísticas no significa que se haya realizado la comprobación sin errores.

Conviene comprobar tres escenarios durante la operación:

- **Flujo relativo uniforme** — la escala del archivo sigue el reloj del streamer y se conservan los intervalos entre fotogramas.
- **Fuente con un reloj incorrecto** — un calendario claramente no fiable se sustituye por una vinculación al servidor; al recuperarse el reloj, la fuente puede volver a proporcionar su propia hora.
- **Flujo que llega en ráfagas** — el búfer suaviza la entrega; las pausas largas y los desbordamientos se ven en las estadísticas y no se ocultan reasignando constantemente el tiempo de cada fotograma.

© Flussonic 2026

Conmutación de fuentes

`media.source.failover`

Un stream puede tener varias fuentes. Sapsan vigila la fuente activa, conmuta automáticamente a la siguiente cuando esta se degrada y vuelve a la principal cuando se recupera. Esta lógica se llama LSI (Live Source Input).

VARIAS ENTRADAS

Las entradas se enumeran en orden de prioridad — la primera es la principal:

```
streams:
  - name: tv1
    inputs:
    - udp:
        host: 239.0.0.1
        port: 5000
    - tshttp:
        url: http://backup-origin.example.com/tv1/mpegts
```

CÓMO FUNCIONA LA CONMUTACIÓN

Propiedades clave del algoritmo, importantes para el aire:

- **Una fuente se vuelve activa solo cuando está lista**: debe entregar los parámetros del stream (MediaInfo) y un punto inicial adecuado (fotograma clave, cuadro de audio o límite de fragmento). Hasta entonces los espectadores siguen recibiendo la fuente actual.
- **Retorno tras una ventana limpia**: una entrada recuperada de mayor prioridad se comprueba en segundo plano. Para entradas que abre Sapsan, debe alcanzar un punto inicial adecuado y pasar `return_alive_secs` sin infringir reglas antes de conmutar.
- Una entrada caída se reconecta con demoras crecientes (`reconnect_initial_delay` → `reconnect_delay_step` → `reconnect_max_delay`); cuando se agotan todas las entradas, la rotación vuelve a empezar desde la primera.
- Una fuente cuyo bitrate supera `max_bitrate` se detiene forzosamente.
- Tiempos de espera separados para la ausencia de vídeo y de audio: el audio perdido con vídeo vivo también cuenta como degradación.

POLÍTICA DE CONMUTACIÓN

La sección `lsi` fija la política para todo el stream. `source_options` permite redefinir los tiempos de espera, límites de bitrate y ventana limpia de una entrada; las demoras de reconexión y el intervalo de comprobación en segundo plano se definen solo en `lsi` :

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    - udp:
        host: 239.0.0.2
        port: 5000
      source_options:
        source_timeout: 10
    lsi:
      source_timeout: 5
      video_timeout: 3
      audio_timeout: 3
```


© Flussonic 2026

Los tiempos de espera y el intervalo de comprobación se expresan en segundos; las demoras de reconexión, en milisegundos. Parámetros principales:

Parámetro	Descripción
`source_timeout`, `video_timeout`, `audio_timeout`	tiempos de espera por ausencia de datos/vídeo/audio
`max_bitrate`	límite de bitrate de la fuente
`reconnect_initial_delay`, `reconnect_delay_step`, `reconnect_max_delay`	política de reconexión (solo `lsi`), milisegundos
`recheck_secondary_inputs_interval`	intervalo de comprobación de entradas inactivas (solo `lsi`), segundos
`allow_if`, `deny_if` (solo en `source_options`)	habilitar/deshabilitar una entrada mediante un archivo de bandera externo

`allow_if` / `deny_if` permiten controlar las entradas desde fuera sin editar la configuración: una entrada se usa solo mientras el archivo de bandera lo permite.

ESTADO DE LA FUENTE SECUNDARIA

Aplique la configuración mediante central. Defina explícitamente el intervalo de comprobación en segundo plano en `lsi`; este ejemplo comprueba las entradas inactivas con un intervalo de 10 segundos:

```
streams:
- name: tv1
  inputs:
  - udp: {host: 239.0.0.1, port: 5000}
  - udp: {host: 239.0.0.2, port: 5000}
  lsi:
    source_timeout: 5
    recheck_secondary_inputs_interval: 10
```

Lea el estado mediante la API de gestión de central. Con dos entradas configuradas, `inputs[1]` es la segunda fuente y `current` indica si alimenta el stream ahora:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://central:8080/central/api-v4/streams/stats/tv1 \
  | jq '{active: [.inputs[] | select(.current)], secondary: .inputs[1] | {url, current, status, problems, checked_at, error, failure_cause}}'
```

`status` es el último estado observado, no una garantía de conmutación sin cortes. `checked_at` contiene la hora de observación RFC 3339 UTC; si falta, todavía no hubo comprobación. Valores:

- **`working`** — llegan frames, ventana limpia sin infracciones.
- **`starting`** — entrada abierta, todavía sin frames.
- **`degraded`** — llegan frames, se infringe una regla de calidad.
- **`dead`** — sin frames por error o timeout.
- **`unchecked`** — entrada todavía no observada.

Consulte `problems`, `error` y `failure_cause` para conocer la causa. `current: false` no significa por sí solo un fallo: una reserva sana no está seleccionada para emisión. Compruebe la antigüedad de `checked_at` y compare `media_info` de ambas entradas: códecs o geometrías distintos no permiten asumir una conmutación sin cortes.

TIEMPO EN LAS ENTRADAS PRIMARIA Y DE RESERVA

Lea la sección detallada `input` directamente del streamer que captura `tv1`. Central entrega el estado actual compacto y no solicita esta sección al streamer:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/streams/stats/tv1 \
  | jq '.input | {primary_s: (.num_sec_on_primary_input // 0), secondary_s: (.num_sec_on_secondary_input // 0), no_data_s: (.num_sec_no_data // 0), switches:
(.input_switches // 0), retries: (.retries // 0)}'
```


© Flussonic 2026

Estos valores acumulativos usan ticks de un segundo durante la vida de las estadísticas del stream. `num_sec_on_secondary_input` suma todas las entradas posteriores a la primera, no solo la segunda. `num_sec_no_data` cuenta los segundos después del timeout sin datos. Los campos a cero pueden omitirse; use `// 0` solo dentro de una sección `input` realmente recibida. Si falta la sección, no significa tiempo cero.

Ejemplo de dos muestras:

```
{"t0":{"primary_s":3600,"secondary_s":120,"no_data_s":0},
 "t1":{"primary_s":3640,"secondary_s":140,"no_data_s":0}}
```

Durante el intervalo, la primaria se usó 40 s y las reservas 20 s; la proporción de reserva del tiempo con datos es `20 / (40 + 20) = 33.3%`. Reste muestras consecutivas. Si los contadores disminuyen tras un reinicio o reasignación del stream, inicie un nuevo intervalo; no sume diferencias negativas.

SIGUIENTES PASOS

- Prepare un archivo de cortinilla
- Monitoree la conmutación de fuentes

© Flussonic 2026

Cortinilla de reserva

`media.source.slate`

Un stream puede tener varias fuentes. Sapsan vigila la fuente activa, conmuta automáticamente a la siguiente cuando esta se degrada y vuelve a la principal cuando se recupera. Esta lógica se llama LSI (Live Source Input).

EL ARCHIVO DE CORTINILLA (BACKUP)

Si todas las fuentes caen, en lugar de una pantalla negra se muestra un archivo (vea cómo preparar el archivo):

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    lsi:
      backup:
        file: /storage/technical-difficulties.mp4
        timeout: 5
```

La cortinilla entra en acción `timeout` segundos después de la caída de las entradas (de inmediato si no hay entradas en absoluto) y tiene sus propios `audio_timeout` / `video_timeout`.

© Flussonic 2026

Quality

`media.source.quality`

REGLAS DE MUERTE Y DEGRADACIÓN

Cada entrada lleva una placa **Opciones de la fuente**; los campos vacíos heredan las reglas comunes del stream del bloque **Tiempos de espera de entrada**. Así los umbrales se fijan una vez por stream y se anulan uno a uno, mientras que la placa plegada muestra con chips qué es exactamente lo configurado.

Los campos se agrupan de a tres, según lo que hace la estación cuando se rompe una regla.

- **Cuando la fuente está muerta** — fallos duros, conmutación inmediata: **Tiempo de espera de fuente (s)** (no hay ni un fotograma), **Tiempo de espera de vídeo** (no hay vídeo), **Bitrate máximo** (bitrate por encima del techo — protección contra basura en la entrada).
- **Cuando la fuente está degradada** — fallos blandos, conmutación solo si existe una fuente mejor: **Tiempo de espera de audio** (el audio se fue mientras el vídeo vive), **Bitrate mínimo** (bitrate por debajo del suelo), **Límite de errores CC** (violaciones de continuidad MPEG-TS dentro de la ventana limpia; el campo existe solo para protocolos de transporte — UDP, HTTP MPEGTS, SRT, RIST).
- **Recuperación y prioridad** — **Ventana limpia (s)** y **Prioridad**.

La degradación va de la calidad de la señal, y dejar una entrada degradada por una muerta no tiene sentido. Por eso una regla blanda no tira el aire, solo baja la entrada en la cola. El motivo de una marca la estación lo nombra directamente: «errores CC por encima del límite», «bitrate por debajo del suelo», «sin audio», «sin cuadros».

LA VENTANA LIMPIA

La **ventana limpia** (`return_alive_secs` , 15 segundos por defecto) es un periodo sin infracciones. Tras una infracción de calidad, la entrada debe mantener esta ventana para recuperar el estado operativo. Las pruebas en segundo plano de entradas que abre Sapsan mantienen la ventana antes del retorno a una entrada prioritaria o de una conmutación por calidad sin un veredicto operativo reciente. La conmutación de emergencia tras un fallo duro puede ocurrir en el primer punto inicial adecuado (fotograma clave, cuadro de audio o límite de fragmento), sin esperar esta ventana.

Es la protección contra el péndulo: una fuente que se levanta diez segundos y vuelve a caer ya no sacude el aire. La contrapartida es honesta y predecible — el retorno a una entrada principal que abre Sapsan espera esta ventana. Las publicaciones entrantes siguen otras reglas; consulte la gestión de fuentes LSI.

QUÉ MIRAR DESPUÉS

Lea la sección detallada `input` en el streamer que recibe el stream mediante `/streamer/api-v4/streams/stats/{name}` ; central no devuelve estos contadores detallados. Por stream se ve cuántos segundos vivió con la entrada principal y cuántos con las de reserva, cuánto tiempo estuvo sin datos, cuántas conmutaciones y reintentos hubo. Una proporción creciente de tiempo en la reserva significa que la entrada principal es sistemáticamente mala, aunque el aire nunca se haya interrumpido.

Los errores desglosados por causa están en la página Errores de entrada.

© Flussonic 2026

6.2.7 Transcodificación (media.transcode)

Codificación de vídeo

`media.transcode.video`

La función `media.transcode.video` codifica la pista de vídeo seleccionada en el códec de salida requerido y permite definir su bitrate, resolución y estructura GOP.

La creación de varias calidades de salida y los detalles de su procesamiento conjunto se describen en la transcodificación multibitrate.

CÓDECS DE SALIDA

La función `media.transcode.video` incluye la codificación en los siguientes códecs de vídeo de salida:

Códec de vídeo	Valor de `codec`
H.264 / AVC	`{set: h264}`
H.265 / HEVC	`{set: hevc}`
AV1	`{set: av1}`
MPEG-2 Video	`{set: mp2v}`

CONFIGURACIÓN

En `transcoder.output`, indique la pista de vídeo de origen con `source`, el códec de destino con `codec` y el bitrate con `bitrate`:

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    transcoder:
      output:
      - source: !best_quality video
        codec: !set h264
        bitrate: 2800000
```

SELECCIÓN DE LA PISTA DE ORIGEN

Para una sola salida de vídeo, use `source: {best_quality: video}`: selecciona la pista de vídeo con el mayor bitrate declarado. Si conoce el ID de entrada, use `source: {exact: v1}`. Con una fuente de reserva, ese ID debe identificar la pista deseada en cada entrada.

Consulte la descripción detallada de la selección de pistas para ver todos los selectores, el comportamiento al pasar a reserva y los ejemplos.

Si no hay coincidencias, la regla en directo no crea salida. El procesamiento en directo no implementa los modos `if_missing: blank` y `substitute`; no sirven para sustituir una señal ausente.

CÓDECS Y PARÁMETROS

- **`codec`** — códec de vídeo de destino de la lista anterior. `codec: same` conserva los fotogramas originales sin recodificarlos.
- **`bitrate`** — bitrate de destino en bits/s.
- **`gop_size`** — longitud del GOP en fotogramas.
- **`gop_structure`** — estructura de la secuencia de fotogramas, por ejemplo `ip` o `ibbbp`.

HEVC

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    transcoder:
      output:
      - source: !best_quality video
        codec: !set hevc
        bitrate: 2800000
```


© Flussonic 2026

AV1

Para la codificación por software, el servidor busca los codificadores disponibles en este orden:

- **SVT-AV1**;
- **libaom**;
- **rav1e**.

Si ninguno está disponible, prueba el codificador AV1 predeterminado disponible. El preset de velocidad se convierte en los ajustes del codificador elegido. En NVIDIA se usa AV1 NVENC; la GPU debe admitir codificación AV1.

Para la entrega segmentada, el servidor fija el intervalo entre fotogramas clave y desactiva la detección de escenas en los codificadores AV1 por software. No ofrecen GOP abierto: se usa GOP cerrado y se registra una advertencia si se solicita uno abierto.

AV1 por software usa VBR con GOP de acceso aleatorio. Si se solicita CBR, el servidor continúa en VBR y registra una advertencia. Es una limitación del codificador.

CAMBIO DE TAMAÑO

`resize`	**Qué hace**
`!scale`	escalado exacto a las dimensiones indicadas
`!fit`	encajar conservando la proporción de aspecto (con relleno, color de fondo configurable)
`!crop`	recortar a las dimensiones indicadas

COMPROBACIÓN

Compruebe el códec y los parámetros de vídeo de salida en las estadísticas del stream. Después abra `http://server/streaming/v/tv1/index.m3u8` en un reproductor y verifique la reproducción.

SIGUIENTES PASOS

- Distribuya el stream transcodificado
- Grabe el stream de salida en el archivo

La forma del píxel y su interacción con el redimensionamiento se describen en SAR y vídeo anamórfico.

© Flussonic 2026

SAR y vídeo anamórfico

`media.transcode.sar`

QUÉ DESCRIBE EL SAR

Sample Aspect Ratio (SAR) es la relación entre la anchura y la altura de una muestra de imagen. SAR `1:1` indica píxeles cuadrados; cualquier otra relación indica vídeo anamórfico. El reproductor utiliza el SAR al mostrar la imagen.

- **Ráster** — dimensiones del fotograma codificado, por ejemplo 720×576.
- **SAR** — forma del píxel, por ejemplo 64:45.
- **DAR** — relación de aspecto de la imagen mostrada: `(anchura del ráster / altura del ráster) × SAR`.

Por ejemplo, 720×576 con SAR 64:45 se muestra a 1024×576 (16:9). SAR 16:11 en el mismo ráster da una anchura de unos 1048: corresponde a otra convención de área activa PAL. Estas relaciones no son intercambiables.

INTERACCIÓN ENTRE SAR Y REDIMENSIONAMIENTO

El SAR de entrada describe la fuente. La estrategia de redimensionamiento determina cómo encajar, recortar o estirar la imagen. El SAR de salida explícito determina el ráster para las dimensiones de visualización resultantes. Cambiar solo el SAR conserva las proporciones de visualización y remuestrea el ráster; requiere transcodificación, no solo modificar metadatos.

Para una fuente de 720×576 con SAR 64:45 y H.264:

Redimensionamiento	SAR de salida	Ráster	Dimensiones de visualización
Ninguno	Automático	720×576	1024×576
Ninguno	1:1	1024×576	1024×576
Ninguno	16:11	704×576	1024×576
Fit, altura 360	Automático	450×360	640×360
Fit, altura 360	1:1	640×360	640×360
Fit, altura 360	4:3	480×360	640×360

El redondeo del ráster a dimensiones pares puede introducir una diferencia de unos píxeles. Con SAR inferior a uno, Sapsan reduce la altura del ráster en lugar de aumentar la anchura.

Con SAR explícito, la anchura y la altura de resize definen el **marco de visualización**:

- **Fit** conserva toda la imagen y añade bandas si hace falta.
- **Crop** recorta por el centro.
- **Scale** estira hasta llenar el marco.

Por ejemplo, Scale 640×480 con SAR 4:3 produce un ráster de 480×480 que se muestra a 640×480: una entrada 16:9 se estira a 4:3.

En modo Automático, H.264/HEVC sin redimensionamiento, con Fit o con Crop conservan el SAR de entrada. En Scale las dimensiones definen el **ráster**: para una entrada anamórfica se recalcula el SAR para conservar el DAR; para una entrada de píxeles cuadrados permanece en 1:1 y puede haber distorsión. Por tanto, pasar de Automático a SAR explícito con las mismas dimensiones Scale puede cambiar el resultado.

CONFIGURACIÓN Y COMPROBACIÓN

En la consola Mcaster, abra una calidad del transcodificador, pulse **Más** y ajuste **Forma del píxel (SAR)**. Consulte los pasos y capturas en la guía de Mcaster.

© Flussonic 2026

Calidad H.264 real: SAR 16:11, sin redimensionamiento; fuente 720×576, SAR 64:45, salida 704×576, SAR 16:11.

El ajuste de la API es `transcoder.output[].params.video.sar`:

```
{"sar": {"width": 16, "height": 11}}
```

La ausencia del campo significa Automático; `1:1` explícito solicita píxeles cuadrados. Elimine el campo para volver a Automático; envíe `sar: null` con JSON Merge Patch. Cada componente debe ser un entero de 1 a 65535.

PATCH sustituye el array `transcoder.output` completo: envíe todas sus calidades y conserve los demás ajustes; no se pueden modificar elementos del array por índice.

En las estadísticas de pista, `width` y `height` describen las dimensiones de visualización, `pixel_width` y `pixel_height` el ráster, y `sar_width` y `sar_height` el SAR. Si el ráster coincide con la visualización y el SAR es unitario, esos campos pueden omitirse: utilice las dimensiones de visualización y 1:1. La vista previa de la consola muestra dimensiones de visualización y SAR configurado, no el ráster codificado real.

RESTRICCIONES Y CONTABILIZACIÓN DEL USO

- **Transcodificación** — el SAR explícito requiere un códec de salida; se rechaza el paso directo `same` con SAR.
- **AV1 y JPEG 2000** — admiten Automático o SAR cuadrado.
- **MPEG-2** — requiere un SAR representable mediante un DAR admitido para el ráster seleccionado.

También se rechaza una relación que reduzca un lado del ráster por debajo de dos píxeles.

La función `media.transcode.sar` se contabiliza por stream: al menos una calidad con SAR explícito, incluido 1:1, cuenta como una unidad. Más calidades no aumentan el número. Se excluyen Automático y la entrada anamórfica sin ajuste explícito. La telemetría envía el número actual como `usage.media_transcode_sar`; se omite el cero. Es una función independiente de la codificación de vídeo `media.transcode.video`.

© Flussonic 2026

VOD

`media.transcode.vod`

Un archivo subido puede servirse tal cual o desplegarse en una **escalera de calidades**: varios archivos de distinto tamaño y bitrate más un descriptor para el reproductor. Cómo desplegarlo lo define una **plantilla de asset**; el trabajo en sí es una **tarea** y se ejecuta en el streamer donde está el original. Las salidas quedan en el mismo asset, junto al original.

Para esto la sección **VOD** tiene dos pestañas: **Plantillas de activos** y **Tareas**. El avance de un asset concreto se ve en su tarjeta; el cuadro general, en el panel de control.

PLANTILLAS DE ASSET

La pestaña **Plantillas de activos** es el catálogo de escaleras. Las columnas: **Nombre**, **Descripción** — las calidades en palabras, por ejemplo `720p H.264, 360p H.264`, y **Assets** — cuántos assets están vinculados a la plantilla.

Las plantillas del sistema (`mbr-1080`, `mbr-720`, `source-like`) no se editan ni se eliminan; su única acción es **Clonar**, que abre el formulario de una plantilla propia con una copia de la escalera. Las plantillas propias tienen **Editar** y **Eliminar**; el botón **Nueva plantilla** abre un formulario con una sola calidad.

Una plantilla vinculada a assets también se puede eliminar: la confirmación indica cuántos son, tras la eliminación los assets quedan sin plantilla y sus archivos no se tocan. La siguiente subida a un asset así ofrece elegir una plantilla de nuevo.

EL FORMULARIO DE LA PLANTILLA

El formulario de la plantilla es la escalera, calidad a calidad. **Nombre** es cómo aparece la plantilla en las listas. **Original** es qué hacer con el archivo de origen tras el procesamiento: **conservar** o **eliminar tras procesar**. Después, las **calidades**: cada una tiene su resolución (el campo **Altura**), **Códec** y **Bitrate**, y cada calidad es un archivo de salida; el preset del codificador, la longitud del GOP y la estructura de cuadros se definen para toda la escalera. Los botones **Añadir calidad** ofrecen las resoluciones habituales, **Otra…** — cualquier resolución, **original tal cual** — una calidad sin cambio de tamaño. La regla de audio es una para toda la escalera: todas las pistas de audio del original se recodifican a AAC y van a cada salida conservando la etiqueta de idioma, así que los doblajes y traducciones no se pierden. El campo **Copias en otros servidores** no forma parte de la escalera: encarga copias del asset en otros streamers y se describe en Copias VOD y balanceo; en una plantilla de transcodificación queda en **sin copia**.

Los nombres de las salidas no se escriben: se construyen con el nombre del origen y la resolución de la calidad. La ficha **Qué saldrá de movie.mp4** a la derecha muestra el resultado para el ejemplo: `movie-mp4-720p.mp4`, `movie-mp4-360p.mp4` y el descriptor `movie-mp4.smil`. El punto del nombre de origen se convierte en guion, así que `movie.mp4` y `movie.mkv` en un mismo asset producen salidas distintas y no se sobrescriben.

Dos calidades con la misma resolución y el mismo códec no se guardan: es un mismo archivo con dos nombres. Las calidades se distinguen por la resolución.

El campo **Copias en otros servidores** indica en cuántos streamers más debe estar el asset; después de la transcodificación, central elige los streamers de destino y crea tareas de copia independientes. Los detalles, en Copias VOD y balanceo.

Debajo está la ficha **Activos con esta plantilla**: enlaces a las tarjetas de los assets vinculados; a partir de cien, la ficha muestra los primeros cien. Una plantilla creada por API sigue las mismas reglas: el servidor solo acepta una escalera así, por lo que cualquier plantilla se abre en el formulario.

SUBIR CON UNA PLANTILLA

La plantilla se vincula al asset en la subida: en la subida VOD, junto al nombre del asset nuevo, está el campo **Plantilla del activo**. Cada archivo de vídeo lleva la marca **Por plantilla**; sin la marca, el archivo se guarda tal cual, sin tarea. Una subida a un asset con plantilla sigue su plantilla, no hay nada que elegir; un asset sin plantilla ofrece elegirla en la propia vista previa.

La tarea aparece sola en cuanto termina la subida. No hay nada que lanzar a mano.

TAREAS

La pestaña **Tareas** es el registro del trabajo de todos los streamers. Sobre la lista está el resumen por estado: **en cola**, **enviado**, **en curso**, **cancelando**, **error**.

Las columnas: **Tarea** — el archivo de origen, el enlace abre la tarjeta de la tarea; **Activo** — la clave, el enlace abre la tarjeta del asset; **Estado**; **Progreso** — el porcentaje y la velocidad respecto al tiempo real; **Ejecutor** — el streamer y su cola; **Intento**; **Error** — en palabras; **Creada** — cuándo se lanzó la tarea. En una tarea en cola la columna **Error** indica el motivo de la espera: el streamer no está conectado o no ejecuta tareas, su cola no tiene sitio,

© Flussonic 2026

un intento falló y se reintentará, no hay streamer, espacio o recepción de copias para la copia. En una tarea en curso que espera a que el streamer de destino acepte los archivos de la copia, la misma columna dice **el ejecutor terminó, esperando la subida de las salidas**.

Estados de una tarea:

- **en cola** — espera un ejecutor;
- **enviado** — aceptada por un streamer, espera su turno allí;
- **en curso** — codificando;
- **cancelando** — la cancelación se envió al ejecutor, se espera su confirmación;
- **listo**;
- **error** — se agotaron los intentos o el origen no sirve;
- **cancelado**.

Por defecto la lista muestra **activas y errores**; las tareas terminadas y canceladas se abren con el filtro **Estado**, y la opción **todos** muestra el registro completo. El filtro **Activo** acepta una clave de asset, **Nodo** — el streamer ejecutor, **Restablecer** quita todos los filtros.

Acciones de la fila:

- **Cancelar** en una tarea en cola la cierra de inmediato; en una tarea en curso la pasa a **cancelando** hasta que el streamer confirme. En una tarea fallida, **Cancelar** quita el registro.
- **Reintentar** se ofrece en los errores que un nuevo intento puede resolver: el streamer no estaba disponible, se acabó el espacio, el codificador se cayó. Los errores del origen — el archivo no se lee, el formato no es compatible — no se resuelven reintentando, y el botón no aparece.

Las tareas terminadas y fallidas se conservan siete días y luego salen del registro por sí solas. El botón **Limpiar** las quita antes: aparece con el filtro por listas, con error o canceladas y, tras confirmar, quita todas las tareas de ese estado. Las tareas de un asset eliminado salen junto con él.

LA TARJETA DE LA TAREA

La tarjeta de la tarea se abre por el nombre del origen en la lista. La cabecera lleva el archivo, el estado y el enlace **Volver a los trabajos**.

El bloque **Progreso** — la barra de avance, **Procesado** respecto a la duración del archivo, **Velocidad** e **Intento**; mientras se sube la copia, en lugar de lo procesado y la velocidad están **Subido** y **Velocidad de subida**, y una tarea terminada con copia muestra ambos. El bloque **Detalles** — la plantilla, qué pasa con el original tras el procesamiento, el origen, las salidas y, en una tarea con copia, la fila **Subidas** con el número de copias; en una tarea fallida aquí está también el mensaje completo del ejecutor. El bloque **Resultado** aparece cuando al menos una salida está lista o la tarea lleva copia, y enumera los archivos listos, el descriptor del reproductor y las subidas de la copia: el streamer, los archivos y el estado — pendiente, subiendo, subido, sin copia. Una tarea en cola muestra el motivo de la espera en lugar del avance; una tarea con copia muestra el chip **subida de copia** junto al estado mientras sube, y si no había nada que copiar, la marca **sin copia** con una pista del porqué. A la derecha, **Ejecución** — el asset, el ejecutor con su cola y quién creó la tarea, y **Cronología** — cuándo se creó, se envió y se inició; **Fin** en una tarea en curso es una estimación por la velocidad actual, marcada con ≈, en una terminada la hora real, y en una fallida o cancelada la fila se llama **Detenida**. Una tarea cancelada no muestra salidas ni resultado: no produjo nada.

UN ASSET EN PROCESO

En la tarjeta del asset, el bloque **Plantilla del activo** muestra la plantilla vinculada. **Guardar** cambia solo la plantilla y no lanza tareas. **Lanzar tarea…** abre la elección de originales: los marcados se procesan de nuevo con la plantilla actual y las salidas se regeneran. El enlace **Tareas del activo** abre el registro filtrado por este asset con todos los estados, incluidas las terminadas; al lado, cuántas de sus tareas están activas y con error.

Mientras la tarea está en curso, su original va primero en la lista de archivos y está resaltado, y su **Eliminar** está inactivo: el archivo está en proceso. Al terminar, las salidas y el descriptor quedan en la lista junto al resto de archivos, sin marcas; el descriptor `.smil` es el punto de entrada del reproductor para toda la escalera.

EL PANEL DE CONTROL

En el panel de control, la fila **Transcodificación de archivos VOD** muestra cuántas tareas hay en cola, en curso y con error, y cuántos streamers pueden ejecutarlas; el enlace **a las tareas** abre el registro. El ejecutor de una tarea concreta es siempre el streamer donde está el original; las tareas de copia descargan los archivos en los streamers de destino, véase Copias VOD y balanceo.

© Flussonic 2026

UNA TAREA POR API

La consola lanza tareas solo con la plantilla del asset. Una tarea puntual para archivos concretos, con otra plantilla o con un plan propio se lanza por API:

```
curl -X POST 'http://<central>/central/api-v4/vod/jobs' \
  -H 'Authorization: Bearer <clave>' -H 'Content-Type: application/json' \
  -d '{"asset_id": "<asset id>", "files": ["movie.mp4"], "template_id": "mbr-720"}'
```

- **files** — qué originales procesar; sin el campo, todos los originales del asset.
- **template_id** o **plan** — una plantilla o un plan en la propia petición; no se vincula al asset y la plantilla del asset no cambia.
- **target_asset_id** deja las salidas en otro asset, **trim** corta un fragmento.

Un plan propio no está limitado por la forma de la escalera: su descriptor, los nombres de salida y el conjunto de pistas son libres. Esa tarea aparece en la pestaña **Tareas** como cualquier otra.

El registro por API — `GET /central/api-v4/vod/jobs` con los filtros **asset_key**, **status** (un estado o varios separados por comas, por ejemplo `queued,running`) y **node**; el registro lleva el motivo de la espera como código (**wait_reason**) y en palabras (**reason**). `DELETE /vod/jobs/{id}` en una tarea lista, fallida o cancelada quita el registro; `DELETE /vod/jobs?status=error` — todos los registros de ese estado.

© Flussonic 2026

Codificación de audio

`media.transcode.audio`

Sapsan recodifica las pistas de audio, por ejemplo G.711 de una cámara IP, y conserva la línea de tiempo del stream.

CÓDECS DE SALIDA

La función `media.transcode.audio` incluye la codificación en los siguientes códecs de audio de salida:

Códec de audio	Valor de `codec`
AAC	`{set: aac}`
Opus	`{set: opus}`
MP3	`{set: mp3}`

CONFIGURACIÓN

`source: {content: audio}` selecciona todas las pistas de audio. Para una pista por su ID de entrada, use `source: {exact: a1}`.

```
streams:
  - name: camera
    inputs:
    - rtsp:
        url: rtsp://camera/stream
    transcoder:
      output:
      - source: !content audio
        codec: !set aac
        bitrate: 128000
```

PARÁMETROS

Parámetro	Descripción
`source`	Regla de selección de las pistas de audio de entrada.
`codec`	Códec de audio de destino de la lista anterior. `codec: same` conserva el códec de origen.
`bitrate`	Bitrate de destino en bits/s.
`if_missing`	En directo, una regla sin coincidencias no crea salida; `blank` y `substitute` no proporcionan medios de sustitución.

Use `source: {language: eng}` para mantener la selección del idioma cuando cambie el códec de la fuente. Si también importa el códec de entrada, use `source: {language_codec: [eng, aac]}`. Consulte la selección de pistas para MBR para ver ejemplos de fuentes principal y de reserva y los límites de `if_missing`. Compruebe las pistas y el códec de audio en las estadísticas del stream y escuche el resultado.

© Flussonic 2026

MBR

`media.transcode.mbr`

El transcodificador de Sapsan cambia los códecs, el bitrate y la resolución y prepara una escalera multibitrate (MBR). Varias calidades de salida pueden utilizar una misma pista de vídeo de origen: se decodifica una sola vez y los fotogramas resultantes se envían a los codificadores de las calidades de salida.

Cada salida se describe mediante una regla: **qué pista de origen seleccionar y qué obtener de ella**. La selección puede basarse en el idioma o el tipo de contenido para que la configuración siga funcionando cuando cambie la composición de la entrada. Esto resulta importante al conmutar entre las fuentes principal y de reserva, cuyos identificadores de pista y códecs pueden ser distintos.

CÓMO CONSTRUYE SAPSAN UNA ESCALERA SINCRONIZADA

La transcodificación multibitrate crea varias representaciones de un mismo vídeo. Difieren en el tamaño del fotograma y el bitrate, y pueden utilizar distintos códecs, pero deben mostrar la misma imagen en el mismo instante. Sapsan las procesa en un único pipeline de vídeo con una escala de tiempo común.

Una sola decodificación para todas las calidades

Cada pista de vídeo de origen tiene un único decodificador para todas sus calidades de salida. El fotograma comprimido de entrada se decodifica una sola vez; la imagen resultante se envía al escalado y la codificación de cada calidad. Por ejemplo, para tres calidades de salida, Sapsan realiza una decodificación de la entrada y tres codificaciones de salida.

Al aumentar el número de calidades, aumenta el trabajo de los codificadores de salida y de las transformaciones de imagen. El coste de decodificar la fuente no se multiplica por el número de calidades. Esto es especialmente importante con vídeo de entrada que exige muchos recursos: el decodificador compartido atiende a toda la escalera.

Marcas de tiempo idénticas para los fotogramas correspondientes

Una misma imagen de origen recibe una marca de tiempo de presentación **PTS** común en todas sus calidades de salida. Coinciden las propias marcas, no solo la frecuencia nominal de fotogramas. El reproductor puede relacionar las imágenes correspondientes de distintas calidades mediante un mismo tiempo de medios.

Por ejemplo, a 25 fotogramas/s, los tiempos de presentación en segundos son los siguientes:

Imagen	1080p H.264	720p HEVC	360p AV1
Primera	2.000	2.000	2.000
Siguiente	2.040	2.040	2.040
Otra más	2.080	2.080	2.080

Para las calidades de una misma pista de origen, el servidor utiliza la precisión de la escala de tiempo de la fuente: los PTS originales no se redondean a un número entero de fotograma solo para realizar la codificación. Las distintas velocidades de procesamiento de los codificadores tampoco crean relojes independientes para las calidades.

El **DTS** establece el orden de decodificación y depende de la reordenación de fotogramas de cada codificador. Al utilizar fotogramas B, el DTS puede diferir del PTS y entre códecs. Para relacionar la misma imagen entre calidades se utiliza el PTS común.

Fotogramas clave sincronizados

Con ajustes de GOP coordinados, los fotogramas clave de las distintas calidades coinciden en los mismos instantes de presentación. El reproductor dispone de puntos de conmutación sincronizados: al seleccionar otra calidad, continúa desde la misma posición temporal. El segmentador puede establecer los límites de los segmentos en los fotogramas clave coincidentes.

Para una escalera sincronizada, configure los mismos valores de `gop_size` y `gop_structure` en sus salidas de vídeo. Por ejemplo, a 25 fotogramas/s y con `gop_size: 50`, el intervalo previsto entre fotogramas clave es de dos segundos para todas las calidades. En el ejemplo siguiente, ambas calidades utilizan una misma pista de origen y la misma estructura de GOP.

© Flussonic 2026

Combinación de las pistas en un stream de salida común

Los codificadores pueden necesitar distintos tiempos para preparar un fotograma, especialmente si utilizan distintos códecs o fotogramas B. Sapsan combina los resultados de los codificadores de vídeo y las pistas de audio según una escala de tiempo de medios común: el intercalador de salida almacena los fotogramas en un búfer y los ordena por DTS. Por tanto, el tiempo que tarda en completarse cada codificación no determina el orden de los fotogramas de todo el stream.

El decodificador compartido, los PTS coincidentes y los fotogramas clave sincronizados proporcionan al reproductor una escalera preparada para la reproducción adaptativa. Puede seleccionar el bitrate adecuado conservando la posición temporal del vídeo. La sincronización original del audio y el vídeo también es importante: ordenar los fotogramas de salida no corrige por sí solo un desfase de audio incorrecto en la señal de entrada.

CONFIGURACIÓN

La configuración se realiza mediante la API de central. Los ejemplos utilizan la forma de objeto de YAML; por ejemplo, `source: {content: audio}` se transmite en JSON como `"source": {"content": "audio"}`. También se admite la notación YAML con etiqueta: `source: !content audio`.

Supongamos que un stream recibe las siguientes variantes de señal:

Entrada	Vídeo	Audio
Principal, MPEG-TS por UDP	`v1`, HEVC, 1920×1080	`a1`: MPEG Audio en inglés, `eng`; `a2`: AC-3 en ruso, `rus`
Reserva, HLS	`v3`, H.264, 1280×720	`a4`: AAC en ruso, `rus`

Las reglas siguientes seleccionan el vídeo y el audio en ruso por sus propiedades. La entrada de reserva no requiere una configuración del transcodificador independiente:

```
streams:
  - name: tv1
    inputs:
      - udp: {host: 239.0.0.1, port: 5000}
      - hls: {url: https://backup.example.com/tv1/index.m3u8}
    transcoder:
      output:
        - source: {best_quality: video}
          codec: {set: h264}
          bitrate: 2800000
          params:
            video:
              gop_size: 50
              gop_structure: ibbbp
              resize: {fit: {width: 1920, height: 1080}}
        - source: {best_quality: video}
          codec: {set: h264}
          bitrate: 1200000
          params:
            video:
              gop_size: 50
              gop_structure: ibbbp
              resize: {fit: {width: 1280, height: 720}}
        - source: {language: rus}
          codec: {set: aac}
          bitrate: 128000
```

A partir del vídeo de cualquiera de estas entradas se generan dos calidades H.264, y la pista de audio en ruso seleccionada se prepara para su entrega en AAC. La pista en inglés de la entrada principal no coincide con la regla de audio. LSI realiza la conmutación entre entradas; las reglas `source` seleccionan las pistas de la entrada ya activa.

© Flussonic 2026

SELECCIÓN DE LA PISTA DE ORIGEN

`source`	**Cómo se seleccionan las pistas en un stream en directo**
`{exact: v1}` o `{exact: a2}`	Una pista con el ID de entrada indicado. Es el ID lógico de Sapsan, no el PID de MPEG-TS.
`{content: video}` o `{content: audio}`	Todas las pistas del tipo indicado. Una regla puede crear varias salidas.
`{best_quality: video}` o `{best_quality: audio}`	Una pista del tipo indicado con el mayor `bandwidth` declarado.
`{language: rus}`	Todas las pistas de audio cuyo idioma sea exactamente `rus`, independientemente del códec y del ID.
`{language_codec: [rus, aac]}`	Todas las pistas de audio que tengan a la vez el idioma `rus` y el códec `aac`.
`{codecs: [pcma, pcmu]}`	Todas las pistas cuyo códec pertenezca al conjunto no vacío indicado.

El idioma se obtiene de los metadatos de la fuente. La comparación es exacta: `rus` y `ru` son cadenas distintas. Si falta la etiqueta de idioma, la regla de idioma no coincide. Si la entrada tiene varias pistas en ruso, `language: rus` selecciona cada una; para restringir la selección, combine el idioma con el códec o utilice un ID concreto.

`best_quality` compara el bitrate declarado, no la resolución ni la calidad visual. Si falta `bandwidth`, se considera cero; en caso de empate, se selecciona la última pista coincidente de la descripción actual de la entrada. La selección se realiza al configurar el transcodificador. Actualizar únicamente la estimación del bitrate no provoca una conmutación automática entre pistas.

Cada regla se ejecuta de forma independiente. Varias reglas con una misma fuente crean varias calidades de salida. Las reglas `language: rus` y `content: audio` procesarán ambas la pista en ruso; esta configuración no significa «ruso y, si no existe, cualquier audio».

Qué cambia al conmutar entre entradas

Cuando cambia la composición o algún parámetro significativo de la entrada, Sapsan vuelve a comparar las reglas con la nueva descripción de las pistas y reconfigura el procesamiento. Se tienen en cuenta los cambios de ID y orden de las pistas, códecs y parámetros de decodificación. Para el audio también se tiene en cuenta el idioma.

En el ejemplo anterior, la regla `language: rus` selecciona `a2` de la entrada principal y `a4` de la reserva. La regla `exact: a2` no selecciona nada en la reserva. La regla `language_codec: [rus, ac3]` limita intencionadamente el procesamiento al AC-3 en ruso y no coincide con el AAC de reserva.

Los identificadores de las pistas de salida pueden cambiar cuando cambia la entrada. En particular, una pista AAC-LC que ya cumpla los requisitos puede pasar sin volver a codificarse y conservar su ID de entrada; esta optimización se aplica después de seleccionar la fuente. La reconfiguración de decodificadores y codificadores también puede producir una transición en la salida. Las reglas de selección permiten trabajar con fuentes de distintas estructuras, pero por sí solas no garantizan una conmutación sin interrupciones.

Transcodificar todas las pistas de audio

Para un canal de televisión multilingüe, puede conservar todos los idiomas disponibles y convertir el audio a AAC:

```
transcoder:
  output:
    - source: {best_quality: video}
      codec: {set: h264}
      bitrate: 2000000
    - source: {content: audio}
      codec: {set: aac}
      bitrate: 128000
```

Si la entrada principal contiene dos pistas de audio, la regla de audio procesa ambas. Si la reserva solo tiene una, la salida tendrá una. La regla utiliza el conjunto de pistas de audio de origen disponible; no establece un número fijo de salidas.

La selección por idioma utiliza los metadatos de la entrada. En el procesamiento en directo actual, la etiqueta de idioma no se transfiere al audio que se vuelve a codificar; en el paso directo sí se conserva. Téngalo en cuenta al comprobar los nombres de los idiomas en el reproductor.

Transcodificar solo determinados códecs

Para fuentes cuyo audio puede llegar en PCMA, PCMU o AAC, puede configurar por separado la conversión de G.711 y el paso directo de AAC:

```
transcoder:
  output:
```


© Flussonic 2026

```
    - source: {best_quality: video}
      codec: {set: h264}
      bitrate: 2000000
    - source: {codecs: [pcma, pcmu]}
      codec: {set: aac}
      bitrate: 128000
    - source: {codecs: [aac]}
      codec: same
```

La regla con el conjunto de códecs selecciona tanto PCMA como PCMU sin depender del identificador de la pista de audio. La otra regla conserva el AAC existente sin transcodificarlo. Si G.711 y AAC están presentes a la vez, se ejecutan ambas reglas. Los códecs ajenos a estos conjuntos, como AC-3, no se seleccionan mediante las reglas de audio de este ejemplo.

Si no hay ninguna pista coincidente

En el transcodificador en directo, una regla sin coincidencias no crea ninguna pista de salida. Por ejemplo, la ausencia de `rus` no provoca la selección del audio en inglés ni la generación de silencio. Las demás reglas que sí coinciden siguen generando sus salidas.

El campo `if_missing` existe en el modelo de configuración, pero el procesamiento en directo no ejecuta sus modos `fail`, `blank` y `substitute`. No se puede depender de ellos para sustituir una pista ausente al conmutar entre fuentes.

Los ejemplos anteriores incluyen codificación real. Si todas las reglas utilizan únicamente `codec: same`, se activa el modo de paso directo general; esta configuración no permite filtrar pistas mediante `source`.

CÓDECS Y PARÁMETROS

- **`codec: {set: h264}`:** transcodifica al códec indicado. Los parámetros se detallan en codificación de vídeo. La notación `codec: same` conserva los fotogramas originales sin transcodificarlos.
- **Escalera sincronizada:** las salidas con distintos códecs procedentes de una misma pista de vídeo mantienen alineadas las marcas de tiempo y los fotogramas clave.
- **`gop_size` y `gop_structure`:** controlan la longitud y la estructura del GOP para obtener segmentos MBR sincronizados.
- **Audio G.711:** la transcodificación a AAC u Opus conservando la escala de tiempo también se describe en codificación de audio.
- **Pistas auxiliares:** las pistas de subtítulos y las vistas previas JPEG admitidas pueden conservarse mediante paso directo. Compruebe la composición final del stream, además de las reglas de codificación.

CAMBIO DE TAMAÑO

`resize`	Función
`{scale: {width: 1280, height: 720}}`	Escala exactamente al tamaño indicado.
`{fit: {width: 1280, height: 720}}`	Ajusta la imagen conservando sus proporciones y añade bandas si es necesario.
`{crop: {width: 1280, height: 720}}`	Recorta la imagen al tamaño indicado.

VERIFICACIÓN

Compruebe los metadatos de entrada y la composición final de las pistas en las estadísticas del stream. Después abra la playlist maestra `http://server/streaming/v/tv1/index.m3u8` y compruebe las calidades de salida y los idiomas del audio.

Conmute el stream a la reserva y compruébela por separado: qué vídeo se selecciona, cuántas pistas de audio se generan y si se conservan los idiomas necesarios. Para diagnosticar la reconfiguración, consulte los mensajes `encode backend selected input tracks` y la tabla de correspondencia entre las pistas de entrada y salida `encode backend stream mapping` en los logs del streamer.

© Flussonic 2026

CBR

`media.transcode.cbr`

QUÉ HACE EL MULTIPLEXOR

- **La tasa objetivo se deriva de los bitrates de las pistas** — con un margen de alrededor del 5% para el crecimiento y un presupuesto para las tablas de servicio.
- **Los huecos se rellenan con paquetes nulos** (PID 0x1FFF): el ancho de banda queda ocupado de forma uniforme aunque los fotogramas lleguen a ráfagas.
- **El PCR se coloca desde un reloj sin deriva**, por eso la divergencia entre PCR y DTS no se acumula ni en muchas horas de emisión.
- **Se tiene en cuenta el modelo de búfer del decodificador (HRD)** — los desbordamientos y los vaciamientos se vigilan en lugar de descubrirse en el receptor.

La uniformidad se mide contra el reloj interno del stream (27 MHz), no contra el reloj de pared: la disposición no depende de lo irregularmente que lleguen los fotogramas desde la fuente.

LO QUE CUESTA LA UNIFORMIDAD

Las tres exigencias de un multiplex — disponer los paquetes de forma uniforme, no detenerse nunca, no perder nada — se satisfacen por pares, pero no las tres a la vez. La uniformidad es una afirmación sobre el futuro, mientras que los datos son un hecho sobre el pasado: para disponer los paquetes uniformemente, la programación tiene que conocer el ritmo por adelantado, y el ritmo solo puede medirse por lo que ya ha llegado.

En la práctica, esto significa:

- **Un PID sin nada que enviar cede su hueco** en vez de detener el multiplex. Una pista de audio anunciada en la PMT pero silenciosa no debe colgar el vídeo — y no lo cuelga.
- **Un PID rezagado se recupera dentro de la ventana** en lugar de ser descartado. Un mux de difusión clásico pierde datos en este punto; aquí el intercambio se resuelve al revés.
- **El ritmo se mide por el PES empaquetado**, no se deduce del ancho de banda declarado multiplicado por una constante.

Lo que de ahí se sigue para el operador: fije el bitrate de salida con honestidad, con margen para los picos, y no al bit justo. Un ancho de banda recortado no es «calidad un poco peor», es un PID rezagado de forma sistemática.

CONFIGURE LAS PISTAS DE SALIDA

Abra el stream, la pestaña **Transcodificador**. Puede empezar desde un preset — por ejemplo **Escalera estándar 1080/720/360 + AAC** — y editar los peldaños a partir de ahí.

En un peldaño del bloque **Vídeo (escalera)**:

- **Bitrate (kbps)** — la tasa objetivo de la pista. La suma de las pistas más las tablas de servicio es el ancho de banda que ocupará el flujo de transporte.
- **Longitud GOP** y **Estructura GOP** — para difusión, el GOP suele ser de exactamente un segundo: un punto de entrada para el receptor y para el empalme. El botón **Unificar GOP** lo alinea en todos los peldaños.
- **Códec** — con el valor **Tal cual**, el peldaño no se recodifica. Si la fuente ya llegó con la forma adecuada, al multiplexor no le hace falta un transcodificador para disponerla uniformemente.
- **Altura**, **Rate control** y **Preset del codificador** — las demás propiedades del peldaño.

El bloque **Codificación** elige quién hace el trabajo: **CPU** o **GPU NVIDIA** (con una tarjeta elegida, o automáticamente la menos cargada).

> **Warning**
>
> Un peldaño passthrough junto a peldaños codificados degrada la escalera: su GOP y sus fotogramas clave no están alineados con los demás peldaños. El panel marca ese peldaño por separado.

© Flussonic 2026

COMPROBACIÓN

Qué mirar en la salida:

- **la tasa** en el receptor — no debe flotar; una caída significa que el multiplex no tenía nada que enviar;
- **el jitter del PCR** y los **contadores de continuidad** — véase TR 101 290; un stream uniforme con un PCR incorrecto no le sirve de nada a un modulador;
- **las tablas de servicio** — el receptor debe ver el servicio bajo el número esperado.

Anular las tablas en un destino individual **no rompe el CBR**: el destino recibe una copia de la salida compartida con PID y tablas reescritos, mientras que la disposición de los paquetes y el PCR siguen siendo los del multiplex compartido.

© Flussonic 2026

HW

`media.transcode.hw`

CODIFICACIÓN: CPU O GPU

La sección **Codificación** decide qué hace el trabajo. El campo **Transcodificar en** tiene dos valores: **CPU** y **GPU NVIDIA**. La ayuda enuncia la diferencia sin rodeos: **CPU es el valor predeterminado; NVIDIA ejecuta todo el vídeo en la GPU** — a la tarjeta se va toda la ruta de vídeo, no solo la codificación.

Al elegir NVIDIA aparece un campo **GPU** con las tarjetas que informó el nodo. El valor **Auto (menos cargada)** elige tarjeta **al iniciar el stream** — un stream ya en marcha no se traslada a otra tarjeta. En un clúster, el layouter coloca un stream con codificación NVIDIA solo en un streamer con GPU — no hacen falta etiquetas para eso (Colocación de streams).

Aquí mismo está el **Preset del codificador**; por defecto, **veryfast**.

© Flussonic 2026

Desentrelazado de entrada

`media.transcode.deinterlace`

El desentrelazado convierte los fotogramas entrelazados de entrada en progresivos. Trabaja con fotogramas decodificados, incluidas las entradas H.264 y MPEG-2. Es un ajuste del decodificador: las pistas de salida progresivas usan el resultado filtrado.

CONFIGURACIÓN

Defina `deinterlace` en `transcoder.decoder`. También está disponible en el bloque **Decodificador** de la pestaña **Transcodificador**.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    transcoder:
      decoder:
        deinterlace: auto
      output:
      - source: !content video
        codec: !set h264
```

Modo	Comportamiento
`auto`	Predeterminado: procesa solo los fotogramas marcados como entrelazados; los progresivos pasan sin filtro.
`force`	Procesa todos los fotogramas, aunque la fuente no indique el entrelazado.
`off`	Desactiva el filtro.

La frecuencia de fotogramas no cambia: se produce un fotograma progresivo por cada fotograma de entrada. Compruebe que no haya efecto peine en los objetos en movimiento y revise las propiedades de la pista de salida.

ENTRADA SIN COMPRIMIR

El modo también se aplica a la entrada sin comprimir SDI, ST 2110 y NDI. El barrido de esa entrada lo declara la propia señal: con `auto`, la entrada entrelazada se desentrelaza para las pistas progresivas y la progresiva pasa sin filtro.

Una pista sin comprimir `v210` o `uyvy422` para la salida SDI también toma los fotogramas después del desentrelazado. Con entrada entrelazada y `auto`, sale progresiva: 1080i50 se emite como 1080p25. Con `off`, sigue entrelazada, pero las pistas progresivas de la misma entrada muestran efecto peine.

La salida entrelazada se describe por separado: configuración interlace.

© Flussonic 2026

Salida entrelazada

`media.transcode.interlace`

Sapsan codifica H.264 entrelazado para receptores de difusión que necesitan 1080i o 576i. Esta pista usa fotogramas de la fuente anteriores al desentrelazado.

CONFIGURACIÓN

Aplique la configuración del stream mediante central. El ejemplo crea una pista H.264 entrelazada de 1080 líneas y conserva la pista de audio original. La fuente debe tener una frecuencia de fotogramas adecuada.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    transcoder:
      hardware: cpu
      output:
      - source: !content video
        codec: !set h264
        bitrate: 6000000
        params: !video
          interlace: true
          resize: !fit
            height: 1080
      - source: !content audio
        codec: same
```

En la consola de central, abra la pestaña **Transcodificador**, pulse **Más** en la calidad y active **Salida entrelazada**.

COMPORTAMIENTO Y LIMITACIONES

- **Orden de campos** — La entrada entrelazada conserva su orden de campos. Una entrada progresiva sin orden de campos se codifica con el campo inferior primero.
- **Procesamiento de campos** — El escalado y la conversión de color procesan cada campo por separado. Si la altura de una fuente 4:2:0 no es divisible por 4, los campos no pueden dividirse por igual: se procesa el fotograma entero con la advertencia `interlaced source cannot be split into fields`.
- **Escalera de calidades mixta** — La pista entrelazada recibe los fotogramas antes del filtro; las pistas progresivas, después. `transcoder.decoder.deinterlace` solo afecta a las salidas progresivas.

Solo se admite H.264 en CPU. La altura de salida debe ser divisible por 4. Sin escalado, se usa la altura de la fuente y se comprueba al iniciar.

Entrada sin comprimir

La salida entrelazada también funciona con una entrada entrelazada sin comprimir. El orden de campos lo declara la fuente:

- **SDI** — el modo de la tarjeta o del controlador; si no lo indica, el estándar de la trama: 1080i y PAL, campo superior primero; NTSC, campo inferior primero.
- **ST 2110** — el SDP del flujo: campo superior primero en cualquier trama, NTSC incluido (ST 2110-20).
- **NDI** — campo superior primero.

VERIFICACIÓN

Tras aplicar la configuración, compruebe que el transcodificador funcione y que la pista tenga la altura y el formato entrelazado necesarios. Compruebe el orden de campos y la fluidez en el receptor de destino. Los errores de configuración aparecen al guardar; los de inicio, en el estado del transcodificador. El orden de campos de la entrada se muestra en las estadísticas del stream, en el campo `media_info.tracks[].field_order`.

© Flussonic 2026

Superposición de logo

`media.transcode.logo`

El transcodificador de Sapsan recodifica el stream de entrada: cambia códecs, bitrate y resolución, y prepara una escalera de calidades multibitrate (MBR) a partir de una sola fuente.

SUPERPOSICIÓN DEL LOGOTIPO

El transcodificador superpone una imagen de logotipo al vídeo — el logotipo del canal en la esquina del fotograma o una marca de agua. El logotipo queda «grabado» en la pista de vídeo: se ve en cualquier dispositivo y llega al archivo. La sección `logo` vive en los parámetros de vídeo del peldaño y se aplica al fotograma terminado de cada peldaño por separado — después del escalado y de las bandas de `fit` —, así que el tamaño en píxeles de la imagen no depende de la resolución de salida.

En un clúster gestionado por central, utilice la biblioteca de logotipos como método principal de entrega. Cargue la imagen una vez en la consola de central y selecciónela en la tarjeta **Logotipo de la escalera**. Central guarda la imagen y la configuración del stream contiene `image: logo@<sha256>`. Sapsan descarga las imágenes que faltan mediante el canal autenticado de sincronización con central, verifica su SHA-256 y las guarda en una caché local. No hace falta copiar el archivo a cada streamer.

Para configurar mediante la API, envíe JSON con `name` y `bytes` (bytes de la imagen en base64 estándar) a `POST /central/api-v4/logos`, con autenticación de gestión. La respuesta contiene `key` y `created`; establezca `logo.image` como `logo@` seguido de esa clave. `GET /central/api-v4/logos` enumera las entradas y `GET /central/api-v4/logos/{key}` devuelve una entrada con la imagen en base64 para la vista previa. Una ruta local o una URL HTTP(S) son alternativas, también para un Sapsan independiente sin central.

En el ejemplo, sustituya `<sha256>` por la clave devuelta por central.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    transcoder:
      output:
      - source: !content video          # 1080p — la cima de la escalera
        codec: !set h264
        bitrate: 2800000
        params: !video
          logo:
            image: logo@<sha256>
            position: top_right
            x: 24
            y: 16
            height: 72
      - source: !content video          # 720p
        codec: !set h264
        bitrate: 1200000
        params: !video
          resize: !fit
            height: 720
          logo:
            image: logo@<sha256>
            position: top_right
            x: 16
            y: 10
            height: 48
```

Campo	Qué define
`image`	la dirección de la imagen: una ruta en el nodo, una URL http(s) o una referencia a una entrada de la biblioteca de logotipos `logo@<sha256>`
`position`	el punto de anclaje en el fotograma — uno de cinco, obligatorio
`x`, `y`	desplazamientos en píxeles desde el anclaje, 0 por defecto
`width`, `height`	el tamaño de la imagen en píxeles del fotograma del peldaño

Los anclajes son `top_left`, `top_right`, `bottom_left`, `bottom_right` y `center`. Los desplazamientos se cuentan desde el anclaje hacia dentro: en las esquinas, desde la esquina correspondiente; en `center`, un desplazamiento desde el centro.

El tamaño se da en píxeles del fotograma del peldaño. Con los dos campos — la imagen se ajusta a ellos; con uno — el otro se deriva de la proporción de la imagen, sin distorsión; con ninguno — la imagen se usa en su tamaño nativo. Para que el logotipo ocupe la misma porción del

© Flussonic 2026

fotograma en toda la escalera, los tamaños de los peldaños se calculan a partir de su altura: en el ejemplo anterior, 72 px a 1080p y 48 px a 720p son la misma porción del fotograma. Los peldaños sin sección `logo` se quedan como estaban.

Requisitos de la imagen:

- Formato — PNG o JPEG según la firma de los bytes. Un PNG con canal alfa conserva la transparencia: los píxeles transparentes no tocan el fotograma y los semitransparentes se mezclan con él.
- Límites — como máximo 1 MB de bytes y 4096 píxeles por lado.
- La URL se descarga al configurar el peldaño, con un tiempo de espera de 10 segundos.
- La misma imagen en varios peldaños se carga y se decodifica una sola vez.

Restricciones:

- El logotipo se superpone solo en una salida con codificación: un peldaño con `codec: same` y los códecs de emisión en bruto (`j2k` , `v210` , `uyvy422`) no lo soportan — esa sección se rechaza al escribir.
- `width` y `height` son enteros positivos no mayores que 4096; el cero y lo que supera el límite se rechazan al escribir, indicando el campo.
- La imagen se lee al montar el peldaño, no en cada fotograma: cambiar la dirección, el anclaje o los desplazamientos es una edición ordinaria de la configuración — el peldaño se rearma y el stream sigue.

> **Un logotipo roto es un error del peldaño**
>
> Si la imagen no se puede cargar o decodificar — no hay archivo, la URL no responde, se supera el límite, los bytes no se leen —, el peldaño no se arma: en el estado de la salida y en el registro aparece un error con nombre, mientras los demás peldaños y el propio stream siguen funcionando. El transcodificador nunca servirá el stream sin el logotipo en silencio. La recuperación es editar la dirección de la imagen: la nueva configuración rearma el peldaño sin reiniciar el proceso.

La referencia `logo@<sha256>` apunta a una entrada de la biblioteca de logotipos de central: el nodo descarga la imagen al aplicar el corte de configuración y entrega al transcodificador un archivo listo. En un nodo suelto (sin central) esa referencia da un error de peldaño con nombre — allí funcionan la ruta en el nodo y la URL http(s). En la consola de central el logotipo de la escalera se define con la tarjeta **Logotipo de la escalera** (Superposición de logotipo).

© Flussonic 2026

6.2.8 Publicidad (media.ads)

Scte35

`media.ads.scte35`

DESCRIPCIÓN GENERAL

Una marca publicitaria indica al equipo siguiente dónde empieza y dónde termina una pausa publicitaria en el stream. Sapsan acepta las marcas en la entrada, las lleva a través del stream junto con el vídeo y el audio, y las entrega en cada salida en el formato que esa salida entiende. Pase lo que pase con el stream en el camino — alineación, transcodificador, cambio de fuente —, el punto de empalme en la salida apunta al mismo fotograma que en la entrada.

Las marcas llegan por las entradas:

- **MPEG-TS** — secciones SCTE-35 en un PID con `stream_type 0x86`, véase Recepción de multicast y MPTS;
- **SDI** — mensajes SCTE-104 en VANC, véase Recepción SDI;
- **HLS** — etiquetas de la lista de reproducción `EXT-X-DATERANGE` con los atributos `SCTE35-OUT`, `SCTE35-IN`, `SCTE35-CMD`, y etiquetas `EXT-X-CUE-OUT` / `EXT-X-CUE-IN`.

Y salen:

- **a MPEG-TS** — como secciones SCTE-35 en su propio PID;
- **a SDI** — como mensajes SCTE-104 en el VANC del fotograma de salida;
- **a HLS** — como etiquetas `EXT-X-DATERANGE` y `EXT-X-CUE-OUT` / `EXT-X-CUE-OUT-CONT` / `EXT-X-CUE-IN`;
- **a DASH** — como eventos `EventStream` en el periodo del manifiesto;
- **al archivo** — junto con los segmentos, de modo que la lista del archivo lleva las mismas marcas que llevaba la lista en directo.

PAUSA PUBLICITARIA

Una fuente rara vez envía una sola marca por pausa. Primero llegan los anuncios — «salimos a publicidad en 8 segundos», a menudo repetidos varias veces —, después, a veces, una marca de salida inmediata, y minutos más tarde una marca de regreso. Sapsan reúne las marcas de un mismo evento (`splice_event_id`) en una sola pausa publicitaria y la recuerda mientras dura.

Si la fuente envió solo un anuncio y ningún empalme propio en el momento de salir, Sapsan pone el empalme por su cuenta, en el fotograma al que ha llegado el punto anunciado. Ese empalme se queda dentro de Sapsan: no sale ninguna marca adicional en ningún formato, y cada salida lleva exactamente las marcas que envió la fuente. Si el anuncio declara el regreso al cumplirse la duración (`auto_return`), Sapsan pone el regreso de la misma manera.

En cada empalme el segmento HLS y DASH se cierra en el primer fotograma clave no anterior al punto de empalme, sin esperar su duración habitual. Así la publicidad empieza con un segmento nuevo, y un reproductor o un sistema de inserción de publicidad en el servidor puede sustituir enteros los segmentos de la pausa.

Comprobación de la previsión. El punto de empalme del anuncio es una previsión. Sapsan la compara con el punto donde el segmento se cerró de verdad y muestra la diferencia. El límite cae en un fotograma clave, así que, si en el punto de empalme no hay fotograma clave, la diferencia llega hasta un GOP.

Cambio de fuente. Cuando el stream pasa a una fuente de respaldo, los empalmes que solo se esperaban se descartan: los anuncios de la fuente anterior no tienen nada que ver con la nueva. Una pausa ya empezada continúa hasta el regreso.

HLS

Una pausa sale como un solo rango `EXT-X-DATERANGE`: mientras dura, con su inicio y `SCTE35-OUT`; tras el regreso, el mismo rango recibe su fin, su duración y `SCTE35-IN`. Junto a él, la lista lleva `EXT-X-CUE-OUT` en el primer segmento de la pausa, `EXT-X-CUE-OUT-CONT` en los siguientes y `EXT-X-CUE-IN` en el primer segmento después de la pausa: unos reproductores y sistemas de inserción de publicidad leen los rangos, otros estas etiquetas.

En una entrada HLS las marcas se toman de las mismas etiquetas. Las etiquetas que ya estaban en la lista en el momento de conectarse cuentan como pasado y no se convierten en marcas. Si los segmentos de la fuente llevan su propio PID SCTE-35, las etiquetas de la lista no se usan: el tiempo de la marca dentro del segmento es más preciso.

© Flussonic 2026

LA PESTAÑA PUBLICIDAD

Lo que pasa con las marcas del stream se ve en la página del stream, pestaña **Publicidad**. Se actualiza sola cada dos segundos.

El bloque **Ahora** muestra:

- si el stream está **en pausa publicitaria** o **en programa**;
- la pausa en curso: número de evento, cuándo empezó, duración declarada y cuánto queda;
- los empalmes pendientes — anuncios cuyo punto aún no ha llegado;
- la última comprobación de la previsión: dónde se esperaba el empalme, dónde se cerró el segmento y cuánto se desviaron.

La tabla **Historial de marcas** enumera las marcas del stream, las nuevas arriba: hora, de dónde viene el registro — **entrada** (marca de la fuente) o **gestor** (empalme puesto por Sapsan, o cambio de fuente) —, tipo de marca, número de evento, punto de empalme, duración declarada y tipos de segmentación SCTE-35.

El historial vive en la memoria del streamer donde corre el stream y empieza de nuevo al reiniciarse. Si el stream no está en marcha, la pestaña lo dice. Las marcas en sí no se pierden: se graban en el archivo junto con los segmentos.

CONTADORES

Las marcas aceptadas y perdidas en la entrada y en la salida las cuentan los contadores de Prometheus: `stream_input_ad_marks_total`, `stream_input_errors_ad_marks_dropped_total`, `stream_out_ad_marks_total`, `stream_out_errors_ad_marks_lost_total`. Una marca perdida cuenta en el total de errores de la entrada, véase Errores de entrada.

© Flussonic 2026

SSAI

`media.ads.ssai`

Server-Side Ad Insertion (SSAI) sustituye segmentos del stream por anuncios en el servidor. Los anuncios pueden seleccionarse por sesión usando los datos de autorización del espectador. La inserción se activa por horario o marcas SCTE.

Los anuncios se transcodifican previamente con los perfiles de salida del stream. El servidor crea un manifiesto para la sesión y el reproductor recibe contenido y anuncios por el mismo mecanismo. Un manifiesto personalizado no debe almacenarse en caché como común para todos los espectadores.

© Flussonic 2026

Bypass

`media.ads.bypass`

DESCRIPCIÓN GENERAL

Una marca publicitaria indica al equipo siguiente dónde empieza y dónde termina una pausa publicitaria en el stream. Sapsan acepta las marcas en la entrada, las lleva a través del stream junto con el vídeo y el audio, y las entrega en cada salida en el formato que esa salida entiende. Pase lo que pase con el stream en el camino — alineación, transcodificador, cambio de fuente —, el punto de empalme en la salida apunta al mismo fotograma que en la entrada.

Las marcas llegan por las entradas:

- **MPEG-TS** — secciones SCTE-35 en un PID con `stream_type 0x86`, véase Recepción de multicast y MPTS;
- **SDI** — mensajes SCTE-104 en VANC, véase Recepción SDI;
- **HLS** — etiquetas de la lista de reproducción `EXT-X-DATERANGE` con los atributos `SCTE35-OUT`, `SCTE35-IN`, `SCTE35-CMD`, y etiquetas `EXT-X-CUE-OUT` / `EXT-X-CUE-IN`.

Y salen:

- **a MPEG-TS** — como secciones SCTE-35 en su propio PID;
- **a SDI** — como mensajes SCTE-104 en el VANC del fotograma de salida;
- **a HLS** — como etiquetas `EXT-X-DATERANGE` y `EXT-X-CUE-OUT` / `EXT-X-CUE-OUT-CONT` / `EXT-X-CUE-IN`;
- **a DASH** — como eventos `EventStream` en el periodo del manifiesto;
- **al archivo** — junto con los segmentos, de modo que la lista del archivo lleva las mismas marcas que llevaba la lista en directo.

© Flussonic 2026

6.2.9 Almacenamiento del archivo (media.dvr)

Acceso remoto al archivo

`media.dvr.remote`

Sapsan reúne las grabaciones de un mismo stream almacenadas en distintos servidores del clúster en un archivo común disponible para el espectador. El streamer que atiende la solicitud localiza el fragmento necesario en otros servidores y lo obtiene por la red. El reproductor no necesita saber qué servidor grabó el momento seleccionado ni en qué disco se guarda.

Esto permite trasladar una cámara entre streamers conservando el acceso a su historial. Las grabaciones antiguas permanecen en el servidor anterior y las nuevas aparecen en el nuevo. **No es necesario copiar previamente el archivo para reunir el historial.**

LA CÁMARA SE HA TRASLADADO Y EL SERVIDOR ANTERIOR HA VUELTO

Supongamos que la cámara `cam1` se grababa en el streamer A. Tras su fallo, la captura y la grabación continuaron en el streamer B. Más tarde, A se reparó y volvió al clúster con sus discos conservados.

Etapa	Dónde están las grabaciones	Cómo se accede al archivo
La cámara se graba en A	En A	Las solicitudes leen el archivo de A.
A no está disponible y la cámara se graba en B	Historial anterior en A y nuevas grabaciones en B	Se puede acceder a las grabaciones de B y a otras copias disponibles.
A vuelve y recibe la asignación de entrega de restos del archivo	El historial anterior permanece en A y las nuevas grabaciones en B	La reproducción obtiene los fragmentos necesarios de ambos servidores como historial de un mismo stream.

Cuando A vuelve, no es necesario esperar a que todo su historial se copie en B. El clúster utiliza las grabaciones conservadas donde están: las solicitudes de momentos antiguos leen fragmentos de A y las de momentos nuevos, fragmentos de B. Para ello, A no necesita volver a conectarse a la cámara ni reanudar la grabación.

Un traslado planificado funciona igual: el streamer anterior puede permanecer en el clúster como fuente del archivo antiguo desde el principio. Si una cámara se ha grabado sucesivamente en varios servidores, su historial puede incluir las grabaciones de cada uno de ellos.

CÓMO ENCUENTRA EL CLÚSTER LAS GRABACIONES ANTERIORES

Central registra qué streamers tienen el archivo de un stream. Tras el traslado, el archivo conservado en la máquina anterior se sirve mediante una asignación de **entrega de restos del archivo** — `dvr_parts`.

Esta asignación mantiene el stream en el servidor para leer lo ya grabado. No captura la fuente ni crea nuevas grabaciones, pero permite a otros streamers solicitar sus fragmentos. El almacenamiento sigue sujeto a la profundidad del archivo del stream.

Central establece las conexiones con los servidores de archivo y autoriza las solicitudes entre streamers. Trasladar una cámara no requiere preparar manualmente para el espectador una nueva lista de las direcciones de sus servidores anteriores.

El streamer que vuelve debe participar de nuevo en el funcionamiento del clúster y cumplir los requisitos para entregar restos del archivo. En particular, se tienen en cuenta el espacio de nombres y las etiquetas del stream, así como una dirección interna del clúster accesible. Si el archivo solo tiene asignado `dvr_cleanup`, se limpia según las reglas de retención, pero no se incluye en las lecturas remotas. Las condiciones de asignación se describen en restos del archivo, y las direcciones, en gestión de streamers.

DIRECCIONAMIENTO COMÚN DE FRAGMENTOS

Un fragmento se identifica por el nombre del stream, la pista y una marca de tiempo entera con su `timescale`. Su dirección no depende del disco físico ni del servidor que lo grabó. Consulte el direccionamiento del archivo.

Por tanto, una solicitud del «fragmento de la cámara `cam1` en el momento seleccionado» conserva su significado después de trasladar la cámara. Cambia el servidor que proporciona los bytes, no la identidad de la grabación.

© Flussonic 2026

Al construir la respuesta, Sapsan realiza varias operaciones:

- **Combina intervalos** — reúne los intervalos disponibles de los archivos locales y remotos. Los huecos reales sin grabación se conservan: combinarlos no los convierte en vídeo grabado.
- **Construye listas de fragmentos** — reúne los resultados de los servidores, los ordena por tiempo y elimina las entradas duplicadas de un mismo fragmento.
- **Lee los datos** — utiliza una fuente local disponible u obtiene el fragmento necesario de un streamer remoto, junto con la información necesaria para reproducirlo.

Para el reproductor, el archivo sigue siendo el del mismo stream. Sapsan gestiona el paso entre servidores de almacenamiento; el cliente no necesita combinar por su cuenta los archivos de A y B.

ACCESO SIN TRASLADO PREVIO DE DATOS

Reunir el historial no requiere una tarea para trasladar todo el archivo, espacio adicional para una copia completa en el nuevo servidor ni volver a grabar el vídeo antiguo. Los streamers transfieren los datos solicitados a medida que se leen. Volver a hacer accesible un archivo grande no depende del tiempo que llevaría copiarlo por completo.

La lectura remota puede interactuar con otras dos funciones:

- **Replicación del archivo** — crea copias locales. En el pipeline actual, un streamer con grabación DVR configurada y discos de archivo puede guardar en su propio archivo los fragmentos leídos remotamente. Esta replicación bajo demanda no es un requisito previo para acceder al historial.
- **Caché del archivo** — guarda los datos leídos en discos de caché para solicitudes posteriores y reduce el tráfico de red.

Por tanto, que no haya una migración obligatoria no impide crear copias locales durante la reproducción. El acceso transparente permite utilizar el archivo donde ya está; la replicación y la caché modifican además la ubicación de los datos leídos.

DISPONIBILIDAD DEL HISTORIAL

Está disponible el material que se ha conservado en el archivo y reside en una fuente accesible. Mientras el único servidor con un tramo antiguo esté apagado, sus discos no pueden leerse por la red. Cuando el servidor vuelve y reanuda la entrega de grabaciones, estas se incorporan de nuevo al historial común, dentro del periodo de retención restante.

Para acceder a las grabaciones también durante un fallo del servidor, se necesitan copias disponibles del archivo. La lectura remota utiliza esas copias, pero no crea grabaciones que no existen ni recupera intervalos durante los que nadie grabó la cámara.

Tras trasladar la cámara y recuperar el servidor, compruebe lo siguiente:

1. Se conserva el nombre del mismo stream: el archivo está vinculado a su identidad, no solo a la dirección de la cámara.
2. El servidor anterior es visible en el clúster y tiene una asignación de entrega de restos del archivo para ese stream.
3. Su dirección interna del clúster es accesible desde otros streamers y la autorización entre servidores funciona.
4. En el historial aparecen los intervalos antiguos y los nuevos, y se reproducen momentos de ambos lados del traslado.

La configuración de grabación se describe en DVR, y la selección del momento de reproducción, en reproducción del archivo.

© Flussonic 2026

Grabación del archivo

`media.dvr.record`

El DVR de Sapsan es un almacenamiento propio en disco: el stream se escribe append-only en archivos por horas, por pista, en varios discos a la vez. La lectura del archivo no interfiere con la escritura.

RENDIMIENTO Y DIRECCIONAMIENTO ESTABLE

El almacenamiento se ha probado con éxito a 25 Gbit/s de escritura en discos mecánicos, simultáneamente con la limpieza del archivo y la lectura. Es el resultado de una configuración probada; dimensione el hardware para la carga conjunta de escritura y lectura.

Cada fragmento y cada fotograma tiene una dirección estable: nombre del stream, identificador de pista y timestamp entero. La línea de tiempo usa ticks enteros y su frecuencia (`timescale`), sin redondear a segundos fraccionarios. La dirección no depende del disco o servidor físico y se conserva en todo el clúster, durante la replicación y la migración del archivo.

La analítica de vídeo con IA puede guardar referencias a eventos detectados, y los sistemas ERP pueden acceder después a los mismos fotogramas individuales. Un fotograma de una pista JPEG usa `/streaming/v/<stream>/image/<track>/<utc_ms>.jpg` ; las direcciones de fragmentos incluyen la pista y un timestamp de la forma `<ticks>-<timescale>` . La disponibilidad de los datos depende de la retención del archivo y de los permisos de acceso.

ALMACENAMIENTO

La sección global `dvr` describe el almacenamiento entero:

```
dvr:
  root: /storage           # raíz del archivo
  catalog: /storage/catalog # catálogo de blobs (por defecto <root>/catalog)
  disks:
  - path: d1               # discos relativos a root
  - path: d2
  - path: d3
```

Parámetros del disco:

Parámetro	Descripción
`path`	ruta del disco relativa a `root`
`mode`	`Active` (por defecto) o `Degraded` — el disco se lee, pero no se escribe en él
`min_free_bytes`	reserva mínima de espacio libre

`check_mount` está activado por defecto y exige que la raíz de un disco de archivo esté en un sistema de archivos montado y no en la partición `/` : Sapsan no escribe en un disco que no lo cumple, y los demás discos siguen escribiendo. La raíz no tiene que ser por sí misma un punto de montaje: los directorios dentro de un volumen montado pasan la comprobación. Una vez corregido el montaje de los discos, reinicie Sapsan.

> **Warning**
>
> Los discos de escritura se enumeran explícitamente. `root` es la casa del catálogo y la base de las rutas relativas, no un disco: con `disks` vacío, la escritura de cada fragmento termina en un error visible en las estadísticas, y en `root` no se escribe nada.

Además de los discos de archivo, el almacenamiento puede llevar discos de caché (`cache_disks`) — en ellos se asientan las partes del archivo que se leyeron: el ajeno traído de otro servidor o el propio levantado de los discos de archivo. La grabación en vivo nunca los elige, y su espacio no lo libera la limpieza, sino el desalojo según la antigüedad del último acceso. Eso es la caché del archivo en un nodo edge.

ACTIVAR LA GRABACIÓN EN UN STREAM

```
streams:
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://admin:password@10.0.0.5/stream0
    dvr:
      max_depth: 168        # profundidad del archivo, horas
```


© Flussonic 2026

`dvr: {}` activa la grabación con los ajustes por defecto.

REPARTO DE LA ESCRITURA ENTRE DISCOS

Sapsan reparte solo la grabación entre los discos activos:

- las horas consecutivas de un mismo stream se alternan entre discos;
- los streams vecinos se reparten de forma pareja entre los discos;
- las pistas de un stream multibitrate se escriben en paralelo en discos distintos.

Un disco en modo `degraded` sigue sirviendo lecturas pero no recibe datos nuevos — así se retira un disco sin perder el archivo.

CÓMO FUNCIONA EL ALMACENAMIENTO

El archivo está dispuesto en los discos como `<disk>/<hash>/<stream>/<Y>/<M>/<D>/<hour>.mp4`. Los ficheros solo se añaden, nunca se reescriben; los segmentos init idénticos se deduplican dentro de un blob.

Un indexador en segundo plano sincroniza sin descanso el catálogo con los discos y cura las divergencias sin intervención del administrador:

- los blobs que aparecieron en el disco a espaldas del catálogo (por ejemplo, tras mover discos de otro servidor) se añaden al catálogo;
- los registros de ficheros desaparecidos se eliminan;
- el índice sidecar de un blob dañado se reconstruye escaneando el propio blob;
- los tamaños en bytes y los filtros bloom de los blobs se rellenan en segundo plano (a los blobs más jóvenes de una hora no se les toca).

Las lecturas toleran fallos: si el catálogo apunta al disco equivocado, el fragmento se busca en todos los discos de esa hora; los registros huérfanos del catálogo se saltan sin error.

OBSERVABILIDAD

El DVR entrega estadísticas por disco (espacio libre/total, contadores de operaciones y de bytes de lectura/escritura, número de blobs, bytes ocupados por el archivo) y del catálogo — véanse el Admin API y la monitorización.

SIGUIENTES PASOS

- Acceso remoto al archivo
- Reproducir el archivo
- Cachear el archivo ajeno en un edge
- Obtener capturas del archivo

© Flussonic 2026

Retención del archivo

`media.dvr.depth`

El DVR de Sapsan es un almacenamiento propio en disco: el stream se escribe append-only en archivos por horas, por pista, en varios discos a la vez. La lectura del archivo no interfiere con la escritura.

LIMPIEZA Y RETENCIÓN

La limpieza se ejecuta una vez por hora (5 minutos después del cambio de hora):

- `max_depth` actúa con granularidad de hora: solo se borran las horas plenamente vencidas;
- los blobs con episodios protegidos nunca se borran por retención;
- el blob de la hora actual (la que se está escribiendo) nunca se borra.

La protección contra el desbordamiento del disco funciona aparte: cuando el espacio libre cae por debajo de `min_free_bytes` (por defecto, el 1 % de la capacidad del disco), Sapsan borra los blobs más viejos **al margen de la retención** hasta liberar espacio suficiente.

RETENCIÓN DE EPISODIOS EN CENTRAL

Central recopila el último estado de los episodios y sus imágenes de los streamers y los guarda en PostgreSQL. Los episodios retenidos siguen disponibles cuando el streamer de origen no responde. Este almacenamiento es independiente de la grabación de vídeo.

En **DVR** del stream o de su plantilla, **Retención adicional de episodios (horas)** configura `dvr.episodes_additional_depth`; la API y YAML usan segundos. El total es `dvr.max_depth + dvr.episodes_additional_depth`. La profundidad adicional no definida equivale a cero. Si existe la sección DVR pero falta `max_depth`, se usan diez años; un cero explícito sigue siendo cero. Sin sección DVR, o con un total de cero, central no retiene episodios.

El plazo empieza en `closed_at` para episodios cerrados y en `updated_at` para los abiertos. Cambiar la configuración DVR efectiva, incluida la plantilla, recalcula la retención de todos los episodios restantes. La sección DVR propia del stream sustituye por completo la sección de la plantilla. Eliminar un stream elimina sus episodios e imágenes. Aumentar la retención no recupera episodios ya eliminados físicamente por la limpieza en segundo plano.

© Flussonic 2026

Replicación del archivo

`media.dvr.replication`

La replicación guarda copias adicionales de los fragmentos del archivo en otro streamer. La reproducción transparente del historial que permanece en los servidores anteriores después de trasladar una cámara se describe en Acceso remoto al archivo.

Las tareas típicas que esto resuelve:

- migrar a un servidor nuevo sin perder historia — el nuevo graba el archivo fresco, el viejo conserva el pasado;
- leer el archivo de servidores réplica;
- recuperarse tras sustituir un disco.

En un clúster, el layouter de central gestiona la autorización de las solicitudes entre servidores. Genera la configuración de comunicación; no es necesario asignar permisos manuales a cada par de servidores.

CONFIGURACIÓN

```
streams:
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://10.0.0.5/stream0
    dvr: {}
    remotes:
    - url: http://old-server:5080
      timeout_ms: 5000
```

Parámetro	Descripción
`url`	dirección del servidor Sapsan remoto donde está el archivo de este stream
`timeout_ms`	tiempo de espera de las solicitudes al servidor remoto

Puede haber varios servidores remotos — Sapsan los interroga todos.

REPLICACIÓN DIFERIDA

Si la grabación DVR está configurada y hay discos de archivo, los fragmentos históricos leídos de servidores remotos pueden guardarse en el archivo local. El contenido cifrado y los fragmentos del extremo del directo quedan excluidos de esta escritura. El archivo local incorpora progresivamente las partes del historial remoto que ven los espectadores; un error al guardarlas no impide su entrega al espectador.

La replicación escribe la historia ajena en los discos de **archivo** y la mantiene ahí hasta que la retención la limpia. Si el servidor no debe llevarse el archivo consigo, sino solo ahorrar red en lo popular, eso es la caché del archivo: discos de caché aparte, desalojo por antigüedad del último acceso y funcionamiento sin discos de archivo en absoluto.

COMPORTAMIENTO ANTE FALLOS

- Si un servidor remoto no está accesible pero los datos existen localmente, la reproducción continúa; el problema solo se registra en el log.
- Sapsan memoriza qué rangos faltan en un remoto (caché negativa) y no vuelve a interrogarlo por los datos ausentes.
- Los fragmentos init de los streams remotos se cachean.

CÓMO SE LOGRA LA UNIÓN SIN COSTURA

En cada acceso al archivo Sapsan fusiona los datos locales y los remotos:

- **Rangos del archivo** — se combinan los intervalos disponibles de todos los servidores. Se conservan los huecos reales sin grabación.
- **Playlists** — las listas de fragmentos del DVR local y de los servidores remotos se combinan, se ordenan y se deduplican. Las grabaciones ausentes no se reconstruyen.
- **Fragmentos** — se leen en cascada: búfer del directo → DVR local → servidor remoto. La URL del fragmento es la misma para el reproductor independientemente de dónde vivan los datos.

© Flussonic 2026

Esto es posible gracias al direccionamiento estable: el fragmento se identifica por el nombre del stream, la pista y un timestamp entero con su timescale. La dirección se conserva en todos los servidores durante la replicación y la organización en clúster, sin recalcular la línea de tiempo al unir archivos.

SIGUIENTES PASOS

- Configurar la grabación DVR
- Cachear el archivo ajeno en un edge
- Reproducir el archivo

© Flussonic 2026

Object Storage

`media.dvr.object_storage`

La configuración siguiente corresponde al almacenamiento de objetos en Flussonic Media Server. Para el archivo y la caché de Sapsan, consulte DVR y caché del archivo.

El archivo puede grabarse en almacenamiento de objetos. En FMS, `s3://` selecciona S3 por HTTP y `s3s://` por HTTPS. Se indican el bucket y el límite de tamaño del archivo.

`dvr /storage copy=s3://USER:PASSWORD@HOST/BUCKET 10G;` graba primero en disco local y transfiere los datos cada hora. Tras migrar el stream, otro servidor puede continuar en el mismo directorio. Varios servidores no deben grabar el mismo stream simultáneamente.

© Flussonic 2026

Caché del archivo

`media.dvr.cache`

Un nodo que sirve el archivo ajeno en lugar del propio — a través de remotes o de una fuente legacy — va a la fuente por absolutamente cada fragmento. La caché del archivo cambia esto: el fragmento leído se asienta en un disco de caché local, y la siguiente solicitud de la misma parte del archivo se sirve localmente, sin ida y vuelta por la red.

Así se construye un edge de CDN: el origin guarda el archivo, los edges guardan lo que los espectadores de verdad ven.

La caché es transparente. Cambia de dónde vienen los bytes, no lo que se sirve: las URL de los fragmentos, las playlists y las fronteras del archivo siguen exactamente iguales para el espectador.

QUÉ DISTINGUE A UN DISCO DE CACHÉ DE UN DISCO DE ARCHIVO

Un disco de caché es un *papel* del disco, no una bandera en él:

- **la grabación en vivo nunca lo elige** — el archivo se escribe solo en los discos listados en `disks`;
- **la retención del stream no lo gobierna** — `max_depth` del stream no se aplica a la caché;
- **el espacio lo libera el desalojo** — según cuánto hace que se accedió por última vez a un blob, no según la antigüedad de su contenido;
- **está aislado en el catálogo** — los blobs de caché viven en sus propias tablas, y los recorridos del archivo (limpieza, backfill, el reparto del espacio ocupado en cubos) no los ven en absoluto.

Por eso la caché de un edge no aparece en la consola como un archivo huérfano, y por eso el desalojo de la caché no compite con la grabación del archivo por el mismo mutex.

CONFIGURACIÓN

La caché se configura en dos sitios: los discos en el nodo, el interruptor en el stream.

```
dvr:
  root: /storage
  cache_disks:
  - path: ssd1              # relativo a root, como un disco de archivo
    max_bytes: 536870912000 # techo de la caché en este disco, bytes
    min_free_bytes: 10737418240
  - path: ssd2

streams:
  - name: cam1
    remotes:
    - url: http://origin:5080
    cache: {}               # activa el cacheo de las lecturas del archivo
```

Parámetros del disco de caché:

Parámetro	Descripción
`path`	ruta del disco relativa a `root`
`max_bytes`	cuánto puede ocupar la caché en este disco; vacío — limitada solo por el espacio libre
`min_free_bytes`	cuánto espacio libre dejar en el volumen

Los límites son independientes: el desalojo funciona hasta que el disco cumpla ambos. El disco de caché no tiene `mode` — `Degraded` describe a un miembro del RAID del archivo, y la caché no replica nada.

La sección `cache` del stream es una **sección de primer nivel del documento, hermana de `dvr`**, no un campo dentro de esta: el cacheo y la grabación son propiedades independientes. Un edge cachea el archivo ajeno sin grabar el propio. La sección no tiene parámetros; su presencia activa el cacheo.

La validación de la configuración rechaza:

- una ruta listada a la vez en `disks` y en `cache_disks`;
- un duplicado dentro de `cache_disks`;
- un disco de caché cuya ruta sea igual a `root` — el desalojo borraría la casa del catálogo.

© Flussonic 2026

Rutas distintas en el mismo volumen físico son una configuración válida, pero al arrancar se registra un aviso: el desalojo libera espacio que el archivo ocupa de inmediato.

Una sección `cache` en un nodo sin discos de caché es un no-op válido con un aviso en el log. Es normal en un clúster, donde un mismo documento de stream se reparte entre nodos distintos.

UN EDGE PURO

Un almacenamiento hecho de `root` y discos de caché solamente, sin discos de archivo, es una configuración válida: hay catálogo, no hay grabación en vivo, y el archivo se va trayendo de los remotes.

```
dvr:
  root: /storage
  cache_disks:
  - path: ssd1
    max_bytes: 536870912000

streams:
  - name: cam1
    remotes:
    - url: http://origin:5080
    cache: {}
```

> **Warning**
>
> `root` es la casa del catálogo, no un disco de escritura. Una lista `disks` vacía solía significar «escribir directamente en `root`»; ahora es un error de escritura visible en las estadísticas, en lugar de una escritura silenciosa por fuera del RAID. Si el nodo debe grabar un archivo, enumere los discos explícitamente en `disks`.

ADMISIÓN EN LA CACHÉ

Cachearlo todo desde la primera solicitud significa desalojar lo popular en favor de lo aleatorio: un solo salto hacia lo profundo del archivo echaría del disco lo que todo el mundo está viendo. Por eso la admisión tiene dos ejes, y se configuran en el nodo, para la caché entera:

Parámetro	Descripción
`cache_fresh_age_secs`	un fragmento más joven que esta profundidad (segundos hacia atrás desde ahora) se cachea en la primera solicitud; por defecto, un día
`cache_admission_hits`	a partir de qué solicitud repetida empieza a cachearse un fragmento más antiguo que la frontera de frescura; por defecto, la segunda; `1` desactiva la barrera

```
dvr:
  root: /storage
  cache_fresh_age_secs: 86400
  cache_admission_hits: 2
  cache_disks:
  - path: ssd1
```

Casi todo el mundo ve la cola fresca del archivo, así que entra en la caché de inmediato. Las profundidades del archivo interesan a unos pocos espectadores — entran en la caché solo cuando se las vuelve a pedir.

ORDEN DE LAS FUENTES DE LECTURA

Un fragmento del archivo se busca a lo largo de una cadena, y la primera fuente que responde cierra la solicitud:

1. el segmentador del stream en vivo;
2. **la caché**;
3. el archivo local;
4. el clúster (remotes);
5. la fuente legacy `old_m4f`;
6. el archivo legacy en disco.

Un acierto de caché responde sin tocar nada por debajo. Si un fragmento está a la vez en un disco de archivo y en un disco de caché, la lectura la sirve la caché: la caché en SSD gana a los discos mecánicos. Un stream sin sección `cache` no tiene ningún paso de caché en la cadena.

© Flussonic 2026

QUÉ PONE UN FALLO EN LA CACHÉ

Una lectura con éxito de una fuente por debajo de la caché se escribe en un disco de caché junto con su fragmento init. Las diferencias respecto a la grabación del archivo:

- **el recorte de `max_depth` no se aplica** — en la caché entra lo que pidió el espectador, aunque sea más profundo que cualquier profundidad de archivo configurada;
- **un segmento multipista se guarda entero, con todas las pistas** — así el cambio de bitrate de un espectador lo sirven aciertos y no fallos;
- **la escritura es idempotente** — varios espectadores que fallen a la vez sobre el mismo fragmento dejan exactamente una copia en la caché;
- **un error de escritura en la caché no afecta a la respuesta del cliente** — este ya tiene los bytes; el error solo se registra y se cuenta en las métricas.

El texto cifrado se guarda en la caché tal cual, junto con su fragmento init: el edge no tiene con qué descifrarlo y entrega exactamente los bytes que llegaron. Ese medio se reconoce por la descripción de su pista — el esquema y el identificador de la clave llegan a `MediaInfo` desde el fragmento init y forman parte de la identidad de la pista igual que el códec, así que un cambio de clave se resuelve con el mismo mecanismo que un cambio de códec. El texto cifrado nunca llega al archivo: el archivo alimenta vistas previas, miniaturas y la línea de tiempo, y todas ellas leen dentro de la muestra.

LECTURA ANTICIPADA

Ver el archivo es secuencial, así que tras servir una lectura Sapsan descarga en segundo plano el fragmento siguiente de la misma pista, si todavía no está en la caché. La lectura anticipada funciona en un edge puro y al leer de una fuente remota, se limita a un fragmento por delante y nunca retrasa la respuesta al cliente.

Esto no es un precalentamiento: aquí no existe rellenar la caché por calendario ni por EPG.

DESALOJO

El espacio en un disco de caché lo liberan blobs horarios enteros, en orden de último acceso — el primero en irse es el usado menos recientemente — hasta que el disco vuelve a entrar en ambos límites.

- La edad del contenido no es un criterio: el programa popular de ayer sobrevive al impopular de hoy, y la hora actual se desaloja en igualdad de condiciones con las demás.
- La única excepción es un blob al que un escritor está añadiendo datos justo ahora: ese se queda, y en su lugar se va el siguiente menos recientemente usado.
- La limpieza del archivo (`max_depth` del stream) y la protección de episodios no consideran los discos de caché en absoluto.

El orden de desalojo lo mantiene su propio índice del catálogo y sobrevive a un reinicio. La marca de acceso se vuelca al catálogo de forma diferida — como mucho una vez cada 10 minutos por blob — para que la lectura no se convierta en escritura.

La contabilidad de la caché (volumen ocupado, tiempos de acceso) se restaura al arrancar mediante un escaneo de rangos del catálogo, sin recorrer los ficheros de datos. Un disco de caché borrado es un estado frío válido: el servidor arranca, la caché se llena desde cero, el archivo queda intacto.

LÍNEA DE TIEMPO Y PROFUNDIDAD EN UN EDGE

La profundidad del rebobinado y los rangos del archivo de un stream cacheado se calculan como la unión del catálogo local (archivo y caché) y de todas las fuentes remotas — y se anuncian en las playlists de rebobinado y en las respuestas sobre los rangos del archivo. Un espectador en el edge ve la misma profundidad que en el origin.

Las ventanas de las fuentes se fusionan **en una lista, conservando los huecos entre ellas**: un intervalo continuo desde el inicio hasta el fin de una fuente no debe sustituir a esa lista — de lo contrario la línea de tiempo pinta de verde intervalos sin datos, y un clic en ellos acaba en un 404. Los huecos no mayores que la resolución solicitada se siguen fusionando por la regla general.

Si el origin no está accesible pero el rango pedido está entero en la caché, la playlist se construye a partir del catálogo local y los fragmentos se sirven de la caché — la reproducción continúa.

OBSERVABILIDAD

Cada **lectura del archivo** se atribuye al eslabón de la cadena que la sirvió:

- **cache** — la sirvió el disco de caché del nodo (un acierto);
- **local** — la sirvió el disco de archivo del nodo;
- **remote** — hubo que ir a una fuente remota (un fallo).

© Flussonic 2026

Solo la lectura de un fragmento por un cliente cuenta como lectura del archivo. La salida del segmentador en vivo y las descargas de lectura anticipada no entran en la atribución — de lo contrario, la lectura anticipada inflaría la tasa de aciertos cuanto mejor funcionara; las descargas se cuentan aparte.

A Prometheus y a local-prom van los contadores acumulativos:

Métrica	Significado
`dvr_archive_reads_total{source}`	lecturas del archivo según los ejes `cache` / `local` / `remote`
`dvr_archive_read_bytes_total{source}`	bytes de esas lecturas
`dvr_read_ahead_fetches_total`	descargas de lectura anticipada
`dvr_disk_reads_total{disk_id}`, `dvr_disk_read_bytes_total{disk_id}`	accesos al disco; en un disco de caché son sus aciertos
`dvr_disk_cache_bytes{disk_id}`	volumen de caché en el disco

La dirección local `GET /streamer/api-v4/dvr` reporta el papel, el volumen y el límite de cada disco de caché, y las velocidades de lectura del nodo según los tres ejes. Los contadores acumulativos no se exponen en la management API: ahí van velocidades y gauges.

La tasa de aciertos nunca se entrega como un número listo — es `cache / (cache + local + remote)`, y la calcula el consumidor: las velocidades se suman, la división va después. Un promedio de promedios miente cuando los nodos pesan distinto.

Para configurar el clúster, vea discos de caché y direcciones de entrega. El rol de asignación edge describe la entrega desde un streamer edge.

EN QUÉ SE DIFERENCIA DE LA REPLICACIÓN DIFERIDA POR REMOTES

Los remotes tienen un comportamiento parecido: un fragmento leído de un servidor remoto se añade al archivo local. La diferencia es fundamental:

	Replicación diferida por remotes	Caché del archivo
Dónde se escribe	en discos de archivo	en discos de caché
Para qué	traer aquí para siempre la historia ajena	servir localmente las solicitudes repetidas
Cuándo se libera	con la limpieza según la retención del stream	con el desalojo según el último acceso
Discos de archivo necesarios	sí	no, un edge puede funcionar sin ellos

La replicación consiste en reubicar el archivo; la caché, en ahorrar red en lo popular.

QUÉ VIENE DESPUÉS

- El archivo de otro servidor (remotes)
- Grabación DVR
- Reproducción del archivo

© Flussonic 2026

Catchup

`media.dvr.catchup`

GUÍA DE PROGRAMACIÓN

El botón amarillo abre la parrilla. Los canales a la izquierda, sus programas a la derecha; las flechas recorren la parrilla y **OK** abre el programa elegido.

Lo que muestra la parrilla depende por completo del horario cargado: sin fuente de la guía la columna de programas está vacía, y eso son datos que faltan, no una vitrina rota — véase Guía de programación.

Los programas que no están en el archivo no se ofrecen. La regla es sencilla: un programa está disponible si su intervalo **se solapa** con la ventana de archivo del canal. El motivo del rechazo se dice con palabras:

- **This channel is not recorded, so only live is available** — el stream no tiene archivo.
- **This programme is no longer in the archive** — el programa ha quedado fuera de la profundidad de grabación.
- **This channel has no broadcast configured** — el canal no tiene stream asignado.

Los límites del archivo llegan junto con la lista de canales, así que la decisión se toma antes de pulsar: el espectador no se topa con una pantalla en negro.

© Flussonic 2026

Exportación del archivo

`media.dvr.export`

Un segmento del archivo grabado se descarga como un único archivo MP4 que se abre en los reproductores habituales, por ejemplo en VLC. El archivo MP4 se ensambla a partir de la grabación en el momento de la solicitud; no hace falta preparar nada de antemano.

La exportación funciona solo en streams con la grabación activada (`dvr`).

URL DE EXPORTACIÓN

```
http://server/streaming/v/<stream>/archive-<from>-<duration>.mp4
```

- `from` — inicio del segmento, UTC en segundos;
- `duration` — duración del segmento en segundos.

Por ejemplo, `archive-1789546139-600.mp4` son diez minutos de archivo a partir del momento indicado.

Si el stream requiere autorización, el token se pasa igual que para la reproducción del archivo, por ejemplo `?token=<token>` — consulte Autorización.

SELECCIÓN DE PISTAS

Sin parámetros, el archivo MP4 recibe todas las pistas de vídeo y audio del segmento. Las pistas se eligen con el parámetro `filter.tracks` :

```
http://server/streaming/v/<stream>/archive-<from>-<duration>.mp4?filter.tracks=v1a1
```

Las pistas se nombran igual que en las listas de reproducción: `v1` , `v2` son vídeo, `a1` es audio. Si ninguna pista de vídeo entra en el archivo MP4, se entrega como `audio/mp4` .

Los subtítulos y las pistas de servicio aún no se admiten en la exportación; una solicitud de una pista así se rechaza con `422` .

QUÉ ENTRA EN EL ARCHIVO MP4

- **Inicio** — el archivo MP4 empieza en el fotograma clave anterior o igual al momento solicitado, por lo que puede durar unos segundos más de lo pedido. Si el inicio cae en un hueco de la grabación, empieza en el primer fotograma clave después del hueco.
- **Huecos de la grabación** — se unen: el archivo MP4 contiene solo tiempo grabado y puede ser más corto que el segmento solicitado.
- **Cambio de parámetros del stream** — si los parámetros de una pista cambiaron dentro del segmento, por ejemplo la resolución de la cámara, el archivo MP4 termina en el punto del cambio.

El servidor propone el nombre del archivo en la cabecera `Content-Disposition` : `<stream>_<AAAA-MM-DD>_<HH-MM-SS>Z_<segundos>s.mp4` . La hora del nombre está en UTC y la duración es la realmente grabada.

El tamaño se conoce antes de descargar: una solicitud `HEAD` a la misma URL devuelve `Content-Length` sin el archivo.

CONFIGURACIÓN

Los límites se definen en la sección `dvr` del servidor:

```
dvr:
  root: /storage
  export_max_duration_secs: 10800   # duración máxima de una exportación, segundos
  export_max_concurrent: 4          # cuántas exportaciones se ejecutan a la vez
```

Parámetro	Por defecto	Descripción
`export_max_duration_secs`	`10800` (3 horas)	una solicitud más larga se rechaza con `422` antes de leer el archivo
`export_max_concurrent`	`4`	una solicitud por encima del límite recibe `503` con la cabecera `Retry-After`

No se admite el valor cero.

© Flussonic 2026

RESPUESTAS DEL SERVIDOR

Código	Cuándo
`200`	se está entregando el archivo MP4
`404`	no hay grabación en el segmento
`422`	la solicitud no se puede cumplir; el motivo se indica en el cuerpo de la respuesta
`503`	ya se ejecuta el número máximo de exportaciones; reintentar tras `Retry-After` segundos
`401`, `403`	la autorización no se superó; `403` también para un stream cifrado

Motivos del código `422`:

- **duración por encima del límite** `export_max_duration_secs`;
- **se nombra una pista inexistente**, o una pista sin grabación en el segmento;
- **se solicitan subtítulos**, o una pista de servicio;
- **`duration` es `now`** — la exportación «hasta el momento actual» no está soportada;

ESTADÍSTICAS

Los contadores de exportación se entregan junto con las métricas del proceso (`/streamer/api-v4/runtime/metrics`, consulte Monitorización):

- `archive_export_requests_total` — número de exportaciones;
- `archive_export_recorded_secs_total` — segundos grabados exportados;
- `archive_export_bytes_total` — bytes exportados;
- `archive_export_failures_total` — rechazos con la etiqueta `reason`.

SIGUIENTES PASOS

- Reproducción del archivo
- Configurar la grabación
- Proteger el archivo con autorización
- Compatibilidad con Flussonic

© Flussonic 2026

6.2.10 Procesamiento (media.process)

OCR de subtítulos

`media.process.ocr_subtitles`

Sapsan convierte los **subtítulos DVB de mapa de bits** del stream de entrada en pistas de texto independientes. El emisor envía una imagen de cada intervención; OCR reconoce los caracteres y asocia el texto obtenido con su tiempo de presentación. El espectador recibe subtítulos que puede seleccionar en el menú del reproductor, aunque este no pueda mostrar las imágenes DVB originales.

El procesamiento se realiza en el streamer que captura la señal durante el funcionamiento del live stream. No es necesario grabar el archivo para ejecutar OCR. Si DVR está activado, la pista de texto generada puede grabarse junto con el stream.

QUÉ SUBTÍTULOS SE RECONOCEN

La entrada es un servicio independiente de subtítulos gráficos DVB EN 300 743. En MPEG-TS, su descripción se transmite en la PMT: de ella Sapsan obtiene el idioma y los identificadores de página. Se reconoce el mapa de bits de los subtítulos, no toda la imagen de vídeo.

Cada fuente de texto tiene su propio método de procesamiento:

- **DVB bitmap subtitles** — imágenes con texto; la sección `subtitles` y el OCR descrito aquí están destinados a ellas.
- **Teletexto** — ya contiene códigos de caracteres; la extracción se configura en la sección `teletext`.
- **CEA-608/708** — subtítulos ocultos dentro del vídeo, controlados mediante la sección `closed_captions`.
- **Voz en la pista de audio** — una función independiente de reconocimiento de voz.

Este pipeline de OCR no extrae texto ya incrustado en la imagen de vídeo. El texto reconocido conserva el idioma del servicio de origen; aquí no se realiza traducción automática a otro idioma.

CÓMO SE CONVIERTE LA IMAGEN EN UNA PISTA DE TEXTO

1. **Descubrimiento del servicio.** Sapsan lee la descripción de las pistas de entrada y descubre los idiomas de los subtítulos DVB. No es necesario enumerar manualmente los PID y las páginas en la configuración.
2. **Composición de la imagen.** El decodificador DVB reconstruye la página a partir de los objetos y las paletas transmitidos. Las regiones de una misma composición se reúnen en una imagen para su reconocimiento.
3. **Reconocimiento.** La imagen se envía al motor seleccionado. La retransmisión de una composición sin cambios no requiere repetir el OCR.
4. **Preparación de la intervención.** Se eliminan las líneas vacías y algunos artefactos de reconocimiento del resultado. El texto recibe tiempos de presentación en la escala temporal del stream. Cambiar o borrar la página finaliza la intervención anterior.
5. **Entrega.** Las intervenciones pasan al pipeline común de procesamiento de pistas de texto y se convierten a WebVTT para su entrega. Pueden reproducirse a través de HLS y DASH.

El reconocimiento lo realiza un proceso de trabajo en segundo plano. La cola de trabajo es limitada: en caso de sobrecarga, pueden descartarse tareas y perderse el texto correspondiente. Por eso, al evaluar el rendimiento, conviene observar no solo el vídeo, sino también el retraso de los subtítulos y el contador de pérdidas de la cola.

El texto se entrega en ventanas vinculadas a la duración del segmento. Una intervención ya reconocida puede transmitirse en ventanas sucesivas mientras permanece en pantalla: no es necesario esperar a que desaparezca para entregarla por primera vez. También se tienen en cuenta los intervalos vacíos para que la pista de subtítulos siga avanzando durante las pausas.

La latencia depende del tiempo de OCR y segmentación, además del almacenamiento en búfer del reproductor. Cuando aparece una pista de texto, Sapsan desactiva LL-HLS **para todo el stream**: la entrega HLS pasa a usar segmentos normales sin partes LL-HLS. Téngalo en cuenta al elegir el modo para una emisión de baja latencia. Compruebe cuándo aparece el texto en el trayecto de salida real.

MOTORES DE RECONOCIMIENTO

Sapsan admite dos motores.

Tesseract reconoce la imagen mediante un modelo de idioma. En el streamer deben estar disponibles una biblioteca Tesseract compatible y los archivos de modelos de idioma `traineddata` para los idiomas del flujo de entrada. La calidad del resultado depende de la fuente tipográfica y del mapa de bits original; disponer del modelo no garantiza un texto sin errores.

© Flussonic 2026

`glyphtable` compara las imágenes de caracteres con una tabla de glifos etiquetados. La tabla está integrada en Sapsan; este motor no necesita una biblioteca OCR independiente. Este método es adecuado para emisores cuyos glifos conocidos estén representados en la tabla. Los glifos desconocidos se convierten en caracteres de sustitución; una nueva tipografía u otro método de renderizado requieren comprobar la cobertura de la tabla.

Si no se especifica `engine`, Sapsan intenta primero inicializar Tesseract para el idioma descubierto y después `glyphtable` si Tesseract no se ha podido inicializar. Esta elección se realiza al iniciar el proceso de trabajo. Una intervención reconocida con baja calidad no provoca por sí sola que se repita el reconocimiento con otro motor.

Con `engine: tesseract` o `engine: glyphtable` explícito, solo se utiliza el motor elegido. Si no está disponible, Sapsan muestra el motivo en las estadísticas y no lo sustituye por otro.

CONFIGURACIÓN

La configuración se envía a través de la API de central. Para activar OCR, añada una sección `subtitles` al stream:

```
streams:
  - name: tv1
    inputs:
      - udp:
          host: 239.0.0.1
          port: 5000
    subtitles:
      mode: extract
```

En este caso, Sapsan descubre los idiomas del flujo de entrada y selecciona automáticamente un motor disponible. Para elegir explícitamente la tabla de glifos, utilice esta sección:

```
subtitles:
  mode: extract
  engine: glyphtable
```

Hay tres modos disponibles:

- **`passthrough`** — modo predeterminado. Los subtítulos de mapa de bits originales se conservan en MPEG-TS, no se ejecuta OCR y no se crean pistas de texto.
- **`extract`** — el reconocimiento crea pistas de texto y conserva los subtítulos de mapa de bits originales. Este modo permite evaluar OCR mientras se sigue atendiendo a dispositivos con decodificador DVB.
- **`replace`** — el reconocimiento crea pistas de texto y excluye de la salida MPEG-TS el PID de subtítulos original vinculado a un proceso de trabajo en funcionamiento.

En el modo `replace` se excluye **todo el PID**, incluidos los servicios de otros idiomas que contiene. Si algunos idiomas no se han podido reconocer, sus subtítulos de mapa de bits también pueden desaparecer de la salida. Compruebe todos los idiomas antes de elegir este modo.

Si el motor no se inicia, no se activa la eliminación de su PID de origen. Si se pierde la vinculación con el servicio o el proceso de trabajo falla de forma definitiva, el PID de origen vuelve a la salida, salvo que otro proceso en funcionamiento en el mismo PID siga excluyéndolo. Un error individual de OCR o el desbordamiento de la cola no detienen necesariamente el proceso de trabajo, por lo que no se garantiza la restauración automática del mapa de bits para cada intervención perdida.

Recursos en el streamer que realiza la captura

La biblioteca Tesseract y sus datos de idioma deben ser accesibles al proceso Sapsan en el servidor que captura el stream. Instalar estos recursos solo en central no permite realizar el reconocimiento.

Para recursos en ubicaciones personalizadas, se utilizan variables de entorno del proceso:

Variable	Finalidad
`TESSERACT_LIB`	Ruta explícita a la biblioteca compartida de Tesseract. Si se establece, se prueba exactamente esa ruta.
`TESSDATA_PATH`	Directorio con archivos de modelos de idioma `.traineddata`.
`DVB_GLYPHS_PATH`	Ruta opcional a una tabla de glifos propia: un archivo TSV o un directorio con `glyphs.tsv`.

© Flussonic 2026

Una tabla propia **sustituye por completo a la integrada**, en lugar de ampliarla. Si el archivo no está disponible o no contiene el idioma necesario, no se recurre automáticamente a la tabla integrada.

Después de instalar o sustituir la biblioteca Tesseract y cambiar el entorno, reinicie el proceso del streamer: el resultado de carga de la biblioteca se guarda en caché. La incorporación de archivos de modelo tampoco significa que un proceso de trabajo ya no disponible vaya a intentar cargarlos de nuevo automáticamente; tras resolver la causa, vuelva a crear el proceso de trabajo modificando la propia sección `subtitles` o reiniciando el streamer.

IDIOMAS Y CAMBIO DE FUENTE

Sapsan crea una pista OCR por idioma descubierto, hasta **nueve pistas por stream**. El idioma se obtiene de la descripción del servicio DVB. Si el emisor indica un idioma incorrecto, esto afecta a la selección de datos para el reconocimiento y a la etiqueta de la pista de salida.

Si hay varios servicios del mismo idioma, se selecciona el primero en el orden de la PMT. Los subtítulos normales y una variante para personas con discapacidad auditiva del mismo idioma no se convierten en dos pistas OCR independientes.

Cuando cambia la descripción de entrada, Sapsan revisa la vinculación con los servicios. Si un idioma desaparece temporalmente, su pista de salida se conserva y el proceso de trabajo pasa a `no_source_match`. Conservar las pistas mantiene su identidad para los consumidores, pero implica que el límite de nueve pistas incluye las ya creadas, no solo los idiomas presentes en ese momento.

Al configurar la redundancia de fuentes, compruebe la presencia y el etiquetado correcto de los servicios DVB en cada entrada. Una conmutación de vídeo correcta no confirma por sí sola que se hayan conservado los subtítulos.

SUPERVISIÓN

El estado de OCR está disponible en el array `subtitles` de las estadísticas del stream en el streamer que realiza la captura:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/streams/stats/tv1 \
  | jq '{name, subtitles: [(.subtitles // [])[] | {
      track, producer, lang, state,
      source_pid, source_page_id, source_lang,
      coverage_lag_ms, error
    }]}'
```

Los campos `source_pid` y `source_page_id` identifican el servicio DVB vinculado al proceso de trabajo. `coverage_lag_ms` es el retraso, en milisegundos, del límite del intervalo de subtítulos procesado respecto a la escala temporal del stream. Este campo puede no aparecer antes de la primera entrega; la ausencia de un valor no significa un retraso nulo.

Estados principales:

- **`running`** — el proceso de trabajo está activo y vinculado a una fuente. No es una evaluación de la precisión del reconocimiento.
- **`no_source_match`** — el proceso de trabajo existe, pero el servicio correspondiente no está presente. La vinculación se revisa cuando cambia la descripción de entrada.
- **`no_model`** — los datos de idioma no están disponibles o la selección automática no ha podido iniciar ningún motor. Los detalles se encuentran en `error`.
- **`feature_missing`** — el motor seleccionado no está disponible. Por ejemplo, puede que la biblioteca Tesseract no esté instalada o no se haya podido cargar.
- **`failed`** — el proceso de trabajo en segundo plano se ha detenido por un fallo. Tras resolver la causa, es necesario volver a crearlo.

Un array `subtitles` vacío no demuestra por sí solo que falte un modelo: con `extract` activado, también puede aparecer antes de descubrir el primer servicio DVB.

© Flussonic 2026

Para supervisar los errores de procesamiento se utilizan las métricas de `/streamer/api-v4/live-metrics`, si su exportación está disponible en la distribución:

Métrica	Qué comprobar
`stream_subtitle_up`	Disponibilidad del proceso de trabajo. El valor también puede ser `1` en `no_source_match`; tenga en cuenta la etiqueta `state`.
`stream_subtitle_coverage_lag_seconds`	Retraso de la pista de subtítulos, en segundos.
`stream_subtitle_decode_errors_total`	Aumento de los errores de análisis de los subtítulos DVB entrantes.
`stream_subtitle_ocr_errors_total`	Aumento de los errores de reconocimiento.
`stream_subtitle_queue_drops_total`	Tareas perdidas por desbordamiento de la cola OCR.

Los contadores de errores son acumulativos: compare su variación durante un intervalo para evaluar el funcionamiento actual. No se incluyen en el array `subtitles` de la consulta API anterior.

COMPROBACIÓN DEL RESULTADO

1. Active `extract` y confirme que el flujo de entrada realmente contiene subtítulos DVB de mapa de bits.
2. Compruebe los idiomas descubiertos y las vinculaciones a PID/páginas. Confirme que los procesos de trabajo hayan pasado a `running`.
3. Seleccione la pista de texto en el reproductor. Compare las intervenciones con la imagen original en distintos programas; compruebe los caracteres y saltos de línea, así como la sincronización con el vídeo.
4. Compruebe el retraso y el aumento de los contadores de errores con la carga de producción. Evalúe por separado los caracteres de sustitución al utilizar `glyphtable`.
5. Si necesita `replace`, compruebe sus consecuencias para todos los idiomas del PID de origen. Si necesita un archivo, confirme que un fragmento grabado se reproduce con la pista de texto.

Se admite la codificación de mapa de bits de objetos DVB con `coding_method = 0`; este decodificador no procesa otros métodos de codificación de objetos. Los datos de entrada dañados y una capacidad de procesamiento insuficiente pueden provocar la pérdida de intervenciones. La comprobación del estado `running` debe complementarse con la revisión del propio texto.

© Flussonic 2026

Speech To Text

`media.process.speech_to_text`

La estación no inventa teletexto ni subtítulos ocultos: transporta el texto que trajo consigo la imagen. Un canal que no lleva texto alguno no obtiene subtítulos por esa vía, y es justamente ese canal el que suele necesitarlos: una desconexión regional, una emisión corporativa, una fuente ajena sin datos de servicio.

El reconocimiento de voz cubre ese caso: la estación lee el audio del canal grabado y le agrega una pista de subtítulos.

La señal en directo no se toca en absoluto. El reconocimiento trabaja **sobre el archivo**: lee el audio grabado y escribe allí mismo los subtítulos. De ahí dos consecuencias que conviene conocer de antemano.

- El canal **necesita grabación de archivo**. Sin ella el reconocimiento no tiene de dónde tomar el audio ni dónde escribir el resultado.
- El retraso es inofensivo. Si el reconocimiento no sigue el ritmo del directo, los subtítulos se retrasan y luego se ponen al día; la salida del canal no se ve afectada.

CÓMO ACTIVARLO

El reconocimiento vive en la pestaña **Procesamiento** de la ficha del canal, en la sección **Reconocimiento de voz**. Lo activa el interruptor **Reconocer voz en subtítulos**.

Hay cuatro campos y ninguno es obligatorio: los valores por defecto bastan para activar el reconocimiento con un solo interruptor.

Campo	Qué define
Modelo	El tamaño del reconocedor. Un modelo mayor es más preciso, pero más lento y necesita más memoria.
Idioma	Código ISO-639-1, por ejemplo `es`. Vacío significa detectar por el audio.
Fuente de audio	La pista de la emisión que se escucha. La primera por defecto.
Ventana de reconocimiento, s	Hasta 120 segundos. Una ventana larga entiende mejor el sentido, pero la línea aparece más tarde.

Conviene indicar el idioma cuando se conoce. La detección decide con los primeros segundos de audio y se equivoca en un canal que empieza con una cortinilla musical, y un idioma equivocado estropea toda la pista, no una sola línea.

La **fuente de audio** es asunto cotidiano en una cabecera: el canal lleva varias pistas de idioma y el reconocimiento escucha exactamente una. Si la pista elegida no está al aire, el estado lo dice: «La pista de audio elegida no está al aire».

SIN GRABACIÓN NO ARRANCA

Si se activa el reconocimiento en un canal sin grabación, el panel responde de inmediato, sin esperar a guardar:

Esto no impide guardar el ajuste: avisa de que no va a funcionar. Active la grabación del canal y el reconocimiento arrancará solo.

ESTADO DEL RECONOCIMIENTO

Bajo el formulario el panel muestra lo que ocurre en realidad. Es el único sitio donde se ve el trabajo vivo del reconocimiento: la ficha del canal toma el estado del nodo donde el canal se graba.

- **Funcionando / Detenido** — si el reconocimiento está en marcha. «Detenido» siempre viene acompañado del motivo en palabras.
- **Retraso** — cuánto se queda lo reconocido por detrás del momento actual. En el borde en directo son segundos; en un archivo que el reconocimiento recupera desde el principio son horas, y eso es un modo de trabajo normal, no un fallo.
- **Audio reconocido** — cuánto audio ha pasado ya por el reconocedor.
- **Líneas** — cuántas líneas de subtítulos se han escrito.
- **Errores** — fallos de lectura del archivo y de reconocimiento. Cero aquí es lo habitual.
- **Última línea** — cuándo escribió algo el reconocimiento por última vez.

© Flussonic 2026

El panel nombra directamente los motivos por los que el reconocimiento no arranca:

Qué dice	Qué hacer
El stream no tiene grabación de archivo	Activar la grabación.
Falta el archivo del modelo o no se pudo cargar	Poner el archivo del modelo en el servidor (más abajo).
La pista de audio elegida no está al aire	Elegir otra fuente de audio o dejar el valor por defecto.
El reconocedor no conoce ese idioma	Revisar el código de idioma o vaciar el campo.
Esta compilación del servidor no tiene reconocimiento de voz	Contactar con el soporte: el servidor está compilado sin el motor.

Un reconocimiento que funciona y tiene un error reciente es un aviso, no una avería: un fragmento no leído no detiene el trabajo, el reconocimiento sigue con el siguiente.

EL ARCHIVO DEL MODELO

El modelo no se entrega con el servidor: uno utilizable pesa más de un gigabyte y cuál hace falta lo decide el dueño de la estación. El servidor lo busca en el directorio de estado del producto — `/var/lib/mcaster/whisper-models` — y espera allí un archivo `ggml-<modelo>.bin`, es decir `ggml-medium.bin` para el modelo `medium`.

Mientras el archivo no está, el estado del reconocimiento dice honestamente «Falta el archivo del modelo o no se pudo cargar», y el canal sigue funcionando y grabando como si nada.

Un solo archivo sirve a todos los canales de la estación: el modelo se lee en memoria una vez por proceso, por muchos canales que lo pidan.

QUÉ SALE A LA SALIDA

Lo reconocido se convierte en una pista de subtítulos normal del canal: aparece en la información de medios y viaja a la reproducción HLS como WebVTT — justo donde hace falta una pista de texto, porque ningún reproductor OTT decodifica teletexto.

El flujo de transporte no llevará el texto reconocido: una pista de subtítulos no tiene en qué viajar dentro de MPEG-TS, la misma limitación que la extracción de teletexto.

Tiene sentido ver la pista sobre el archivo. En el borde en directo los subtítulos aún no están: aparecen con el mismo retraso que muestra el estado.

© Flussonic 2026

Miniaturas por HTTP

`media.process.thumbnails_http`

Sapsan obtiene capturas JPEG de la cámara por HTTP.

CONFIGURACIÓN

```
streams:
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://admin:password@10.0.0.5/stream0
    thumbnails:
    - http:
        url: http://10.0.0.5/snapshot.jpg
        fetch_interval: 10  # período de captura, s (por defecto 10)
        timeout: 5          # tiempo de espera de la petición, s (por defecto 5)
      dvr: true             # guardar los fotogramas en el archivo (por defecto false)
```

`fetch_interval` y `timeout` deben ser mayores que 0. Las respuestas que no son JPEG o superan el límite de tamaño se descartan; la vista previa no se actualiza.

Véase también: Miniaturas de fotogramas clave.

OBTENCIÓN DE CAPTURAS

Qué	URL
Fotograma actual del stream	`http://server/streaming/live-preview-jpeg/<stream>`
Fotograma de una pista concreta	`http://server/streaming/track-preview-jpeg/<stream>/<track>`
Fotograma del archivo	`http://server/streaming/v/<stream>/image/<track>/<utc_ms>.jpg`

CAPTURAS EN EL ARCHIVO

Con `dvr: true` los fotogramas se escriben en el archivo como una pista JPEG aparte con las dimensiones reales del fotograma. En el manifiesto DASH del archivo se expone como un AdaptationSet estándar de thumbnail tile (`http://dashif.org/guidelines/thumbnail_tile`) — los reproductores con soporte de miniaturas en la línea de tiempo lo recogen automáticamente.

SIGUIENTES PASOS

- Configure la grabación DVR

© Flussonic 2026

Miniaturas de fotogramas clave

`media.process.thumbnails_keyframe`

Sapsan genera miniaturas JPEG a partir de los fotogramas clave de la pista de vídeo del stream.

CONFIGURACIÓN

```
streams:
  - name: cam1
    inputs:
    - rtsp:
        url: rtsp://admin:password@10.0.0.5/stream0
    thumbnails:
    - keyframe:
        track: v1           # pista de vídeo de origen
        width: 320          # por defecto 320 si no se fija ningún tamaño; mínimo 32
        height: 180
        every: 3gop         # capturar cada tercer GOP (por defecto: cada GOP)
      dvr: true
```

`every` debe ser mayor que 0. Las dimensiones del fotograma deben ser de al menos 32.

Véase también: Miniaturas por HTTP.

OBTENCIÓN DE CAPTURAS

Qué	URL
Fotograma actual del stream	`http://server/streaming/live-preview-jpeg/<stream>`
Fotograma de una pista concreta	`http://server/streaming/track-preview-jpeg/<stream>/<track>`
Fotograma del archivo	`http://server/streaming/v/<stream>/image/<track>/<utc_ms>.jpg`

CAPTURAS EN EL ARCHIVO

Con `dvr: true` los fotogramas se escriben en el archivo como una pista JPEG aparte con las dimensiones reales del fotograma. En el manifiesto DASH del archivo se expone como un AdaptationSet estándar de thumbnail tile (`http://dashif.org/guidelines/thumbnail_tile`) — los reproductores con soporte de miniaturas en la línea de tiempo lo recogen automáticamente.

SIGUIENTES PASOS

- Configure la grabación DVR

PREVIEWS DE FOTOGRAMAS DEL ARCHIVO

- Un fotograma JPEG de la pista de capturas: `http://server/streaming/v/<stream>/image/<track>/<utc_ms>.jpg`
- Un fragmento MP4 de un momento del archivo: `http://server/streaming/dvr-preview-mp4/<stream>/<utc_ms>` — un fragmento corto desde un fotograma clave; para el preview, el servidor elige por sí solo la pista de menor resolución.

© Flussonic 2026

6.2.11 Reproducción (media.play)

Sesiones de visualización

`media.play.sessions`

Una sesión de visualización es el estado que Sapsan mantiene para el acceso de un espectador a un stream. Vincula las solicitudes del reproductor con la autorización y registra la actividad, los bytes entregados y el uso del archivo. Esto permite observar la visualización en su conjunto, aunque la entrega consista en muchas solicitudes HTTP independientes.

Por ejemplo, un reproductor HLS solicita una lista maestra, las calidades y una secuencia de segmentos. La adaptación de calidad y las nuevas conexiones HTTP no deberían convertir por sí solas una visualización en cientos de espectadores. Sapsan encuentra la sesión existente y asocia con ella las solicitudes posteriores.

CÓMO SE AGRUPAN LAS SOLICITUDES HTTP

Para la reproducción HTTP habitual, la identidad se compone de los siguientes datos:

- **Nombre del stream** — el recurso multimedia solicitado.
- **Tipo de acceso** — la reproducción se distingue de la publicación.
- **Dirección IP del cliente** — teniendo en cuenta los proxies de confianza configurados.
- **Token de acceso** — para un ticket JWT validado, se utiliza el identificador de su concesión `jti`.

Sapsan utiliza estos datos para buscar la sesión en cada solicitud. Si ya existe una sesión correspondiente, la solicitud actualiza su actividad. Si no, se crea una nueva y se comprueba el acceso. Una solicitud de fragmento también puede iniciar una sesión: obtener la lista maestra no es un primer paso obligatorio.

El servidor asigna el identificador de sesión. Un cliente HTTP habitual no necesita generar su propio UUID de sesión ni conservar una cookie especial. `User-Agent` se registra en las estadísticas, pero no forma parte de la clave de agrupación indicada.

El token debe llegar a las solicitudes relacionadas. En la reproducción HTTP, el parámetro `?token=` tiene prioridad sobre `Authorization: Bearer …`. La emisión y entrega de JWT se describen en tokens de acceso.

Este registro depende de la identidad elegida, no del dispositivo físico. Varios reproductores con la misma IP, el mismo token y el mismo stream pueden compartir una sesión. Cambiar la IP o el identificador del token puede crear una nueva. Para contabilizarlos por separado, emita tickets distintos y configure correctamente los proxies de confianza.

CICLO DE VIDA Y ESTADÍSTICAS

Las solicitudes y los datos entregados mantienen la actividad de la sesión. Si no hay actividad durante **60 segundos**, la sesión caduca; la limpieza se realiza periódicamente, por lo que el cierre no tiene por qué coincidir con el de la pestaña del navegador.

Un registro de sesión abierto y un espectador activo son indicadores distintos. La implementación actual activa la sesión después de entregar **10 KiB** de datos. Por tanto, una sola solicitud de una lista pequeña no aumenta necesariamente el número de espectadores activos. El umbral mide bytes entregados y no demuestra que una persona haya visto la imagen. El registro de sesiones cerradas solo incluye las que llegaron a estar activas.

Se registra la siguiente información:

- **Tiempo** — apertura, última actividad, duración y cierre.
- **Entrega** — protocolo y número de bytes entregados.
- **Cliente** — IP, `User-Agent` e identificador de usuario, si lo proporciona la autorización.
- **Archivo** — un indicador de acceso a DVR. Después de retroceder al archivo, permanece activado incluso al volver al directo.

La transición directo → archivo → directo no crea por sí sola tres visualizaciones. Del mismo modo, las solicitudes de segmentos y las conexiones de red no deben sumarse como si fueran espectadores.

© Flussonic 2026

CALIDAD DE ENTREGA SRT

Una sesión abierta por SRT lleva un corte protocolar de la entrega — contadores actuales de la conexión en la respuesta de sesiones activas:

Campo	Significado
`rtt_ms`, `rtt_max_ms`	RTT actual y máximo durante la vida de la sesión
`send_buffer_packets`	paquetes en el búfer de envío — señal temprana de congestión en el espectador
`packets_sent`	paquetes enviados — el denominador de la proporción de pérdidas
`lost_packets`	pérdidas: huecos en la secuencia saliente de paquetes
`drop_too_late_packets`	de las pérdidas — paquetes saltados como caducados: la latencia no alcanzó a recuperarlos
`drop_requests`	de las pérdidas — paquetes descartados por orden del receptor: el espectador informó de que ya no necesita ese rango
`retransmitted_packets`	paquetes reenviados — respuestas a los informes NAK del espectador
`naks_received`	informes NAK del espectador: cuántas veces reportó pérdidas
`health`	veredicto de salud de la conexión: `ok`, `degraded` o `bad`

Los valores de una sesión activa se actualizan: el nodo fija periódicamente los contadores en el registro de la sesión. Al cerrarse, el registro del nodo recibe los totales finales y el veredicto de la última ventana.

El veredicto `health` describe la ventana de entrega — el último minuto de la conexión, no toda su vida: la proporción de pérdidas y el RTT actual frente al doble de la latencia de entrega. Los fallos pasados quedan en el pasado: cuando la ventana se limpia, el veredicto vuelve a `ok`, y el cuándo se caía se lee en la historia del corte (véase la historia por minutos). Un corte de conexión no estropea el veredicto: una sesión limpia cerrada por un espectador que enmudeció conserva el veredicto de su última ventana, y la marcha misma la describe la razón de cierre.

Las sesiones de otros protocolos no llevan sección `srt` en absoluto — la ausencia del campo no significa pérdidas cero.

HISTORIA POR MINUTOS DE LA SESIÓN

Las sesiones con propietario `user_id` — del mapa de clientes de la entrega o de la etiqueta de propietario — se escriben en el almacén de métricas de central en puntos por minutos: el gráfico de una sesión se construye con el mismo mecanismo que los gráficos de la entrada del stream. Cada serie lleva las etiquetas `name` (stream), `user_id` y `session`:

Serie	Qué muestra
`stream_play_srt_session_rtt_seconds`, `..._rtt_max_seconds`	RTT — actual y máximo
`stream_play_srt_session_send_buffer_packets`	búfer de envío, paquetes
`stream_play_srt_session_packets_sent_total`, `..._bytes_total`	paquetes y bytes enviados
`stream_play_srt_session_lost_packets_total`	pérdidas
`stream_play_srt_session_drop_too_late_packets_total`, `..._drop_requests_total`	pérdidas por causa: saltados como caducados y descartados por orden del receptor
`stream_play_srt_session_retransmitted_packets_total`, `..._naks_received_total`	retransmisiones e informes NAK

Las series se leen en el servidor Prometheus integrado de central con la misma consulta de rango que las series del stream. La profundidad del detalle es la ventana del almacén, seis horas por defecto; un reinicio del plano de métricas borra la historia. Más profundo que la ventana están los totales generales del registro cerrado — bytes, duración, razón de cierre. Las sesiones anónimas no crean series: sin propietario no hay gráficos, y no ceros.

AUTORIZACIÓN DE LA SESIÓN

El método principal recomendado para permitir el acceso es un ticket JWT. El streamer lo valida localmente, mientras el estado de sesión vincula las solicitudes y su tráfico. Que ya exista una sesión abierta no elimina la validación del JWT en la siguiente solicitud HTTP.

© Flussonic 2026

Para las integraciones antiguas con un backend de autorización externo, el permiso se conserva durante un periodo definido. Dentro de esta ventana, las solicitudes posteriores reutilizan la decisión; cuando caduca, se requiere una nueva autorización. `auth_duration`, con un valor predeterminado de 180 segundos, corresponde a la vigencia de la decisión de autorización, no al tiempo de espera por inactividad de 60 segundos.

Comprobar las solicitudes HTTP posteriores y finalizar una respuesta continua ya abierta son operaciones distintas. La existencia de una sesión no significa que un cambio en la decisión de autorización interrumpa al instante cualquier entrega MPEG-TS en curso.

DIFERENCIA RESPECTO A LA VALIDACIÓN DE ENLACES EN UN EDGE NGINX

La validación de un enlace firmado determina si se admite una solicitud concreta. Por ejemplo, el módulo `secure_link` de nginx comprueba la autenticidad del enlace y su caducidad. El registro de la visualización requiere un estado que relacione las solicitudes individuales del reproductor.

Tarea	Solo validación del enlace	Registro de sesiones de Sapsan
Permitir una solicitud de segmento	Valida el enlace presentado	Comprueba el acceso y vincula la solicitud con una visualización.
Contabilizar una visualización	Las solicitudes individuales no definen por sí solas sus límites	Registra la actividad y el ciclo de vida de la sesión.
Contabilizar el tráfico	Hay datos de cada solicitud	Los bytes se acumulan en la sesión.
Analizar el rebobinado y el protocolo	Requiere agregación adicional	Esta información forma parte del registro de sesión.
Recopilar datos de varios edges	Requiere un mecanismo común de registro	Central recopila datos de los streamers y proporciona un historial del clúster.

Nginx puede integrarse con estado y analítica externos. La comparación se refiere a una configuración en la que el edge solo valida enlaces: esa comprobación por sí sola no equivale al registro de sesiones de visualización.

SESIÓN DEL CLÚSTER Y CAMBIO ENTRE EDGES

Con balanceo de carga, un espectador puede iniciar la reproducción en el edge A y continuar en el edge B. Para analizar esa visualización, central recopila datos de los servidores, guarda las sesiones cerradas en un registro común y genera un historial con una etapa de combinación de registros.

En el historial del clúster, los registros con una misma identidad reconocida se combinan si sus intervalos se solapan o están separados por no más de **30 minutos**. Los bytes entregados se suman, el intervalo temporal abarca la unión y `nodes` enumera los servidores participantes. Esto proporciona un modelo de visualización lógica sobre los registros individuales de los servidores.

La implementación actual tiene una limitación: la lectura del historial se basa en la práctica en valores completos de `id` coincidentes. Al abrir la reproducción en otro edge, este ID puede cambiar porque incluye la hora de apertura. Por tanto, todavía no se garantiza la combinación de todos los cambios entre edges. El identificador del ticket JWT por sí solo no elimina esta limitación de la salida actual del historial.

La lista actual de sesiones activas y el historial combinado de sesiones cerradas son vistas distintas. El número de filas de la lista activa no permite deducir automáticamente el número de personas únicas durante un periodo ni el resultado de la combinación histórica.

CONSULTA MEDIANTE LA API

Sesiones actuales del stream en central:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  'http://central:8080/central/api-v4/play-sessions/active?stream=cam1&limit=100'
```

Historial con combinación de registros:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  'http://central:8080/central/api-v4/play-sessions/closed?stream=cam1&limit=100'
```

La respuesta contiene un array `sessions`. Los campos `id`, `bytes`, `duration` y `dvr` describen la visualización; `node` y `nodes`, la participación de los streamers. `uncovered_nodes` indica los servidores cuya recopilación de datos está incompleta. Un servidor sin cobertura no debe considerarse un servidor con cero espectadores.

© Flussonic 2026

Para el diagnóstico local, el streamer ofrece las rutas correspondientes `/streamer/api-v4/play-sessions/active` y `/streamer/api-v4/play-sessions/closed`. La recopilación del historial en central es asíncrona: cerrar una sesión localmente no significa que aparezca de inmediato en el registro común.

© Flussonic 2026

Ports

`media.play.ports`

TRANSPORTE

MPEG-TS sobre SRT/UDP; el cliente se conecta en modo caller al puerto dedicado del stream.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**

Audio

- **AAC**
- **MP2**
- **MP3**
- **AC-3**
- **E-AC-3**
- **SMPTE 302M**

Otras pistas

- **SCTE-35**
- **KLV**
- **ID3**
- **tx3g**
- **Teletext**
- **DVB subtitles**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

AV1 en MPEG-TS utiliza la convención `stream_type = 0x06` con el descriptor de registro `AV01` ; el receptor debe entender esta forma. Esto no garantiza compatibilidad con todas las implementaciones de AV1 en TS. Opus no se incluye en la lista: reconocer o anunciar un códec no basta para entregar una salida utilizable, y la ruta actual no proporciona el empaquetado TS/PES necesario para Opus.

DISTRIBUCIÓN DESDE EL PUERTO DEL STREAM

Mediante central, configure `srt_play.port` y, opcionalmente, `srt_play.passphrase` en la config del streamer. Consulte el ejemplo completo y los parámetros en reproducción SRT. El cliente se conecta a `srt://server:4010` en modo caller; sustituya el puerto por el valor configurado.

Cada puerto pertenece a un stream; los puertos duplicados entre streams se rechazan al validar la configuración. `srt_play.mpegts` define parámetros independientes de salida MPEG-TS para este puerto.

© Flussonic 2026

HLS

`media.play.hls`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**

Audio

- **AAC**
- **MP3**
- **MP2**
- **AC-3**
- **E-AC-3**
- **Opus**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

Todo stream de Sapsan está disponible de inmediato por HLS (fMP4/CMAF) sin configuración adicional. Los streams en vivo reciben por defecto LL-HLS con segmentos parciales.

Las pistas de texto estructuradas generadas por Sapsan se entregan como WebVTT; el teletexto o los subtítulos DVB en bitmap de entrada no constituyen por sí solos una pista de texto lista. Primero debe activarse la extracción o el reconocimiento correspondiente.

Cuando hay una pista de subtítulos de texto, LL-HLS se desactiva para todo el stream: HLS utiliza segmentos convencionales. La compatibilidad del códec con el reproductor y la disponibilidad de LL-HLS son condiciones distintas.

URL DE REPRODUCCIÓN

Qué	URL
Playlist maestro (todas las calidades)	`http://server/streaming/v/<stream>/index.m3u8`
Playlist de una sola pista	`http://server/streaming/v/<stream>/variant/<track>/index.m3u8`
Rebobinado (rewind)	`http://server/streaming/v/<stream>/rewind/<segundos-atrás>/index.m3u8`
Archivo por tiempo absoluto	`http://server/streaming/v/<stream>/index.m3u8?from=<utc_ms>&to=<utc_ms>`

El playlist maestro lleva las calidades con `RESOLUTION`, `FRAME-RATE` y `CODECS`; el audio pasa a grupos rendition (`EXT-X-MEDIA`). El audio G.711 (PCMA/PCMU) no se sirve por HLS — transcodifique primero esos streams a AAC.

© Flussonic 2026

HLS SOBRE MPEG-TS

Para los decodificadores que no reproducen fMP4, el mismo stream se entrega por HLS con segmentos MPEG-TS. Las direcciones son las mismas de arriba, con un segmento de forma después del nombre del stream:

Forma	Playlist maestro	Para quién
audio en pistas aparte	`http://server/streaming/v/<stream>/ts/index.m3u8`	decodificadores que entienden `EXT-X-MEDIA`
audio dentro del segmento de vídeo	`http://server/streaming/v/<stream>/ts/mux/index.m3u8`	decodificadores antiguos: vídeo y audio como un solo programa en cada segmento

Seleccione el archivo en ambas disposiciones con `?from=<utc_ms>&to=<utc_ms>`. El rebobinado está disponible para TS con pistas separadas: `http://server/streaming/v/<stream>/ts/rewind/<seconds>/index.m3u8`. La disposición mixta `ts/mux/` no tiene URL de rebobinado.

En `ts/mux/`, cada vídeo forma una variante con todo el audio publicado, hasta diez pistas de audio. Se omiten los subtítulos. MPEG-TS no admite LL-HLS. Ambas disposiciones TS HLS tienen una lista de códecs más reducida que la salida MPEG-TS continua:

- **H.264**
- **HEVC**
- **AAC**
- **MP3**
- **MP2**
- **AC-3**
- **E-AC-3**

Qué forma entregan las direcciones de Flussonic (`/<stream>/index.m3u8`, `video.m3u8` y otras) se describe en la página de compatibilidad con Flussonic. Los segmentos `ts/mux/` incompletos y los ensamblajes fuera de la caché se ven en las estadísticas.

TOKENS EN LOS PLAYLISTS

Si la reproducción está protegida con un token, `?token=` se añade automáticamente a todas las URLs anidadas del playlist: variantes, segmentos init, segmentos, partes y preload hints. Al reproductor le basta con abrir el playlist maestro con el token.

CANTIDAD DE SEGMENTOS

La longitud del playlist en vivo la fija el parámetro `segments` del stream:

Configure estas opciones mediante central en la config del streamer.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    segments: 6
```

QUÉ VIENE DESPUÉS

- Proteja la reproducción con un token
- Reproduzca el archivo
- Reproducción LL-HLS

© Flussonic 2026

LLHLS

`media.play.llhls`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**

Audio

- **AAC**
- **MP3**
- **MP2**
- **AC-3**
- **E-AC-3**
- **Opus**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

Todo stream de Sapsan está disponible de inmediato por HLS (fMP4/CMAF) sin configuración adicional. Los streams en vivo reciben por defecto LL-HLS con segmentos parciales.

Las pistas de texto estructuradas generadas por Sapsan se entregan como WebVTT; el teletexto o los subtítulos DVB en bitmap de entrada no constituyen por sí solos una pista de texto lista. Primero debe activarse la extracción o el reconocimiento correspondiente.

Cuando hay una pista de subtítulos de texto, LL-HLS se desactiva para todo el stream: HLS utiliza segmentos convencionales. La compatibilidad del códec con el reproductor y la disponibilidad de LL-HLS son condiciones distintas.

URL DE REPRODUCCIÓN

Qué	URL
Playlist maestro (todas las calidades)	`http://server/streaming/v/<stream>/index.m3u8`
Playlist de una sola pista	`http://server/streaming/v/<stream>/variant/<track>/index.m3u8`
Rebobinado (rewind)	`http://server/streaming/v/<stream>/rewind/<segundos-atrás>/index.m3u8`
Archivo por tiempo absoluto	`http://server/streaming/v/<stream>/index.m3u8?from=<utc_ms>&to=<utc_ms>`

El playlist maestro lleva las calidades con `RESOLUTION`, `FRAME-RATE` y `CODECS`; el audio pasa a grupos rendition (`EXT-X-MEDIA`). El audio G.711 (PCMA/PCMU) no se sirve por HLS — transcodifique primero esos streams a AAC.

© Flussonic 2026

LL-HLS

Para los streams en vivo Sapsan sirve un LL-HLS completo (versión 9 del protocolo):

- segmentos parciales `EXT-X-PART` en el borde en vivo y `EXT-X-PRELOAD-HINT` para la siguiente parte;
- blocking playlist reload: los parámetros de cliente `_HLS_msn` y `_HLS_part` retienen la petición hasta que aparece el segmento o la parte solicitada;
- actualizaciones delta del playlist: `_HLS_skip=YES` acorta el playlist con la etiqueta `EXT-X-SKIP`;
- `EXT-X-RENDITION-REPORT` para un cambio de calidad rápido;
- `PART-HOLD-BACK`, `CAN-SKIP-UNTIL`, `HOLD-BACK` se calculan automáticamente a partir de las duraciones del segmento y de la parte.

El LL-HLS se puede desactivar por cliente con `?llhls=false` — el playlist maestro lo propaga a todas las URLs de las variantes, y el reproductor recibe un HLS clásico (versión 7).

TOKENS EN LOS PLAYLISTS

Si la reproducción está protegida con un token, `?token=` se añade automáticamente a todas las URLs anidadas del playlist: variantes, segmentos init, segmentos, partes y preload hints. Al reproductor le basta con abrir el playlist maestro con el token.

© Flussonic 2026

DASH

`media.play.dash`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**

Audio

- **AAC**
- **MP3**
- **MP2**
- **Opus**
- **AC-3**
- **E-AC-3**
- **PCMA**
- **PCMU**

Subtítulos

- **WebVTT**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

DASH está disponible para directo y archivo cuando las pistas pueden empaquetarse en un formato de salida compatible.

La lista describe el empaquetado directo de pistas de Sapsan en fMP4. La admisión en el manifiesto no basta: el empaquetador actual no crea los segmentos de inicialización necesarios para VP9, JPEG 2000 o L16. La descripción de una pista no debe interpretarse como una reproducción lista. El AAC recibido mediante `MP4A-LATM` se empaqueta en MP4 como AAC, no como RTP LATM.

Las pistas de texto estructuradas generadas por Sapsan se entregan como WebVTT; el teletexto o los subtítulos DVB en bitmap de entrada no constituyen por sí solos una pista de texto lista. Primero debe activarse la extracción o el reconocimiento correspondiente.

URL DE REPRODUCCIÓN

Qué	URL
Manifiesto en vivo	`http://server/streaming/v/<stream>/Manifest.mpd`
Archivo por tiempo absoluto	`http://server/streaming/v/<stream>/Manifest.mpd?from=<utc_ms>&to=<utc_ms>`

MANIFIESTO EN VIVO

Un MPD dinámico (perfil `isoff-live`): todas las calidades de un stream MBR viven en un único AdaptationSet de vídeo como Representation separadas, ordenadas por bitrate ascendente — los reproductores construyen correctamente la escalera ABR. La ventana `timeShiftBufferDepth` y el `suggestedPresentationDelay` se calculan automáticamente (el retraso es de unos dos segmentos respecto al borde en vivo).

© Flussonic 2026

MANIFIESTO DEL ARCHIVO

El MPD del archivo es estático. Los huecos de la grabación se convierten en `<Period>` separados, y la línea de tiempo se elige con el parámetro `?timeline=`:

- `compact` (por defecto) — los huecos se contraen y el archivo se reproduce de forma continua;
- `wallclock` — los huecos se conservan y la línea de tiempo coincide con el tiempo real.

El `bandwidth` de cada Representation se mide a partir de los datos reales del rango solicitado. Si el archivo tiene una pista JPEG de capturas, esta se expone como un AdaptationSet estándar de thumbnail tile.

QUÉ VIENE DESPUÉS

- Proteja la reproducción con un token
- Reproduzca el archivo

© Flussonic 2026

MSS

`media.play.mss`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Vídeo

- **H.264**

Audio

- **AAC-LC**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

CONFIGURACIÓN

Sapsan entrega Microsoft Smooth Streaming para streams en vivo y archivos mediante el puerto HTTP. MSS no tiene un puerto separado.

URL DE REPRODUCCIÓN

Modo	URL
Live	`http://server/streaming/v/cam1/mss/Manifest`
Archivo	`http://server/streaming/v/cam1/mss/archive/1651829645-120/Manifest`

La dirección del archivo contiene el inicio en segundos Unix y la duración en segundos; requiere grabación DVR. El reproductor solicita los fragmentos indicados en el manifiesto. `filter.tracks=a1` selecciona la primera pista de audio; `filter.tracks=v1`, la primera de vídeo. Envíe el token como `?token=<token>`; se incluye en las URL de fragmentos.

Las pistas necesitan un bitrate positivo. Se admite el perfil básico H.264/AAC-LC; las pistas incompatibles se omiten. No se admite la forma rewind de `.isml`.

SIGUIENTES PASOS

- Compatibilidad de URL legacy
- Grabación DVR

© Flussonic 2026

MPEGTS

`media.play.mpegts`

TRANSPORTE

MPEG-TS continuo por HTTP/HTTPS (TCP).

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**

Audio

- **AAC**
- **MP2**
- **MP3**
- **AC-3**
- **E-AC-3**
- **SMPTE 302M**

Otras pistas

- **SCTE-35**
- **KLV**
- **ID3**
- **tx3g**
- **Teletext**
- **DVB subtitles**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

Sapsan multiplexa cualquier stream en un SPTS MPEG-TS y lo sirve por HTTP — cómodo para set-top boxes, transcodificadores y sistemas heredados.

AV1 en MPEG-TS utiliza la convención `stream_type = 0x06` con el descriptor de registro `AV01`; el receptor debe entender esta forma. Esto no garantiza compatibilidad con todas las implementaciones de AV1 en TS. Opus no se incluye en la lista: reconocer o anunciar un códec no basta para entregar una salida utilizable, y la ruta actual no proporciona el empaquetado TS/PES necesario para Opus.

URL DE REPRODUCCIÓN

```
http://server/streaming/mpegts/<stream>
```

QUÉ SE PRODUCE

- PAT/PMT/SDT correctas con contadores de continuidad continuos; el número de programa, los PID y los nombres de servicio los define la sección `mpegts`; los PID en conflicto y los reservados se reasignan automáticamente.
- Vídeo H264/HEVC, audio AAC/AC3/EAC3, descriptores de idioma de las pistas, teletexto (descriptor teletext con páginas e idiomas).
- Las pistas JPEG de capturas nunca entran en el TS.

© Flussonic 2026

QUÉ VIENE DESPUÉS

- Capture MPEG-TS
- Retransmita el stream

© Flussonic 2026

RTSP

`media.play.rtsp`

TRANSPORTE

RTSP 1.0 y RTP/RTCP interleaved en una conexión TCP. UDP no es compatible: SETUP con UDP devuelve 461.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**

Audio

- **AAC**
- **Opus**
- **PCMA**
- **PCMU**

La lista describe la paquetización RTP de Sapsan; la decodificación depende del cliente. La entrega correcta requiere parámetros de códec y un formato de payload RTP compatible.

Sapsan actúa como servidor RTSP y entrega pistas compatibles a NVR, sistemas de analítica y otros servidores multimedia. La disponibilidad por RTSP depende de los códecs del stream y de la paquetización RTP.

AAC utiliza `MPEG4-GENERIC` con cabeceras AU; AV1 utiliza fragmentación RTP y un sequence header. Existen declaraciones SDP para `MP4A-LATM` y MPEG Audio (`MPA`, MP2/MP3), pero la salida actual no proporciona su paquetización RTP correspondiente completa; no se incluyen en la lista anterior. Si es necesario, convierta el audio a AAC u Opus. El nombre de un códec en SDP no garantiza por sí solo la compatibilidad.

CONFIGURACIÓN

Basta con activar el listener RTSP:

Configure estas opciones mediante central en la config del streamer.

```
listeners:
  rtsp:
    - port: 554
```

REPRODUCCIÓN

```
ffplay rtsp://server:554/cam1
```

Lo que soporta el servidor:

- los métodos OPTIONS, DESCRIBE, SETUP, PLAY, TEARDOWN;
- el transporte RTP/AVP/TCP (interleaved) — funciona por un solo puerto TCP, amigable con los firewalls;
- un RTCP Sender Report antes del primer paquete RTP de cada pista — el cliente obtiene de inmediato el anclaje al tiempo absoluto;
- vídeo H.264, HEVC y AV1, audio AAC (RFC 3640);
- las marcas de tiempo originales del stream se conservan.

La ruta URL es el nombre exacto del stream: `rtsp://server:554/group/cam1` selecciona `group/cam1`. Para reproducción protegida, añada `?token=<token>`: la solicitud usa el mecanismo compartido de autorización. El acceso sin token depende de la configuración de autorización. Envíe el token en la URL; un encabezado HTTP Bearer no lo sustituye en RTSP.

© Flussonic 2026

Para cada pista, el cliente envía SETUP con `RTP/AVP/TCP;unicast;interleaved=0-1` (otro par de canales para la siguiente pista). PLAY inicia la reproducción en vivo. También se admiten PAUSE y GET_PARAMETER para mantener la sesión. NPT numérico no selecciona el archivo: utilice el modo RTSP DVR.

QUÉ VIENE DESPUÉS

- Capture un stream por RTSP
- Archivo por RTSP

© Flussonic 2026

RTSP DVR

`media.play.rtsp_dvr`

Sapsan entrega el archivo grabado mediante RTSP. El cliente obtiene los periodos de grabación disponibles, selecciona una hora UTC y controla la reproducción dentro de una misma sesión. La lectura de un archivo existente no requiere una fuente de entrada activa.

TRANSPORTE

RTSP 1.0 y RTP/RTCP interleaved comparten una conexión TCP. UDP no es compatible: `SETUP` con UDP devuelve `461`.

CÓDECS COMPATIBLES

- **H.264**
- **HEVC / H.265**
- **AV1**
- **AAC**
- **Opus**
- **PCMA**
- **PCMU**

La lista corresponde a la paquetización RTSP; el cliente debe admitir los códecs seleccionados.

AAC utiliza `MPEG4-GENERIC` con cabeceras AU; AV1 utiliza fragmentación RTP y un sequence header. Existen declaraciones SDP para `MP4A-LATM` y MPEG Audio (`MPA`, MP2/MP3), pero la salida actual no proporciona su paquetización RTP correspondiente completa; no se incluyen en la lista anterior. Si es necesario, convierta el audio a AAC u Opus. El nombre de un códec en SDP no garantiza por sí solo la compatibilidad.

PROTOCOLO Y URL

El control utiliza RTSP 1.0: RFC 2326. La descripción de las pistas utiliza SDP: RFC 4566. Los medios se transmiten mediante RTP/RTCP: RFC 3550. Para H.264 se utiliza el formato de carga RTP de RFC 6184.

La lista de periodos de archivo en un único atributo SDP y el parámetro `position` son extensiones de la interfaz de archivo de Sapsan. Sus formatos se describen a continuación. Se indican las capacidades admitidas y las diferencias conocidas; admitir RTSP 1.0 no implica admitir todos los modos de estos RFC.

El acceso requiere un archivo DVR grabado, un listener RTSP configurado y una licencia que permita `media.play.rtsp_dvr`.

```
rtsp://sapsan.example.com:554/camera?dvr=true
```

La URL es estática: el tiempo se indica en la cabecera `Range`, no en la URL. Se requiere el valor `dvr=true`; otros valores se rechazan. Una URL normal sin este parámetro sirve la reproducción en directo y no acepta un `Range: clock=...` de archivo. Si se requiere autorización mediante token, añada `&token=<token>`.

LICENCIA

La función `media.play.rtsp_dvr` utiliza el límite de licencia independiente `media_play_rtsp_dvr`. RTSP en directo utiliza `media_play_rtsp`; el permiso para reproducir en directo no concede acceso al archivo.

La admisión se comprueba antes de leer el archivo, también en un `PLAY` directo sin `DESCRIBE` previo. Un límite de cero prohíbe el acceso y un límite numérico agotado impide crear nuevas sesiones: el servidor responde `453 Not Enough Bandwidth`.

La sesión de archivo conserva su admisión por licencia al terminar el intervalo seleccionado. El siguiente `PLAY` en la misma conexión utiliza esa sesión. Libérela con `TEARDOWN` o cerrando la conexión; EOF por sí solo no libera la plaza ocupada. Consulte licencias.

PERIODOS DE GRABACIÓN DISPONIBLES

El cliente comienza con `DESCRIBE`:

```
DESCRIBE rtsp://sapsan.example.com:554/camera?dvr=true RTSP/1.0
CSeq: 1
Accept: application/sdp
```


© Flussonic 2026

La respuesta `200 OK` contiene un cuerpo `application/sdp`. Antes de las descripciones de pistas `m=`, incluye este atributo de sesión:

```
a=range:clock=20261019T080000Z-20261019T093000Z;clock=20261019T100000Z-
```

El ejemplo muestra dos periodos de grabación separados por una interrupción. El formato temporal es `YYYYMMDDThhmmssZ`, en segundos enteros UTC. La ausencia de hora tras el guion del último periodo indica que la grabación continúa.

- **Orden.** Los periodos se enumeran en orden cronológico ascendente.
- **Unión.** Se unen los periodos contiguos; al construir la lista también se unen las interrupciones de hasta 10 segundos inclusive. Un periodo así no garantiza fotogramas en cada segundo que abarca.
- **Cantidad.** Se anuncian como máximo los 100 periodos más recientes. Las grabaciones anteriores que todavía existan se pueden solicitar con un `PLAY` explícito, aunque no aparezcan en esta lista.
- **Archivo vacío.** Se omite el atributo `a=range`. El servidor puede utilizar una fuente en directo disponible para describir las pistas.

Si hay grabaciones, las descripciones de las pistas se obtienen del archivo. Por tanto, `DESCRIBE` y la reproducción siguen disponibles tras desconectar la fuente de entrada.

PREPARACIÓN DEL TRANSPORTE

Envíe `SETUP` para cada pista necesaria utilizando su identificador del SDP. Se admite RTP/RTCP dentro de la misma conexión TCP que RTSP: *interleaved TCP*. En el ejemplo, `<track-url>` representa la URL de la pista seleccionada:

```
SETUP <track-url> RTSP/1.0
CSeq: 2
Transport: RTP/AVP/TCP;unicast;interleaved=0-1
```

Cabeceras relevantes de la respuesta:

```
RTSP/1.0 200 OK
CSeq: 2
Session: example-session;timeout=60
Transport: RTP/AVP/TCP;unicast;interleaved=0-1
```

Utilice el `Session` recibido en las solicitudes posteriores. Seleccione otro par de canales interleaved para la siguiente pista. El transporte UDP se rechaza con `461`.

Los ejemplos muestran los campos relevantes del intercambio. En la red, las líneas terminan en CRLF, una línea vacía separa las cabeceras del cuerpo y `Content-Length` indica la longitud del cuerpo en bytes.

SELECCIÓN TEMPORAL Y BÚSQUEDA

Para reproducir un intervalo, envíe `PLAY` a la URL de toda la sesión:

```
PLAY rtsp://sapsan.example.com:554/camera?dvr=true RTSP/1.0
CSeq: 3
Session: example-session
Range: clock=20261019T081000Z-20261019T081500Z
Scale: 1
```

Campos esenciales de la respuesta:

```
RTSP/1.0 200 OK
CSeq: 3
Session: example-session;timeout=60
Range: clock=20261019T081000Z-20261019T081500Z
Scale: 1
```


© Flussonic 2026

La respuesta `RTP-Info` contiene `seq` y `rtptime` para las pistas preparadas que se entregan, también al iniciar el archivo y después de una búsqueda. Al reanudar tras una pausa, `rtptime` corresponde al siguiente fotograma real, incluido un salto sobre una interrupción de grabación.

Solicitud `PLAY`	Comportamiento de Sapsan
`Range: clock=<from>-<to>`	Comenzar en un fragmento independiente y entregar fotogramas con DTS inferior a `<to>`
`Range: clock=<from>-`	Seleccionar fragmentos existentes desde la hora indicada, sin límite superior
Un nuevo `Range: clock=...` en la misma sesión	Cambiar inmediatamente la posición; no se necesita otro `SETUP`
`Range: npt=now-`	Cambiar a la reproducción en directo
Sin `Range`, con una fuente ya seleccionada	Continuar la reproducción, también después de `PAUSE`
Sin `Range`, en el primer inicio	Seleccionar la fuente en directo
NPT numérico abierto, como `npt=10-`	`451 Parameter Not Understood`; la reproducción actual continúa

El `Range` de respuesta informa de los límites seleccionados. Si `<from>` está dentro de un fragmento independiente, la reproducción comienza al inicio de ese fragmento, que puede ser anterior a lo solicitado. La búsqueda hacia atrás se limita a 60 segundos; si no hay un inicio adecuado, se selecciona el siguiente fragmento independiente. El final se limita por `<to>` y por la grabación disponible. No se entregan fotogramas con DTS igual o superior a `<to>`.

El guion se conserva también con un extremo abierto. Las solicitudes aceptan segundos UTC enteros, pero un límite real de la respuesta puede incluir milisegundos, por ejemplo `clock=20261019T081000.250Z-20261019T081500Z`.

La reproducción es hacia delante a velocidad normal. Omita `Scale` o indique `1`; una solicitud con `Scale: 1` recibe `Scale: 1`. Otras velocidades y la forma inversa `clock=-<to>` devuelven `451 Parameter Not Understood` y la reproducción actual continúa.

Si hay una interrupción de grabación entre fragmentos seleccionados, el servidor pasa al siguiente fragmento disponible sin esperar toda la duración de la interrupción. Las marcas de tiempo saltan el periodo ausente.

Un intervalo abierto lee la selección de fragmentos disponible al recibir `PLAY` y termina al agotarla. No incorpora nuevos fragmentos ni cambia al directo automáticamente. Para el directo se necesita un `PLAY` explícito con `npt=now-`.

POSICIÓN ACTUAL

Para obtener la posición de entrega, envíe `GET_PARAMETER` con `position` en el cuerpo:

```
GET_PARAMETER rtsp://sapsan.example.com:554/camera?dvr=true RTSP/1.0
CSeq: 4
Session: example-session
Content-Type: text/parameters
Content-Length: 10

position
```

El cuerpo es `position` seguido de CRLF, 10 bytes en total. Respuesta:

```
RTSP/1.0 200 OK
CSeq: 4
Session: example-session;timeout=60
Range: clock=20261019T081012Z
Supported: position
```

El tiempo aparece en una **cabecera**, no en el cuerpo de la respuesta. Es una única marca temporal sin guion, a diferencia del intervalo de la respuesta `PLAY`. El valor corresponde al DTS del último fotograma entregado, truncado a segundos enteros, no al momento en que el cliente lo muestra.

Antes del primer fotograma entregado de la fuente seleccionada, también después de una búsqueda, `position` devuelve `503 Service Unavailable` sin `Range`, con `Supported: position`. Durante una pausa y después de EOF del archivo devuelve `200 OK` y la última posición confirmada. El código `200` por sí solo no indica si siguen llegando fotogramas. Un parámetro desconocido en un cuerpo no vacío devuelve `200 OK` y `Supported: position`, sin `Range`.

© Flussonic 2026

PAUSA Y FINALIZACIÓN DE LA SESIÓN

`PAUSE` en la URL de la sesión suspende la entrega conservando la posición dentro del fragmento. El siguiente `PLAY` sin `Range` reanuda los fotogramas restantes. Un nuevo `Range: clock=...` cambia inmediatamente la posición, también al enviar varias solicitudes `PLAY` en pipeline; no se crea una cola de intervalos.

Durante la pausa, mantenga la sesión activa con un `GET_PARAMETER` vacío. El servidor anuncia `timeout=60`; envíe solicitudes keepalive con mayor frecuencia que ese intervalo. Las solicitudes RTSP y los paquetes RTCP entrantes cuentan como actividad.

Cuando un intervalo que ya ha entregado medios RTP llega a su final natural, el servidor envía RTCP BYE para las pistas preparadas y termina la entrega actual. La conexión y la sesión permanecen abiertas: se puede solicitar otro intervalo sin repetir `SETUP`.

Para liberar la sesión, envíe `TEARDOWN` con su `Session`. Después de `200 OK`, la conexión se cierra y se libera la plaza de licencia.

MEDIOS Y LÍMITES DE COMPATIBILIDAD

Para H.264, los parámetros del decodificador se transmiten en SDP y los SPS/PPS que falten se insertan antes de los fotogramas clave. Los parámetros se actualizan cuando cambia la configuración de los fragmentos del archivo. Las marcas RTP se calculan a partir del DTS del archivo; RTCP Sender Report las relaciona con el tiempo NTP.

RTCP Sender Report contiene los contadores acumulados de paquetes RTP y octetos de carga útil. Las cabeceras RTP y la envoltura interleaved no se incluyen en los octetos. Una búsqueda con el mismo SSRC no reinicia los contadores.

El perfil admitido tiene los siguientes límites:

- **Transporte.** Solo interleaved TCP; UDP devuelve `461`.
- **Velocidad.** Solo reproducción normal hacia delante; otros valores de `Scale` devuelven `451`.
- **Escala temporal.** Para buscar en el archivo, utilice `clock` con segundos UTC enteros. No se admiten NPT numérico ni SMPTE. Las fracciones de segundo solo se permiten en las respuestas.
- **Secuencia de intervalos.** Cada nuevo `PLAY` de archivo cambia inmediatamente la posición; no hay cola de intervalos.
- **Grabación en curso.** Un intervalo abierto no lee fragmentos creados después del inicio de la reproducción.

CÓDIGOS DE RESPUESTA

Código	Causa
`400 Bad Request`	`dvr` o URL incorrectos; falta `Transport` en `SETUP`
`403 Forbidden`	Autorización denegada o stream cifrado no disponible para entrega RTSP
`404 Not Found`	No se encuentra el stream, el archivo o la pista
`451 Parameter Not Understood`	`Range` o `Scale` no admitidos; archivo solicitado mediante una URL normal de directo
`453 Not Enough Bandwidth`	Función prohibida por la licencia o límite de sesiones alcanzado
`454 Session Not Found`	Falta el contexto de sesión necesario
`457 Invalid Range`	Ventana vacía o inversa; no hay fragmentos de archivo adecuados
`461 Unsupported Transport`	Se ha solicitado un transporte distinto de interleaved TCP
`503 Service Unavailable`	Fuente no disponible; `position` solicitado antes del primer fotograma de la fuente seleccionada. EOF del archivo por sí solo no provoca `503` para `position`

© Flussonic 2026

VERIFICACIÓN DEL CLIENTE

Compruebe la integración con una grabación de límites temporales conocidos y una interrupción:

1. Obtenga el SDP y compare `a=range` con los periodos de grabación, teniendo en cuenta la unión de interrupciones cortas.
2. Envíe `SETUP` para las pistas necesarias. Compruebe los pares de canales interleaved y el identificador de sesión.
3. Solicite `position` antes del primer fotograma: se espera `503` sin `Range`.
4. Inicie `PLAY` dentro de un fragmento con `Scale: 1`. Compruebe el `Range` elegido, el `Scale` devuelto y que `rtptime` / `seq` correspondan al primer paquete RTP. Ningún fotograma debe tener DTS >= `<to>`.
5. Pause dentro de un fragmento y reanude con `PLAY` sin `Range`. Compruebe que se conservan los fotogramas restantes y la posición.
6. Compruebe `451` para una velocidad no admitida y NPT numérico: la reproducción actual debe continuar.
7. Espere EOF y RTCP BYE y solicite otro intervalo en la misma conexión. Compruebe la posición tras EOF y los contadores acumulados de Sender Report.
8. Compruebe el rechazo por licencia `453` con un límite de cero o agotado y termine con `TEARDOWN`.

© Flussonic 2026

RTMP

`media.play.rtmp`

TRANSPORTE

RTMP por TCP con mensajes multimedia continuos.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**

Audio

- **AAC**
- **MP3**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

HEVC y AV1 utilizan Enhanced RTMP con FourCC `hvc1` y `av01`; el cliente debe admitir estas extensiones. La compatibilidad convencional con RTMP/FLV no garantiza la reproducción de estos códecs de vídeo.

CONFIGURACIÓN

Sapsan entrega streams en vivo a clientes RTMP. Active el puerto mediante central en la config del streamer:

```
listeners:
  rtmp:
    - port: 1935
```

REPRODUCCIÓN

```
rtmp://server:1935/live/cam1
```

La aplicación del cliente es `live` y el play path es `cam1`. Sapsan selecciona el stream por el play path sin añadir la aplicación. Para `group/cam1`, envíe `group/cam1` completo como play path. Este endpoint entrega vídeo en vivo sin selección del archivo. HEVC y AV1 requieren un cliente Enhanced RTMP.

Para reproducción protegida, envíe el token en el play path (`cam1?token=<token>`). También se admite un token en los metadatos connect; el del play path tiene prioridad.

SIGUIENTES PASOS

- Autorización de reproducción
- Captura RTMP

© Flussonic 2026

SRT

`media.play.srt`

TRANSPORTE

MPEG-TS sobre SRT/UDP; el cliente se conecta en modo caller al puerto dedicado del stream.

CÓDECS COMPATIBLES

Vídeo

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**

Audio

- **AAC**
- **MP2**
- **MP3**
- **AC-3**
- **E-AC-3**
- **SMPTE 302M**

Otras pistas

- **SCTE-35**
- **KLV**
- **ID3**
- **tx3g**
- **Teletext**
- **DVB subtitles**

Esta lista describe la salida de Sapsan; la decodificación depende del reproductor. Las pistas también necesitan parámetros de códec válidos para el contenedor seleccionado.

Sapsan sirve un stream por SRT: cada stream recibe un puerto UDP dedicado al que se conectan los clientes SRT en modo caller.

AV1 en MPEG-TS utiliza la convención `stream_type = 0x06` con el descriptor de registro `AV01`; el receptor debe entender esta forma. Esto no garantiza compatibilidad con todas las implementaciones de AV1 en TS. Opus no se incluye en la lista: reconocer o anunciar un códec no basta para entregar una salida utilizable, y la ruta actual no proporciona el empaquetado TS/PES necesario para Opus.

CONFIGURACIÓN

Configure estas opciones mediante central en la config del streamer.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    srt_play:
      port: 4010
      passphrase: verysecretpass
```

`passphrase` activa el cifrado; sin ella, el stream se sirve sin cifrar.

© Flussonic 2026

La sección `srt_play` puede llevar una subsección `mpegts` — número de programa, PID de las pistas y SDT propios, solo para la reproducción desde este puerto.

MAPA DE CLIENTES

Un `passphrase` común al puerto es una única cerradura compartida: la llave la tiene cualquiera al que se la haya entregado. El mapa de clientes define cerraduras personales: cada espectador recibe su propio secreto, y la sesión recibe automáticamente un propietario — el nombre del registro coincidente.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    srt_play:
      port: 4010
      clients:
        ivan:
          passphrase: ivan-personal-secret
        office:
          ips: [198.51.100.0/24]
```

Un registro del mapa es un nombre de cliente con dos campos opcionales:

- **`passphrase`** — el secreto personal del registro: si está definido, el handshake se cifra y el espectador debe presentar exactamente ese; si no lo está, el registro acepta handshakes sin cifrar.
- **`ips`** — un filtro de direcciones: una dirección o subred desde las que se aceptan conexiones. Un conjunto ausente o vacío no restringe las conexiones.

La admisión se compone de los campos de los registros: una conexión cifrada se acepta cuando su material de clave se despliega con el `passphrase` de uno de los registros con contraseña cuyas direcciones incluyen la del espectador; una sin cifrar — por un registro sin contraseña con dirección coincidente, o por un registro abierto (sin contraseña y sin direcciones) desde cualquier dirección. Los registros con y sin contraseña conviven en un mismo mapa: el cifrado es una propiedad del handshake, no del puerto.

La sesión del espectador lleva el propietario — el nombre del registro del mapa coincidente. En una entrega sin mapa, el propietario procede de la etiqueta `user_id_<nombre>` en `labels`; las sesiones de una entrega sin etiquetar y sin mapa quedan sin sujeto.

La configuración se valida al guardar:

- **`passphrase` idénticos** de dos clientes de una misma entrega se rechazan: el secreto es lo que permite a la sesión identificar a su cliente.
- **Conjuntos de `ips` solapados** de dos clientes se rechazan: un filtro de direcciones debe pertenecer a un único cliente.

En la consola de gestión el servidor genera el secreto al añadir un cliente. En YAML, un registro abierto es una forma válida de acceso libre.

El espectador se conecta con su propio secreto:

```
ffplay "srt://server:4010?passphrase=ivan-personal-secret"
```

OBSERVABILIDAD DE LOS HANDSHAKES

Una entrega con mapa de clientes acepta handshakes desde la red, y su carga principal es la comprobación de secretos. El nodo cuenta los handshakes como series en el scrape del nodo (`GET /streamer/api-v4/runtime/metrics`):

Serie	Qué cuenta
`srt_handshake_attempts_total`	intentos que llegaron a una decisión de admisión
`srt_handshake_secret_candidates_total`	candidatos de secreto probados — el número total de comprobaciones de material de clave
`srt_handshake_bad_secret_total`	rechazos: ningún candidato coincidió
`srt_handshake_address_filter_rejects_total`	rechazos del filtro de direcciones

```
# un espectador normal gasta un par de candidatos por intento; la fuerza bruta prueba todo el mapa
rate(srt_handshake_secret_candidates_total[5m])
  / rate(srt_handshake_attempts_total[5m])
```


© Flussonic 2026

REPRODUCCIÓN

```
ffplay "srt://server:4010?passphrase=verysecretpass"
```

QUÉ VIENE DESPUÉS

- Capture un stream por SRT
- Retransmita el stream por SRT a otro servidor

© Flussonic 2026

NDI

`media.play.ndi`

Sirve un stream activo a receptores NDI. La tarifa contabiliza esta capacidad por stream: una fuente anunciada puede servir a varios receptores. La salida requiere el listener del streamer y el permiso de salida NDI del stream.

TRANSPORTE

NDI High Bandwidth con SpeedHQ, servido por TCP. Use **Auto** en el receptor. El catálogo compartido del servidor utiliza TCP `5960`; los puertos de medios se asignan automáticamente dentro de TCP `5961` – `6060`. Cada fuente anunciada necesita un puerto de medios. El descubrimiento local utiliza mDNS por UDP `5353`.

La salida NDI no tiene autenticación. Restrinja el acceso de red a receptores de confianza; no se solicita un token de reproducción. Esta capacidad no proporciona NDI HX ni un push a una dirección de destino preconfigurada.

PREPARACIÓN DE PISTAS

El codificador NDI utiliza pistas raw preparadas de un stream activo:

- **Vídeo:** la primera pista `v210` o `uyvy422` apta en el orden de MediaInfo.
- **Audio:** la primera pista apta en uno de estos formatos:
  - **PCM de 16 bits**
  - **S32P**
  - **FLTP**
- **Sin audio:** se admite la salida solo de vídeo.

Las pistas comprimidas se omiten. Configure por separado la decodificación de la entrada comprimida y el formato de píxel necesario, como para la salida SDI. Las pistas raw no se combinan y la salida NDI no permite seleccionarlas por número. Un stream existente puede conservar pistas comprimidas para otros receptores y añadir pistas raw aptas.

VÍDEO ENTRELAZADO

El vídeo entrelazado se emite entrelazado. NDI solo admite el orden «campo superior primero», por lo que la salida desplaza una línea el vídeo con campo inferior primero (480i) y lo emite con el campo superior primero.

CONFIGURACIÓN

Configure los ajustes a través de central. En la configuración actual del clúster, fije el stream a un servidor concreto: asigne al streamer un valor único de `labels`, `ndi-studio-a`, y al stream `required_ndi-studio-a`. Active el listener en ese mismo servidor. Si varios servidores tienen la etiqueta, se permite asignar el stream a cualquiera de ellos. El procedimiento está en la guía de Agora.

Fragmento de la configuración aplicada al streamer con entrada SDI raw:

```
listeners:
  ndi:
    bind: 192.168.10.20
    port: 5960
    ports: {from: 5961, to: 6060}
    queue: 4
streams:
  - name: conference
    inputs:
      - sdi:
          device: /dev/video0
          pixel_format: v210
    ndi_output:
      quality: 90
```


© Flussonic 2026

Sustituya la dirección y el dispositivo por los suyos. Para una entrada comprimida, prepare las pistas raw con el transcodificador. La conexión de un receptor NDI no inicia un stream detenido.

Campo	Valor y restricciones
`listeners.ndi`	Omitir la sección desactiva el listener.
`bind`	IP opcional de la interfaz; si se omite, se usan todas las interfaces IPv4.
`port`	Puerto del catálogo, por defecto `5960`; debe quedar fuera del rango de medios.
`ports.from`, `ports.to`	Límites inclusivos de los puertos de medios, por defecto `5961` y `6060`.
`queue`	Límite positivo de cola por cliente en imágenes completas, por defecto `4`; un par de campos cuenta como una imagen.
`ndi_output`	Omitir la sección desactiva la reproducción NDI de este stream.
`quality`	Calidad SpeedHQ fija de `52` a `99`, por defecto `90`, sin ajuste adaptativo.

Permita que los receptores accedan al catálogo y a los puertos de medios. Un puerto de catálogo distinto del predeterminado puede impedir la búsqueda solo por IP en NDI Access Manager.

REPRODUCCIÓN Y ESTADO

Seleccione `nombre-servidor (nombre-stream)` en el receptor NDI. El receptor abre la conexión; varios receptores comparten un codificador NDI. El primer espectador inicia el codificador, que se detiene tras 20 segundos sin espectadores. El anuncio y el puerto asignado se mantienen hasta que se desactive la salida.

El estado aparece en `ndi_output` de las estadísticas del stream. `address` y `viewers` indican la dirección del puerto de medios y el número de espectadores admitidos; `encoder_running` y `signal` indican el estado del codificador y si se preparó una imagen completa durante el último segundo. `error` describe los errores de preparación o publicación.

Las conexiones se mantienen si dejan de llegar fotogramas; la salida NDI no genera vídeo negro ni silencio. Prepare una cortinilla mediante la fuente de reserva del stream. La cola limitada de un cliente lento descarta primero imágenes completas; tras 5 segundos de descartes continuos se desconecta a ese cliente. Desactivar `ndi_output` o el listener cierra inmediatamente las conexiones y retira los anuncios.

FUNCIONES RELACIONADAS

- Captura NDI — capacidad independiente `media.ingest.ndi`.
- Gestión de puertos.
- Asignación y etiquetas.

© Flussonic 2026

Reproducción WebRTC (WHEP)

`media.play.webrtc`

CONFIGURACIÓN

Configure estas opciones mediante central. Ejemplo de config del streamer:

`listeners.webrtc` define puertos UDP compartidos por WHIP y WHEP. Si falta o está vacío, el sistema operativo selecciona un puerto libre. `port: 0` también activa la selección automática. Los candidatos ICE del SDP anuncian el puerto asignado; no se reserva un puerto por sesión. Volver a aplicar la misma lista conserva el puerto. Un reinicio puede cambiarlo.

Parámetro	Valor	Comportamiento
`listeners.webrtc`	Ausente o `[]`	Un puerto UDP automático
`listeners.webrtc[].port`	Obligatorio por entrada, `0` – `65535`	`0` selecciona el puerto automáticamente; otro valor lo fija

```
listeners:
  webrtc:
  - port: 5005
  - port: 5006
```

Para reglas fijas del firewall o reenvío detrás de NAT, configure puertos explícitos y permita el acceso de los clientes. El proxy HTTP gestiona el SDP; los medios UDP deben llegar al propio streamer. La dirección ICE se obtiene del enrutamiento IPv4 local; `listeners.webrtc` no ofrece ajustes de dirección ICE pública ni TURN. La selección automática del puerto no configura NAT.

URL DE REPRODUCCIÓN

URL directa del streamer:

```
https://streamer.example.com/streaming/whep/studio
```

La URL corresponde a un cliente WHEP que envía SDP por POST. Abrirla en la barra del navegador ejecuta GET y no inicia la reproducción.

AUTENTICACIÓN

WHEP utiliza la autorización de reproducción. Envíe un token permitido en `Authorization: Bearer PLAY_TOKEN` o `?token=PLAY_TOKEN`. El acceso se comprueba al crear la sesión; los bytes se contabilizan en ella. Un token de publicación no concede por sí solo acceso de reproducción.

TRANSPORTE

HTTP(S) gestiona el SDP y el control de la sesión. Los medios utilizan UDP con ICE y DTLS-SRTP/SRTCP. Use HTTPS en el navegador; el acceso HTTP y el acceso al puerto UDP son requisitos independientes.

Si la página del navegador y el endpoint WHIP/WHEP usan distintos orígenes, añada en el proxy HTTPS la cabecera de respuesta `Access-Control-Expose-Headers: Location` para que el cliente pueda leer la URL de sesión. No es necesario si la página y la API comparten origen.

CÓDECS COMPATIBLES

- **H.264**
- **AV1**
- **Opus**

El vídeo utiliza H.264 o AV1; el audio, Opus. H.264 tiene prioridad en la negociación; WHEP también depende de los códecs de origen y del reproductor. WebRTC no transcodifica: convierta previamente otros códecs de audio a Opus. HEVC y VP8/VP9 no están disponibles en este procesamiento. WHEP requiere una pista de vídeo compatible; el audio es opcional.

© Flussonic 2026

API HTTP

Método y ruta	Resultado
`POST /streaming/whep/{stream}`	Oferta SDP (`application/sdp`); devuelve `201 Created` , respuesta SDP y URL de sesión en `Location`
`GET /streaming/whep/{stream}`	`204 No Content` ; no devuelve vídeo ni una página de reproductor
`OPTIONS /streaming/whep/{stream}`	CORS y tipos de contenido aceptados
`DELETE /streaming/whep/session/{id}`	Cierra la sesión; devuelve `204 No Content`
`PATCH /streaming/whep/session/{id}`	Un fragmento SDP válido no vacío recibe `501 Not Implemented`

La oferta debe incluir todos los candidatos ICE del cliente; no se admite añadirlos después mediante PATCH. Guarde `Location` para cerrar la sesión. El servidor responde `recvonly` para WHIP y `sendonly` para WHEP.

REQUISITOS Y COMPORTAMIENTO

El stream debe estar activo y proporcionar `MediaInfo` válido. El cliente necesita un puerto UDP accesible y códecs compatibles. SRTP protege la conexión, pero WHEP no entrega contenido con cifrado DRM activado.

GESTIÓN DE ERRORES

Código	Causa
`400`	SDP incorrecto u oferta no negociable
`403`	Acceso denegado; WHEP también rechaza contenido cifrado
`404`	El stream no está activo; WHEP también requiere información de pistas disponible
`415`	Content-Type incorrecto; WHEP también requiere vídeo compatible
`503`	No se pudo registrar la ruta multimedia; un grupo de espectadores WHEP también puede estar temporalmente no disponible

LIMITACIONES

La oferta y la respuesta iniciales contienen todos los candidatos ICE. No se admiten Trickle ICE, reinicio ICE, contraoferta WHEP ni alternativa TCP. Cambiar la lista de puertos UDP reconstruye el transporte: vuelva a conectar los clientes. Una lista sin cambios conserva los sockets.

SIGUIENTES PASOS

- Publicación WHIP
- Ver mediante Agora

© Flussonic 2026

Media over QUIC

`media.play.moq`

Sapsan entrega medios en directo por Media over QUIC: el cliente se conecta al servidor, elige un stream y se suscribe a sus pistas. Sapsan actúa como publicador de medios. La fuente de entrada puede usar otro protocolo, por ejemplo, RTSP.

La implementación actual utiliza MoQT transport draft-19. Es una versión concreta de un Internet-Draft; la compatibilidad del cliente debe comprobarse con esa versión. El formato del catálogo sigue MSF draft-01, y los medios se transmiten en CMAF.

CÓMO FUNCIONA LA ENTREGA

Para cada stream activo, Sapsan publica un broadcast cuyo espacio de nombres coincide con el nombre del stream. Por ejemplo, `cam` se convierte en el nombre que el cliente elige al suscribirse.

- **Descubrimiento** — una suscripción a un prefijo vacío del espacio de nombres permite obtener la lista de streams publicados.
- **Catálogo** — la pista `catalog` contiene JSON con la descripción de los medios y los datos de inicialización.
- **Pistas** — el vídeo y el audio se publican con nombres como `v1` y `a1`.
- **Objetos** — las partes CMAF se transmiten a medida que están disponibles. El límite de un segmento inicia un grupo nuevo, y sus partes se convierten en objetos consecutivos.

Un cliente que se conecta a mitad de un segmento espera un inicio de grupo adecuado. Una suscripción establecida recibe los objetos siguientes a medida que se producen. La latencia final depende de la fuente, la codificación, la estructura de segmentos y partes, la red y el búfer del reproductor.

Hay dos formas de conexión a un mismo listener UDP:

Cliente	Transporte
Cliente MoQT nativo	QUIC con ALPN `moqt-19`.
Navegador	WebTransport sobre HTTP/3, ALPN `h3`; el cliente utiliza una variante compatible de MoQT draft-19.

Esta ruta está destinada a la entrega de vídeo y audio en directo. No ofrece reproducción de DVR/VOD, captura de cámaras por MoQ ni entrega de subtítulos. La suscripción MoQ no sustituye la configuración de la fuente o del transcodificador.

CONFIGURACIÓN DEL LISTENER

Defina `listeners.moq` en los ajustes del streamer. En una instalación operativa, la configuración se gestiona mediante central. Para una comprobación local, puede usar este fragmento de configuración del servidor:

```
listeners:
  http:
    - port: 8080
  moq:
    - bind: "127.0.0.1:4443"
      certificate: /etc/sapsan/tls/moq-cert.pem
      private_key: /etc/sapsan/tls/moq-key.pem

streams:
  - name: cam
    inputs:
      - rtsp:
          url: rtsp://camera.example/live
```

Sustituya la fuente y las rutas del certificado y de la clave. `bind` es una dirección IP numérica y un puerto UDP; el puerto 4443 se ha elegido para el ejemplo. Permitir solo TCP en ese puerto no permite establecer una conexión QUIC.

Para la primera comprobación en navegador, utilice una fuente con H.264 y AAC compatibles con el navegador elegido. El catálogo describe las pistas existentes: MoQ por sí solo no convierte un códec incompatible con el navegador en uno compatible. Si hace falta, configure previamente la transcodificación.

ACCESO AL LISTENER

El listener MoQ actual permite acceder a los streams en directo publicados por el servidor sin integración con la autorización HTTP ni con las sesiones de reproducción. La validación de un ticket JWT o de `?token=` para HLS no se aplica a MoQ. TLS cifra el transporte, pero no determina quién puede suscribirse a un stream concreto.

© Flussonic 2026

Al elegir la dirección del listener, tenga en cuenta qué clientes pueden alcanzarlo. En este ejemplo, solo es accesible desde la máquina local. Para un entorno de prueba remoto, limite el acceso de red al puerto UDP a los participantes de la prueba.

VERIFICACIÓN DE LA RECEPCIÓN DE MEDIOS

El árbol de fuentes de Sapsan incluye la utilidad `moq-probe`. Establece una conexión MoQT, descubre el stream y lee el catálogo. Si se especifica una pista, también verifica la recepción de un objeto multimedia.

Después de iniciar Sapsan con el stream `cam` activo y la pista de vídeo `v1`, ejecute lo siguiente desde la raíz del árbol de fuentes:

```
MOQ_PROBE_SERVER=127.0.0.1:4443 \
MOQ_PROBE_BROADCAST=cam \
MOQ_PROBE_TRACK=v1 \
cargo run -p moq --bin moq-probe
```

Para compilar la utilidad se necesita el entorno Rust del proyecto. `MOQ_PROBE_SERVER` acepta una IP y un puerto, sin el prefijo `https://`; el nombre del broadcast y de la pista deben coincidir con el stream en ejecución.

Resultado esperado:

```
MOQT probe OK: SETUP + SUBSCRIBE_NAMESPACE + catalog + media object
```

El parámetro `MOQ_PROBE_TRACK` es importante: sin él, un resultado correcto significa que se ha comprobado el catálogo, pero no se ha recibido un objeto de vídeo. La utilidad comprueba la estructura del objeto CMAF, sin decodificar la imagen ni el audio. Esta utilidad de diagnóstico también omite la verificación del certificado del servidor, por lo que su éxito no confirma la confianza TLS de un cliente normal.

Un HTTPS GET convencional o abrir la dirección del listener MoQ como una página web no realiza una suscripción a los medios ni sustituye esta comprobación.

VERIFICACIÓN DE LA REPRODUCCIÓN EN EL NAVEGADOR

El árbol de fuentes incluye un entorno de prueba con el reproductor estándar `@moq/watch` y una versión compatible fijada de la biblioteca cliente. Los siguientes comandos se ejecutan desde la raíz del repositorio Sapsan.

1. Cree un certificado de prueba:

```
bash
    crates/moq/tests/gen-browser-cert.sh
```

El script crea `crates/moq/tests/wt-cert.pem` y `wt-key.pem`. Especifique sus rutas absolutas en `certificate` y `private_key` del listener. Es un certificado ECDSA de 14 días, adecuado para la comprobación mediante el hash fijado del certificado en el entorno de navegador.

1. Inicie Sapsan con esta configuración y una fuente compatible. Por ejemplo, para una ejecución local desde las fuentes:

```
bash
    CONFIG_PATH=/absolute/path/to/moq.yaml \
    cargo run --no-default-features --bin sapsan
```

Este comando está destinado al entorno de prueba local. Las capacidades adicionales de esta ejecución dependen de la compilación; para la comprobación, elija una fuente compatible que ya funcione.

1. Inicie la parte del navegador:

```
bash
    crates/moq/tests/watch-sapsan/serve.sh
```

Se necesitan `bun`, `git` y `openssl`; en el primer arranque, el script obtiene las dependencias. Inicia el entorno de prueba cliente, no el propio Sapsan.

1. Abra `http://localhost:5173/sapsan.html?name=cam` en un navegador compatible con WebTransport. Por defecto, el entorno se conecta a `https://127.0.0.1:4443`. Inicie la reproducción y active el audio con un clic si el navegador lo exige.

La comprobación del navegador tiene éxito cuando se reproducen imagen y audio, no solo cuando aparece el nombre del stream en la lista. El uso de `localhost` es intencional: WebTransport exige un contexto seguro, y una página HTTP convencional accedida por una IP de la LAN no equivale a este entorno. El indicador del búfer del reproductor tampoco mide la latencia total desde la cámara hasta la pantalla.

© Flussonic 2026

DVR

`media.play.dvr`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Dependen del protocolo de entrega: HLS y DASH. Sapsan empaqueta las pistas existentes; compruebe la compatibilidad de decodificación en el reproductor de destino.

El archivo grabado está disponible por los mismos protocolos que el directo: HLS y DASH. Además existe una API para consultar los rangos grabados y los previews.

La reproducción del archivo funciona solo en streams con la grabación activada (`dvr`) — de lo contrario, la solicitud se rechaza con un error claro.

REPRODUCCIÓN POR TIEMPO ABSOLUTO

El intervalo del archivo se fija con los parámetros `from` / `to` (UTC en milisegundos) en las URL habituales:

```
http://server/streaming/v/<stream>/index.m3u8?from=<utc_ms>&to=<utc_ms>
http://server/streaming/v/<stream>/Manifest.mpd?from=<utc_ms>&to=<utc_ms>
```

La playlist HLS del archivo es una playlist VOD cerrada. En DASH, los huecos de grabación se convierten en Periods, y la línea de tiempo se elige con `?timeline=compact|wallclock` — véase DASH.

QUÉ RANGOS ESTÁN GRABADOS

Los intervalos grabados los entrega la HTTP API:

```
http://server/streaming/dvr-range/<stream>
http://server/streaming/dvr-ranges/<stream>
```

> **Note**
>
> TODO: formato de respuesta de dvr-range / dvr-ranges.

SIGUIENTES PASOS

- Configurar la grabación
- Reproducir un archivo de otro servidor
- Proteger el archivo con autorización

© Flussonic 2026

Reproducción desde el inicio

`media.play.startover`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Dependen del protocolo de entrega: HLS y DASH. Sapsan empaqueta las pistas existentes; compruebe la compatibilidad de decodificación en el reproductor de destino.

El archivo grabado está disponible por los mismos protocolos que el directo: HLS y DASH. Además existe una API para consultar los rangos grabados y los previews.

La reproducción del archivo funciona solo en streams con la grabación activada (`dvr`) — de lo contrario, la solicitud se rechaza con un error claro.

REBOBINADO

Una playlist rebobinada N segundos hacia atrás desde el momento actual:

```
http://server/streaming/v/<stream>/rewind/<seconds>/index.m3u8
```

Por ejemplo, `rewind/3600/` es un timeshift de una hora. El rebobinado une sin costura el archivo con el borde del directo: la playlist empieza en el archivo y continúa con el directo, sin duplicados ni huecos en la costura; LL-HLS (blocking reload, actualizaciones delta) sigue funcionando. Los huecos de grabación se convierten en un `EXT-X-DISCONTINUITY` estándar.

© Flussonic 2026

VOD

`media.play.vod`

TRANSPORTE

HTTP o HTTPS; las listas y los fragmentos se entregan mediante solicitudes separadas por TCP.

CÓDECS COMPATIBLES

Dependen del protocolo de entrega: HLS y DASH. Sapsan empaqueta las pistas existentes; compruebe la compatibilidad de decodificación en el reproductor de destino.

Sapsan guarda archivos como assets VOD, mantiene su catálogo en central y los entrega bajo demanda. Los assets no necesitan una fuente en vivo. Gestione cargas, almacenamientos, plantillas y tareas en la consola de central.

ALMACENAMIENTO Y CARGA

Mediante central, defina una raíz VOD con nombre en la config del streamer, que corresponde a un directorio del disco:

```
vod:
  roots:
    - name: movies
      path: /srv/vod
```

Los nombres contienen letras latinas minúsculas, dígitos, `_` y `-` y deben ser únicos en la config. El proceso necesita acceso de lectura y escritura al directorio. Central recibe disponibilidad y espacio libre en los informes del streamer.

En **VOD**, abra **Cargas**, seleccione los archivos y una clave del asset. Central elige un almacenamiento disponible con espacio libre y emite un permiso de carga; los bytes van directamente al streamer. **Almacenamientos** muestra disponibilidad y capacidad. **Tokens de carga** da acceso restringido de carga a una aplicación externa; la revocación impide seguir utilizándolo.

CATÁLOGO Y ARCHIVOS

- **Clave:** nombre modificable como `movies/2026/film.mp4`; sus prefijos forman el árbol del catálogo.
- **Asset id:** identificador permanente que se conserva al renombrar la clave.
- **Revisión:** versión de la composición de archivos; cambia al añadir o eliminar archivos.
- **Archivos:** vídeo y otros archivos del asset, con tamaño, hash e información multimedia.

ARCHIVOS DEL ASSET

Abra un asset en **Catálogo** para inspeccionar archivos y ubicaciones. Un asset disponible tiene una ubicación lista en un almacenamiento disponible. Un archivo que no puede analizarse como multimedia solo permite descarga **RAW**. **Renombrar clave** cambia los metadatos y conserva archivos y enlaces por Asset id; una clave ocupada se rechaza.

REPRODUCCIÓN

En la ficha del archivo, **Play** abre un reproductor; **HLS**, **DASH** y **RAW** copian un enlace. Use enlaces de central para reproducir mediante el clúster:

Formato	URL de central
HLS	`/playback/vod/<asset-id>/<file>/index.m3u8`
DASH	`/playback/vod/<asset-id>/<file>/Manifest.mpd`
Archivo	`/playback/vod/<asset-id>/<file>`

Central selecciona una ubicación lista y redirige al cliente al streamer. La URL nativa del streamer es `/streaming/vod/<root>/<asset-id>/<file>/index.m3u8` (para DASH, `Manifest.mpd`; para descargar, sin sufijo). No sustituya Asset id por la clave del catálogo. La reproducción protegida requiere un token que permita acceder al asset; acceso denegado devuelve 403 y un asset o archivo inexistente, 404. Sin ubicaciones disponibles, central devuelve 503 con `Retry-After`.

© Flussonic 2026

COPIAS Y BALANCEO

Configure **Copias en otros servidores** en la plantilla de asset. El valor `1` significa el original y una copia en otro streamer. Una plantilla sin calidades copia archivos preparados sin transcodificar. Ejecute una tarea desde la ficha del asset o asigne una plantilla al cargar.

Central selecciona streamers conectados con una raíz VOD disponible, un diario de archivos legible, el rol **Copia** activado y espacio suficiente. Solo se permite una ubicación del asset por streamer. Se requiere el tamaño de los archivos más la reserva `vod_min_free_bytes` (1 GiB por defecto); se elige el almacenamiento elegible con más espacio libre. Sin candidatos, la tarea espera en la cola.

La copia es una tarea separada en el streamer de destino: descarga los archivos por la dirección interna, verifica tamaño y hash y publica la copia. Las copias no ocupan puestos de transcodificación. Desactivar el rol impide nuevas tareas; las que ya están en curso continúan. Los archivos añadidos después llegan a las copias existentes con la siguiente tarea.

La tabla de ubicaciones muestra original y copias con revisión y espectadores. Central dirige nuevas reproducciones a la ubicación lista con menos sesiones; excluye un streamer desconectado cuando vence su liveness. El enlace por Asset id no cambia. Una redirección no transfiere una conexión abierta: el reproductor debe reconectarse tras un fallo.

ELIMINACIÓN

- **Archivo:** elimina un archivo e incrementa la revisión; al eliminar el último se elimina el asset.
- **Asset:** elimina el asset y sus archivos del catálogo.
- **Carpeta:** elimina los assets bajo el prefijo de clave seleccionado.

El catálogo cambia inmediatamente. Si el almacenamiento no está disponible, la eliminación física espera a que se restablezca la conexión. Compruebe el estado de tareas y ubicaciones en central; una entrada del catálogo no garantiza una copia lista.

SIGUIENTES PASOS

- HLS
- DASH
- Autorización de reproducción

© Flussonic 2026

6.2.12 Publicaciones salientes (media.push)

RTMP

`media.push.rtmp`

Sapsan inicia una publicación saliente por RTMP. Un stream puede tener varios pushes con nombre.

CONFIGURACIÓN

Aplique la configuración del stream mediante central.

Los pushes de un stream son un mapa de «nombre del push — configuración», no una lista. El nombre es único dentro del stream y sirve de clave: por él se encuentra el push en la configuración, en las estadísticas y en las métricas. Un push tiene exactamente un protocolo.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    pushes:
      delivery:
        rtmp:
          url: rtmp://a.rtmp.youtube.com/live2/xxxx-xxxx
```

Protocolo	Parámetros
`rtmp`	`url`, `connect_timeout_ms`

El nombre es la identidad del push. Renombrarlo no traslada ni la configuración ni los contadores acumulados: el push antiguo desaparece y el nuevo empieza de cero.

CAPACIDADES Y COMPORTAMIENTO

- El push RTMP soporta H.264, AAC/MP3, HEVC y AV1 (enhanced RTMP), así como streams de solo audio; las marcas de tiempo en el push empiezan de cero, como esperan las CDN.
- Los errores del push se distinguen y quedan visibles en el estado: conexión rechazada, tiempo de espera de conexión, rechazo de publicación (`NetStream.Publish.BadName` — clave ocupada o incorrecta), servidor colgado (tiempo de espera de escritura de fotogramas).

Reconexión

El pusher no tiene política de reintentos propia — lo vuelve a levantar el stream. Cada cinco segundos el stream coteja los pushes en marcha con la configuración y reemplaza un pusher muerto por uno nuevo; el contador `reconnects` cuenta exactamente esos reemplazos. Por eso el envío a un destino inalcanzable nunca «se rinde» ni necesita reinicio manual: se seguirá levantando mientras el push esté activado.

Lo que **no** cuenta como reconexión:

- **editar la configuración de un push** — el pusher también se reinicia, pero es una decisión del operador, no un fallo del canal;
- **desactivar y activar** (`enabled: false`) — desactivar no es eliminar: la configuración y los contadores acumulados se quedan, solo se detiene el envío.

Desactivación y eliminación

Un push desactivado permanece en la configuración con su nombre: las estadísticas lo muestran como desactivado, sus contadores quedan congelados en los últimos valores, y al activarlo el conteo continúa desde ellos. Eliminar la clave elimina también las estadísticas: los contadores pertenecen al nombre.

MONITORIZACIÓN

Cada push tiene estadísticas propias — destino, estado, causa del fallo, velocidad, bytes, errores, reconexiones. La referencia de campos, las series de Prometheus y las recetas de alertas están en la página Estadísticas.

TRANSPORTE

RTMP sobre TCP; Sapsan abre una conexión saliente a `url`.

© Flussonic 2026

CÓDECS COMPATIBLES

- **H.264**
- **HEVC (Enhanced RTMP)**
- **AV1 (Enhanced RTMP)**
- **AAC**
- **MP3**

El push empaqueta las pistas codificadas en FLV sin transcodificarlas. Para HEVC y AV1, el receptor debe admitir Enhanced RTMP; admitir H.264/AAC no lo garantiza.

© Flussonic 2026

SRT

`media.push.srt`

Sapsan inicia una publicación saliente por SRT. Un stream puede tener varios pushes con nombre.

CONFIGURACIÓN

Aplique la configuración del stream mediante central.

Los pushes de un stream son un mapa de «nombre del push — configuración», no una lista. El nombre es único dentro del stream y sirve de clave: por él se encuentra el push en la configuración, en las estadísticas y en las métricas. Un push tiene exactamente un protocolo.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    pushes:
      delivery:
        srt:
          host: 203.0.113.10
          port: 9000
          passphrase: verysecretpass
```

Protocolo	Parámetros
`srt`	`host`, `port`, `passphrase`, `stream_id`, tiempos de espera

El nombre es la identidad del push. Renombrarlo no traslada ni la configuración ni los contadores acumulados: el push antiguo desaparece y el nuevo empieza de cero.

Un push por UDP, SRT o RIST puede llevar su propia sección `mpegts` — número de programa, PID de las pistas y SDT propios, solo para ese push.

CAPACIDADES Y COMPORTAMIENTO

- Un push SRT con una `passphrase` que no coincide recibe un rechazo en el handshake; un receptor silencioso (sin ACK) se registra como peer timeout.

Reconexión

El pusher no tiene política de reintentos propia — lo vuelve a levantar el stream. Cada cinco segundos el stream coteja los pushes en marcha con la configuración y reemplaza un pusher muerto por uno nuevo; el contador `reconnects` cuenta exactamente esos reemplazos. Por eso el envío a un destino inalcanzable nunca «se rinde» ni necesita reinicio manual: se seguirá levantando mientras el push esté activado.

Lo que **no** cuenta como reconexión:

- **editar la configuración de un push** — el pusher también se reinicia, pero es una decisión del operador, no un fallo del canal;
- **desactivar y activar** (`enabled: false`) — desactivar no es eliminar: la configuración y los contadores acumulados se quedan, solo se detiene el envío.

Desactivación y eliminación

Un push desactivado permanece en la configuración con su nombre: las estadísticas lo muestran como desactivado, sus contadores quedan congelados en los últimos valores, y al activarlo el conteo continúa desde ellos. Eliminar la clave elimina también las estadísticas: los contadores pertenecen al nombre.

MONITORIZACIÓN

Cada push tiene estadísticas propias — destino, estado, causa del fallo, velocidad, bytes, errores, reconexiones. La referencia de campos, las series de Prometheus y las recetas de alertas están en la página Estadísticas.

TRANSPORTE

MPEG-TS sobre SRT/UDP. Sapsan se conecta al servidor receptor; permita UDP saliente y tráfico de retorno SRT.

CÓDECS COMPATIBLES

- **H.264**

© Flussonic 2026

- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio**
- **SMPTE 302M**

Estos códecs permiten reempaquetar pistas ya codificadas en MPEG-TS; el propio push no las transcodifica. AAC se transmite con ADTS. La compatibilidad de un códec con fMP4 o HLS no implica la misma compatibilidad en esta salida.

AV1 se anuncia como flujo privado con el registro `AV01`; el receptor debe admitir este empaquetado concreto. Opus no se incluye en el conjunto de códecs verificado para esta salida: su presencia en la entrada no demuestra una señalización y un empaquetado completos de Opus en MPEG-TS a la salida.

© Flussonic 2026

RIST

`media.push.rist`

Sapsan inicia una publicación saliente por RIST. Un stream puede tener varios pushes con nombre.

CONFIGURACIÓN

Aplique la configuración del stream mediante central.

Los pushes de un stream son un mapa de «nombre del push — configuración», no una lista. El nombre es único dentro del stream y sirve de clave: por él se encuentra el push en la configuración, en las estadísticas y en las métricas. Un push tiene exactamente un protocolo.

```
streams:
  - name: tv1
    inputs:
    - udp: {host: 239.0.0.1, port: 5000}
    pushes:
      delivery:
        rist:
          host: 203.0.113.10
          port: 9000
```

Protocolo	Parámetros
`rist`	`host`, `port`, `cache_ms`, `retransmit_rate_percent`, `ttl`, `multicast_interface`

El nombre es la identidad del push. Renombrarlo no traslada ni la configuración ni los contadores acumulados: el push antiguo desaparece y el nuevo empieza de cero.

Un push por UDP, SRT o RIST puede llevar su propia sección `mpegts` — número de programa, PID de las pistas y SDT propios, solo para ese push.

MONITORIZACIÓN

Cada push tiene estadísticas propias — destino, estado, causa del fallo, velocidad, bytes, errores, reconexiones. La referencia de campos, las series de Prometheus y las recetas de alertas están en la página Estadísticas.

TRANSPORTE

MPEG-TS sobre RIST/UDP con retransmisión de paquetes perdidos.

CÓDECS COMPATIBLES

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio**
- **SMPTE 302M**

Estos códecs permiten reempaquetar pistas ya codificadas en MPEG-TS; el propio push no las transcodifica. AAC se transmite con ADTS. La compatibilidad de un códec con fMP4 o HLS no implica la misma compatibilidad en esta salida.

AV1 se anuncia como flujo privado con el registro `AV01`; el receptor debe admitir este empaquetado concreto. Opus no se incluye en el conjunto de códecs verificado para esta salida: su presencia en la entrada no demuestra una señalización y un empaquetado completos de Opus en MPEG-TS a la salida.

© Flussonic 2026

UDP TS

`media.push.udp_ts`

Sapsan envía MPEG-TS a un receptor por UDP. Use una dirección de grupo para multicast o la dirección del receptor para unicast.

CONFIGURACIÓN

Aplique esta configuración del stream mediante central. `delivery` identifica el push en la configuración y las estadísticas.

```
streams:
- name: tv1
  inputs:
  - udp: {host: 239.0.0.1, port: 5000}
  pushes:
    delivery:
      udp:
        host: 239.1.1.1
        port: 5500
        ttl: 10
        interface: eth0
```

Parámetro	Descripción
`host`	Dirección del grupo o receptor
`port`	Puerto UDP de destino
`encapsulation`	MPEG-TS directo (predeterminado) o en RTP
`ttl`	TTL multicast; predeterminado 10
`interface`	Nombre de la interfaz multicast de salida
`bind_ip`	Dirección local del socket

TRANSPORTE

UDP no recupera paquetes perdidos. Permita el puerto de destino y compruebe las rutas multicast y la interfaz seleccionada. `encapsulation: rtp` añade una cabecera RTP; configure el receptor con el mismo formato.

CÓDECS COMPATIBLES

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio**
- **SMPTE 302M**

Estos códecs permiten reempaquetar pistas ya codificadas en MPEG-TS; el propio push no las transcodifica. AAC se transmite con ADTS. La compatibilidad de un códec con fMP4 o HLS no implica la misma compatibilidad en esta salida.

AV1 se anuncia como flujo privado con el registro `AV01`; el receptor debe admitir este empaquetado concreto. Opus no se incluye en el conjunto de códecs verificado para esta salida: su presencia en la entrada no demuestra una señalización y un empaquetado completos de Opus en MPEG-TS a la salida.

Aquí RTP transporta paquetes MPEG-TS, no flujos RTP independientes de cada códec.

© Flussonic 2026

SIGUIENTES PASOS

- TR 101 290
- Push UDP de reserva
- Estadísticas de pushes

© Flussonic 2026

Push UDP de reserva

`media.push.udp_standby`

El push de reserva escucha su grupo multicast. No transmite mientras detecta otro emisor. Cuando el grupo permanece en silencio durante la ventana de relevo, Sapsan empieza a transmitir; al detectar otro emisor vuelve a detener la entrega. Sus propios paquetes no cuentan como otro emisor.

CONFIGURACIÓN

Aplique esta configuración del stream mediante central. `delivery` identifica el push en la configuración y las estadísticas.

```
streams:
- name: tv1
  inputs:
  - udp: {host: 239.0.0.1, port: 5000}
  pushes:
    delivery:
      udp:
        host: 239.1.1.1
        port: 5500
        interface: eth0
        standby:
          takeover_ms: 1000
```

`standby: {}` usa una ventana de 1000 ms. `takeover_ms` es un entero positivo en milisegundos; cero no es válido. Sin `standby`, el push transmite continuamente. Asigne ventanas distintas a varios emisores de reserva: una ventana menor tiene prioridad cuando esperan a la vez.

TRANSPORTE

Solo multicast UDP; se rechaza unicast. El servidor de reserva debe recibir el mismo grupo y puerto: el relevo depende de detectar un emisor, sin coordinación mediante central.

CÓDECS COMPATIBLES

- **H.264**
- **HEVC**
- **AV1**
- **JPEG 2000**
- **MPEG-2 Video**
- **AAC**
- **AC-3**
- **E-AC-3**
- **MPEG Audio**
- **SMPTE 302M**

Estos códecs permiten reempaquetar pistas ya codificadas en MPEG-TS; el propio push no las transcodifica. AAC se transmite con ADTS. La compatibilidad de un códec con fMP4 o HLS no implica la misma compatibilidad en esta salida.

AV1 se anuncia como flujo privado con el registro `AV01`; el receptor debe admitir este empaquetado concreto. Opus no se incluye en el conjunto de códecs verificado para esta salida: su presencia en la entrada no demuestra una señalización y un empaquetado completos de Opus en MPEG-TS a la salida.

Aquí RTP transporta paquetes MPEG-TS, no flujos RTP independientes de cada códec.

SIGUIENTES PASOS

- UDP TS
- TR 101 290

© Flussonic 2026

Salida SDI

`media.push.sdi`

FINALIDAD

La función `media.push.sdi` envía el vídeo y el audio embebido a una salida física SDI. Se configura por separado de la captura SDI y requiere un puerto de salida.

REQUISITOS DEL STREAM

La salida requiere una pista de vídeo sin comprimir `v210` o `uyvy422`. SDI push empaqueta fotogramas preparados: el transcodificador realiza la decodificación y el cambio de tamaño. Si el stream solo contiene vídeo codificado, añada una pista sin comprimir mediante `!set v210` o `!set uyvy422`. Una entrada SDI con formato adecuado puede alimentar otro puerto SDI sin transcodificación.

El estándar de la pista debe corresponder al modo del puerto de salida. Para una tarjeta nueva, verifique la salida de vídeo y audio con la carga prevista.

CONFIGURACIÓN

Envíe la configuración mediante la API de central. Ejemplo de stream con puertos distintos para captura y salida:

```
streams:
  - name: tv1
    inputs:
      - sdi:
          device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
          pixel_format: v210
    pushes:
      sdi-out:
        sdi:
          device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index1
          queue: 4
          audio:
            - track: 1
              channels: [1, 2]
```

Sustituya las rutas de ejemplo por las de entrada y salida de su tarjeta. `standard` es opcional: si se omite, se deduce de la pista de vídeo. Si se indica y no coincide con la pista, el push genera un error. El estándar de salida `ntsc` está admitido, consulte Señal de 525 líneas. Antes de volver a una versión sin salida NTSC, quite `standard: ntsc` del push: una versión anterior lo rechaza. Un push sin el campo deduce el estándar de la pista y funciona igual.

Indique `pixel_format` en la entrada; el formato de salida procede de la pista de vídeo. `queue` establece la cola del dispositivo de salida entre 2 y 8 fotogramas, con 4 por defecto. En `audio`, `track` es el número de pista de audio del stream, desde 1, y `channels` es el par de canales de audio SDI de destino. Sin asignación explícita, las pistas ocupan los pares en orden, hasta ocho pares.

SEÑAL ENTRELAZADA

La salida toma el barrido de la pista de vídeo. Una pista sin comprimir del transcodificador con entrada entrelazada es progresiva con `deinterlace: auto` (predeterminado): una entrada 1080i50 se emite como 1080p25. Para emitir una señal entrelazada, defina `deinterlace: off`. Una entrada PsF se emite como progresiva: 1080PsF25, como 1080p25.

© Flussonic 2026

SEÑAL DE 525 LÍNEAS

El estándar `ntsc` (525 líneas, 29,97 fps, entrelazado) se emite desde una pista sin comprimir de 720x486 o 720x480. Las filas de la pista se colocan en las líneas SDI según la altura y el orden de campos de la pista, de modo que los campos salen en el orden de captura:

Pista	Fuente	Líneas SDI	Pérdidas
486, campo inferior primero	entrada SDI	21–263, 283–525	ninguna
486, campo superior primero	ST 2110	20–262, 283–525	línea 20; la línea 263 repite la línea vecina del mismo campo
480, campo superior primero	ST 2110, MPEG-2, H.264	23–262, 286–525	5 líneas arriba y 1 abajo en negro
480, campo inferior primero	DV	23–262, 285–524	4 líneas arriba y 2 abajo en negro

Una pista sin orden de campos declarado se emite como campo superior primero. No hace falta redimensionar 486 → 480 para la salida. Un ráster de 720x487 no se emite. El dispositivo de salida debe admitir un modo de 525 líneas; si no, el push pasa al estado `Error`.

PÉRDIDA DE LA FUENTE

Una vez abierta la línea, el push envía fotogramas negros con silencio cuando faltan temporalmente fotogramas de la fuente. Los fotogramas de relleno se contabilizan por separado. La apertura inicial requiere una descripción del stream con una pista de vídeo sin comprimir. Un dispositivo ocupado o una discrepancia de estándar pone el push en `Error`; `last_error` contiene la causa.

FUNCIONES RELACIONADAS

12G-SDI ingest es una ampliación de captura independiente; su disponibilidad no confirma la salida 12G en la tarjeta elegida. ASI transporta MPEG-TS y tiene documentación separada; la salida ASI corresponde a `media.push.asi`.

TRANSPORTE

Una salida física SDI de la tarjeta transmite vídeo y audio integrado por un enlace.

CÓDECS COMPATIBLES

- **v210**
- **uyvy422**
- **PCM (16-bit)**
- **S32P**
- **FLTP**

El vídeo debe llegar sin comprimir en formato v210 o uyvy422. Para el audio embebido se admiten PCM de 16 bits, S32P y FLTP; las pistas comprimidas H.264/HEVC/AAC no se decodifican automáticamente para SDI: prepare antes las pistas sin comprimir necesarias.

© Flussonic 2026

Supervisión del push

`media.push.supervisor`

Sapsan puede enviar un stream de manera activa: a las CDN y plataformas de streaming por RTMP, a otro servidor por SRT o RIST, o a la red por UDP (multicast/unicast). Un stream puede tener varios pushes a la vez.

RECONEXIÓN

El pusher no tiene política de reintentos propia — lo vuelve a levantar el stream. Cada cinco segundos el stream coteja los pushes en marcha con la configuración y reemplaza un pusher muerto por uno nuevo; el contador `reconnects` cuenta exactamente esos reemplazos. Por eso el envío a un destino inalcanzable nunca «se rinde» ni necesita reinicio manual: se seguirá levantando mientras el push esté activado.

Lo que **no** cuenta como reconexión:

- **editar la configuración de un push** — el pusher también se reinicia, pero es una decisión del operador, no un fallo del canal;
- **desactivar y activar** (`enabled: false`) — desactivar no es eliminar: la configuración y los contadores acumulados se quedan, solo se detiene el envío.

DESACTIVACIÓN Y ELIMINACIÓN

Un push desactivado permanece en la configuración con su nombre: las estadísticas lo muestran como desactivado, sus contadores quedan congelados en los últimos valores, y al activarlo el conteo continúa desde ellos. Eliminar la clave elimina también las estadísticas: los contadores pertenecen al nombre.

MONITORIZACIÓN

Cada push tiene estadísticas propias — destino, estado, causa del fallo, velocidad, bytes, errores, reconexiones. La referencia de campos, las series de Prometheus y las recetas de alertas están en la página Estadísticas.

QUÉ VIENE DESPUÉS

- Estadísticas de push
- Monitorice los pushes

ESTADÍSTICAS DE PUSHES

La entrega saliente se cuenta aparte de la captura y por cada push: la clave de la estadística es el par «nombre del stream, nombre del push», el mismo que en la configuración. Un stream puede salir a cinco destinos, y «el stream envía 40 Mbit/s» no responde a si llega a alguno de ellos en concreto.

> **La opción «estadísticas de pushes»**
>
> La vista de pushes es una opción de licencia aparte. Corta solo los canales de salida: la sección `pushes` y los campos planos `push_*` del Admin API, las series `stream_push_*` del scrape del nodo y del servidor Prometheus integrado. La recolección en sí nunca se detiene, y los contadores siempre van a la telemetría de Retroview — el análisis en Retroview está disponible también sin la opción.

La sección `pushes`

`GET /streamer/api-v4/streams/stats/{name}` (y cada elemento de `GET /streamer/api-v4/streams/stats`) trae un objeto indexado por nombre de push:

```
"pushes": {
  "cdn-main": {
    "status": "running",
    "proto": "udp",
    "url": "udp://127.0.0.1:5500",
    "opened_at": "2026-08-13T11:17:53Z",
    "bitrate_kbit": 2113.4,
    "bytes_total": 19053048,
    "datagrams_total": 14478
  },
  "youtube": {
    "status": "error",
    "proto": "rtmp",
    "url": "rtmp://198.51.100.7/live/secret-key",
    "error": "rtmp connect failed: timeout after 5s",
```


© Flussonic 2026

```
    "reconnects_total": 22
  },
```


© Flussonic 2026

```
  "backup-dc": {"status": "disabled", "proto": "srt", "url": "srt://203.0.113.10:9000"}
}
```


© Flussonic 2026

Campo	Qué significa	Cómo usarlo
`status`	`running`, `error` o `disabled`	fuera de `running` durante más de N minutos — alerta; `disabled` es normal, es un push que el operador apagó
`error`	la causa del estado `error`	lo primero que hay que leer al diagnosticar: «connect failed», rechazo de publicación, handshake rechazado
`proto`, `url`	el transporte y el destino tal como los definió el operador	se ve adónde va realmente el stream
`opened_at`	el momento en que arrancó el pusher actual	se renueva en cada reconexión — el canal se cae y vuelve
`bitrate_kbit`	la velocidad de envío medida	claramente por debajo del bitrate del stream — el envío no da abasto
`errors_rate`	errores de envío por segundo	crece con la conexión viva — el receptor o el canal se están degradando
`bytes_total`	bytes enviados, acumulados	conciliar el volumen de salida por dirección
`errors_total`	errores de envío, acumulados	—
`reconnects_total`	resurrecciones de un pusher muerto	crece con `status: running` — el canal se corta y se recupera
`retransmitted_packets_total`	retransmisiones del transporte (SRT, RIST)	crecen — pérdidas en el camino hacia el receptor
`rtt_ms`, `rtt_max_ms`	RTT hasta el destino: el actual y el máximo durante la vida del pusher actual; solo en pushes SRT, una reconexión reinicia el máximo	RTT claramente por encima de la latencia configurada — suba la latencia del push
`error_events`	anillo de eventos recientes: errores de envío (momento, texto) y paradas del pusher (momento, causa); se mantiene solo para pushes con la etiqueta de propietario `user_id_<nombre>` en `labels`	la cronología junto al campo `error` — la primera causa del fallo actual
`closures_by_reason`	totales de paradas del pusher por causa: `error`, `peer_closed`, `disabled`, `reconfigure`, `server_restart`	las paradas del operador se distinguen de un

© Flussonic 2026

Campo	Qué significa	Cómo usarlo
		destino muerto por la causa
`datagrams_total`, `frames_total`	las unidades enviadas: datagramas en los transportes de paquetes, fotogramas en RTMP	—

Los campos en cero se omiten de la respuesta: un push RTMP no tiene datagramas, uno UDP no tiene ni fotogramas ni retransmisiones.

Junto a los contadores del stream, como campos de nivel superior de la misma respuesta, hay tres sumas sobre todos los pushes — `push_bytes_total`, `push_errors_total`, `push_reconnects_total`. Responden «cuánto envió el stream en total» sin recorrer el mapa.

Pushes activos

La lista de pushes en funcionamiento de todo el clúster no es un recorrido por los streams, sino una proyección ya lista del plano de estadísticas:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  'http://central:8080/central/api-v4/pushes/active?limit=100'
```

El mismo camino existe en el streamer: `/streamer/api-v4/pushes/active`. La clave del registro es la terna «stream, push, nodo»: un grupo standby mantiene dos registros por par stream×push, el activo y el en espera, distinguibles por el nodo y el campo `standby_status`. El registro lleva el destino sin secretos, el transporte, el propietario de la etiqueta `user_id_<nombre>` cuando está definido, y valores actuales: estado, momento de arranque del pusher actual, velocidad, errores, reconexiones, retransmisiones, bytes y datagramas, el retraso del pusher respecto al anillo de distribución y, en SRT, el RTT.

```
{
  "pushes": [
    {
      "stream": "tv1",
      "push": "cdn-main",
      "node": "node-7",
      "transport": "srt",
      "endpoint": "srt://203.0.113.10:9000",
      "user_id": "ivan",
      "status": "running",
      "opened_at": "2026-10-21T09:15:02Z",
      "bitrate_kbit": 2113.4,
      "rtt_ms": 34,
      "rtt_max_ms": 61,
      "error_events": [
        {"at": 1792845002, "close_reason": "error"},
        {"at": 1792845062, "error": "srt connect failed: timeout after 5s"}
      ],
      "closures_by_reason": {"error": 22, "reconfigure": 1}
    }
  ]
}
```

Los eventos de `error_events` van de más antiguos a más recientes y están limitados por un anillo: el desbordamiento expulsa a los más antiguos y la crónica completa no se guarda en ninguna parte. El anillo pertenece al push, no al pusher — los eventos se conservan tras una reconexión. El recurso acepta los filtros `stream`, `push`, `user_id`, `node`, `transport` y `limit`; los registros van del más reciente hacia arriba, por el momento de arranque del pusher. Un push apagado y un push sin pusher no entran en la lista; una respuesta vacía significa «no hay pushes funcionando».

Los eventos, los totales de paradas por causa y la profundidad SRT de protocolo (`stream_push_srt_*`) se mantienen solo para los pushes destacados — aquellos cuya configuración lleva la etiqueta de propietario `user_id_<nombre>` en `labels`. Un push sin propietario vive con la familia básica de series y los campos de estado; retirar la etiqueta detiene los puntos nuevos, y los ya escritos permanecen hasta que la ventana del almacén los expulse.

© Flussonic 2026

Series de Prometheus

Cada contador de push es una serie propia con la etiqueta `push`:

Serie	Etiquetas
`stream_push_bytes_total`	`stream`, `push`, `proto`
`stream_push_errors_total`	`stream`, `push`, `proto`
`stream_push_reconnects_total`	`stream`, `push`, `proto`
`stream_push_retransmitted_packets_total`	`stream`, `push`, `proto`
`stream_push_closures_total`	`stream`, `push`, `proto`, `reason`
`stream_push_datagrams_total`, `stream_push_frames_total`	`stream`, `push`, `proto`

En el servidor Prometheus integrado los mismos contadores están disponibles con las etiquetas `name` (el stream) y `push`; en central se les añade `node`. La profundidad de protocolo — `stream_push_srt_*` y `stream_push_rist_*` — llega en el scrape del nodo junto con la opción de contadores extendidos. La serie `stream_push_closures_total` cuenta las paradas del pusher por causa del diccionario cerrado y es más amplia que `reconnects_total`: aquella cuenta solo los arranques tras una caída, esta — todas las paradas; también se mantiene solo para los pushes destacados.

```
# velocidad de envío por destino
rate(stream_push_bytes_total{stream="tv1"}[1m])

# la salida total del stream por todos los pushes
sum by (name) (rate(stream_push_bytes_total{name="tv1"}[1m]))

# canales que se cortan: reconexiones en cinco minutos
increase(stream_push_reconnects_total[5m]) > 0

# pérdidas en el camino hacia el receptor por SRT/RIST
rate(stream_push_retransmitted_packets_total[1m])

# destino inaccesible: una racha de paradas por error de envío
increase(stream_push_closures_total{reason="error"}[5m]) > 0
```

Cómo leer el estado

- **`status: error` con un `error` no vacío** — no se está enviando nada y la causa está nombrada. El push, aun así, sigue levantándose solo, una vez cada cinco segundos.
- **`status: running` pero `bitrate_kbit` en cero** — la conexión está, los datos no: compruebe si el propio stream está corriendo.
- **`status: running` con un `reconnects_total` creciente** — el canal se corta y se recupera; mire `opened_at`, muestra la edad de la conexión actual.
- **un `closures_by_reason` creciente con causa `error`** — el destino no responde y el pusher sigue reiniciándose; el ritmo se ve en los momentos de los eventos de `error_events`. Las paradas con `disabled` y `reconfigure` son del operador, no fallos de red.
- **`rtt_max_ms` de un push SRT claramente por encima de la latencia configurada** — la latencia no da para recuperar las pérdidas: suba la latencia del push.
- **`status: disabled`** — el push está apagado en la configuración. Sus contadores quedan congelados en sus últimos valores; no se pierde nada.

CONFIGURACIÓN

Configure los `pushes` con nombre del stream mediante central. `enabled: false` pausa la entrega y conserva la configuración.

TRANSPORTE

Cada push con nombre selecciona su transporte. La supervisión no añade un protocolo de transporte.

CÓDECS COMPATIBLES

Los códecs dependen del protocolo seleccionado: UDP, RTMP, SRT, RIST, SDI.

© Flussonic 2026

6.2.13 Protección de la reproducción (media.access)

Acceso condicional DVB SimulCrypt

`media.access.cas`

Sapsan admite la **aleatorización de la salida MPEG-TS en directo mediante DVB SimulCrypt**. Los medios se protegen con el algoritmo DVB-CSA2, mientras que un sistema externo de acceso condicional — CAS — genera los mensajes que permiten al receptor obtener las claves.

La configuración se aplica a un stream y a su salida SPTS común: un programa en el flujo de transporte. Esta es la función `media.access.cas`. El cifrado de un múltiplex de varios programas, identificado en la lista de funciones tarifarias como `media.mux.simulcrypt`, es una capacidad independiente; la implementación descrita aquí no implica que esté disponible en Sapsan.

CÓMO FUNCIONA EL SISTEMA

La comunicación con los componentes de CAS utiliza las interfaces ECMG–SCS y EMMG–MUX de ETSI TS 103 197.

Participante	Función
Sapsan	Genera palabras de control CW, gestiona la rotación de claves y aleatoriza MPEG-TS.
ECMG del proveedor de CAS	Recibe CW y criterios de acceso, y genera mensajes ECM para los receptores.
EMMG del proveedor de CAS	Genera mensajes EMM con derechos de acceso; si es necesario, los envía a Sapsan para insertarlos en el flujo.
Receptor compatible con CAS	Recibe ECM y utiliza sus mecanismos de acceso condicional para recuperar las CW y decodificar el programa autorizado.

Una **CW — control word** — es una clave de aleatorización para un criptoperiodo determinado. En esta implementación, Sapsan genera la CW; el ECMG la utiliza para generar un mensaje ECM según los criterios de acceso recibidos.

El funcionamiento se repite de forma cíclica:

1. Sapsan establece una conexión TCP con el ECMG y abre una sesión para el stream.
2. Sapsan genera las CW con antelación y las envía junto con `access_criteria` en mensajes `CW_provision`.
3. El ECMG devuelve ECM para los criptoperiodos correspondientes.
4. Sapsan aleatoriza la carga útil de los PID de medios e inserta ECM en el flujo de transporte. En la PMT aparece un descriptor CA que identifica el sistema de acceso condicional y el PID de ECM.
5. En los límites de los criptoperiodos se utilizan las siguientes CW, alternando los indicadores even/odd. El momento del cambio se determina mediante el PCR del flujo.

ECM y EMM cumplen funciones distintas. ECM permite a un receptor autorizado obtener la CW de un programa. EMM entrega derechos a los mecanismos de acceso condicional del suscriptor. Sapsan recibe EMM ya generados por el proveedor de CAS; las suscripciones y el contenido de estos mensajes se gestionan en el sistema externo.

A QUÉ SALIDAS SE APLICA LA ALEATORIZACIÓN

La sección `mpegts.encryption` se configura **a nivel de stream**. La aleatorización se realiza una sola vez antes de distribuir la salida MPEG-TS común a los consumidores:

- **UDP, incluida la encapsulación en RTP** — push de salida MPEG-TS.
- **SRT** — push MPEG-TS y entrega SRT en directo.
- **RIST** — push MPEG-TS.
- **HTTP MPEG-TS** — entrega continua en directo de la salida TS común.

Varios destinos de este tipo comparten un contexto CAS. Esta sección no permite configurar un ECMG o una clave independiente para cada push: se rechaza `encryption` dentro de la configuración de un canal de salida. Tampoco se admite seleccionar pistas individuales mediante `tracks` en un push MPEG-TS de un stream cifrado.

© Flussonic 2026

Los ajustes de PID y PSI de salida pueden aplicarse sobre el flujo aleatorizado común.

La aleatorización de esta salida no activa la protección de otros métodos de entrega. En concreto, no cifra DASH/fMP4 ni los segmentos HLS, no protege la salida RTMP ni cifra las grabaciones DVR. Para proteger HLS/DASH existe una función independiente de DRM. El acceso a las demás salidas debe configurarse por separado.

CONFIGURACIÓN DE ECMG

La configuración se envía a través de la API de central. El siguiente fragmento YAML define un stream con entrada UDP y salida UDP aleatorizada:

```
streams:
  - name: tv1
    inputs:
      - udp:
          host: 239.0.0.1
          port: 5000
    mpegts:
      pnr: 1
      pmt_pid: 1020
      pids:
        v1: 101
        a1: 102
      service_name: TV1
      encryption:
        type: simulcrypt
        ecmg: "192.0.2.10:2222"
        cas_id: "4ae10006"
        ecm_pid: 100
        ecm_id: 1
        access_criteria: "0303010000000000"
        ecm_private_data: "15"
        nominal_cp_duration: 15
    pushes:
      encrypted-out:
        udp:
          host: 239.1.1.1
          port: 5500
```

Sustituya las direcciones y los valores de CAS por los acordados con el proveedor. Los identificadores `v1` y `a1` del ejemplo deben corresponder a las pistas de su stream. Los derechos del dispositivo del suscriptor para el programa elegido deben estar configurados en CAS.

Parámetro de `mpegts.encryption`	Significado
`type`	`simulcrypt`; único tipo admitido en esta sección, utilizado de forma predeterminada.
`ecmg`	Dirección obligatoria del ECMG externo en formato `host:port`. Sapsan se conecta como cliente TCP.
`cas_id`	`super_CAS_id` hexadecimal obligatorio, de hasta 32 bits. En el ejemplo, `4ae1` es `CA_system_id` y `0006` es `CA_subsystem_id`.
`ecm_pid`	PID obligatorio para ECM en el MPEG-TS de salida.
`ecm_id`	Identificador del stream ECM en el proveedor de CAS, con valor predeterminado `1`. Utilice valores distintos para los streams cifrados de un mismo servidor. No es un PNR ni un PID.
`access_criteria`	Criterios de acceso en hexadecimal, definidos por el proveedor de CAS. Si se omiten, se envía un valor vacío.
`ecm_private_data`	Datos adicionales del proveedor en el descriptor CA de la PMT, en hexadecimal.
`nominal_cp_duration`	Duración del criptoperiodo **en unidades de 100 ms**. El valor predeterminado es `15`, es decir, **1,5 segundos**. No se admite cero.

Envíe los valores hexadecimales como cadenas entre comillas. Los PID asignados explícitamente deben estar en el intervalo 32–8190 y no entrar en conflicto: los PID de medios, PMT, ECM y EMM deben ser distintos.

El cliente ECMG actual utiliza la versión de protocolo **3**. Acuerde con el proveedor los parámetros del criptoperiodo y la compatibilidad con el ECMG concreto: recibir su `channel_status` no implica admitir todas las variantes de temporización anunciadas. En particular, `lead_cw`, `min_cp_duration` y `delay_start` / `delay_stop` no constituyen un mecanismo completamente implementado de configuración automática de tiempos.

© Flussonic 2026

ENTREGA DE EMM

Si los derechos deben entregarse a través de este flujo de transporte, añada los siguientes campos **a la misma sección** `mpegts.encryption`, conservando los ajustes de ECMG:

```
emmg_listen: "0.0.0.0:2233"
emm_pid: 200
emm_private_data: "15"
emm_bitrate: 500
```

Aquí se invierte el sentido de la conexión: Sapsan escucha en un puerto TCP y el inyector EMM del proveedor se conecta y envía los mensajes ya generados. Sapsan los inserta en el PID indicado y lo anuncia mediante un descriptor CA en la CAT.

Parámetro	Significado
`emmg_listen`	Dirección y puerto locales del listener EMMG–MUX. Sin este parámetro, la entrega de EMM está desactivada.
`emm_pid`	PID de EMM. Debe configurarse junto con `emmg_listen`; no se admite uno sin el otro.
`emm_private_data`	Datos del proveedor en el descriptor CA de la CAT. Si se omiten, se utiliza `ecm_private_data`.
`emm_bitrate`	Ancho de banda para insertar EMM, en kbit/s; valor predeterminado `500`. No es la tasa de bits de los medios.

Cada stream con EMM crea su propio listener. Para streams que funcionan simultáneamente, elija direcciones y puertos libres que no entren en conflicto. La conexión con ECMG requiere acceso TCP saliente; la entrega de EMM requiere acceso TCP entrante desde el inyector.

La entrega de EMM es opcional: la aleatorización puede funcionar si los derechos llegan al dispositivo por otro medio. La presencia de ECM sin derechos válidos no garantiza la decodificación en el dispositivo del suscriptor.

INSERCIÓN DE MENSAJES Y COMPORTAMIENTO ANTE FALLOS

ECM y EMM ocupan el lugar de paquetes nulos. El filtro conserva el número y el orden de los paquetes TS, así como la posición del PCR. ECM tiene prioridad sobre EMM. Para entregar los mensajes a tiempo se necesita suficiente capacidad de paquetes nulos; si es insuficiente, se aplaza la inserción de ECM.

Compruebe la generación de CBR y el relleno al dimensionar el flujo de salida. Una sesión correcta con ECMG no confirma por sí sola que ECM ya esté entrando en el transporte con la frecuencia necesaria.

Si no hay una CW válida, Sapsan **anula la carga útil de los paquetes de medios**, conservando las cabeceras y los campos de adaptación que contienen PCR. Un fallo del ECMG no hace que esta salida MPEG-TS pase a transmitir los medios en claro. PSI/SI y los mensajes de acceso condicional permanecen sin aleatorizar para que el receptor pueda analizar el transporte.

El cliente se reconecta automáticamente al ECMG con un retraso creciente. Las claves preparadas pueden cubrir una interrupción breve de la conexión. Si se agotan las claves para los siguientes criptoperiodos, la reproducción se interrumpe hasta que se reanuda su recepción.

Cambiar la dirección del ECMG o los parámetros de la sesión CAS vuelve a crear la sesión y puede provocar una pausa hasta recibir nuevos ECM. Eliminar la sección `encryption` desactiva la aleatorización: es un cambio de configuración distinto de un fallo de red del ECMG.

SUPERVISIÓN

En el streamer que realiza la aleatorización, solicite las estadísticas de un stream:

```
curl -fsS -u "$ADMIN_USER:$ADMIN_PASSWORD" \
  http://streamer:8080/streamer/api-v4/streams/stats/tv1 \
  | jq '{name, cas}'
```

La sección `cas` proporciona tres contadores acumulativos:

Campo	Qué indica
`scrambled_packets_total`	Número de paquetes TS aleatorizados.
`ecm_received_total`	Número de ECM recibidos.
`ecmg_errors_total`	Número de errores de comunicación con ECMG.

© Flussonic 2026

Compare los valores durante un intervalo. Los contadores se conservan al volver a crear el componente CAS, y la sección `cas` puede permanecer después de desactivar el cifrado. Su presencia no confirma por sí sola que la aleatorización esté activada en ese momento. Las estadísticas compactas de central no sustituyen una consulta al streamer que realiza la aleatorización para obtener estos contadores detallados.

Si la exportación de métricas está disponible, `/streamer/api-v4/live-metrics` contiene:

- **`stream_cas_scrambled_packets_total`** — paquetes aleatorizados.
- **`stream_cas_ecm_received_total`** — ECM recibidos.
- **`stream_cas_ecmg_errors_total`** — errores de ECMG.

Los detalles de la rotación de CW y de los retrasos de inserción de ECM se encuentran en los logs del streamer. Allí también se informa de la anulación de los medios y los contadores de EMM. Estos indicadores adicionales no se incluyen en la sección pública `cas`.

COMPROBACIÓN ANTES DE LA PUESTA EN SERVICIO

1. Acuerde los identificadores y criterios de acceso con el proveedor de CAS. Compruebe la compatibilidad con la versión de protocolo ECMG y el criptoperiodo elegido.
2. Aplique la configuración y confirme que `ecm_received_total` aumenta y que no hay errores de ECMG o que su causa se conoce.
3. Capture el MPEG-TS de salida y compruebe el descriptor CA de la PMT y la presencia de ECM en el PID anunciado. Si se entregan EMM, compruebe la CAT y el PID correspondiente.
4. Compruebe la aleatorización de los PID de medios y la reproducción real en un receptor autorizado. Los contadores del servidor no bastan para esta comprobación.
5. Interrumpa la conexión con ECMG de forma controlada. Compruebe la anulación de los medios al agotarse las claves preparadas y la recuperación de la reproducción cuando vuelve el ECMG.
6. Si hay varios destinos de entrega, compruebe cada salida, incluidos los PID y los parámetros PSI personalizados.

La configuración actual define un contexto CAS y un ECMG por stream; no hay una lista de proveedores CAS conectados simultáneamente. Se admite DVB-CSA2; esta sección no permite elegir CSA3 o AES. La compatibilidad con el ECMG y los equipos de suscriptor de un proveedor concreto debe verificarse mediante una prueba de extremo a extremo.

© Flussonic 2026

Backend de autorización

`media.access.auth_backend`

El backend HTTP de autorización es un **mecanismo obsoleto (deprecated), no recomendado para nuevas integraciones**. Para nuevas integraciones, use el JWT del espectador: la aplicación obtiene un token firmado mediante la API de central y los edges lo validan localmente. El backend HTTP se mantiene por compatibilidad con las instalaciones existentes.

Con la autorización HTTP, el streamer consulta al backend al autorizar inicialmente la sesión y al reautorizarla, no en cada segmento HLS/DASH. Los inicios masivos de reproducción y las comprobaciones repetidas generan carga en el backend. Este servicio requiere capacidad para procesar solicitudes, alta disponibilidad y baja latencia: sus respuestas afectan al inicio y la continuidad de la reproducción. JWT elimina esta dependencia mediante la validación local en el streamer.

CÓMO SE TRANSMITE EL TOKEN

El token se acepta de dos formas:

- **Parámetro de URL** — `?token=<token>`, tiene prioridad.
- **Cabecera** — `Authorization: Bearer <token>`, se utiliza si la URL no contiene el parámetro `token`.

Para HLS, basta con abrir la lista maestra con el token: Sapsan añade `?token=` a todas las URL anidadas (variantes, segmentos, partes, init).

BACKENDS DE AUTORIZACIÓN EXTERNOS

La decisión puede delegarse en backends HTTP:

```
auth:
  upstreams:
    main:
      url: http://backend.example.com/auth
      timeout_ms: 3000
```

Sapsan envía un `POST` a la `url` del backend con un cuerpo JSON:

```
{
  "kind": "play",
  "stream_name": "cam1",
  "proto": "hls",
  "user_agent": "Mozilla/5.0 ...",
  "token": "viewer-secret-1",
  "session_id": "..."
}
```

La decisión se toma según el estado HTTP de la respuesta (el cuerpo no se analiza):

- `2xx` — permitir;
- `401` o `403` — denegar (la denegación se almacena en caché; una solicitud idéntica repetida no consulta al backend);
- cualquier otro estado, indisponibilidad o superación de `timeout_ms` (1000 ms por defecto) — el backend se considera caído;
- **varios backends se consultan en paralelo**: basta con una respuesta que permita el acceso; si solo hay denegaciones, se deniega; si todos los backends están indisponibles, se devuelve un error de backend no disponible, en lugar de una denegación silenciosa.

SESIONES

Una sesión HTTP agrupa las solicitudes por medio, tipo de acceso (play/publish), IP y token. User-Agent no forma parte de esta identidad. Varios dispositivos detrás de un mismo NAT con el mismo token pueden compartir una sesión; no es un identificador de dispositivo físico.

Las solicitudes de medios repetidas reutilizan la autorización de la sesión hasta que termina su ventana de autorización. Por defecto, el intervalo de reautorización es de 180 segundos; `auth_duration` en la configuración define otro intervalo y la cabecera de respuesta del backend `X-AuthDuration` lo sustituye. Si una nueva comprobación deniega el acceso, la sesión se cierra. Consulte autorización de la sesión y estadísticas.

SIGUIENTES PASOS

- Configurar la captura de publicaciones

© Flussonic 2026

Tokens de acceso

`media.access.token`

Para las nuevas integraciones, use **JWT de espectador**. Su aplicación comprueba los permisos del espectador y obtiene un token de reproducción firmado mediante la API de administración de central. El reproductor presenta este token al streamer, que lo valida localmente sin consultar a su aplicación ni a un backend HTTP de autorización.

Un mismo JWT funciona en todos los edges del clúster que hayan recibido la clave pública correspondiente. Al pasar a otro edge, el espectador no necesita un token nuevo. Este método de admisión elimina la carga de solicitudes de autorización cuando muchos espectadores empiezan a ver contenido a la vez y evita que la admisión de cada espectador dependa de la disponibilidad y la latencia de un backend HTTP externo.

El backend HTTP de autorización es un mecanismo heredado, obsoleto y no recomendado para nuevas integraciones. Se mantiene por compatibilidad.

CÓMO SE ENVÍA EL TOKEN

El token se puede enviar de dos maneras:

- **Parámetro de URL** — `?token=<token>`, que tiene prioridad.
- **Cabecera** — `Authorization: Bearer <token>`, que se usa si la URL no contiene el parámetro `token`.

Para HLS, abra la playlist maestra con el token: Sapsan lo añade a las URL de las playlists y los medios anidados. Si usa una cabecera, el cliente debe enviarla en las solicitudes protegidas. No entregue la clave de administración al reproductor: solo su aplicación la necesita para emitir JWT.

OBTENER UN JWT MEDIANTE CENTRAL

Primero cree un stream llamado `news` y conecte los streamers a central. Espere a que la configuración y las claves públicas lleguen a los edges. El ejemplo requiere `curl` y `jq`; sustituya los dominios y la clave de administración por sus valores.

```
CENTRAL_URL='https://central.example.com'
EDGE_1='https://edge1.example.com'
EDGE_2='https://edge2.example.com'
ADMIN_KEY='replace-with-management-api-key'

ISSUED=$(curl --fail-with-body --silent --show-error \
  -X POST "$CENTRAL_URL/central/api-v4/play-sessions/grants" \
  -H "Authorization: Bearer $ADMIN_KEY" \
  -H 'Content-Type: application/json' \
  --data '{"streams":["news"],"user_id":"viewer-42","ttl_secs":3600,"max_concurrent":1}')

TOKEN=$(printf '%s' "$ISSUED" | jq -er '.token')
JTI=$(printf '%s' "$ISSUED" | jq -er '.jti')
```

Central crea una **concesión**, un registro del acceso otorgado, y devuelve **201 Created**. Los campos de la concesión y `token` están en el mismo nivel de la respuesta. `jti` identifica la concesión y `expires_at` indica cuándo caduca, en UTC. El JWT solo se devuelve al crear la concesión; consultarla mediante la API no devuelve el token.

Campos de la solicitud:

Campo	Significado
`streams`	Lista no vacía de nombres exactos de medios permitidos. Para el stream `news`, indique `news`; no se admiten comodines.
`user_id`	Identificador del espectador en su sistema. Se incluye en las estadísticas de la sesión.
`ttl_secs`	Validez en segundos desde la emisión. Por defecto, `14400`, es decir, cuatro horas.
`max_concurrent`	Límite de sesiones activas simultáneas de esta concesión. Si se omite, es `1`; `null` significa sin límite; `0` se rechaza.
`bind_ip`	Vinculación opcional a una dirección IP o CIDR. Sin ella, el token permite que cambie la dirección del cliente.
`note`	Nota opcional para el operador.

© Flussonic 2026

REPRODUCCIÓN Y CAMBIO DE EDGE

Obtenga la playlist HLS del primer edge y después del segundo usando **el mismo** `TOKEN`:

```
curl --fail-with-body --silent --show-error \
  "$EDGE_1/streaming/v/news/index.m3u8?token=$TOKEN"

curl --fail-with-body --silent --show-error \
  "$EDGE_2/streaming/v/news/index.m3u8?token=$TOKEN"
```

Estos comandos obtienen playlists; para reproducir el contenido, abra la URL correspondiente en un reproductor HLS. Ambos edges deben poder servir el stream `news`. La validación del JWT por sí sola no asigna el stream ni garantiza la disponibilidad de su fuente.

Para que el balanceador elija el streamer, use `https://central.example.com/playback/v/news/index.m3u8?token=<JWT>`. La disponibilidad de esta URL también depende de central: selecciona un streamer y redirige la solicitud o actúa como proxy de los medios. Una URL directa del edge la valida el propio streamer.

Cambiar de edge puede crear una nueva sesión local. Central asocia las sesiones con su concesión y tiene en cuenta la actividad para que el edge anterior, que ya no recibe solicitudes de medios, no ocupe el límite indefinidamente. Vea sesión del clúster y cambio de edge.

ALCANCE Y VALIDEZ

El mecanismo de autorización descrito aquí no se aplica al listener MoQ.

El JWT se firma con Ed25519 mediante el algoritmo `EdDSA`; `kid` en la cabecera selecciona la clave pública. Su contenido incluye:

- **`jti`** — identificador de la concesión.
- **`user_id`** — identificador del espectador.
- **`exp`** — caducidad en segundos Unix.
- **`streams`** — nombres de medios permitidos.
- **`bind_ip`** — dirección IP o red, si se ha configurado la vinculación.

El directo y el archivo de un stream usan el mismo nombre: `news`. Para VOD, indique `vod/<asset-id>`: la concesión cubre los archivos de ese asset con independencia del root de almacenamiento y del nombre del archivo. El claim estándar `aud` no se usa para este alcance. El límite `max_concurrent` se guarda en el registro de central, fuera del JWT.

El edge comprueba localmente la firma, la validez y el alcance, así como la vinculación a IP si está configurada. Se admite un margen de 30 segundos por desfase de reloj. Un JWT no válido se rechaza sin intentar autorizarlo mediante el backend HTTP. Tras la caducidad, se rechazan las siguientes solicitudes HTTP; ampliar el acceso requiere una concesión nueva. No se garantiza la interrupción inmediata de una respuesta HTTP continua ya abierta; esta limitación se explica en autorización de la sesión. Planifique la renovación del token en su aplicación si el espectador puede seguir viendo contenido después de que caduque.

LÍMITE DE SESIONES Y REVOCACIÓN DEL ACCESO

`max_concurrent` se aplica a la concesión en todo el clúster. Central recibe información sobre las sesiones y termina de forma asíncrona las sesiones activas más antiguas que superen el límite; la nueva sesión permanece. No se trata de un rechazo inmediato de la segunda solicitud en cada edge. El límite cuenta sesiones del servidor, no necesariamente dispositivos físicos distintos: las solicitudes con los mismos medios, IP y concesión pueden agruparse en una sola sesión HTTP.

Cambiar el límite sin volver a emitir el JWT:

```
curl --fail-with-body --silent --show-error \
  -X PATCH "$CENTRAL_URL/central/api-v4/play-sessions/grants/$JTI" \
  -H "Authorization: Bearer $ADMIN_KEY" \
  -H 'Content-Type: application/json' \
  --data '{"max_concurrent":2}'
```

Revocar toda la concesión:

```
curl --fail-with-body --silent --show-error \
  -X DELETE "$CENTRAL_URL/central/api-v4/play-sessions/grants/$JTI" \
  -H "Authorization: Bearer $ADMIN_KEY"
```


© Flussonic 2026

La revocación devuelve **204 No Content**. Central la entrega a los streamers mediante un canal independiente de terminaciones. Al recibirla, el streamer cierra las sesiones de esa concesión y rechaza las nuevas solicitudes. La revocación no es una comprobación global inmediata en cada solicitud de medios: si pierde la conexión con central, el edge no conoce una nueva revocación hasta que se restablece el canal.

ENTREGA DE CLAVES Y FUNCIONAMIENTO SIN CONEXIÓN CON CENTRAL

Central conserva la clave privada de firma y entrega las claves públicas a los streamers en `auth.ticket_keys`, junto con la configuración. Si la política no especifica una lista de claves, central inserta las suyas automáticamente. Un edge con una clave pública puede validar JWT, pero no puede emitir tokens nuevos.

La validación en un edge ya configurado no requiere que central esté disponible. Mientras el JWT sea válido y los medios estén disponibles, la admisión local sigue funcionando con las claves recibidas. Las siguientes operaciones siguen dependiendo de central:

- **Emisión de nuevas concesiones** — requiere la API de central y su almacenamiento.
- **Nuevas claves y cambios de política** — deben llegar a los streamers.
- **Revocaciones y límite global de sesiones** — requieren el canal de terminaciones y la información de sesiones.
- **Balanceo mediante `/playback`** — lo realiza central.

Durante una rotación, la clave pública anterior debe seguir disponible para validar los JWT que aún sean válidos. No la elimine antes de que caduquen esos tokens. La compatibilidad con varias claves no implica que exista una API pública de rotación automática.

TOKENS ESTÁTICOS

Los tokens estáticos se mantienen por compatibilidad y para instalaciones existentes sencillas. No sustituyen la emisión de concesiones individuales a los espectadores:

```
auth:
  play_tokens:
  - viewer-secret-1
  publish_tokens:
  - publisher-secret-1
```

- **`play_tokens`** — tokens de reproducción.
- **`publish_tokens`** — tokens de publicación.

El JWT de espectador está destinado a la reproducción; no lo use como permiso de publicación. En un clúster administrado, central entrega la política `auth`; el YAML anterior muestra la estructura de configuración de un streamer independiente.

`opaque_tokens` vale `true` por defecto para mantener la compatibilidad. Tras la migración, puede establecerlo en `false` para desactivar la ruta general de tokens opacos y backends HTTP, incluidos los tokens estáticos de la política global. Los tokens de reproducción de la configuración de cada stream siguen funcionando. Un `open_playback` explícito sigue abriendo el acceso a la reproducción. Compruebe toda la política de acceso durante la migración.

© Flussonic 2026

Geo

`media.access.geo`

Los ajustes de acceso geográfico de esta página corresponden a **Flussonic Media Server**. Para autorizar la reproducción en Sapsan, consulte backends de autorización y tokens de acceso.

Flussonic Media Server determina el país del espectador por su dirección IP y lo compara con los países permitidos. Los ajustes GeoIP y `allowed_countries` corresponden a Flussonic Media Server.

© Flussonic 2026

DRM

`media.access.drm`

El titular de los derechos de un contenido premium exige que el stream nunca llegue al espectador sin cifrar. El streamer puede cifrar lo que sale de él: el espectador recibe texto cifrado y la dirección en la que su reproductor pide la licencia. Los ajustes de cifrado del stream se envían mediante la API de central.

Se cifra la salida, no la grabación. El archivo del streamer queda sin cifrar, por lo que el rebobinado, las miniaturas, la transcodificación y el reconocimiento de subtítulos siguen funcionando igual que en un stream normal.

Cifra el origin. El cifrado lo hace el streamer que recibió la fuente. Los edges reciben de él el texto cifrado ya listo y lo entregan tal cual: no tienen la clave ni llegan a tenerla — más abajo.

La tarea inversa — tomar una **fuente cifrada** — se resuelve en la entrada del stream, no aquí: véase Fuente MPEG-DASH.

ACTIVE EL CIFRADO

El interruptor **Cifrar la entrega** habilita el bloque. Después elija el **Origen de la clave**:

- **En los ajustes** — usted mismo indica la clave. Sirve cuando la clave proviene de un acuerdo con el titular de los derechos y no de un servidor.
- **Servidor de claves (SPEKE)** — la clave se pide a un servidor externo mediante SPEKE. Así funcionan los servicios comerciales de DRM.
- **EZDRM** — la clave se pide a EZDRM con el usuario y la contraseña o con el Secure Token de su cuenta de EZDRM.

VÉALO FUNCIONAR, CON CLEARKEY

Para ver el cifrado de principio a fin — del fragmento a la imagen en el navegador — no hace falta un DRM comercial: **ClearKey** se reproduce en un navegador corriente y la licencia la emite el propio streamer. Este modo no protege nada, la clave viaja al reproductor en claro, pero todo el recorrido queda comprobado en un minuto y sin una sola suscripción.

1. En la pestaña **Entrega** active **Cifrar la entrega**.
2. **Origen de la clave** — **En los ajustes**. En **Valor de la clave** escriba 32 caracteres hexadecimales, por ejemplo `00112233445566778899aabbccddeeff`. Deje vacío el **Identificador de la clave**.
3. En **Sistemas de protección** deje solo **ClearKey**.
4. Guarde y abra la **Página del reproductor** — el enlace está en el bloque **Cómo obtener el flujo**, más abajo en la misma pestaña.

La imagen arranca como siempre, y ese es justamente el resultado: los fragmentos salieron cifrados, el reproductor pidió la licencia al streamer, recibió la clave y los descifró por su cuenta.

Dos comprobaciones muestran lo que ocurre de verdad.

En la lista de reproducción apareció una clave. Un stream cifrado lleva una línea del sistema de protección antes de la lista de calidades; uno normal no la tiene:

```
curl -s http://<dirección>/streaming/v/<nombre>/index.m3u8 | grep KEY
```

```
#EXT-X-SESSION-KEY:METHOD=SAMPLE-AES,URI="data:text/plain; base64,...",KEYFORMAT="org.w3.clearkey"
```

Un protocolo sin cifrado se niega. El servidor no sabe cifrar MPEG-TS y por eso no entrega ese stream en absoluto:

```
curl -s -o /dev/null -w '%{http_code}\n' http://<dirección>/streaming/mpegts/<nombre>
403
```

La consola ya no muestra una clave guardada — el campo contiene `<redacted>`. El valor de la clave no se entrega ni por la API ni a los registros; para poner otra, escriba una clave nueva encima.

LA CLAVE EN LOS AJUSTES

- **Valor de la clave** — 32 caracteres hexadecimales (AES-128).
- **Identificador de la clave** — puede quedar vacío: entonces se deriva del identificador de contenido y de la sal.

© Flussonic 2026

EL SERVIDOR DE CLAVES

- **Dirección del servidor de claves** — allí se envía un documento CPIX. Después la clave no sale de este streamer.
- **Versión de SPEKE** — la versión del protocolo acordada con el servidor.

EZDRM

EZDRM pide la clave a su manera, así que es un origen de clave propio y no un servidor SPEKE: elija **EZDRM** en **Origen de la clave**.

Autorización fija cómo se presenta el streamer ante EZDRM, de una de dos maneras:

- **Usuario y contraseña** — las credenciales de la cuenta de EZDRM, las mismas que en Flussonic. Es la opción predeterminada: rellene **Usuario de EZDRM** y **Contraseña de EZDRM**.
- **Secure Token** — el token de la sección **My Account** de accounts.ezdrm.com. Se puede revocar allí sin cambiar la contraseña de la cuenta.

Dirección del servidor de claves puede quedar vacía: entonces se usa la dirección predeterminada de EZDRM. Solo hace falta si EZDRM le dio una dirección propia.

La contraseña y el token guardados no se vuelven a mostrar: el campo queda vacío y marcado con puntos. Para cambiar el valor, escriba uno nuevo; si no lo toca, se conserva el guardado.

QUÉ CIFRAR

El campo **Qué cifrar** define el alcance:

- **Todas las muestras de vídeo** — el valor por defecto y lo que suele exigir el titular de los derechos.
- **Solo fotogramas clave** — bastante más barato en CPU: los fotogramas diferenciales no se decodifican sin el fotograma clave de todos modos.

El audio se cifra en ambos modos: es barato, y dejarlo en claro vuelve inútil la protección del contenido.

SISTEMAS DE PROTECCIÓN

Sistemas de protección enumera aquellos a los que se anuncia la protección: **Widevine**, **PlayReady**, **FairPlay**, **ClearKey**. Hay que seleccionar al menos uno: cifrar sin decirle a ningún reproductor con qué descifrar es una entrega que nadie podrá ver.

ClearKey es un modo de demostración y no ofrece protección: la clave se entrega al reproductor tal cual. Existe para que todo el recorrido — del cifrado a la reproducción — se compruebe con un navegador corriente, sin comprar un DRM comercial. Para contenido real elija un sistema cuya licencia se entregue de forma protegida.

DISPOSITIVOS ANTIGUOS

El streamer cifra con el esquema `cbcs`: lo entienden los tres sistemas de protección, así que una sola copia cifrada sirve tanto para HLS como para DASH. Los dispositivos que solo conocen el esquema anterior, `cenc`, no pueden descifrarlo — televisores y decodificadores con PlayReady anterior a 4.0 y dispositivos Android anteriores a 7.1. Revise los dispositivos de su público antes de pasar un canal.

IDENTIFICADOR DE CONTENIDO Y SAL

- **Identificador de contenido** — vacío significa el nombre del stream. Varios streams que deban cifrarse con una misma clave indican aquí un identificador común.
- **Sal del identificador de la clave** — no es un secreto: solo evita que coincidan los identificadores de dos instalaciones.

Ambos campos alimentan el identificador de la clave, por lo que editarlos cambia lo que se anuncia al reproductor.

UN STREAM TIENE UNA SOLA CLAVE

Al stream se le asigna la clave una vez y no cambia mientras el stream funciona: no hay rotación de claves.

- **Toda la escalera de calidades se cifra con una clave** — 1080p, 360p y el audio por igual.
- **El archivo se cifra con la misma clave que el borde en vivo.** Rebobinar una semana atrás no exige ni una segunda clave ni un segundo viaje al servidor de claves.
- **La lista de reproducción lleva una sola etiqueta de clave para toda la ventana** — no aparecen grupos de segmentos bajo claves distintas.

© Flussonic 2026

La clave cambia solo junto con los ajustes: editar **Valor de la clave**, **Identificador de contenido** o **Sal del identificador de la clave** significa una clave nueva, y el stream pasa a cifrarse con ella. Quien esté viendo en ese momento sigue viendo: el reproductor vuelve a pedir la licencia, igual que ante un cambio de códec.

SIN CLAVE NO HAY ENTREGA

El cifrado en sí no se rompe — lo que se rompe es la obtención de la clave, y eso es lo que hay que vigilar.

Mientras no se tenga la clave, el stream cifrado no se entrega en absoluto: la petición de medios termina en error. Un stream así jamás responde con contenido en claro: un servidor de claves inaccesible apaga la entrega, no retira el cifrado.

Lo que puede pasarle al servidor de claves:

- **No responde** — el stream no se entrega. En cuanto el servidor responda, la entrega se reanuda sola: sin reiniciar el stream y sin un hueco en el archivo.
- **No devolvió todos los sistemas pedidos** — la respuesta se rechaza entera. Si se eligieron Widevine y FairPlay y en la respuesta solo viene Widevine, la clave no se acepta: una señalización incompleta significaría que parte del público no puede ver, en silencio.
- **Respondió otra cosa** — no un documento CPIX, o un XML que no se puede analizar: la clave no se acepta y el motivo va al registro del streamer. Los valores de las claves nunca se escriben en los registros.
- **Entregó una clave para otro esquema** — el servidor de claves marcó la clave con un esquema distinto de `cbcs`: la respuesta se rechaza y el esquema se nombra en el registro. Una señalización para otro esquema no descifraría el stream para ningún espectador.

Hay una petición al servidor de claves por stream, no por espectador ni por fragmento: el streamer conserva la clave recibida. Mil espectadores que llegan a la vez producen una sola petición.

QUÉ DEJA DE FUNCIONAR

En un stream cifrado se niegan las superficies que no tienen cifrado: la extracción MPEG-TS y su subárbol HLS, la vista previa MP4 del archivo, el flujo del archivo para RTSP, la entrega WebRTC y SRT. La negativa es deliberada: entregar ese stream en claro sería una forma de sortear la protección.

Siguen funcionando: las vistas previas JPEG y las miniaturas, los subtítulos WebVTT y los pushes del stream.

STREAMERS QUE ENTREGAN

La clave se queda en el streamer que recibió la fuente, el origin. Los streamers vecinos, los edges, reciben el mismo texto cifrado que el espectador y lo siguen entregando, anunciando la protección en listas y manifiestos — pero no tienen la clave, y tampoco aparece en sus ajustes. Un streamer de entrega comprometido no reparte medios en claro.

Solo el streamer que recibió la fuente habla con el servidor de claves, de modo que el número de peticiones depende del número de streams y no del tamaño del clúster.

Si el streamer de entrega tiene discos de caché configurados, el texto cifrado se deposita en ellos tal cual, y una petición repetida del mismo fragmento se sirve localmente. El streamer sigue sin poder descifrar esa caché: entrega exactamente los bytes que llegaron. El texto cifrado no llega al archivo — el archivo alimenta vistas previas, miniaturas y la línea de tiempo, y todo eso mira dentro de la imagen.

MIGRACIÓN DESDE FLUSSONIC

Un stream que tenía el cifrado configurado en Flussonic Media Server se importa desactivado, y el motivo se nombra en el informe — siempre, pase lo que pase con sus ajustes.

Los ajustes de EZDRM viajan: el usuario, la contraseña y el identificador de recurso llegan al origen **EZDRM**, y **Qué cifrar** conserva el valor de Flussonic — solo fotogramas clave, salvo que allí estuviera fijado el cifrado completo. El identificador de clave del stream cambia, eso sí: el streamer lo deriva con su propia regla, y EZDRM entrega al canal una clave nueva. Active el stream después de comprobar que se reproduce.

Los ajustes de los demás proveedores de claves no viajan: Flussonic tiene una quincena de ellos frente a tres orígenes de clave aquí, y no se pueden trasladar literalmente. El cifrado de un stream así se configura de nuevo — más seguro que un canal que tras la importación quedara emitiendo en claro.

© Flussonic 2026

6.2.14 Episodios (media.episodes)

Gestión de episodios

`media.episodes.management`

Un episodio vincula un evento con un stream y un tiempo del vídeo. `media.episodes.management` abarca gestión, almacenamiento y acceso a episodios; es una función por stream de la tarifa existente.

Central guarda el último estado del episodio y sus imágenes en PostgreSQL. El recolector recibe actualizaciones de los streamers; la API de gestión lee el historial guardado sin necesitar que el streamer de origen esté disponible. El `payload` de origen y las anotaciones externas de `meta` se guardan por separado.

LECTURA

`GET /central/api-v4/episodes` devuelve una selección conjunta; `GET /central/api-v4/episodes/{episode_id}` lee un episodio. Los filtros de stream, tipo y tiempo se combinan con el filtro JSON `meta`. Siga `next_cursor`; el límite predeterminado es 100 y el máximo 1000. Offset está prohibido.

El acceso de espectadores usa `/playback/v/{stream_name}/episodes` en central y `/streaming/v/{stream_name}/episodes` en un streamer. Central sirve el historial guardado; el streamer, su ventana actual en memoria. Se aplican los permisos y tokens de reproducción habituales del stream. Leer episodios e imágenes no crea ni renueva una sesión de visualización.

Para ejemplos de creación, actualización de `meta` y búsqueda, consulte el manual de Optic. La retención se describe en Archivo de episodios.

© Flussonic 2026

Episodios de cámaras IP

`media.episodes.ipcam`

`media.episodes.ipcam` agrupa la recepción de episodios de cámaras IP. La tarifa asigna estas fuentes a la función:

- **Email** — eventos de cámara recibidos por correo.
- **Sondeo ONVIF** — notificaciones obtenidas mediante sondeo.
- **ONVIF RTSP** — metadatos de cámara en el stream RTSP.

ONVIF

El procesamiento ONVIF actual acepta notificaciones PullPoint y metadatos de cámara en RTSP. Se convierten en episodios tipados de movimiento y detección. Las fuentes de movimiento de una cámara se siguen de forma independiente; los episodios tienen tiempos de apertura y cierre.

Un streamer expone su ventana actual de episodios mediante `GET /streamer/api-v4/episodes`. El recolector de central guarda las instantáneas e imágenes en el historial conjunto. Una indisponibilidad prolongada de central puede causar huecos fuera de la ventana del streamer.

La configuración de cámaras y eventos se describe en Eventos ONVIF. La mención de email en la tarifa no define un endpoint SMTP ni parámetros de conexión; esta guía no los introduce. Los eventos de un emisor externo usan la API de episodios de central.

© Flussonic 2026

Episodios externos

`media.episodes.external`

`media.episodes.external` abarca la creación y actualización de episodios desde una fuente externa. Un sistema de control de acceso o analizador externo envía el evento directamente a central usando el nombre del stream.

CREACIÓN

`POST /central/api-v4/episodes` guarda un episodio directamente en PostgreSQL. El stream debe existir y tener habilitada la retención de episodios. No es necesario localizar ni esperar al streamer asignado.

La creación inicial devuelve 201. Repetir el mismo `(media, opened_at, details.type)` devuelve 200 con el ID original sin modificar los datos. Los episodios custom usan el literal `custom` en la clave. Central asigna `episode_id` y `updated_at`.

ANOTACIONES Y BÚSQUEDA

`PATCH /central/api-v4/episodes/{episode_id}/meta` actualiza JSON externo arbitrario mediante JSON Merge Patch. La recolección conserva este campo. Si meta no está establecido, se omite del JSON del episodio.

El filtro JSON `meta` se envía en la query string de la lectura del historial y busca la estructura indicada antes del límite de página. Escribir requiere permiso de modificación de gestión; un token de reproducción permite leer. Consulte el ejemplo completo de curl en Crear, editar meta y buscar en Optic.

© Flussonic 2026

Archivo de episodios

`media.episodes.dvr`

`media.episodes.dvr` abarca el almacenamiento de episodios separado del flujo de vídeo principal. Un episodio y su vídeo tienen plazos de vida independientes: el evento puede seguir disponible después de eliminar el vídeo.

ALMACENAMIENTO Y RETENCIÓN

En la implementación actual, el historial persistente de episodios y sus imágenes están en PostgreSQL de central; los segmentos de vídeo están en el archivo de vídeo. La retención de episodios usa la sección `dvr` del stream o de su plantilla: `max_depth + episodes_additional_depth`, en segundos. Son ajustes de archivo existentes.

Si `episodes_additional_depth` no está establecido, equivale a cero. Si hay sección `dvr` pero falta `max_depth`, se usan diez años para la retención de episodios; un cero explícito sigue siendo cero. Sin `dvr`, o con una suma cero, central no retiene episodios.

La retención empieza en `closed_at` para episodios cerrados o en `updated_at` para abiertos. Los cambios en los ajustes efectivos se aplican a todos los episodios restantes. El historial devuelve un registro hasta que la limpieza en segundo plano lo elimina físicamente. Eliminar un stream elimina sus episodios e imágenes.

Escribir `meta` no prolonga la retención. Las anotaciones se eliminan con el episodio. Los ajustes de vídeo están en Grabación de archivo y Profundidad de archivo; los procedimientos de episodios están en el manual de Optic.

© Flussonic 2026

6.3 Gestión

6.3.1 API de gestión y telemetría (control.plane)

API

`control.plane.api`

Sapsan expone el estado del servidor por HTTP. La API es compatible en formato con la Flussonic Streamer API v3, así que las integraciones existentes siguen funcionando.

ACCESO

El acceso a la API está protegido por un usuario y una contraseña en la sección `api_auth`:

```
api_auth:
  login: admin
  password: secret
```

ENDPOINTS

El prefijo base es `/streamer/api/v3`:

Método y ruta	Qué devuelve
`GET /streamer/api/v3/streams`	lista de streams y su estado
`GET /streamer/api/v3/config`	configuración actual del servidor
`GET /streamer/api/v3/dvrs`	lista de almacenamientos DVR
`GET /streamer/api/v3/dvrs/{name}`	estado de un DVR concreto
`GET /streamer/api/v3/rproxy`	estado de la pasarela Peeklio y de los agentes
`GET /streamer/api/v3/monitoring/readiness`	sonda de readiness para orquestadores

API NATIVA V4

La API nativa, aún en evolución, vive bajo el prefijo `/streamer/api-v4`:

Ruta	Qué devuelve
`GET /listeners`	lista de listeners
`GET /streams`, `GET /streams/{name}`	streams y su estado
`GET /streams/stats`, `GET /streams-stats`	estadísticas de streams
`GET /sessions/stats`	estadísticas de sesiones
`GET /live-metrics`, `GET /sessions-metrics`	métricas de live y de sesiones
`GET /dvr`, `GET /dvr/catalog`, `GET /dvr/metrics`	estado del almacenamiento y del catálogo DVR
`POST /dvr/rebuild-index`, `GET` (estado), `DELETE` (detener)	reconstrucción del catálogo desde los discos
`POST /dvr/cleanup`, `GET /dvr/cleanup`	iniciar la limpieza del archivo / estado
`GET /runtime/metrics`	métricas del proceso en formato Prometheus
`GET /auth`	estado de la autorización
`GET /license/status`	estado de la licencia
`GET /rproxy/streampoint-agents`	agentes de la pasarela Peeklio

© Flussonic 2026

Note

TODO: ejemplos de respuestas, gestión de streams por API (cuando existan operaciones de escritura).

QUÉ VIENE DESPUÉS

- Monitoreo

ACCESO AL ADMIN API

El Admin API está protegido por una sección aparte `api_auth` — véase Admin API.

SALUD DEL SERVIDOR

- Sonda de readiness: `GET /streamer/api-v3/monitoring/readiness` — para Kubernetes y balanceadores.
- Estado de los streams: `GET /streamer/api-v3/streams`.

ESTADÍSTICAS DEL STREAM EN EL ADMIN API

La lista de streams

`GET /streamer/api-v4/streams/stats` devuelve un elemento por stream — la página de streams de la UI vive de esta lista, y también es lo correcto para sondearla desde scripts.

Campo	Qué significa	Cómo usarlo
`status`, `status_since`	el estado del stream y el momento en que entró en él	fuera de `running` durante más de N minutos — alerta
`bitrate_kbit`	el bitrate de medios medido de la entrada activa, por los timestamps de los fotogramas	caído visiblemente por debajo del nominal — la fuente se degradó. Esto no es el throughput de red: un archivo puede leerse más rápido que el tiempo real
`source.url`	la entrada que está reproduciendo **realmente**	difiere de la primera del config — el stream vive en una reserva
`inputs[]`	el estado de runtime de cada entrada configurada: `url`, `status`, antigüedad de la comprobación `checked_ago_ms`, bitrate, texto del error	véase más abajo
`dvr.from`, `dvr.to`	la ventana del archivo en este servidor	`to` se retrasa respecto al tiempo actual — la grabación se detuvo; `from` dejó de avanzar — la limpieza no funciona, adelante un desbordamiento de disco
`dvr.recorded_hours`, `dvr.bytes`, `dvr.periods`	los totales del archivo: cuántas horas están grabadas, cuánto espacio en disco ocupa el archivo y de cuántos tramos continuos consta; se actualiza una vez por minuto, precisión de una hora	`recorded_hours` notablemente menor que la anchura de la ventana `to - from` — el archivo tiene huecos; `bytes` — el espacio que el stream ocupa en el servidor
`bytes_in`, `bytes_out`	contadores acumulados de bytes desde el arranque del stream	para velocidades e historia use el servidor Prometheus integrado o Retroview: la instantánea se pone a cero con un reinicio

© Flussonic 2026

Estados de los elementos de `inputs[]`:

- `active` — reproduciendo ahora mismo;
- `ok` — una reserva, viva según la última comprobación;
- `error` — una reserva con un problema registrado (el texto, en `error`);
- `unchecked` — una reserva que aún no se ha comprobado nunca.

Esto responde a la pregunta principal del failover: «¿conmutaremos de verdad cuando caiga la principal?». Las reservas se comprueban periódicamente (`recheck_secondary_inputs_interval`, véanse las fuentes de reserva); alerte sobre `error`, sobre `unchecked` y sobre un `checked_ago_ms` demasiado antiguo — una reserva sin comprobar no puede considerarse operativa.

Un solo stream

`GET /streamer/api-v4/streams/stats/{name}` añade a esos mismos campos secciones de diagnóstico — no están en la lista para que la lista siga siendo barata.

La sección `input` — contadores de la entrada actual:

Campo	Para qué
`bytes`, `frames`, `bitrate_kbit`	si la entrada trae datos y a qué velocidad
`retries`	reconexiones a la fuente: crece — la fuente o la red son inestables
`input_switches`	conmutaciones entre entradas: crece — la entrada se cae y se recupera; investigue la principal
`errors`	el contador agregado de errores de entrada: crece — hay un problema; la causa, en los contadores extendidos o en Retroview
`num_sec_on_primary_input`, `num_sec_on_secondary_input`	segundos vividos en la principal y en la reserva: una proporción notable en la reserva — la entrada principal es sistemáticamente mala
`num_sec_no_data`	segundos sin datos en absoluto
`errors_lost_packets`, `errors_connection_closed`, `errors_http_request_error`, …	desglose de errores por causa — **solo con la opción de contadores extendidos**

La sección `dvr` en la respuesta individual se amplía con:

Campo	Para qué
`write.segments_written`, `write.segments_failed`, `write.segments_skipped`	contadores de escritura de fragmentos: `segments_failed` crece — el archivo está perdiendo datos ahora mismo, compruebe el disco
`write.segments_written_slow / _delayed / _collapsed`, `write.segments_discontinuity`	perfilado de la escritura — **solo con la opción de contadores extendidos**

Note

La instantánea del Admin API se pone a cero cuando el servidor se reinicia. Responde a «qué está pasando ahora», pero no sirve ni para conciliar la factura ni para analizar el incidente de ayer — para eso está Retroview.

© Flussonic 2026

Telemetría

`control.plane.telemetry`

RETROVIEW

Retroview es la monitorización en la nube del fabricante. Una vez por minuto Sapsan envía telemetría — el catálogo completo de contadores, incluido todo el detalle extendido — y Retroview la guarda durante meses. En el servidor no hay que configurar nada: el canal funciona junto con la licencia.

Cuándo ir a Retroview y no al servidor Prometheus integrado:

- **historia y tendencias** — el crecimiento del tráfico y del número de streams a lo largo de los meses, la planificación de capacidad;
- **post-mortems** — qué le pasaba al stream de noche: la telemetría sobrevive a los reinicios y no depende de la memoria del servidor;
- **conciliación de consumo** — los contadores son muestreados en el tiempo por un sistema externo, así que la factura se puede verificar aunque los servidores se hayan reiniciado;
- **análisis de causas sin la opción de contadores extendidos** — la telemetría siempre lleva el detalle causal;
- **alertas listas** — reglas de producto en lugar de reglas caseras.

© Flussonic 2026

6.3.2 Observabilidad (control.observability)

Métricas de Prometheus

`control.observability.metrics`

EL SCRAPE DE PROMETHEUS: SU PROPIA MONITORIZACIÓN

> **La opción «contadores extendidos»**
>
> Los endpoints del scrape están disponibles solo con la opción de licencia de contadores extendidos. Sin ella, use el servidor Prometheus integrado y Retroview.

Para las instalaciones con infraestructura de monitorización propia, Sapsan sirve métricas en el formato de texto de Prometheus:

- `GET /streamer/api-v4/live-metrics` — streams y entradas;
- `GET /streamer/api-v4/sessions-metrics` — sesiones de reproducción;
- `GET /streamer/api-v4/dvr/metrics` — discos y catálogo del archivo;
- `GET /streamer/api-v4/runtime/metrics` — el proceso y el asignador.

Diferencias respecto al servidor Prometheus integrado: la historia se guarda de su lado y solo la limita el retention de su TSDB; está disponible la profundidad completa, incluidas las métricas de protocolo por PID/SRT/RTP; el alerting es su alertmanager. Los contadores son monótonos y sobreviven a la observación de los reinicios — este es también el canal para la conciliación de facturas de su lado.

MÉTRICAS PROMETHEUS

`GET /streamer/api-v4/runtime/metrics` sirve métricas del proceso en formato Prometheus (incluido el asignador jemalloc). Al lado están las métricas de live, de sesiones y de DVR: `/live-metrics`, `/sessions-metrics`, `/dvr/metrics` (véase el Admin API). Los endpoints del scrape requieren la opción de licencia de contadores extendidos; sin ella, los gráficos los sirve el servidor Prometheus integrado (un datasource de Grafana listo). El recorrido completo por los canales de estadística, los campos y las recetas de alertas está en la página Estadísticas.

© Flussonic 2026

RECETAS

Pregunta	Dónde mirar	Señal del problema
¿El stream está vivo?	el `status` en la API; `rate(stream_input_bytes_total[1m])` en el Prometheus integrado	el estado no es `running`; la velocidad de captura cayó a cero
¿La reserva está lista?	`inputs[]` en la lista de estadísticas	`status` es `error` o `unchecked`; `checked_ago_ms` muy por encima del intervalo de comprobación
¿Estamos viviendo en la reserva?	`source.url`; `num_sec_on_secondary_input`	la URL no es la primera del config; los segundos en la reserva siguen creciendo
¿La entrada se está degradando?	`input.errors`, `input.retries`; las causas — contadores extendidos o Retroview	los contadores crecen con el stream vivo
¿El push llega?	`pushes[].status` y `bitrate_kbit`; `rate(stream_push_bytes_total[1m])`	el estado no es `running`; la velocidad en cero con el stream vivo
¿Por qué no funciona el push?	`pushes[].error`	la causa en palabras: conexión rechazada, tiempo de espera, rechazo de publicación
¿Se corta el canal hasta el receptor?	`pushes[].reconnects_total`, `opened_at`	las reconexiones crecen; `opened_at` se renueva cada pocos minutos
¿Hay fuerza bruta contra una entrega SRT con mapa de clientes?	scrape del nodo: `srt_handshake_attempts_total`, `srt_handshake_secret_candidates_total`	muchos candidatos de secreto por intento — fuerza bruta; un par — un espectador normal
¿Se está degradando la entrega a un espectador?	`srt.health` en las sesiones; la serie `stream_play_srt_session_lost_packets_total`	veredicto `degraded` o `bad`; las pérdidas crecen
¿El archivo se está escribiendo?	`dvr.to` en la lista; `dvr.write.segments_failed` en el GET individual	`to` se retrasa respecto al tiempo real; `segments_failed` crece
¿Hay huecos en el archivo?	`dvr.recorded_hours` contra la ventana `to - from`	las horas grabadas son notablemente menos que la anchura de la ventana
¿Los discos dan abasto?	`segments_written_slow/_delayed` (opción); `dvr_disk_free_bytes` en el scrape	la proporción de escrituras lentas crece; el espacio libre mengua más rápido de lo esperado
¿El servidor está sano?	`runtime/metrics`: `process_rss_bytes`; el Prometheus integrado: `stream_memory_bytes`	la memoria crece de forma monótona sin crecimiento de carga
¿Qué pasó de noche?	Retroview	—

© Flussonic 2026

ENTREGA DE HLS MPEG-TS

Los segmentos incompletos, ensamblajes fuera de caché, campos de API y ejemplos de alertas se describen en contadores de HLS MPEG-TS.

© Flussonic 2026

Logs

`control.observability.logs`

Sapsan escribe logs estructurados en JSON en la salida estándar (`stdout`). La variable `RUST_LOG` controla el nivel; el valor predeterminado es `info` , con filtros específicos para módulos internos.

INICIO MANUAL

Indique la ruta a su configuración existente. El segundo ejemplo activa mensajes detallados del módulo `hls` .

```
RUST_LOG=info CONFIG_PATH=/etc/sapsan/sapsan.yaml /usr/sbin/sapsan
RUST_LOG=info,hls=trace CONFIG_PATH=/etc/sapsan/sapsan.yaml /usr/sbin/sapsan
```

SERVICIO SYSTEMD

Cuando el paquete se ejecuta como `sapsan.service` , systemd journal recibe su salida. `journalctl` permite leer este registro. Un archivo syslog independiente depende de la configuración de reenvío de registros del sistema operativo.

```
sudo journalctl -u sapsan.service -n 100 --no-pager
sudo journalctl -u sapsan.service -f
sudo journalctl -u sapsan.service --since "1 hour ago" -o cat
```

Los dos primeros comandos muestran mensajes recientes y siguen los nuevos. Para cambiar el nivel del servicio, ejecute `sudo systemctl edit sapsan` y añada:

```
[Service]
Environment="RUST_LOG=info,hls=debug"
```

Aplique el cambio con `sudo systemctl restart sapsan` . Consulte los filtros en la documentación de journalctl.

DOCKER Y ALMACENAMIENTO CENTRAL

Para un contenedor llamado `sapsan` , utilice:

```
docker logs --tail 100 sapsan
docker logs --follow sapsan
```

Docker recibe stdout del proceso; la consulta depende del controlador de logs. Para enviar logs al almacenamiento central, configure la fuente Docker logs de Vector y la salida Loki. Otra opción para Loki es Grafana Alloy con loki.source.docker. Configure la dirección y la retención en el sistema de recopilación de logs.

© Flussonic 2026

Trazas

`control.observability.traces`

Sapsan exporta trazas mediante **OTLP/HTTP con protobuf**. Configure el receptor en Sapsan; el endpoint HTTP debe incluir la ruta completa `/v1/traces`.

INICIAR JAEGER

Para diagnosticar en una máquina, inicie Jaeger all-in-one:

```
docker run --rm --name jaeger \
  -p 127.0.0.1:16686:16686 \
  -p 127.0.0.1:4318:4318 \
  cr.jaegertracing.io/jaegertracing/jaeger:2.22.0
```

Esta configuración guarda las trazas en RAM y las pierde al detener el proceso. De forma predeterminada, el número de trazas es ilimitado: al acumularse, aumenta el consumo de RAM y puede agotarse la memoria disponible. Configure `jaeger_storage.backends.<nombre>.memory.max_traces` en Jaeger para limitar las trazas almacenadas; limita su número, no la memoria en bytes. El almacenamiento en memoria de Jaeger está pensado para diagnósticos breves. Para un uso prolongado, configure almacenamiento persistente.

CONFIGURAR SAPSAN

`telemetry` es una configuración local del proceso: central no entrega esta sección al streamer. Añádala a su configuración de inicio existente `/etc/sapsan/sapsan.yaml` (o al archivo indicado por `CONFIG_PATH`):

```
telemetry:
  otlp:
    enabled: true
    endpoint: http://127.0.0.1:4318/v1/traces
    sample_ratio: "0.01"
```

`enabled` activa la exportación. `sample_ratio` es una cadena con la proporción de muestreo de trazas nuevas: `"0.01"` equivale al 1% y `"1.0"` a todas; se respeta la decisión del span padre. El valor predeterminado es `"0.01"`. Para un diagnóstico breve puede usar `"1.0"` temporalmente.

Recargue la configuración local del servicio. El exportador aplica los cambios sin reiniciar el proceso:

```
sudo systemctl reload sapsan
```

Tras realizar operaciones cubiertas por las trazas, abra `http://127.0.0.1:16686`, seleccione el servicio **sapsan** y busque trazas. La exportación se realiza por lotes, por lo que aparecen con retraso.

Si Sapsan se ejecuta en Docker, `127.0.0.1` apunta a su propio contenedor. Conecte Sapsan y Jaeger a la misma red Docker y utilice `http://jaeger:4318/v1/traces`, donde `jaeger` es el nombre del contenedor receptor.

COMPATIBILIDAD CON VARIABLES DE ENTORNO

Si falta `telemetry.otlp`, Sapsan admite `OTEL_ENDPOINT` y `OTEL_TRACES_SAMPLER_ARG`:

```
OTEL_ENDPOINT=http://127.0.0.1:4318/v1/traces \
OTEL_TRACES_SAMPLER_ARG=0.01 \
CONFIG_PATH=/etc/sapsan/sapsan.yaml /usr/sbin/sapsan
```

La sección `telemetry.otlp` tiene prioridad sobre estas variables. Sin la sección ni `OTEL_ENDPOINT`, la exportación está desactivada; los logs JSON siguen escribiéndose en stdout.

© Flussonic 2026

Servidor Prometheus integrado

`control.observability.prometheus`

El clúster central/Sapsan incluye un plano de datos de estadísticas integrado. Los streamers Sapsan envían sus estadísticas actuales a central; el titular del mandato `stats` construye el historial de métricas en memoria. Las consultas a este historial sirven los gráficos de la consola sin una TSDB en disco ni un servidor de métricas separado.

Un grupo con varias instancias de central tiene un único plano de estadísticas y un único almacén de historial. Las otras instancias reenvían las consultas al titular del mandato: todas consultan el mismo historial. Al reiniciar el titular o transferir el mandato se crea un almacén nuevo; las muestras anteriores no se transfieren.

Los valores predeterminados son un intervalo de muestreo de 30 segundos y una retención de seis horas. Cada serie tiene además un límite de 1024 muestras y el almacén admite hasta 1 200 000 series. Estos límites acotan el uso de memoria; no garantizan seis horas de historial en cualquier configuración. Para conservar el historial utilice un sistema de métricas externo.

MÉTRICAS DE UN SERVIDOR

Un streamer Sapsan independiente exporta los contadores actuales para que los recopile un Prometheus externo. Esto no proporciona la API de consultas históricas descrita aquí, que pertenece al plano de estadísticas del clúster en central.

CONSULTAS AL CLÚSTER

En Grafana, utilice una fuente de datos Prometheus con la URL base `https://central.example/central/api-v4/prometheus` y autenticación de la API de gestión de central. Añada las rutas estándar de la API Prometheus a esta base:

- **Consultas:** `/api/v1/query` y `/api/v1/query_range`.
- **Catálogo:** `/api/v1/series`, `/api/v1/labels` y `/api/v1/label/{name}/values`.
- **Información de compilación:** `/api/v1/status/buildinfo`.

Esta URL usa el backend seleccionado por `central.stats`: historial integrado, Prometheus externo o Retroview. Los límites de memoria y retención anteriores solo se aplican al historial integrado.

Las series de streams usan `name` para el stream y `server_id` para el streamer. El motor integrado admite un subconjunto de PromQL; las construcciones no compatibles devuelven un error explícito.

```
rate(in_media_bytes_total{name="cam1"}[1m])
sum by (server_id) (rate(in_media_bytes_total[1m]))
```

ESTADO DE STREAMERS Y AGENTES

Central registra cuatro series sobre el estado del clúster:

- **`host_cpu_usage_ratio{server_id}`** — carga de CPU del streamer de 0 a 1, mientras el streamer está conectado.
- **`node_uptime_seconds{server_id}`** — tiempo de funcionamiento del streamer en segundos, mientras está conectado. Un reinicio pone el valor a cero; `resets()` cuenta los reinicios.
- **`node_status{server_id, status}`** — estado de la conexión del streamer con central: 1 para el valor actual de `status`, 0 para los demás.
- **`agent_endpoint_connected{agent_id}`** — 1 si el agente está conectado a central, 0 en caso contrario.

Valores de `status`:

- **`online`** — central recibe los informes del streamer.
- **`connection_lost`** — no hay informes desde hace más de 30 segundos; el streamer todavía se considera operativo.
- **`offline`** — no hay informes desde hace más de 90 segundos. Significa que central no recibe informes, no que el streamer esté apagado.

En los gráficos use una ventana `[65s]`: sin ella, una serie que ya no se registra sigue mostrando su último valor durante cinco minutos. Defina la ventana de `resets()` 30 segundos más larga que el paso del gráfico; el ejemplo usa un paso de 5 minutos.

```
last_over_time(host_cpu_usage_ratio[65s]) * 100
sum by (status) (last_over_time(node_status[65s]))
resets(node_uptime_seconds[330s])
count(last_over_time(agent_endpoint_connected[65s])) - sum(last_over_time(agent_endpoint_connected[65s]))
```


© Flussonic 2026

6.3.3 Auditoría (control.audit)

Log

`control.audit.log`

QUIÉN VE EL DIARIO

La sección **Registro de auditoría** la ven dos roles:

- **security** — el rol de supervisión: el registro de auditoría y su propia cuenta (entrar, cambiar su contraseña, salir). Los ajustes, los streams y el estado observado le están cerrados por completo, incluida la lectura.
- **superadmin** — como todo lo demás en el panel.

Los roles **admin** y **viewer** no tienen acceso: la supervisión de los operadores no pertenece a los propios operadores, y a un espectador el diario le revelaría los usuarios de los administradores y las direcciones desde las que entran. La prohibición la aplica el servidor, no el menú: a un rol sin acceso se le rechaza también por enlace directo.

Los roles se asignan en la sección Cuentas. A quien vigila lo que hacen los operadores se le crea una cuenta propia con el rol **security**, en lugar de darle un superadministrador.

QUÉ ENTRA EN EL DIARIO

Deja anotación **un cambio que llegó a producirse** y el inicio de sesión:

- la edición de ajustes — de un stream, de una máquina, de una plantilla, de una fuente de la guía, del acceso, de la propia instalación de control;
- el trabajo con las máquinas del clúster — registro, revocación, eliminación, apertura y cierre de la ventana de unión;
- el trabajo con las cuentas — creación, edición, eliminación, cambio de contraseña;
- la migración de ajustes desde Flussonic v3 — tanto la construcción del plan como su aplicación;
- la ejecución manual de la asignación;
- el inicio de sesión de un administrador y el intento fallido;
- el trabajo con el registro de clientes — crear, editar y eliminar un cliente, vincular y desvincular una cuenta.

Una operación masiva deja una anotación **por cada objeto afectado**, no una por llamada: migrar trescientos streams se ve como trescientas anotaciones, y la búsqueda por nombre encuentra cualquiera de ellas.

Aparte queda la revelación de un token de unión vigente: es una lectura, pero revelar un secreto equivale a entregarlo, así que la anotación se mantiene. El valor del token nunca entra en ella.

EL LISTADO Y SUS COLUMNAS

Las anotaciones van de las nuevas a las antiguas. El botón **Cargar más** trae la siguiente página: el listado no termina en las primeras cincuenta anotaciones.

La tabla tiene cinco columnas:

- **Hora** — el momento de la acción.
- **Actor** — quién actuó. O bien el usuario de un administrador, o bien uno de los tipos de servicio: **clave estática** (entrada con las credenciales de emergencia de `/etc/mcaster/secrets.env`, comunes a todos y por eso sin nombre detrás) y **nodo** (el registro de una máquina del clúster, donde el actor es su identificador).
- **Acción** — qué se hizo, con un nombre de máquina de la forma `<recurso>.<verbo>`: `stream_config.put`, `node_config.patch`, `admin.create`, `auth.login`. Los nombres son estables, lo que resulta cómodo para buscar y filtrar.
- **Recurso** — sobre qué: tipo y nombre, por ejemplo `stream/program-a`.
- **Resultado** — **éxito** en una acción consumada y **fallo** en un inicio de sesión fallido.

La flecha a la izquierda de la fila despliega los **detalles** de la anotación: la dirección desde la que llegó la petición, el rol del administrador, la versión del documento después de la edición, la lista de campos modificados y —si la petición llegó con una cabecera de trazado— su identificador, con el que la anotación se coteja con los logs.

© Flussonic 2026

BÚSQUEDA EN EL DIARIO

Sobre el listado hay cinco campos que funcionan en conjunto:

- **Tipo de recurso** y **Nombre del recurso** — «todo lo que se hizo con este stream».
- **Actor** — «todo lo que hizo esta persona».
- **Desde** y **Hasta** — una ventana de tiempo en formato RFC 3339, por ejemplo `2026-08-14T14:00:00Z`.

Aplicar vuelve a leer el listado desde el principio. Un campo vacío significa «no filtrar».

Un incidente suele analizarse en dos consultas: primero por recurso — qué le pasó a este stream — y después por el actor de la anotación encontrada — qué más hizo esa misma persona ese día.

© Flussonic 2026

Retention

`control.audit.retention`

CONSERVACIÓN Y PROTECCIÓN

El diario vive en la misma base de datos que el resto del estado de la máquina y sobrevive al reinicio del proceso. No crea archivos en disco.

Las anotaciones solo se añaden: editar o borrar una anotación concreta no existe ni en el panel ni en la API. La retención — siete días por defecto — la fija el ajuste de proceso `audit_log_ttl_secs` en `/etc/mcaster/`, y desde el panel no se puede cambiar. Es deliberado: quien quiera esconder su acción no debe poder acortar la retención con el mismo acceso con el que la cometió. El valor cero significa «conservarlo todo», no «borrarlo todo».

Frente a quien tiene acceso a la propia base de datos el diario no ofrece protección ni la promete.

© Flussonic 2026

Export

`control.audit.export`

El equipo de seguridad del cliente reúne los eventos de todos sus sistemas en un único SIEM y los analiza allí, no en el panel de cada sistema por separado. Mcaster entrega el registro de auditoría en formato **CEF** (Common Event Format): un formato de texto, una línea por evento, con los campos separados por delimitadores. ArcSight, KUMA, MaxPatrol SIEM y otros receptores corporativos lo aceptan sin ninguna adaptación.

La exportación entrega la ventana elegida del diario entera, en un solo archivo, de las anotaciones más antiguas a las más recientes. Se puede obtener de dos maneras:

- con el botón **Exportar a CEF** de la sección **Registro de auditoría**: el navegador guarda el archivo;
- con una petición a la API: así recoge el diario el recolector del SIEM según un calendario.

La exportación es la misma lectura del diario que el listado del panel: los mismos permisos, la misma opción de licencia, las mismas anotaciones. Para ella Mcaster no crea archivos en la máquina ni abre conexiones salientes hacia un receptor.

EXPORTAR DESDE EL PANEL

El botón **Exportar a CEF** está sobre el listado, junto a **Aplicar**. Exporta exactamente lo que muestra el listado:

- **Tipo de recurso**, **Nombre del recurso** y **Actor** acotan la exportación igual que acotan el listado;
- **Desde** y **Hasta** fijan la ventana de tiempo;
- los campos vacíos significan «no filtrar»: se exporta todo el diario dentro de su retención.

La exportación toma los filtros **aplicados**. Si después de pulsar **Aplicar** se corrigió un campo pero el listado no se volvió a leer, la exportación usa el valor anterior: el archivo siempre coincide con lo que se ve en pantalla.

El archivo se llama `audit-log.cef`. Contiene todas las anotaciones de la ventana, no solo las páginas que el listado ya ha cargado.

El botón lo ven los mismos roles que el propio diario: **security** y **superadmin**. Cuando el diario no está incluido en la licencia, el botón está desactivado.

El archivo se arma en la memoria de la pestaña del navegador. Para una ventana de varios días es suficiente; el diario de un periodo de retención largo conviene recogerlo por la API.

EXPORTAR POR LA API

```
GET /central/api-v4/audit-log/export?format=cef&from=2026-09-27T00:00:00Z&to=2026-09-27T12:00:00Z
```

La autorización es la misma autorización de operador que en el resto de la API de gestión, y los roles también: **security** y **superadmin**.

Parámetros:

- `format` — obligatorio; por ahora el único valor es `cef`;
- `from`, `to` — los límites de la ventana en formato RFC 3339, ambos incluidos en la ventana;
- `resource_type`, `resource_name`, `actor`, `actor_type` — los mismos filtros que en el panel; `actor_type` admite `admin`, `static` y `node`.

La respuesta es texto (`text/plain`), como adjunto con el nombre `audit-log.cef`. Llega en flujo: las primeras líneas llegan de inmediato y el servidor guarda en memoria una sola página del diario, así que el tamaño de la ventana no está limitado por nada.

Rechazos:

- **400** — un límite de la ventana no está en formato RFC 3339, o el formato falta o es desconocido. Una petición no válida nunca se convierte en una exportación de todo el diario;
- **403** — el rol no tiene acceso al diario;
- **404** — el registro de auditoría no está incluido en la licencia.

Si la exportación se corta a mitad de la transferencia —por ejemplo, falla la base de datos en el lado de la máquina—, la respuesta termina como una transferencia incompleta: curl informa de una conexión cortada y el navegador marca la descarga como fallida. Un archivo que llegó entero contiene toda la ventana.

© Flussonic 2026

Sin `to`, la ventana queda abierta hacia delante: las anotaciones hechas mientras la exportación está en curso también entran. Si hace falta un límite fijo, indique `to`.

LA LÍNEA CEF

Cada evento es una línea:

```
CEF:0|Flussonic|mcaster|26.09|stream_config.patch|stream_config.patch|3|rt=1790590530250 externalId=0f6b8a1e-6a3c-5d7e-9b1a-2c3d4e5f6a7b
dvchost=11111111-2222-3333-4444-555555555555 suid=admin suser=ivanov outcome=success src=10.20.1.11 cs1=stream cs1Label=resourceType cs2=conference-a
cs2Label=resourceName cs3=9c1f07e2 cs3Label=sessionId cs4=4bf92f3577b34da6a3ce929d0e0e4736 cs4Label=traceId cs5={"fields":
["labels"],"role":"admin","src_ip":"10.20.1.11","version":7} cs5Label=details cn1=3672 cn1Label=auditId cn2=1 cn2Label=schemaVersion
```

La cabecera, hasta la séptima barra vertical:

- **Device Vendor** — `Flussonic`;
- **Device Product** — `mcaster`: por él el SIEM elige sus reglas de análisis;
- **Device Version** — la versión de Mcaster instalada;
- **Device Event Class ID** y **Name** — la acción, el mismo nombre de máquina que en la columna **Acción** del listado: `stream_config.patch`, `admin.create`, `auth.login_failed`;
- **Severity** — la gravedad del evento, descrita más abajo.

Después vienen los campos, en forma de `clave=valor`:

Campo CEF	Qué contiene
`rt`	el momento del evento, en milisegundos desde la época
`externalId`	el identificador global del evento: por él el SIEM descarta los duplicados
`dvchost`	el identificador de la instalación de Mcaster: los eventos de dos instalaciones en un mismo SIEM no se mezclan
`suid`	el tipo de actor: `admin`, `static` (clave estática) o `node` (registro de una máquina del clúster)
`suser`	el login del administrador; la clave estática no lo tiene
`outcome`	`success` para una acción que llegó a producirse, `failure` para un inicio de sesión fallido
`src`	la dirección desde la que llegó la petición
`cs1`, `cs2`	tipo y nombre del recurso (`cs1Label=resourceType`, `cs2Label=resourceName`)
`cs3`	el identificador de la sesión del administrador (`cs3Label=sessionId`): une un inicio de sesión con todo lo hecho en esa sesión
`cs4`	el identificador de trazado de la petición (`cs4Label=traceId`): con él el evento se cruza con los logs
`cs5`	los detalles de la anotación en JSON (`cs5Label=details`): rol, versión del documento, lista de campos editados
`cn1`	el número de la anotación en el diario de la instalación (`cn1Label=auditId`)
`cn2`	la versión del esquema del evento (`cn2Label=schemaVersion`)

Los campos que un evento no tiene no aparecen en la línea en absoluto: una clave vacía el SIEM la leería como un valor vacío. Una acción con la clave estática, por ejemplo, no tiene ni `suser` ni `cs3`.

Los caracteres especiales se escapan como exige la especificación CEF: en la cabecera, la barra invertida y la barra vertical; en los campos, la barra invertida y el signo igual; los saltos de línea se convierten en `\n`. Por eso un evento ocupa siempre exactamente una línea, contengan lo que contengan sus valores.

© Flussonic 2026

GRAVEDAD DEL EVENTO

La gravedad no se guarda en el diario: se deduce del evento al exportarlo, con una misma tabla:

- **7** — una acción fallida; por ahora, un intento fallido de inicio de sesión (`auth.login_failed`);
- **5** — una acción que el equipo de seguridad espera por encima del flujo general:
  - el borrado de cualquier cosa, la revocación de una máquina, la reemisión de los tickets internos del clúster, la revelación del token de unión;
  - cualquier acción sobre las cuentas de administradores;
  - la edición de los ajustes de acceso y de los ajustes de la propia instalación de control;
  - la apertura y el cierre de la ventana de unión;
  - cualquier acción con la clave estática: un inicio de sesión con las credenciales de emergencia merece atención por sí mismo;
- **3** — todo lo demás: edición de streams, plantillas, fuentes de la guía, el inicio de sesión de un administrador.

Las reglas se basan en la forma de la acción y no en una lista hecha de antemano, así que las acciones que aparezcan en versiones futuras reciben su gravedad por las mismas reglas.

RECOGIDA PERIÓDICA EN UN SIEM

El recolector del SIEM recoge el diario por ventanas según un calendario: cada petición toma como `from` el `to` de la anterior. Los dos límites pertenecen a la ventana, así que una anotación que cae justo en la junta llega dos veces, y el SIEM descarta el duplicado por `externalId` . Del mismo modo se repite sin pérdidas una ventana tras un corte.

Una ventana no debe ir más allá de la retención del diario: siete días por defecto (el ajuste `audit_log_ttl_secs` , véase Conservación y protección). Un recolector que se ha retrasado más tiempo solo encontrará lo que todavía no se ha borrado.

A un recolector que lee JSON también le sirve el listado del diario con cursor, descrito en la sección Lectura del registro en JSON. Los campos del evento son los mismos; solo cambia la forma.

LECTURA DEL REGISTRO EN JSON

Un recolector externo lee `GET /central/api-v4/audit-log` con la misma autorización de operador que el resto de la API de gestión y sigue el diario por cursor: la respuesta trae el campo `next` , que se pasa como parámetro `cursor` en la petición siguiente. El orden es estable — de las nuevas a las antiguas — y cada evento tiene un identificador global derivado de la identidad de la instalación y del número de anotación. Por eso una página releída tras un corte devuelve los mismos identificadores y el receptor descarta los duplicados por sí solo; los eventos de dos instalaciones distintas nunca coinciden.

Los filtros son los del panel: `resource_type` , `resource_name` , `actor` , `from` , `to` , `limit` (hasta 1000 anotaciones por página).

© Flussonic 2026

6.3.4 Acceso (control.access)

Roles

`control.access.roles`

ROLES

Un administrador tiene exactamente un rol, elegido entre cuatro:

- **superadmin** — todo, incluida esta sección y la sustitución del documento de licencia.
- **admin** — todo el trabajo cotidiano: streams, plantillas, máquinas del clúster, el archivo, la protección de reproducción. Exactamente dos cosas están cerradas: la sección «Cuentas» y la sustitución de la licencia.
- **viewer** — solo observación: el panel de control y las sesiones. La configuración está cerrada por completo, incluida la lectura.
- **security** — supervisión: el registro de auditoría y su propia cuenta. Los ajustes y el estado observado están cerrados por completo; el diario, al revés, no se abre ni a admin ni a viewer.

El viewer merece una nota, porque prohibir la **lectura** de la configuración parece excesivo. No lo es: los documentos de configuración llevan secretos por construcción — contraseñas de fuentes ajenas, claves, credenciales de máquinas. Un documento no se puede entregar con los secretos recortados por el camino, así que a un viewer no se le entrega en absoluto.

El servidor comprueba el rol en cada petición, no solo al iniciar sesión.

AÑADIR UN ADMINISTRADOR

El botón **Añadir cuenta** abre el formulario:

Introduzca aquí una contraseña temporal y entréguela a la persona como usted entregue las contraseñas. Marque la casilla **Exigir cambio de contraseña en el primer inicio de sesión**: una contraseña que usted inventó por otra persona también la conoce usted, lo cual significa que ya no es un secreto solo de ella.

La cuenta nueva aparece en la lista en el estado **Contraseña temporal**:

En su primer inicio de sesión, el dueño de esa cuenta cae no en el panel, sino en el formulario de cambio de contraseña:

No se abre nada más hasta que se cambie la contraseña — es el servidor el que rechaza, no una ventana con aspecto severo: conocer las direcciones no sirve para saltarse el formulario. Una vez cambiada, la persona queda en el panel de inmediato, sin un segundo inicio de sesión.

EDITAR, RESTABLECER CONTRASEÑA Y DESACTIVAR EL ACCESO

El botón **Editar** de una fila abre todo lo que se puede cambiar en una cuenta existente:

- **Rol** — el nuevo rol surte efecto de inmediato: las sesiones de la cuenta se cierran y la persona inicia sesión de nuevo con los nuevos derechos.
- **Nueva contraseña** — así se recupera a quien olvidó la suya. El panel no le mostrará a nadie su propia contraseña, por lo que una contraseña olvidada no se recupera, sino que se sustituye. La contraseña emitida aquí también es temporal: su dueño la cambia en el siguiente inicio de sesión. Un campo vacío significa «no tocar la contraseña».
- **Nota** — para qué sirve esta cuenta y de quién es. Empieza a valer la pena cuando ya hay una docena.
- **Deshabilitado** — el acceso se cierra de inmediato, junto con cualquier sesión ya abierta, mientras que la cuenta en sí permanece.

Desactivar es a lo que se recurre la mayor parte del tiempo: alguien se fue de vacaciones, cambió de puesto dentro de la empresa, perdió el portátil. El botón **Eliminar** borra la cuenta para siempre; elimine cuando la cuenta se creó por error.

QUÉ VE UN VIEWER

Así se ve el mismo panel bajo el rol **viewer**: del menú quedan solo el panel de control y las sesiones, y las secciones de configuración no se muestran.

Es un rol cómodo para el turno de guardia y para cualquiera que necesite ver qué pasa sin cambiar nada.

© Flussonic 2026

6.4 Clúster

6.4.1 Control (cluster.control)

Unión de streamers

`cluster.control.nodes`

Central almacena la configuración del clúster y determina qué streams ejecuta cada streamer. Para añadir una máquina, el operador abre una ventana de unión y le proporciona la dirección de central y el token de unión. Tras registrarse, el streamer obtiene una identidad persistente, sincroniza la configuración e informa de su estado.

No hace falta copiar la configuración de todos los streams al servidor nuevo. Central entrega las asignaciones; el operador administra la máquina mediante la configuración del streamer. Consulte la instalación para desplegar Sapsan.

REQUISITOS

Se necesitan:

- **Dirección de central:** dirección HTTP o HTTPS del grupo de gestión accesible desde el streamer.
- **Token de unión:** secreto de una ventana activa obtenido por el operador en central.
- **Directorio de estado persistente:** directorio escribible que se conserve al reiniciar el proceso o contenedor.
- **Instalación compatible:** la distribución y la licencia deben permitir participar en el clúster seleccionado.

El streamer inicia la conexión de sincronización. Las solicitudes inversas de central y la lectura de medios por otros streamers requieren direcciones correctas de la máquina. Consulte conectividad y entrega.

VENTANA DE UNIÓN

La acción `permit_join` abre una ventana de **una hora** y emite un token de unión. Durante ese plazo pueden registrarse los streamers que tengan el token. Es un permiso temporal de registro, no la contraseña permanente del streamer. No hay aprobación manual individual: el token activo permite el registro.

Abra la ventana mediante la API de gestión de central con una clave administrativa:

```
curl -fsS -X POST \
  -H "Authorization: Bearer $ADMIN_KEY" \
  https://central.example/central/api-v4/nodes/permit-join
```

La solicitud no necesita cuerpo. La respuesta contiene el token en `token`, el estado en `open` y los segundos restantes en `remaining_secs`. Guarde el token para las máquinas nuevas. Un `GET` normal a la misma dirección no revela el secreto; para recuperarlo mientras la ventana siga activa, use `GET /central/api-v4/nodes/permit-join/reveal` con autenticación administrativa.

Repetir `permit_join` con la ventana abierta la prolonga conservando **el mismo token**. Abrir una ventana después de su cierre o caducidad genera un token nuevo. La prolongación conserva el token ya entregado a las máquinas en instalación.

Un token permite unir varios streamers. Cierre la ventana al terminar. Su cierre o caducidad no revoca registros existentes: los streamers unidos siguen sincronizándose con sus propias credenciales.

```
curl -fsS -X DELETE \
  -H "Authorization: Bearer $ADMIN_KEY" \
  https://central.example/central/api-v4/nodes/permit-join
```

El streamer reintenta un registro inicial denegado. Compruebe que la ventana está abierta y que la máquina usa su token actual. Si la ventana anterior caducó, entregue el token nuevo a los streamers aún no registrados.

ACTIVAR EL MODO GESTIONADO

Configure localmente la conexión en `managed_by`:

```
managed_by:
  url: https://central.example
  name: streamer-01
```


© Flussonic 2026

```
  join_token: "JOIN_TOKEN_FROM_CENTRAL"
  state_dir: /var/lib/sapsan/central
```

Campo	Función
`url`	Dirección de central o lista de direcciones de un único grupo de gestión.
`name`	Etiqueta del operador; por defecto, el hostname de la máquina.
`join_token`	Token de registro inicial; no se vuelve a usar si existe una identidad guardada.
`state_dir`	Directorio de identidad y caché de configuración. Por defecto se usa el directorio de estado escribible de la instalación que contiene `server.id`.

La dirección del grupo puede ser una lista:

```
managed_by:
  url:
    - https://central-1.example
    - https://central-2.example
  join_token: "JOIN_TOKEN_FROM_CENTRAL"
```

La lista establece la preferencia de direcciones. Ante un fallo de red, el streamer prueba otras del grupo. También guarda las direcciones recibidas de central para reutilizarlas tras reiniciar. No mezcle clústeres independientes: son entradas alternativas al mismo grupo.

Sin esquema se añade `http://`. Para TLS, indique `https://` y configure el certificado del servidor.

También puede usar variables de entorno:

Variable	Parámetro
`CENTRAL_URL`	`managed_by.url`; varias direcciones se separan con comas.
`NODE_NAME`	`managed_by.name`.
`JOIN_TOKEN`	`managed_by.join_token`.
`STATE_DIR`	`managed_by.state_dir`.

El entorno prevalece sobre los valores de configuración correspondientes. Elimine las variables antiguas al trasladar la instalación a otro clúster.

`managed_by` activa el modo gestionado. La sección `central` configura el propio componente de gestión y no sustituye a `managed_by`.

MIGRACIÓN DESDE CONFIG_EXTERNAL

`config_external` se ha eliminado. Sapsan rechaza esta sección al iniciar y solicita la migración a `managed_by`.

Antes de cambiar la configuración, traslade a central los streams que antes entregaba el backend heredado. En el modo gestionado, central es propietario de la lista de streams; dejar `streams` locales en YAML provoca un error de validación.

Sustituya la sección heredada por `managed_by` como se muestra arriba. El token de unión solo se necesita para el registro inicial. Los campos antiguos `bearer`, `interval_secs`, `timeout_ms`, `client_host` y `fetch_runtime_config` no tienen equivalentes y deben eliminarse.

La caché heredada no se transfiere. Conserve `state_dir`: tras el primer registro contendrá la nueva identidad y la caché de configuración.

IDENTIDAD PERSISTENTE

El registro devuelve `node_id` y el secreto `node_key`. El streamer los guarda en `identity.json`, dentro de `state_dir`, para sincronizaciones posteriores. `node_id` es el identificador de la API; ni `name` ni el hostname constituyen esa identidad.

Al reiniciar normalmente, el streamer carga su identidad sin registrarse de nuevo. No necesita otra ventana aunque el token anterior haya caducado. Cambiar la etiqueta tampoco crea un registro nuevo.

Conserve el directorio al actualizar o recrear el contenedor. Perder `identity.json` supone perder las credenciales; un archivo corrupto produce un error en lugar de crear otro registro silenciosamente. Copiarlo a otra máquina transfiere la misma identidad, no une un streamer nuevo.

© Flussonic 2026

El archivo contiene un secreto y se crea con permisos `0600`. Limite el acceso al proceso y a los administradores. No incluya identidades o tokens en informes públicos de diagnóstico.

AUTENTICACIÓN EN AMBOS SENTIDOS

La gestión y la lectura de medios usan credenciales distintas:

Dirección	Mecanismo de acceso
Streamer → central	Tras unirse, presenta `node_key` como secreto Bearer. Central identifica el registro por esa clave.
Central → API de gestión del streamer	Central crea una cuenta `api_auth` independiente para HTTP Basic y la entrega en la configuración gestionada, por ejemplo para consultar sesiones y realizar acciones de gestión.
Streamer → otro streamer	La lectura de medios y archivo entre servidores usa un billete JWT firmado independiente. `node_key` no se entrega a los streamers vecinos.

Cada sentido tiene su mecanismo de autorización. Esto **no configura automáticamente TLS mutuo**. Con HTTPS, el cliente verifica el certificado del servidor mediante TLS normal; la unión no fija una clave especial de central ni emite un certificado de cliente.

Compruebe la dirección y use HTTPS o una red protegida para transmitir secretos de gestión. Poseer un token de unión no autentica por sí solo al servidor central de destino.

Los billetes entre servidores se firman con **Ed25519** y se verifican localmente. Su vigencia actual es de **24 horas**; central los renueva antes de caducar. Un billete guardado y vigente permite leer de otros streamers durante una interrupción temporal, pero no da acceso indefinido sin renovación.

PROPIEDAD DE LA CONFIGURACIÓN

En modo gestionado, **central determina la lista de streams**: el campo local `streams` no se combina con las asignaciones ni actúa como reserva al perder la conexión. Cree los streams y cambie su ubicación mediante central. Consulte la distribución de streams.

La configuración del servidor tiene varias capas. Central entrega los parámetros del streamer, pero los archivos include locales y el entorno pueden sobrescribir valores individuales. El valor guardado en central puede diferir del efectivo en la máquina. La interfaz señala estas sobrescrituras; consulte el orden de aplicación.

Actualice la configuración y supervise su aplicación desde central. En la máquina permanecen la dirección del grupo y los parámetros de inicio necesarios para conectarse.

SI CENTRAL NO ESTÁ DISPONIBLE

El streamer guarda en disco la configuración y su slice de streams. Los streams ya aplicados siguen funcionando durante pérdidas temporales de la conexión de gestión; la lista local no sustituye su configuración.

Tras reiniciar, puede restaurar el funcionamiento desde la caché y reanudar la sincronización. Sin una caché utilizable, como en el primer arranque, necesita central para recibir asignaciones. El token de unión no contiene la configuración de los streams.

El rechazo de `node_key` tampoco convierte al streamer en un miembro nuevo: informa del error y conserva la configuración aplicada. Recuperar la conexión exige la intervención del operador. Abrir otra ventana no repara un registro existente.

La caché no convierte al streamer en un plano de control independiente. Las asignaciones nuevas y los cambios esperan a que vuelva la conexión; los billetes caducan. La continuidad también depende de las fuentes, los discos y los demás streamers.

VERIFICAR LA UNIÓN

Después del arranque, compruebe:

1. El streamer figura en el registro de central con la etiqueta esperada y el estado de conexión actual.
2. El directorio persiste tras reiniciar y se reutiliza el mismo registro.
3. Central alcanza la dirección API y los demás streamers alcanzan la dirección privada de entrega.
4. La configuración se entrega y confirma como aplicada; el entorno no sobrescribe los valores necesarios.
5. Los streams asignados funcionan y las lecturas entre servidores usan un billete vigente.

© Flussonic 2026

Evalúe la conexión y la aplicación de configuración por separado de los medios. El registro no garantiza una fuente accesible o un almacenamiento sano; perder central tampoco detiene inmediatamente una emisión en curso.

© Flussonic 2026

Management

`cluster.control.management`

La función `cluster.control.management` incluye tres subsistemas de central:

- **Almacenamiento de configuración** — almacenamiento centralizado de la configuración de los streamers gestionados.
- **Gestión del streamer** — entrega de la configuración al streamer y supervisión de su aplicación.
- **Interfaz de usuario (UI)** — interfaz web para editar la configuración y comprobar que se ha aplicado.

En modo gestionado, central almacena la configuración del streamer y la entrega a la máquina. Las sobrescrituras locales de parámetros individuales se describen a continuación. El formulario de configuración vive al pie de la ficha del streamer — ábralo desde el registro con un clic en la fila.

Consulte la unión de streamers para añadir una máquina al clúster y conocer las ventanas de unión y la autenticación.

CÓMO SE APLICAN LOS CAMBIOS

El botón **Guardar cambios** envía solo lo que usted cambió — las ediciones paralelas de otras secciones no se pisan. Desde ahí, una traza de tres indicadores sigue la entrega:

- **Guardado** — central aceptó los cambios.
- **En vuelo** — la configuración va camino del streamer; el indicador pulsa. Si el streamer está offline, el formulario lo dice sin rodeos: la configuración se aplicará cuando vuelva el enlace.
- **Aplicado** — el streamer confirmó la aplicación; al lado — cuántos segundos tomó la entrega.

Si la configuración se cambió en paralelo con usted, el formulario muestra un cartel de conflicto con una elección: **Ver la más reciente** — releer la versión ajena, **Sobrescribir** — insistir en la suya.

Una insignia amarilla **ENV** en un campo significa que el valor lo fija el entorno de la propia máquina: pesa más que lo entregado por central, y editar el formulario no tendrá efecto en este streamer hasta quitar la variable.

LISTENERS

Los puertos que el streamer abre por protocolo: HTTP, RTSP, RTMP, WebRTC. Un puerto se añade como chip; un chip marcado `tls` significa que el puerto lleva certificado.

CERTIFICADOS Y HTTPS

La sección **Certificados** cuelga TLS de los listeners HTTP — el certificado vive en un puerto, así que primero hace falta el listener. El botón **Activar HTTPS** hace ambas cosas de una vez: un listener en 443 y una entrada de Let's Encrypt en él. Un puerto puede llevar varias entradas: los clientes eligen el certificado por SNI.

La fuente del certificado de una entrada:

- **Let's Encrypt** — emisión automática para los dominios listados. Justo al lado aparece el **Correo de contacto ACME**, y es obligatorio: sin él no se emite nada y el guardado se bloquea, y a él el CA envía los avisos de expiración. Bajo **Avanzado** están la URL del directorio ACME y el interruptor del CA de pruebas — una URL vacía es el Let's Encrypt de producción.
- **PEM estático** — su propio certificado y clave privada: PEM completo o ruta a un archivo en el streamer.
- **Autofirmado** — para bancos de pruebas y redes internas.

Los dominios van separados por comas; el formulario sugiere el hostname de las direcciones de entrega del streamer. ACME no emite para una IP pelada — hace falta un dominio.

El estado de la emisión se ve junto a la propia entrada, y se repite arriba en la ficha del streamer: **emitido** (con el plazo hasta expirar), **emitiéndose**, **fallido** — con el texto del error. Dos semanas antes de expirar la fila se pone amarilla.

DVR

Dónde escribe el streamer el archivo: la **Raíz del archivo**, la lista de discos cuando hay varios y la **Ruta del catálogo**. El interruptor **Comprobar montaje antes de escribir** está activado por defecto: escribir en la partición raíz del sistema está prohibido, y un disco sin montaje propio se reporta «no montado». Desactívelo solo a conciencia.

© Flussonic 2026

La raíz del archivo es la base de las rutas relativas y el hogar del catálogo de metadatos, **no un disco de escritura**: sin ningún disco en la lista no hay dónde escribir el archivo, y el streamer reporta un error de escritura honesto en lugar de escribir en silencio en la raíz.

Discos de caché

La segunda lista son los **Discos de caché**. Son discos con otro papel: en ellos se asientan las lecturas del archivo — las partes que el streamer trajo para un espectador de otro streamer o de sus propios discos de archivo — y una solicitud repetida la sirve la caché: localmente y desde un disco rápido. La grabación en vivo nunca los elige, la retención de streams no los gobierna, y el espacio se libera por desalojo según la última lectura. Así un edge se convierte en caché edge.

Cada disco de caché tiene dos límites opcionales:

- **Límite de caché, GiB** — cuánto puede ocupar la caché; vacío significa limitada solo por el espacio libre.
- **Mantener libre, GiB** — cuánto espacio dejar en el volumen; vacío significa el mínimo del sistema.

La misma ruta no puede figurar a la vez como disco de archivo y como disco de caché — al guardar, esa configuración se rechaza.

Debajo está la admisión en caché, común a toda la caché del streamer:

- **Ventana fresca, horas** — los fragmentos más jóvenes se cachean en la primera solicitud; vacío significa un día.
- **Contenido antiguo: número de solicitud** — a partir de qué repetición se cachea un fragmento más antiguo que ese límite; vacío significa la segunda, `1` significa de inmediato.

La razón de la división es simple: casi todos ven la cola fresca del archivo y solo unos pocos su profundidad, así que un solo salto atrás no debe desalojar lo popular.

Un streamer sin discos de caché no se rompe por un stream con la caché activada: la caché simplemente no funciona y va un aviso al log.

El volumen de la caché y la tasa de aciertos se describen en observabilidad de la caché; la configuración y el funcionamiento están en caché del archivo.

La configuración y el acceso se describen en grabación del archivo y reproducción del archivo.

VOD

La sección **VOD** define los **roots VOD que el nodo sirve y en los que acepta subidas**. Sin al menos un root, el Catálogo VOD se queda vacío: no hay dónde subir.

Cada root tiene un **Nombre del root** y una **Ruta**, además del ajuste fino: **Máx. de archivos abiertos**, **Idle timeout, s**, **Máx. de handles de lectura**; y aparte un **Máx. global de archivos abiertos** para todo el nodo.

El nombre no es una etiqueta sino parte de la disposición en disco, así que **renombrar un root no está soportado**. Elija desde el principio el nombre que va a quedarse.

Una errata en el nombre de un campo se rechaza al guardar: **422** con la ruta hasta el campo — por ejemplo `unknown field "vod.root"` si escribe `root` en lugar de `roots`.

PREFIJOS DE PUBLICACIÓN

Los prefijos de nombres bajo los que el streamer acepta publicaciones entrantes. Sin prefijos, la publicación se acepta solo en streams existentes.

ROL Y COLOCACIÓN

La pestaña **Clúster** lleva los campos de control del streamer. El **Rol** define qué hace la máquina en el clúster — solo las máquinas con el rol **Streamer** reciben streams. Los interruptores junto al rol convierten al streamer en entrega (**Nodo edge**) o en grabación del archivo (**Nodo DVR**); no se pueden activar ambos. El **Namespace**, las **Etiquetas** y los límites gobiernan la colocación — su significado está en Colocación de streams.

Los interruptores **Transcodificación** y **Copiado** son los roles VOD del streamer: si recibe tareas de transcodificación y si acepta copias de assets en sus almacenes. Ambos están activos por defecto; un rol desactivado detiene solo la entrega de trabajo nuevo, las tareas en curso terminan, y la consola avisa de ellas antes de desactivarlo. Bajo el interruptor hay una nota cuando el streamer no puede hacerlo, por ejemplo una compilación sin backend de transcodificación. Son campos de control de central; al streamer no se le entregan. Los detalles, en Copias VOD y balanceo.

© Flussonic 2026

CONEXIÓN Y ENTREGA

- **URL de API del streamer** — la dirección por la que central llega al management API de la máquina. Sin ella el streamer lleva un chip de accesibilidad en el registro y sus sesiones desaparecen de la estadísticas de sesiones.
- **Zona CDN** — a qué zona pertenece el streamer; el balanceador de reproducción dirige a los espectadores por ella.
- **URL pública de payload** — la base pública de direcciones para espectadores. Puesta — los reproductores reciben un redirect directo al streamer; vacía — central proxya los medios a través de sí.
- **URL privada de payload** — la base para transferencias entre streamers y el proxeo; por defecto coincide con la URL de API. Sin ella los vecinos no leen de este streamer: los restos del archivo que hay en él no se sirven, solo se borran.

El modo de entrega resultante se ve en la ficha del streamer: *en proceso* (el streamer local), *redirect*, *proxeado por central* o *inaccesible* — en rojo cuando no hay nada puesto. Un aviso aparte aparece cuando hay certificado emitido para un nombre pero la base de entrega sigue en `http://` — el TLS queda sin usar.

REVERSE PROXY Y RESTABLECIMIENTO

La sección **Reverse proxy (JSON)** es un editor temporal de la sección cruda `rproxy` ; un JSON inválido bloquea el guardado.

El botón **Restablecer valores** en la zona de peligro devuelve toda la configuración del streamer a los valores por defecto — el streamer la aplica en su siguiente sync. La acción es irreversible.

© Flussonic 2026

Layouter

`cluster.control.layouter`

Los streams los reparte por los streamers el **layouter**, un subsistema de central. Usted añade streamers y crea streams, y él decide qué máquina captura el stream, cuál guarda una copia de su archivo, cuál lo sirve a los espectadores. La decisión se recalcula ante cada cambio, y si una máquina falla, sus streams se mudan solos a las que siguen vivas.

Una decisión del layouter se llama **asignación** y une un stream, un streamer y un rol — lo que esa máquina hace para el stream. Por cada asignación central arma la config de servidor del stream y la entrega al streamer, y por eso los ajustes de los streams se editan en central y no en las máquinas.

La colocación se gobierna con tres ajustes. Los **namespaces** dividen el clúster en partes que no se cruzan. Las **etiquetas** fijan requisitos y preferencias. Los **límites** acotan cuánto toma una sola máquina.

CÓMO SE REPARTEN LOS STREAMS

Qué hace el layouter

- **Asigna roles y recalcula asignaciones.** La ejecución arranca cuando cambia la entrada — se unió un streamer, se creó un stream, se editaron ajustes. No hay calendario y no hay nada que lanzar a mano.
- **Tiene en cuenta las capacidades del streamer.** Un stream con codificación en GPU se asigna solo a un streamer con tarjetas. Un stream con archivo necesita un streamer con discos de archivo si en su namespace no hay streamer DVR: entonces el archivo lo graba el streamer DVR, y el streamer de ingest no necesita discos.
- **Vigila la carga.** Entre los streamers aptos se elige el menos cargado, por procesador, red y ocupación del archivo. Un streamer sobrecargado se alivia mudando streams fuera de él.
- **No muda streams sin causa.** Mientras la entrada no cambia, las asignaciones se quedan como están. Que una máquina vecina esté poco cargada no es motivo para mudar nada.

Roles de asignación

Los roles no se eligen ni se activan por separado. Se derivan de lo que ya está configurado: el stream tiene fuente, así que hace falta ingest; al stream se le fijó un número de copias de respaldo del archivo, así que aparecen copias; el clúster tiene streamers edge, así que aparece la entrega edge; en un streamer quedó el archivo de un stream mudado, así que aparece dvr parts o dvr cleanup. Conocerlos sirve para leer la tabla de asignaciones en la pestaña **Layout** del stream.

ingest — el rol principal, lo tiene todo stream en funcionamiento. El streamer recibe la fuente del stream, la transcodifica y graba el archivo cuando están configurados. Un stream tiene un streamer de ingest. Aquí es donde se comprueban los requisitos del stream hacia la máquina, por eso la mayoría de los motivos de rechazo pertenecen a este rol.

dvr record — el archivo del stream no lo graba el streamer de ingest sino un streamer DVR de su namespace. Una asignación por stream; la recibe todo stream con archivo en cuanto el namespace tiene un streamer DVR. El streamer DVR recibe el directo del streamer de ingest, como una copia de respaldo.

edge — el stream se sirve a los espectadores desde un streamer edge. El rol se asigna a cada stream en cada edge apto y no hay nada que activar aparte. Ese stream no captura fuente ni graba archivo, recibe el resultado terminado del streamer de ingest.

dvr backup — el archivo se graba en varios streamers más además del de ingest o de dvr record. Cuántas copias mantener lo fija el campo **Copias de respaldo DVR** en la pestaña **Layout**. En un namespace con streamer DVR las copias se colocan solo en streamers DVR.

dvr parts — el stream se mudó a otro streamer y en el anterior quedó archivo grabado. La asignación sirve ese archivo a los vecinos, así el espectador lo ve en la línea de tiempo común, y lo borra según la retención del stream. El streamer no graba nada nuevo: no tiene entradas del stream.

dvr cleanup — en el streamer hay archivo del stream, pero no se puede o no hace falta servirlo. La asignación solo borra el archivo según la retención: el stream no tiene entradas y los vecinos no leen de este streamer. Es el rol más débil: le toca al resto al que no le corresponde ningún otro. Cuándo un resto recibe servicio y cuándo limpieza — en la sección Restos del archivo.

© Flussonic 2026

Tolerancia a fallos

El layouter reacciona al estado de conexión del streamer de la siguiente manera.

- **Señal perdida.** Las asignaciones se congelan — no se retiran, no se crean, y la máquina no recibe streams nuevos. Un corte breve no muda streams, y en cuanto vuelve el enlace la máquina sigue llevando los mismos.
- **Offline.** Los streams se mudan a máquinas vivas en cuanto se encuentran aptas.

La entrega edge es la excepción. Se mantiene con cualquier estado de conexión, porque el edge levanta el stream cuando lo pide un espectador, y no tiene sentido retirar la asignación por el silencio. Un streamer sacado del clúster pierde todas sus asignaciones.

RESTOS DEL ARCHIVO

Cuando un stream se muda a otro streamer, el archivo grabado se queda en la máquina anterior. Central lo contabiliza como **resto** y lo atiende con uno de dos roles hasta que venza la retención del stream.

- **dvr parts** — el streamer encaja con el stream por namespace y etiquetas, no es edge ni la máquina de ingest ni la de dvr backup, y tiene URL privada de payload.
- **dvr cleanup** — en todos los demás casos: no hay dirección, el namespace es otro, las etiquetas requeridas no coinciden, el stream está deshabilitado, la grabación del stream se quitó. La limpieza solo necesita el rol **Streamer**, almacenamiento de archivo y una máquina viva y aprobada. El límite de streams no la cuenta.

Un servicio que corresponde pero no se creó se ve como demanda sin cubrir en la pestaña **Layout** del stream — normalmente con el motivo **sin dirección utilizable**. La limpieza en lugar del servicio fuera del namespace o de las etiquetas no genera demanda: el namespace y las etiquetas funcionaron como se configuraron, y el rol se ve solo en la tabla de asignaciones. Un resto que ni siquiera la limpieza atiende es una demanda sin cubrir del rol **dvr cleanup**, por ejemplo con el motivo **rol del nodo no compatible**.

Grabación quitada y stream deshabilitado

Apagar la grabación no borra el archivo al instante. El stream se queda en los streamers donde está su archivo, y cada uno borra lo grabado con una limpieza de retención por defecto — un día. Vuelva a encender la grabación antes, y el archivo entero vuelve a estar en uso. Un stream deshabilitado recibe solo limpieza: sus restos se borran según la retención del stream.

Huérfanos

Un archivo sin stream es un **huérfano**. Así queda el archivo de un stream eliminado; así se ve también el archivo de un streamer que vuelve al clúster tras una ausencia larga.

El streamer encuentra ese archivo por sí mismo y lo reporta a central, y central le asigna limpieza con el motivo **archivo huérfano**. No hay que hacer nada: en un día el archivo está borrado y el espacio, libre. Si existe un stream con ese nombre, el archivo cuenta como suyo y se atiende según sus reglas.

CÓMO INFLUIR EN LA COLOCACIÓN

Los ajustes del streamer viven en la pestaña **Clúster** de su página.

El stream tiene su propio namespace y sus propias etiquetas, y se fijan en la pestaña **Layout** de su página — el mismo sitio donde se ven sus asignaciones.

Namespaces

El namespace es una frontera estricta. Un stream se asigna solo a streamers con el mismo namespace, y no hay excepciones entre namespaces. El valor vacío también es un namespace, así que un stream sin él se asigna solo a streamers sin él.

Así se divide un clúster en partes que no se cruzan — por inquilino o por sitio, por ejemplo. El namespace del streamer se fija en la pestaña **Clúster**, el del stream en la pestaña **Layout**.

Etiquetas

La etiqueta es una marca libre que usted pone al streamer y al stream. Qué significa lo decide usted: el layouter solo compara etiquetas cuando elige máquina. Así entra en la colocación una condición que el sistema no puede conocer.

La etiqueta y el namespace saben hacer lo mismo — prohibir la colocación — y en ambos casos la frontera no se puede cruzar. La diferencia está en la escala. El namespace divide el clúster entero: hay que marcarlo todo y ningún stream se mueve entre las partes. La etiqueta ata streams concretos

© Flussonic 2026

y admite un repliegue mediante `preferred_`. Si se necesita aislar todo el clúster, tome un namespace; si la condición es de streams concretos, una etiqueta.

Las etiquetas del streamer se fijan en la pestaña **Clúster**, las del stream en la pestaña **Layout**. La forma de escribirla decide qué significa. La palabra `fast` de abajo es solo un ejemplo, en su lugar puede ir cualquier palabra suya.

- **`fast`** — propiedad. Así se marca una máquina o un stream sin exigir nada.
- **`required_fast`** — requisito. Un stream que la lleva se asigna solo a un streamer que también tenga la etiqueta `fast`. La regla funciona en ambos sentidos — un streamer con `required_fast` toma solo streams con esa etiqueta.
- **`preferred_fast`** — preferencia. En igualdad de condiciones gana el streamer con la etiqueta `fast`, pero su ausencia no impide la asignación.

Se compara la palabra en sí, no la forma de escribirla. El requisito `required_fast` lo satisface cualquiera de las tres formas del otro lado — `fast`, `required_fast` o `preferred_fast`. La preferencia `preferred_fast` se cuenta igual.

Ejemplos de colocación

Dos centros de datos con un canal estrecho entre ellos. Mientras el layouter ve el clúster entero, coloca los streams y sus copias de archivo en cualquier máquina: el tráfico de transferencia del archivo cruza el canal entre centros, y al fallar una sede sus streams pasan en masa al otro centro, donde sus fuentes no se pueden recibir de todos modos. Divida el clúster en los namespaces `dc1` y `dc2` — y cada stream con todos sus papeles vive dentro de su sede. Hay que marcarlo todo: cada stream y cada streamer necesitan un namespace, y un streamer sin él es inaccesible para los streams `dc1` y `dc2`.

Una fuente multicast no llega desde cada centro. Diez nodos están repartidos entre dos centros de datos, y el multicast se enruta solo dentro del suyo. Hay que atar a la sede solo un stream, no todo el clúster — por eso no un namespace, sino una etiqueta: `dc1` y `dc2` a los streamers, `required_dc1` al stream. Los demás streams siguen colocándose en todo el clúster.

Las máquinas GPU primero, pero no solo ellas. Un stream puede transcodificar tanto en CPU como en GPU, pero las máquinas con tarjetas son pocas. Ponga a las máquinas con tarjetas la etiqueta `gpu` y al stream `preferred_gpu`: en igualdad de condiciones cae en una máquina con GPU, y cuando estas se llenan rebosa a las demás. La transcodificación en GPU se describe en la página Transcodificador.

El stream está mejor en su centro, pero el traslado es admisible. `preferred_dc1` mantiene el stream en el primer centro mientras la sede tenga capacidad y lo rebose al segundo cuando se llene. Un namespace no puede expresar semejante repliegue.

Una máquina y un stream con varias etiquetas. Una máquina y un stream pueden llevar muchas etiquetas: un streamer del primer centro con GPU lleva `dc1` y `gpu`, y un stream con `required_dc1` y `preferred_gpu` se queda dentro del primer centro, primero en las máquinas con GPU.

Límites y carga

La carga del streamer se mide por la máquina entera — todo su procesador y todo su tráfico, incluida la entrega a espectadores, las transferencias entre streamers y los procesos ajenos. Los roles de asignación no afectan a la medición.

Tres campos en la pestaña **Clúster** fijan los topes.

- **Límite de streams** — cuántos streams puede llevar la máquina. Cero la deja en el clúster, pero vacía.
- **Límite de red, kbit/s** — el tope de tráfico de la máquina. Por encima el layouter no añade streams, y al superarse alivia la máquina.
- **Límite de DVR, %** — la ocupación de los discos de archivo con la que el streamer cuenta como lleno.

Los topes restringen las asignaciones de ingest, de dvr record y de copias de respaldo del archivo. La entrega edge, dvr parts y dvr cleanup no quedan restringidos por ellos — esos roles se asignan sin importar la carga de la máquina.

© Flussonic 2026

6.4.2 Entrega (cluster.delivery)

Zones

`cluster.delivery.zones`

Cuando los streamers están en distintas ciudades y redes, al espectador hay que mandarlo al cercano. Para eso están las zonas CDN: la IP del espectador se asigna a una zona por rutas CIDR, y el balanceador de reproducción elige un streamer vivo de la zona coincidente que tenga el stream.

Mientras no hay zonas, rige el comportamiento por defecto: los reproductores van a los streamers edge, si el clúster los tiene, y si no — al streamer donde el stream se captura. Para una instalación en una sola red suele bastar — las zonas llegan junto con la geografía.

CÓMO SE ELIGE EL STREAMER

1. La IP del espectador se compara con las rutas CIDR de **todas** las zonas; gana el prefijo coincidente más largo.
2. En la zona ganadora se busca un streamer vivo que lleve el stream.
3. Si no lo hay — el espectador se lleva a la **zona de reserva** de esa zona, si está puesta.
4. Desde ahí — el modo de entrega del streamer: con URL pública de payload el espectador recibe un redirect directo a la máquina, sin ella central proxya los medios a través de sí (vea la configuración del streamer).

CREE UNA ZONA

La pantalla **Zonas CDN → Crear zona**, un nombre corto como `eu` o `mad`.

Se abre el editor de la zona:

- **Rutas CIDR** — prefijos IPv4 e IPv6 en una sola lista: `10.0.0.0/8`, `2a02:6b8::/32`. Un chip por prefijo; un CIDR inválido el formulario no lo acepta.
- **Zona de reserva** — adónde llevar al espectador cuando esta zona no tiene streamer vivo con el stream.

Pulse **Guardar**. Si la zona se editó en paralelo, el editor muestra el mismo cartel de conflicto que los demás formularios: releer o sobrescribir.

ASIGNE LOS STREAMERS

Una zona sin streamers no enruta nada. La pertenencia se fija del lado del streamer: el campo **Zona CDN** en su configuración. En el registro la zona se ve junto al nombre de la máquina.

SI LOS ESPECTADORES ACABAN DONDE NO DEBEN

- **Un espectador cae en zona ajena.** Revise las rutas: gana el prefijo más largo entre todas las zonas — una ruta más específica de otra zona pisa la suya.
- **La zona coincidió pero el espectador se fue a otra parte.** En la zona no se halló streamer vivo con el stream — actuó la zona de reserva o el comportamiento por defecto. Mire el registro: ¿viven las máquinas de la zona y está el stream colocado en ellas?
- **Los espectadores reciben la dirección de central en vez de la del streamer.** El streamer no tiene URL pública de payload — central proxya los medios. Es un modo válido, pero todo el tráfico pasa por central.

© Flussonic 2026

6.5 Dispositivos

6.5.1 ONVIF (device.onvif)

Descubrimiento de cámaras

`device.onvif.discovery`

Sapsan puede descubrir cámaras ONVIF en la red, recibir sus eventos (movimiento, detección de personas y vehículos) y comprobar la compatibilidad de una cámara.

DESCUBRIMIENTO DE CÁMARAS

Sapsan descubre las cámaras mediante WS-Discovery (solicitudes Probe multicast y respuestas ProbeMatch) y, además, entiende el protocolo de descubrimiento propietario de Hikvision. De cada cámara recopila las direcciones (XAddrs), el nombre y el modelo; las observaciones repetidas de una misma cámara se fusionan.

Al trabajar a través de NAT, las direcciones que una cámara comunica sobre sí misma (URL de snapshot, URL de RTSP) se reescriben automáticamente a la dirección del dispositivo realmente alcanzable.

CONFIGURACIÓN DE LA CÁMARA MEDIANTE ONVIF

Tras descubrir una cámara, Sapsan puede obtener su URL RTSP y configurar la propia cámara mediante ONVIF. Los parámetros disponibles dependen del modelo y de los permisos de su cuenta.

1. Obtenga los perfiles con `media/GetProfiles` e identifique el token de la configuración del codificador.
2. Lea `media/GetVideoEncoderConfigurations` y los valores permitidos mediante `media/GetVideoEncoderConfigurationOptions`.
3. Modifique los campos necesarios de la configuración completa y envíela con `media/SetVideoEncoderConfiguration`. Permite ajustar resolución y frecuencia de fotogramas, bitrate y GOP, y un códec admitido por la cámara.
4. Vuelva a leer la configuración y compruebe la entrada RTSP. La cámara puede reiniciar el codificador al aplicar los cambios.

Utilice `POST /flussonic/api/onvif_call/{camera}/{service}/{method}` en la interfaz HTTP de Sapsan. `{camera}` es el nombre del stream configurado cuya entrada proporciona la dirección de la cámara; para MBR se utiliza la primera fuente anidada. Ejemplo de consulta de perfiles:

```
curl --user admin -X POST \
  'http://sapsan.example.com:5080/flussonic/api/onvif_call/camera1/media/GetProfiles' \
  -H 'Content-Type: application/json' --data '{}'
```

La autorización en Sapsan y la cuenta de la cámara son independientes. Por defecto, las credenciales de la cámara se toman de la URL de entrada y ONVIF utiliza el puerto 80. `x-onvif-auth: user:password` sustituye las credenciales de la cámara; `x-onvif-port`, su puerto ONVIF. El cuerpo JSON contiene los argumentos del método ONVIF; las claves con `$` representan atributos XML, como `$token`.

La configuración del stream mediante central controla la captura en Sapsan. `SetVideoEncoderConfiguration` modifica la propia cámara. Para PTZ y otras operaciones, consulte control ONVIF.

SIGUIENTES PASOS

- Captura desde una cámara por RTSP

© Flussonic 2026

Eventos de cámaras

`device.onvif.events`

Sapsan puede descubrir cámaras ONVIF en la red, recibir sus eventos y comprobar la compatibilidad de una cámara.

EVENTOS DE LA CÁMARA

Los eventos ONVIF se convierten en episodios:

- **Movimiento** — CellMotion `IsMotion` y MotionAlarm: `true` indica que hay movimiento; `false`, que ha terminado.
- **Detección de personas y vehículos** — Los eventos contienen los resultados de la analítica de la cámara.
- **Entradas digitales** — La cámara comunica las activaciones de las entradas.
- **Tiempo de espera por inactividad** — El streamer cierra los episodios que dejan de recibir eventos.

El streamer mantiene un episodio de movimiento por stream. Tiene en cuenta todas las fuentes de movimiento activas de la cámara; la ventana de espera comienza cuando la última fuente activa indica que el movimiento ha terminado.

AGRUPACIÓN DE EPISODIOS DE MOVIMIENTO (DEBOUNCE)

Tras una señal de fin de movimiento, el streamer espera **30 segundos** antes de cerrar definitivamente el episodio. Esta ventana agrupa las pausas breves de movimiento en un solo episodio.

- **Pausa inferior a 30 segundos** — Una nueva señal de movimiento continúa el episodio anterior con el mismo ID y la misma hora de inicio. Por ejemplo, si el movimiento vuelve tras 5 segundos, no se crea un episodio separado.
- **Pausa de 30 segundos o más** — El episodio anterior se cierra; el siguiente movimiento crea uno nuevo.
- **La cámara no envía una señal de fin de movimiento** — Se aplica otro tiempo de espera de seguridad: 15 segundos sin una nueva señal que indique movimiento.

Las señales `false` repetidas sin una nueva señal `true` no prolongan la ventana de espera.

Cuando vence la ventana, la hora de fin del episodio es la de la última señal de fin de movimiento. Los 30 segundos de espera no se añaden a su duración.

Ejemplo:

Hora	Evento	Resultado
10:00:00	Comienza el movimiento	Comienza el episodio
10:00:10	Termina el movimiento	Comienza la espera
10:00:15	Vuelve el movimiento	Continúa el mismo episodio
10:00:20	Termina el movimiento	Comienza otra ventana de espera
Desde las 10:00:50	El movimiento no ha vuelto	El episodio se cierra con hora de fin 10:00:20

El streamer fija la ventana de 30 segundos; no tiene un campo propio en la interfaz. El área de detección de la cámara se configura mediante una máscara de celdas.

Al detenerse el stream, sus episodios abiertos se cierran de inmediato; la ventana de espera no continúa después.

© Flussonic 2026

Control

`device.onvif.control`

ONVIF permite controlar la cámara y acceder a sus parámetros. Si admite PTZ, Watcher puede activar los controles de giro, inclinación y zoom.

También permite cambiar parámetros de codificación, como el perfil H.264. Los ajustes disponibles dependen de la cámara. La recopilación de eventos de movimiento se activa por separado y requiere un detector activo en la cámara.

© Flussonic 2026

Comprobación de conformidad ONVIF

`device.onvif.compliance`

La comprobación integrada de la cámara genera un informe estructurado: información del dispositivo, autenticación (HTTP Digest), reloj de la cámara (el desfase de tiempo se maneja automáticamente), clasificación de eventos, además de una prueba del stream RTSP con un resumen de errores (pérdida de paquetes, payload dañado, desincronización) y estadísticas de fotogramas clave.

Note

TODO: cómo lanzar el descubrimiento y la comprobación de compatibilidad (config/API/CLI), la configuración de la cámara vía ONVIF, la obtención de snapshots.

© Flussonic 2026

6.5.2 Agente (device.agent)

Runtime

`device.agent.runtime`

Flussonic Agent se ejecuta en una cámara u otro dispositivo local e inicia una conexión cifrada con el servidor. Permite conectar cámaras detrás de NAT sin IP pública ni redirección de puertos.

El Agente registra la cámara automáticamente. En conexiones inestables almacena datos en un búfer para retransmitir paquetes perdidos. La instalación depende del firmware y del fabricante.

© Flussonic 2026

6.6 Gateway

6.6.1 Túneles a dispositivos (gateway.tunnel)

Túneles de agentes

`gateway.tunnel.agents`

Peeklio es un gateway de proxy inverso integrado en Sapsan. Un agente ligero, instalado junto a la cámara (detrás de NAT), se conecta por sí mismo a Sapsan y mantiene un túnel; a través de ese túnel Sapsan toma el video y el tráfico del dispositivo como si estuviera en la red local.

CONFIGURACIÓN DEL GATEWAY

```
rproxy:
  streampoint_key: <clave para conexiones streampoint>
  endpoint_auth_url: http://backend.example.com/agent-auth
  agents:
  - agent_id: a1b2c3
    password: <contraseña del agente>
    salt: <sal>
    streampoint_url: http://this-server:5080
```

Los agentes se pueden autorizar mediante la lista local `agents` o un backend externo `endpoint_auth_url`. Si se define `endpoint_auth_url`, se usa el backend y se ignora `agents`. Sin backend, una lista `agents` no vacía permite la autorización local. El modo «solo streampoint», sin registrar agentes nuevos, se aplica cuando no se configura ni un backend ni credenciales locales de agentes.

En el lado del agente se fijan la dirección del servidor, el `agent_id`, la contraseña, el intervalo de pings y una **ACL de hosts** — la lista de direcciones a las que se puede acceder a través de este agente; todo lo demás se rechaza.

El estado de la pasarela se conserva al recargar la configuración. Los cambios de `streampoint_key` o `endpoint_auth_url` se aplican al recargar y reconectan las sesiones de control de los agentes con los nuevos ajustes.

CAPTURA DE UN STREAM A TRAVÉS DE UN AGENTE

En la entrada de un stream se indica `via_agent` con el identificador del agente:

```
streams:
  - name: cam-behind-nat
    inputs:
    - rtsp:
        url: rtsp://admin:password@192.168.1.10/stream0
        via_agent: "agent://a1b2c3"
```

La dirección en `url` es la dirección local de la cámara en la red del agente. El handshake RTSP y el stream van por el túnel.

COMPORTAMIENTO ANTE ERRORES

Los errores de conexión a través de un agente son distinguibles: puerto cerrado (connection refused), host que no está en la ACL del agente (permission denied), nombre que no resuelve. El servidor puede enviar de forma remota a un agente los comandos `reset` y `reboot`.

MONITORIZACIÓN DE AGENTES

`GET /streamer/api-v4/rproxy/streampoint-agents` devuelve contadores por cada agente: pings, bytes en ambas direcciones, intentos/aperturas/conexiones actuales. Véase el Admin API.

> **Note**
>
> La configuración del propio agente y la emisión de claves quedan fuera del alcance de este manual. Estas operaciones están estrechamente integradas con Flussonic Watcher y se realizan mediante la solución suministrada.

© Flussonic 2026

6.7 ACS

6.7.1 Acceso (acs.access)

Gates

`acs.access.gates`

Un punto de paso se conecta al sistema de control de acceso. Watcher envía un evento de reconocimiento y el integrador transmite una orden de apertura al controlador. El ejemplo de Sigur usa `ALLOWPASS` con el identificador de puerta.

El integrador se suscribe a `person_detected` para rostros o `car_detected` para matrículas. Configure la dirección del controlador y la relación entre cámara y paso. El reconocimiento no sustituye los permisos de acceso.

© Flussonic 2026

6.8 Funciones adicionales

6.8.1 Subtítulos ocultos

Los subtítulos ocultos (closed captions, CC) viajan dentro del propio vídeo y no como una pista aparte: en los datos auxiliares de cada fotograma (SEI para H.264 y HEVC, user data para MPEG-2, metadatos OBU para AV1). Los entienden los televisores y los decodificadores, pero un reproductor de navegador no puede mostrar ese texto: HLS y DASH esperan los subtítulos como una pista aparte.

Sapsan hace las dos cosas: pasa los subtítulos ocultos dentro del vídeo tal como son y los decodifica en una pista de subtítulos WebVTT.

Modos de procesamiento

El modo se fija con el campo `mode` de la sección `closed_captions` del stream. Controla solo la entrega — la decodificación y la grabación en el archivo ocurren siempre:

- **`passthrough`** (por defecto) — los subtítulos ocultos se quedan dentro del vídeo y se anuncian en los manifiestos como closed captions. No hay pista de subtítulos en los manifiestos.
- **`extract`** — una pista de subtítulos **en lugar de** los subtítulos ocultos incrustados: el texto se decodifica en una pista WebVTT, mientras que los subtítulos ocultos se recortan del vídeo de salida y no se anuncian como closed captions.
- **`both`** — ambas cosas a la vez: los subtítulos ocultos viajan dentro del vídeo y se anuncian, y junto a ellos va una pista de subtítulos.

Configuración

```
streams:
  - name: news
    inputs:
      - url: udp://239.0.0.1:1234
    closed_captions:
      mode: extract
      services:
        CC1:
          language: eng
          name: English
        SERVICE3:
          language: spa
          name: Espanol
```

La sección `services` es opcional y hace dos cosas:

- fija el **nombre y el idioma** de la entrada del menú del reproductor. Sin ella, el nombre se deduce del stream: el idioma anunciado por el servicio y, en su defecto, la dirección del servicio (`CC1` , `SERVICE3`);
- **preanuncia** el servicio en los modos `extract` y `both` : un servicio listado obtiene su pista desde el primer segmento, sin esperar al primer cue. Esto importa para los reproductores que leen la lista de pistas una sola vez al arrancar: un servicio que empieza a hablar en el minuto diez aparecería en el menú solo después de reabrir el stream.

La clave del servicio es `CC1` … `CC4` para CEA-608 y `SERVICE1` … `SERVICE63` para CEA-708.

Subtítulos desde SDI

SDI transporta CEA-608/708 en paquetes CDP dentro de ANC. Al codificar vídeo SDI en H.264 o HEVC, Sapsan inserta los subtítulos en el SEI del fotograma correspondiente. La misma señal también puede transportar teletexto.

Configure la entrada SDI y la codificación de vídeo. El modo `both` conserva los subtítulos integrados para receptores MPEG-TS y añade WebVTT para reproductores de navegador:

```
streams:
  - name: studio
    inputs:
      - sdi:
          device: /dev/v4l/by-path/pci-0000:5e:00.0-video-index0
    transcoder:
      output:
        - source: {best_quality: video}
```


© Flussonic 2026

```
          codec: {set: h264}
          bitrate: 2800000
        - source: {best_quality: audio}
          codec: {set: aac}
          bitrate: 128000
    closed_captions:
      mode: both
```

CDP es el portador de subtítulos SDI admitido; no se admite CEA-608 de la línea 21 sin CDP. Los paquetes CDP dañados se descartan por completo. DVR conserva el SEI de subtítulos junto con las muestras H.264/HEVC codificadas. El portador de teletexto SDI usa la pista `t68`, junto con CC1 en `t1` y SERVICE1 en `t5`.

Qué recibe el reproductor

Cada servicio que habla se convierte en su propia pista de texto con un número fijo: `CC1` … `CC4` → `t1` … `t4`, `SERVICE1` … `SERVICE63` → `t5` … `t67`. El número pertenece al servicio y no cambia cuando el stream se reinicia, de modo que los enlaces a una pista siguen funcionando.

En HLS la pista llega como un rendition `EXT-X-MEDIA` con `TYPE=SUBTITLES`; en DASH, como un AdaptationSet con `contentType="text"` y `mimeType="text/vtt"`. Los segmentos de la pista viven en direcciones como:

```
/streaming/v/news/subtitles/t1/1738245600000.vtt
```

El último segmento de la ruta es el inicio de la ventana en milisegundos; el final de la ventana lo decide el servidor según la misma rejilla de segmentos que usa el vídeo. Una ventana vacía es un segmento WebVTT vacío válido, no un error: la pista no debe romperse durante una pausa entre cues.

Los subtítulos se sirven en vivo, en rewind y desde el archivo. Los cues decodificados se escriben siempre en el archivo independientemente del modo, así que activar `extract` funciona con efecto retroactivo: los subtítulos aparecen también en el archivo ya grabado.

La limitación del modo `extract` en las salidas TS

En el modo `extract` los subtítulos ocultos se recortan del vídeo de salida por completo — en todas las salidas a la vez, MPEG-TS incluido: push por udp/rtp/srt, segmentos `.ts` de HLS y tshttp. A una pista de texto no le queda dónde viajar dentro del MPEG-TS, así que en este modo un consumidor de una salida TS se queda sin subtítulos en absoluto.

Si el mismo stream se ve en un navegador y se recoge por MPEG-TS al mismo tiempo, use `both`: mantiene los subtítulos ocultos dentro del vídeo para el consumidor del TS y añade la pista para el navegador.

Migración desde Flussonic Media Server

En el Flussonic legacy la opción `cc.extract` **no** quitaba los subtítulos ocultos del vídeo — activaba la extracción además de los incrustados. Su equivalente directo en Sapsan es, por tanto, `mode: both`, no `mode: extract`.

Flussonic	Sapsan	Comportamiento
opción no definida	`mode: passthrough`	subtítulos ocultos dentro del vídeo, sin pista
`cc.extract`	`mode: both`	subtítulos ocultos dentro del vídeo **y** una pista de subtítulos
—	`mode: extract`	comportamiento estricto nuevo: una pista **en lugar de** los subtítulos ocultos incrustados

`mode: extract` es un comportamiento nuevo sin equivalente en el Flussonic legacy. Elíjalo cuando ninguno de sus consumidores necesite los subtítulos ocultos incrustados: elimina la entrada duplicada del menú del reproductor (los mismos subtítulos como closed captions y como pista) y ahorra espacio en los fotogramas de vídeo.

Comprobación

Para confirmar que la pista ha aparecido, revise el playlist maestro:

```
curl -s http://localhost:8080/streaming/v/news/index.m3u8 | grep SUBTITLES
```


© Flussonic 2026

y el contenido del segmento:

```
curl -s http://localhost:8080/streaming/v/news/subtitles/t1/1738245600000.vtt
```

Los servicios detectados y el número de pista de cada uno se ven en las estadísticas del stream (`GET /streamer/api-v4/streams/stats` , la sección `captions`).

El trabajo del pipeline lo muestran las métricas de Prometheus con el prefijo `stream_captions_` : `stream_captions_cc_pairs_total` dice si los subtítulos ocultos llegan al servidor siquiera; `stream_captions_cues_decoded_total` , cuántos cues se decodificaron; `stream_captions_cues_delivered_total` , cuántos llegaron a la pista. Un `stream_captions_unrecognized_payloads_total` creciente significa que el stream lleva un wrapper de subtítulos ocultos que no conocemos; comuníquelo al soporte.

En SDI, distinga la recepción de CDP de la decodificación del texto. La entrada proporciona estos contadores Prometheus:

- **`stream_input_sdi_vanc_cdp_bad_identifier_total`** — identificador CDP no válido.
- **`stream_input_sdi_vanc_cdp_bad_checksum_total`** — suma de comprobación CDP no válida.
- **`stream_input_sdi_vanc_cdp_malformed_total`** — estructura, marcadores o footer no válidos.
- **`stream_input_sdi_vanc_cdp_without_data_total`** — CDP válido sin cc_data.

`stream_input_transcoder_caption_unsupported_total` cuenta las calidades de salida sin un portador de subtítulos SDI compatible. Seleccione H.264 o HEVC para transportar los subtítulos SDI en el vídeo codificado.

© Flussonic 2026

6.8.2 Compatibilidad con Flussonic

Sapsan responde en las URL habituales de Flussonic Media Server para que los reproductores, las integraciones y el monitoreo existentes sigan funcionando durante la migración. La compatibilidad está activada por defecto y se puede desactivar en el config:

```
flussonic:
  streaming: false   # por defecto true
```


© Flussonic 2026

URL heredadas admitidas

Se reconocen estas formas de dirección (GET/HEAD; OPTIONS atiende CORS preflight):

Legacy URL	Comportamiento
`/<stream>/index.m3u8`	HLS; disposición según el perfil
`/<stream>/index.fmp4.m3u8`, `/<stream>/index.ll.m3u8`	302 al master HLS nativo
`/<stream>/tracks-<tracks>/index.m3u8` (también `index.fmp4.m3u8`, `index.ll.m3u8`)	HLS con selección de pistas
`/<stream>/variant/<track>/index.m3u8`	Lista de una pista
`/<stream>/rewind-<seconds>.m3u8`	Rebobinado HLS; disposición según el perfil
`/<stream>/video.m3u8`	Master HLS con MPEG-TS multiplexado
`/<stream>/video-<from>-<duration>.m3u8`	HLS de archivo con MPEG-TS multiplexado
`/<stream>/video-rewind-<seconds>.m3u8`	Forma legacy; master mixto en vivo sin rebobinado
`/<stream>/index-<from>-<duration>.m3u8`, `/<stream>/archive-<from>-<duration>.m3u8` (también `.fmp4.m3u8`)	HLS de archivo
`/<stream>/tracks-<tracks>/index-<from>-<duration>.m3u8` (también `archive-`, `.fmp4.m3u8`)	HLS de archivo con selección de pistas
`/<stream>/archive-<from>-<duration>.mpd`	302 a DASH de archivo
`/<stream>/archive-<from>-<duration>.mp4`	302 a exportación de archivo
`/<stream>/tracks-<tracks>/archive-<from>-<duration>.mp4`	Exportación con selección de pistas
`/<stream>/info.json`, `/<stream>/ranges.json`, `/<stream>/recording_status.json`	Información del stream, rangos del archivo, estado de grabación
`/<stream>/preview.jpg`, `/<stream>/preview.mp4`	Vista previa en vivo
`/<stream>/<utc>-preview.jpg`, `/<stream>/<utc>-preview.mp4`, `/<stream>/<utc>.jpg`, `/<stream>/YYYY/MM/DD/HH/MM/SS-preview.jpg`, `/<stream>/YYYY/MM/DD/HH/MM/SS-preview.mp4`	Vista previa del archivo
`/<stream>/m4f`, `/<stream>/m4s`	Transporte legacy en vivo
`/<stream>/archive-<from>-<duration>[.m4f]`, `/<stream>/m4f-<from>-<duration>[.m4f]`	Flujo legacy de archivo; duración también admite `now`
`/<stream>/YYYY/MM/DD/HH/MM/SS.m4f`, `/<stream>/YYYY/MM/DD/HH/MM/SS.jpg`	Segmento legacy del archivo o vista previa
`/<stream>.isml/manifest`, `/<stream>.isml/Manifest`	307 al manifiesto MSS nativo
`/<stream>(archive=<from>-<duration>).isml/manifest` (también `Manifest`)	Manifiesto MSS del archivo
`/<stream>.isml/QualityLevels(<bitrate>)/Fragments(<track>=<time>)` (también forma de archivo `.isml`)	Fragmento MSS

En las formas m4f, `[.m4f]` indica una extensión opcional. Las marcas `<utc>` de vista previa pueden usar segundos o milisegundos Unix.

`<from>` y `<duration>` son segundos enteros; la duración debe ser positiva. `<tracks>` usa números como `v1a1`. También se reconocen `info.json` y `ranges.json` anidados. Los selectores antiguos `mono`, `video`, `video<N>`, `track-*` y `txt-*` antes de ventanas del archivo se aceptan sin aplicar esa selección: se devuelve el master completo.

© Flussonic 2026

Las URL HLS neutras (`index.m3u8` , `.m3u8` del archivo, `rewind-…`) con un perfil de salida que contiene `hls` devuelven el master con la disposición del perfil. Sin ese perfil, redirigen al HLS fMP4 nativo. Las formas explícitas `.fmp4.m3u8` y `.ll.m3u8` mantienen fMP4. La familia `video*` entrega MPEG-TS multiplexado.

Reconocer una URL no implica admitir todos los modos: la exportación `.ts` y su duración `now` devuelven 422; rewind de `.isml` no se admite. Los endpoints finales aplican la misma autorización, límites de licencia y disponibilidad del archivo que las URL nativas.

Parámetros de `ranges.json` : `closed_at_gte` , `opened_at_gte` , `opened_at_lte` , `resolution` , `limit` , `cursor` , `sort` .

Las formas `video<N>.m3u8` y sus variantes de archivo/rewind no son compatibles.

DISPOSICIÓN DE LISTAS EN LAS URL FLUSSONIC

Las direcciones neutras `index.m3u8` , ventanas de archivo `.m3u8` y `rewind-<seconds>.m3u8` con un perfil de dispositivo que contiene `hls` devuelven una lista con código 200. Sin perfil o con un perfil sin `hls` , mantienen la redirección 302 a la URL fMP4 nativa con la query original. Las ventanas de archivo conservan sus límites; rewind legacy con ese perfil devuelve un master en vivo **sin rebobinado**. El parámetro `from` puede seleccionar el archivo.

El campo `hls.output` del perfil seleccionado por la solicitud determina la disposición:

- **`fmp4`** — fragmentos fMP4.
- **`ts`** — MPEG-TS con pistas de audio separadas, bajo `/streaming/v/<stream>/ts/…` .
- **`ts_mux`** — MPEG-TS con audio dentro de los segmentos de vídeo, bajo `/streaming/v/<stream>/ts/mux/…` .

`video.m3u8` siempre devuelve la forma mixta `ts_mux` , incluso sin perfil; el perfil filtra las pistas. `video-rewind-<seconds>.m3u8` se reconoce, pero también devuelve un master en vivo sin rebobinado. Las formas explícitas `.fmp4.m3u8` e `index.ll.m3u8` , así como `tracks-…/` , mantienen la redirección.

La admisión, sesiones, inicio ondemand y selección de perfil coinciden con la entrega nativa. Los enlaces son rutas absolutas bajo `/streaming/v/<stream>/…` con parámetros de query, incluido el token. El proxy debe conservar este prefijo. No se admiten las familias independientes `mono` y timeshift (`video-timeshift_rel-…`). Consulte las formas nativas y límites de códecs en HLS sobre MPEG-TS.

Admin API v3

La API bajo el prefijo `/streamer/api/v3` responde en el formato de Flussonic Streamer API v3 — véase Admin API. Al convertir una configuración de v3, algunos campos se descartan de forma deliberada (`static` , `comment` , `title` , `segment_duration` , listeners `https` y otros) — durante la conversión se dispone de un informe de pérdidas.

Tokens

La reproducción legacy acepta `?token=` y `Authorization: Bearer` . En estas URL, el token de query tiene prioridad sobre el encabezado.

Qué viene después

- Admin API
- Autorización

© Flussonic 2026