MANUAL SMART-SRT PASO A PASO

Preparar todo lo necesario

- Descargamos el instalador del **SmartSRT Control** de http://jrweb.todostreaming.es/downloads/SetupSmartSRT.exe y lo instalamos (Windows 7 o superior). Necesitará el NET Framework 4.6.1 que se descargará automáticamente si no lo tienes instalado en tu sistema. **AVISO**: *Este software es gratuito y no esta firmado digitalmente, por lo que aunque Windows le advierta de ello, puede hacer caso omiso e instalarlo sin problemas, ya que garantizamos que la copia que hay en nuestro servidor esta completamente libre de virus.*
- Preparamos dos Raspberry Pi con el último software, que se puede descargar en la zona de descargas de nuestra web http://www.todostreaming.es/ . Uno será el **emisor** y el otro el **receptor**.
- Preparamos el **encoder** que vamos a usar como fuente de video. Puede ser un software de streaming RTMP (FMLE, Wirecast, OBStudio, Vmix o nuestro LMUI con el modulo encoder), pero si quieres la más baja latencia posible, te recomendamos el uso de un encoder hardware tipo Hi3516A, como el URay UHE265-1-Mini con entrada HDMI, o el URay USE265-1-Mini con entrada SDI. A la venta en Aliexpress por unos 240 USD.

El Raspberry Pi que va a hacer de **emisor**, debe de estar en la misma red local que el **encoder**.

Nos damos de alta en el servicio SmartSRT

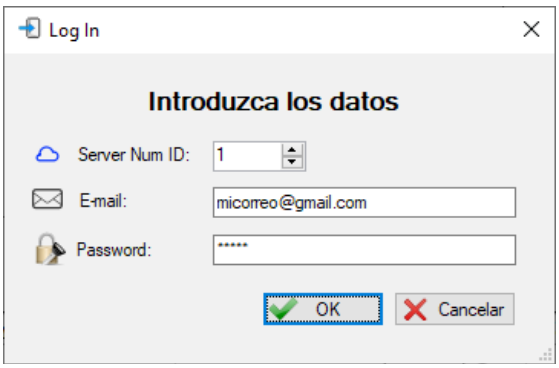
Puedes pedirnos una DEMO gratis en sales@todostreaming.es
Te daremos los siguientes datos de acceso:
- Server Num ID.- Un número entero que dice el server SmartSRT que usarás. (ej: 1)
- E-mail.- Que usarás para el proceso de Log In de tu cuenta. (ej: micorreo@gmail.com)
- Password.- Que usarás para entrar en tu cuenta. (ej: 12345)

Configurar la parte del emisor

Con todo lo necesario instalado, conectado a la red y enchufado a la electricidad, vamos a comenzar con la configuración de la parte del **emisor**.

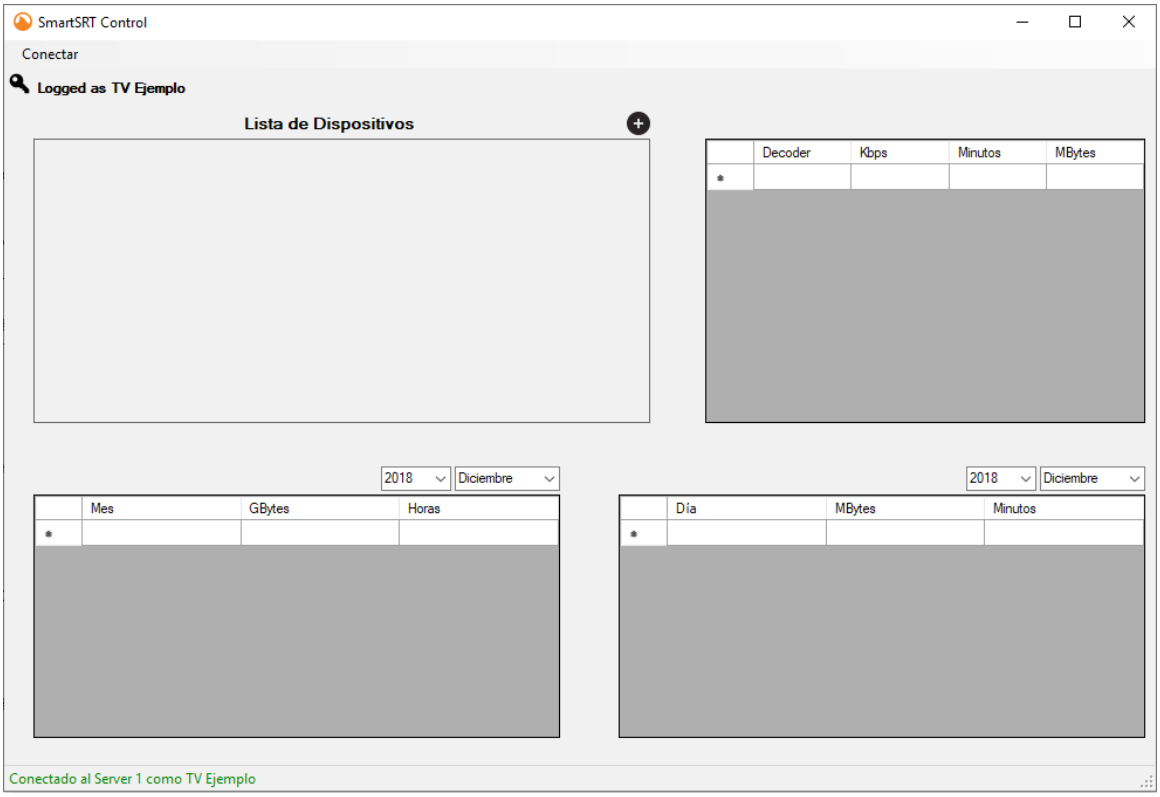
1) Lanzamos la aplicación **SmartSRT**.

Aparecerá la ventana totalmente vacía.



Pulsamos en el menú “Conectar”, y luego en “Servidor SmartSRT”. Rellenamos los datos del alta.

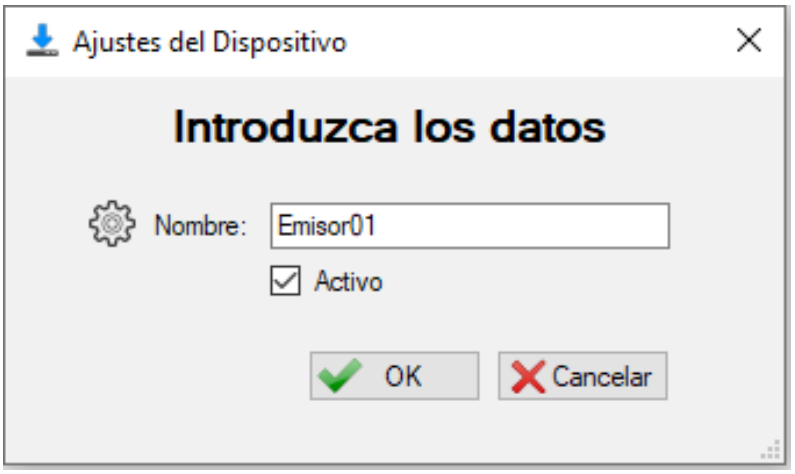
La ventana se llena con varios bloques:



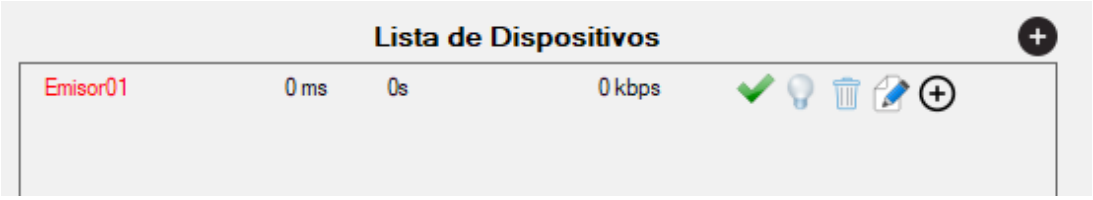
Si nos logeamos correctamente, y tenemos conexión directa con el Server, en la barra de estado inferior aparecerá un mensaje en **verde** indicándolo. En caso contrario aparecerá un mensaje en **rojo** con el error.

2) Dar de alta el dispositivo emisor.

En el bloque con la cabecera “Lista de Dispositivos”, en la parte superior derecha, hay un icono en forma de círculo negro y una cruz blanca dentro. Si situamos el cursor del mouse sobre él, nos aparece un cartel con su función “Añadir un Nuevo Emisor”. Pulsamos sobre él, y nos aparece una ventana emergente para rellenar sus datos.

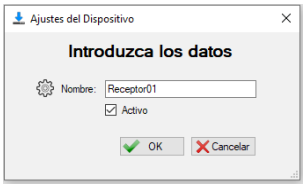


Puedes ponerle el nombre que desees, y que te sea mas sencillo de identificar. Si luego quieres cambiarlo, se puede hacer fácilmente Editando este Equipo.
Aparecerá el nuevo equipo dado de alta en el Listado de Dispositivos.

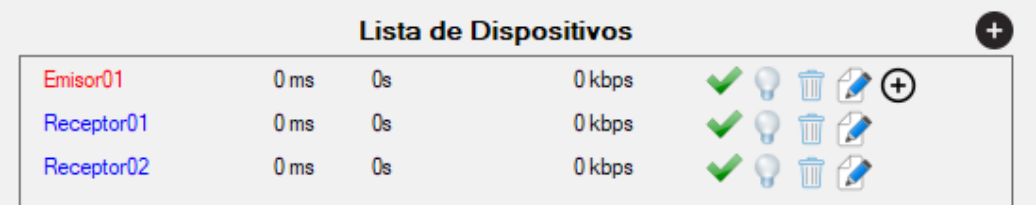


Si situamos el cursor del mouse sobre los diferentes elementos, aparecerá un mensaje que identifica lo que es, y si el cursor cambia de forma a una mano, indicará que ese elemento es un botón activo, que permite cambiar cosas de ese dispositivo.

Vamos a describirlos de izquierda a derecha, pasando el cursor del ratón sobre ellos:
- Copiar encoder smart url: Aparece al pasar sobre el nombre del equipo. Es un botón activo, que si pulsamos sobre él, no parece hacer nada. Nos copia en el portapapeles la dirección **smart url** que vamos a necesitar copiar en el Raspberry Pi que actuará como **emisor**.
- Buffer delay: Nos indica con un número entero seguido de ms, los milisegundos máximos de delay, con que este dispositivo se ha conectado al servidor SmartSRT.
- Tiempo conectado: Nos indica el tiempo ininterrumpido que el dispositivo lleva conectado al servidor SmartSRT. Nos lo indica en días, horas, minutos y segundos (ej: 3d:21h:34m:15s).
- Bitrate en Kbps: Nos muestra el bitrate calculado por el multiplexor local. Es una medida orientativa e inexacta, que con los minutos se va ajustando a su valor real.
- Desactivar/Activar equipo: Este elemento es un botón activo, que cuando esta en forma de icono check **verde**, nos permite desactivar el dispositivo, y si esta en forma de aspa **roja**, nos permite re-activarlo. Antes de realizar el cambio, una ventana emergente nos preguntará si estamos seguros de ello. Un equipo desactivado no podrá conectarse con el servidor SmartSRT mientras esté en ese estado, lo que nos puede servir para controlar el acceso de cada dispositivo, a la red de distribución en tiempo real. Si desactivamos un emisor, todos los receptores conectados al mismo también se desactivarán.
- Equipo conectado/desconectado: Con la forma de una bombilla que se enciende o se apaga, este indicador es muy claro sobre el estado del equipo emisor o receptor en cada momento.
- Borrar equipo: Es un botón activo que nos permite borrar, de manera definitiva, cualquier equipo de la red. El borrado es definitivo, por lo que nos aparecerá antes una ventana para que reafirmemos nuestra intención. Si borramos un emisor que tiene receptores, todos los receptores que tiene asignados, también se borrarán definitivamente. Maneje este botón con cuidado. Es preferible usar el botón de desactivar explicado anteriormente, a tener que lamentar y volver a configurar desde cero los equipos borrados, ya que las direcciones URL smart se generan aleatoriamente, y nunca son iguales.
- Editar equipo: Es un botón activo que muestra una ventana donde podemos editar el estado del equipo y su nombre. Podemos cambiar estas cosas en cualquier momento cuando deseemos.
- Añadir un Nuevo Receptor: Es un botón activo en forma de círculo con una cruz negra interior, que solo aparece a la derecha de los emisores, y permite dar de alta receptores que se conectarán a la señal de este emisor. La ventana y proceso de alta, es igual a la del receptor. Hagamos click en él, y vamos a añadir 2 receptores a este emisor.



Al final del proceso nos quedará algo como esto.



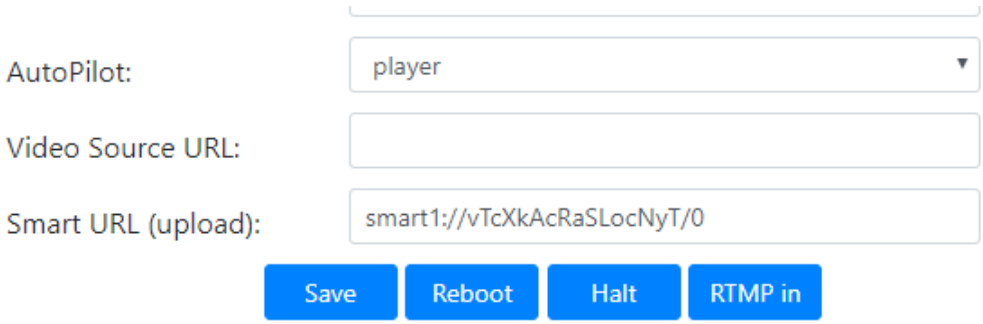
Podemos añadir emisores y receptores, conforme a la necesidad de nuestra red de distribución. Aparecerán en primer lugar el emisor y debajo todos sus receptores asociados. De un solo vistazo podremos controlar en tiempo real el estado de nuestra red completa.

Podemos añadir, cambiar y quitar elementos de la red en pleno funcionamiento, sin ningún problema.

3) Configurar el Raspberry Pi emisor.

Usando nuestro navegador web favorito, podemos entrar en el panel de control del Raspberry Pi conforme esta descrito en el manual de su software, que puedes bajarte de nuestra web. Entramos en la pestaña LAN donde nos centraremos en los 2 campos inferiores.

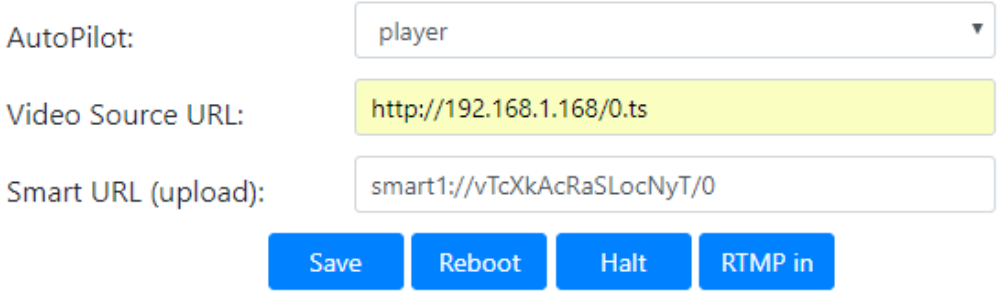
En el campo **Smart URL (upload)** pegaremos del portapapeles el "**encoder smart url**", que se nos copió al pulsar sobre el nombre del **emisor** en la **Lista de Dispositivos** del **SmartSRT Control**.



El contenido del campo **Video Source URL**, va a depender del **encoder** que vamos a usar como fuente de streaming.

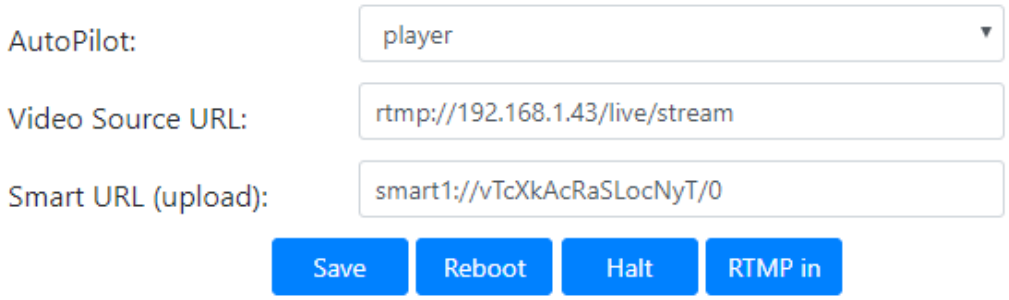
A) Opción del encoder Hi3516A.

Vamos a suponer que lo dejamos en su dirección IP de fábrica que es 192.168.1.168, y que lo hemos configurado para emitir por HTTP. Veremos los detalles en un apartado específico al respecto.
Este sería el contenido del campo **Video Source URL**.



B) Opción del software de streaming RTMP.

Para esta opción pulsamos sobre el botón inferior derecho **RTMP in**. Se nos generará automáticamente la URL RTMP a usar. En nuestro ejemplo sería algo como esto.



Sea una opción o la otra, ya podemos pulsar el botón **Save**. Entonces el **emisor** quedará ya activado, y listo para emitir una vez que el **encoder** esté iniciado. Vamos por tanto a configurar el **encoder**.

4) Configurar el encoder.

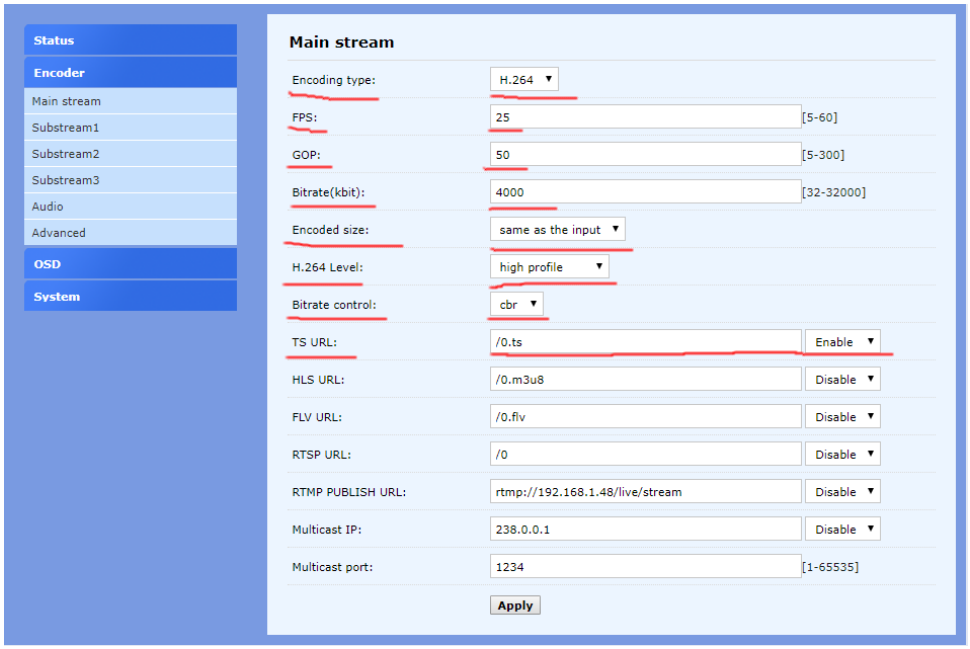
Aquí tenemos 2 posibilidades.

A) Opción del encoder Hi3516A.

Usando el navegador web nos metemos en su panel de configuración, en nuestro ejemplo http://192.168.1.168

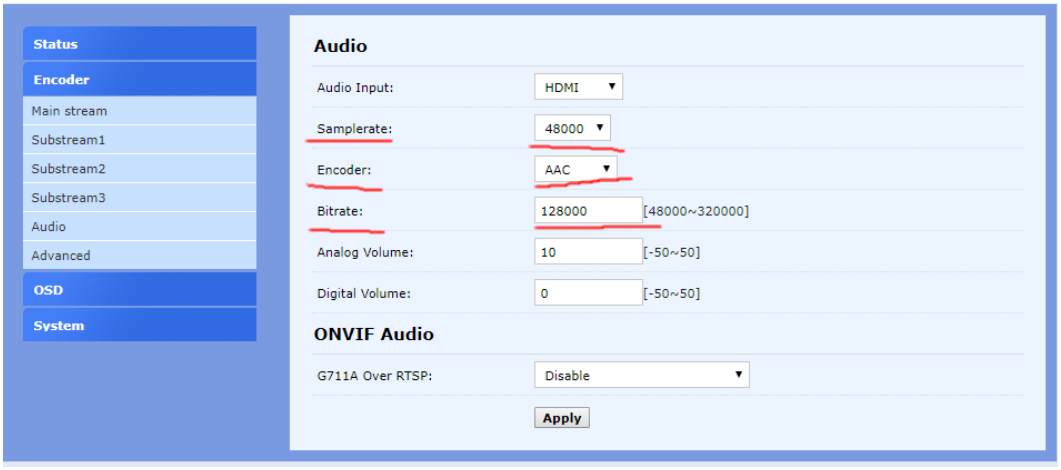
Esta es la configuración que más nos gusta para una fuente 1080i50 en H.264 (resaltamos en **rojo** los valores a ajustar):

Panel Encoder- Main stream:

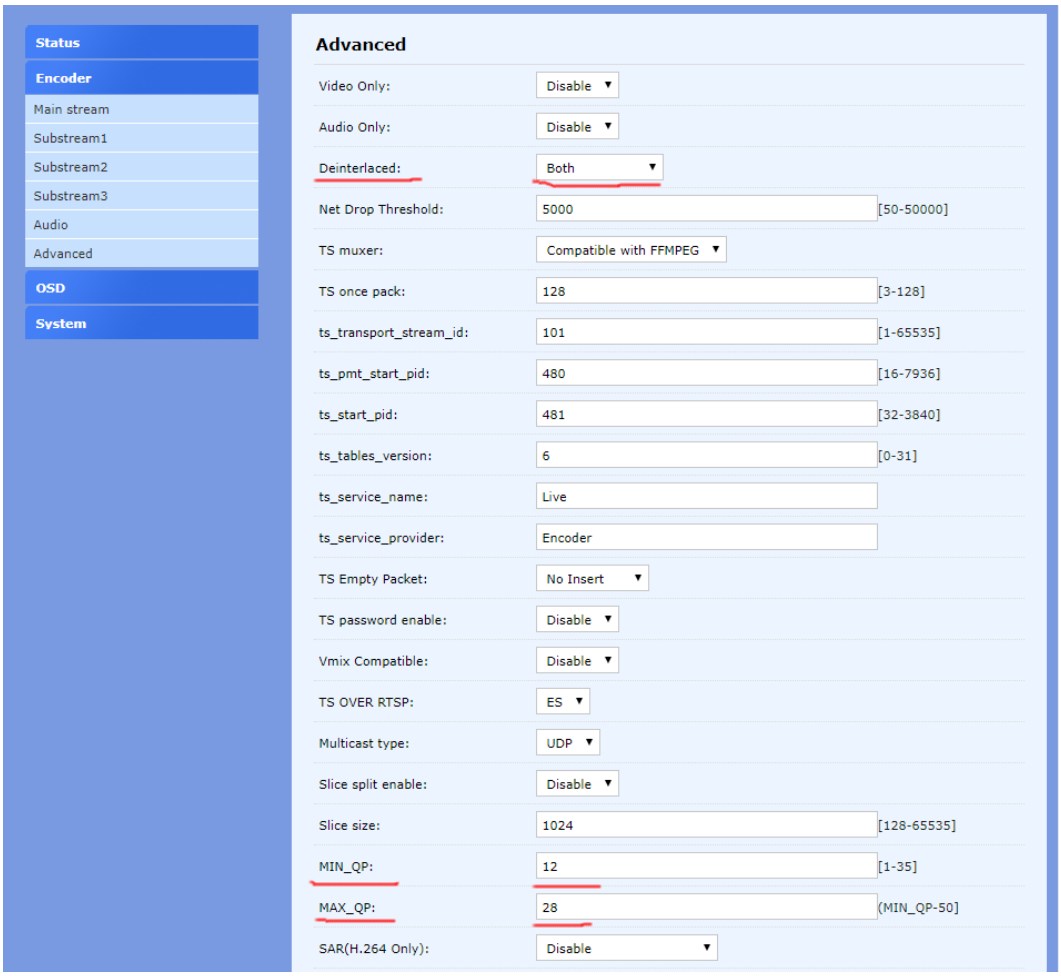


Los FPS los detecta automáticamente el encoder de la fuente de video. Aconsejamos un valor de GOP del doble del FPS, para una buena calidad. El bitrate en una fuente 1080i50 aconsejable es de 4000 kbps. También es importante usar en Bitrate control el valor CBR (constant bitrate), que ajusta el bitrate a la suma del bitrate de video + el de audio que veremos después, y usa siempre el mismo ancho de banda en cualquier situación.

Panel Audio:



Panel Advanced:



Deinterlaced = Both, hace que capture los campos de video en el orden original de la fuente.
Los valores de QP minimo y máximo restringen la calidad de la compresión final.

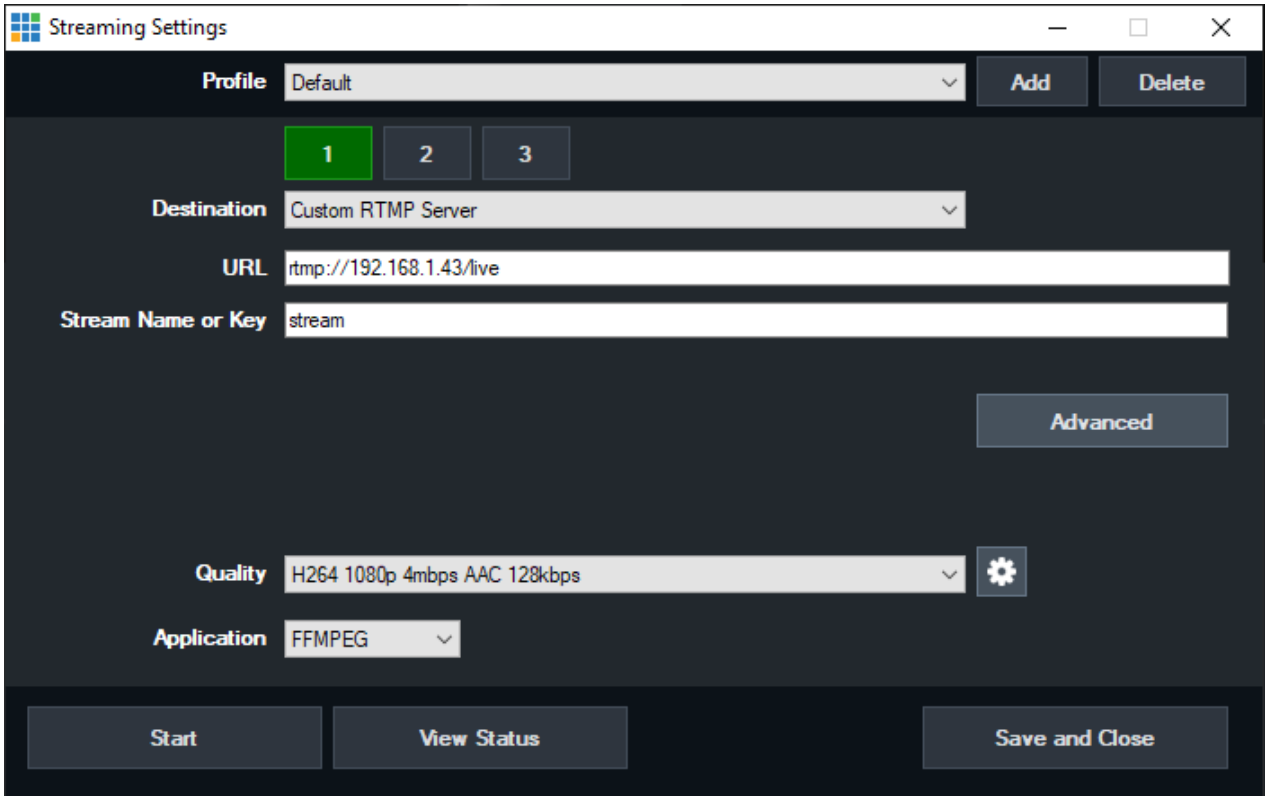
Con estos ajustes, la emisión local se podrá recibir de la URL http://192.168.1.168/0.ts que hemos usado en el paso 3 opción A, para configurar el Raspberry Pi **emisor**.

B) Opción del software de streaming RTMP.

Como todos los softwares de streaming RTMP son similares, vamos a usar como ejemplo el VMix.

En el paso 3 opción B obtuvimos al pulsar el botón RTMP en Video Source la URL siguiente:
rtmp://192.168.1.43/live/stream

Con ese valor, como ejemplo, vamos a rellenar los campos del VMIX (Streaming Settings).



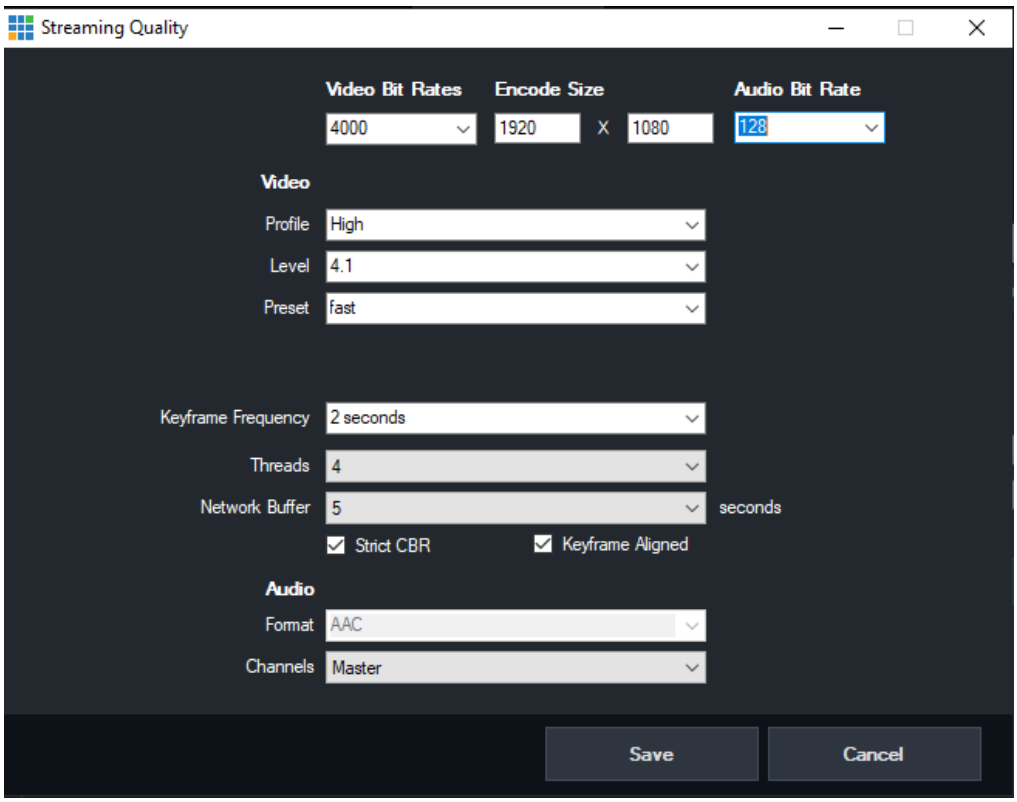
Una vez ajustados los valores de Quality (ponemos un ejemplo más adelante). Al pulsar Start se comienza a enviar señal por RTMP localmente desde el VMIX, directo al Raspberry Pi **emisor**, el cual recogerá esa señal y la convertirá en SRT para enviarla al servidor SmartSRT.

La primera parte de la Video Source URL, es lo que en este tipo de softwares de streaming RTMP se conoce como FMS URL o simplemente URL (en nuestro caso, *rtmp://192.168.1.43/live*).

La segunda parte de la Video Source URL, es lo que se llama en esos softwares Stream Name, Key o clave de emisión (en nuestro caso, *stream*).

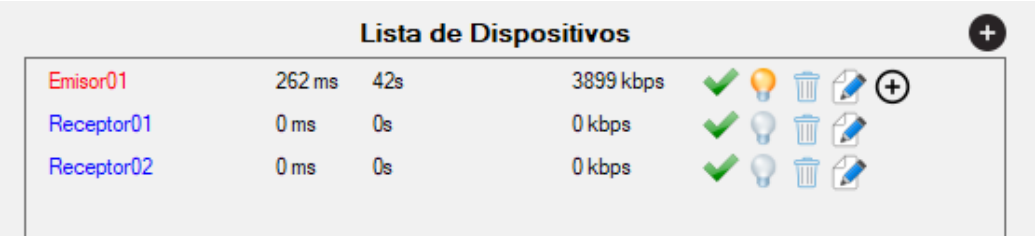
Antes de emitir, es conveniente ajustar la calidad de la emisión.

En la parte de Quality, pulsando sobre la rueda de Settings, podemos ajustar los valores del encoding. Esto es un ejemplo para H.264, con fuente de video 1080i50.



Recomendamos usar un **encoder** hardware antes que ningún software, por su mayor estabilidad y menor latencia.

Una vez el Raspberry Pi recibe señal del **encoder** local, negociará la conexión con el SmartSRT server, la clave AES de cifrado a usar, y empezará a enviar la señal. Entonces podremos ver en el **SmartSRT Control** algo como esto.

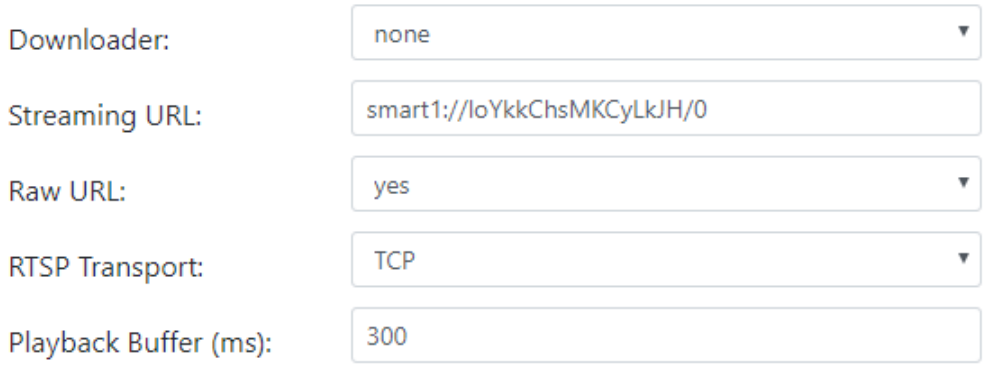


La linea correspondiente al emisor empieza a actualizar cada segundo los valores del tiempo conectado, del bitrate y también el valor del buffer delay usado. Además, la bombilla encendida nos dice que dicho dispositivo esta conectado y enviando datos.

5) Configurar un receptor

En el paso 2, cuando creamos la red de distribución, vimos que haciendo clic sobre el nombre del dispositivo, en nuestro caso, sobre el receptor que queremos configurar, se nos copia en el portapapeles la direccion URL SmartSRT del equipo.

Dicha dirección es la que vamos a pegar en el panel de configuración del Raspberry Pi **receptor**. Concretamente en la pestaña **Player**, en el campo **Streaming URL**.



Es importante que para usar el protocolo SRT, pongamos los campos Downloader en none, Raw URL en yes, el contenido del campo RTSP Transport aquí es completamente indiferente. Finalmente en Playback Buffer, recomendamos usar valores desde los 300 ms a los 1000 ms. Este valor afecta a la latencia total solo en la cantidad que pongamos, pero en redes con un PING muy cambiante (jittering), es mejor usar valores seguros (mayor o igual a 700 ms) para evitar Drops.

El resto de campos, son indiferentes al tema que abordamos en este manual.

Una vez terminada la configuración, pulsamos el botón **Save** para guardar, y luego el **Play**. Tras lo que veremos, en la **Lista de Dispositivos** del **SmartSRT Control**, como los números del dispositivo conectado van cambiando.

De igual manera, se pueden programar y conectar todos los receptores que necesitemos en nuestra red de distribución.

Latencia, búfferes y estabilidad

Es importante conocer los parámetros que determinan la latencia total de cada receptor con respecto a la señal original, y como influyen también en la estabilidad.

SRT es el protocolo que usan todos los dispositivos que se conectan con el servidor SmartSRT a través del Internet público. Este protocolo, aunque esta basado en UDP, tiene un mecanismo avanzado para recuperar paquetes perdidos, confiriéndole una confiabilidad similar a TCP.

Por defecto, el sistema calcula automáticamente los búfferes delay, tanto del emisor como de los receptores, y solamente podemos cambiar el Buffer Playback del receptor tal como hemos visto en el apartado anterior. Aunque este último valor no se refiere a la transmisión SRT en sí, si no a la adaptabilidad del Player del receptor a las fluctuaciones de los timestamps, debidos a cambios bruscos en el PING (jittering). Redes muy estables pueden usar valores de 300 ms, y redes menos estables es conveniente que usen valores de 700 ms o superiores. Aunque el Buffer Playback puede ponerse hasta valores de 5000 ms, con SRT no es necesario superar los 1000 ms.

Suponiendo que trabajamos con un encoder Hi3516A, la latencia total en un receptor concreto sería la suma de los siguientes factores:

- Buffer delay del emisor (aparece en la app Smart Control, en Lista de Dispositivos) (ej: 262ms)
- Buffer delay del receptor (aparece en el mismo sitio) (ej: 289ms)
- Playback Buffer del RPI (visto en el apartado anterior) (ej: 300ms)
- Conversor de protocolos + multiplexor del SmartSRT server = 700 ms.
- Latencia del encoder (al comprimir). Un encoder hardware Hi3516A tiene 300 ms.

262 ms + 289 ms + 300 ms + 700 ms + 300 ms = 1851 ms (1.85 segundos máximo)

En nuestro ejemplo nos sale que la latencia máxima del **Receptor01** es de 1.85 segundos. Sin embargo con las condiciones reales de la red mientras hacíamos este ejemplo, nos ha dado un retardo real de 1.40 segundos.

SRT es un protocolo de baja latencia, donde los valores de los buffer delay van desde 50 ms hasta 5000 ms, dependiendo del PING y de la pérdida de paquetes. Pero aunque el sistema por defecto calcule dichos bufferes, existe una forma de forzarlos a un valor concreto, si por ejemplo queremos bajarlo, porque no nos importa que aparezcan artefactos en el video de vez en cuando, y nos interese tener una menor latencia, o por si al contrario queremos subir su valor, porque nos interesa mucho más la estabilidad de la emisión a largo plazo.

Para forzar esos valores, es necesario que cambiemos el número entero que aparece al final de la URL tipo Smart, que por defecto es un 0 (buffer automático).

En nuestro ejemplo esa URL era: **smart1://IoYkkChsMKCyLkJH/0**
Si por ejemplo queremos usar un valor de 50 ms, la cambiaríamos a:
smart1://IoYkkChsMKCyLkJH/50

Si ponemos un valor fuera del rango permitido, entre 50 y 5000, el sistema pondrá el valor límite.

Nuestra implementación de SRT, da por finalizada la comunicación entre 2 partes, solamente cuando no hay transmisión de ningún paquete durante al menos 3 segundos seguidos. Y al funcionar

sobre UDP, solo depende de los bufferes y de las condiciones de la red para recuperar paquetes perdidos. En caso de no poder hacerlo, no cortará la conexión, si no que simplemente se verán artefactos en la recepción.

En caso de experimentar artefactos indeseados en la recepción, recomendamos elevar el buffer al doble del valor creado automáticamente. Usar un buffer el doble del automático, nos permite ser inmunes a tasas de pérdidas de datos de hasta un 10%, condiciones que dejarían fuera de juego a cualquier otro protocolo de streaming.

A continuación vamos a exponer una tabla de PING de ejemplo, junto con valores de bufferes y la latencia máxima final, como valores de referencia.

PING (ms)	Buffer emisor	Buffer receptor	Buffer Player	Latencia máxima
33	100	100	300	1.5 s
166	500	500	500	2.5 s
230	750	750	500	3 s
500	1500	1500	500	4.5 s

Como observación final, añadir que los búfferes dentro de una misma red no estan relacionados, y no tienen porqué ser iguales, ya que cada dispositivo se encontrará en unas condiciones de red diferentes a los demás, y a una distancia diferente del servidor **SmartSRT**. Si todos los receptores dependientes de un mismo emisor experimentan artefactos, es muy posible que el problema provenga de la conexión del emisor con el servidor, por lo que el buffer del emisor es el que tendríamos que elevar al doble para evitar dicho inconveniente.