Wirecast SRT Support Guide

What Is SRT?

Secure Reliable Transport (SRT) is an open source streaming protocol, originally developed by Haivision and promoted by the SRT Alliance, that optimizes streaming performance across unpredictable networks, such as the Internet.

Secure: Encrypts video streams
Reliable: Recovers from severe packet loss
Transport: Dynamically adapts to changing network conditions

The SRT protocol was designed specifically for live video in an effort to deliver high-quality, low-latency video over the Internet with high reliability.

Who Should Use SRT?

If you are using Wirecast to record or stream to social networks – SRT is probably not for you!

None of the main streaming services support the SRT protocol for ingest. Most still use RTMP (Twitch, YouTube, Facebook...). If you're exclusively using these services, SRT is not for you.

The main category of users who would potentially be interested in SRT in Wirecast is the professional broadcast industry. The configuration of Wirecast for SRT is fairly easy. However, the server setup is more challenging since it requires system/network admin knowledge.

SRT User Guides:

The SRT Alliance has released some useful guides for SRT:

- SRT Explained in 75 Seconds - Quick video explaining SRT.
- The SRT Deployment Guide - This covers different network topologies, calling modes, configuring firewalls, and using statistics to configure settings for optimal performance.
- Haivision's Open-Source repository on GitHub also has some helpful explanations:
- SRT vs. UDT Protocols - Explains the similarities and differences between the SRT and UDT (Reliable UDP) protocols.

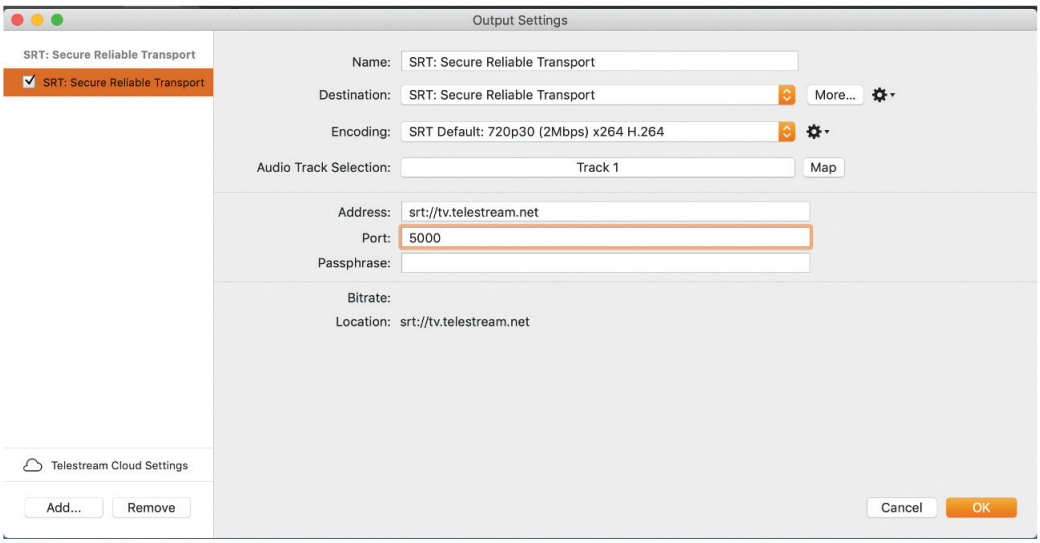
SRT in Wirecast

Calling Modes

SRT connections between products can be made using one of three modes.

- Caller A caller initiates a connection to a listener. The caller must know the public address and port of the listener.
- Listener A listener waits for a caller to initiate a connection on a certain port.
- Rendezvous When both products are in rendezvous mode, they can automatically negotiate a connection on a mutually agreed upon port. (NOTE: this is not related to Wirecast's Rendezvous peer-to-peer conferencing feature)

Wirecast supports Caller mode for both ingest and output. In order to work with another product, that product must support listener mode.



Output Settings

When you choose the "SRT: Secure Reliable Transport" destination, the output settings dialog will look like this

Encoding Presets

Only presets that rely solely on supported codec implementations (Video: x264 or Apple H.264, Audio: AAC) will be visible. All others will be hidden from the Encoding popup menu.

Address

You can specify the domain name or IPv4 address of the server. This is required.

Please note: IPV6 addresses are not supported at this time.

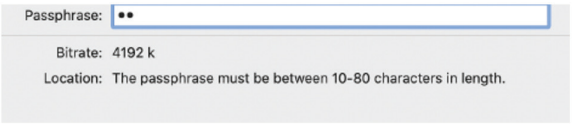
You do not need to include the "srt://" scheme, but you can. If the wrong scheme is entered, it will be corrected automatically in the Location field. The port should be specified in the Port field. If you specify the port in the address field using the colon delimiter (ex. 127.0.0.1:6789), it will be ignored.

Port

Enter the port that the SRT server is listening on. This is required.

Passphrase

If your server requires a passphrase, enter it here. Otherwise, leave it blank.



When present, SRT requires that the passphrase be at least 10 characters and no more than 80 characters. If you enter an invalid number of characters, you will see a message in the Location field explaining the requirements.

If your server does not require a passphrase, and you have specified one, you will see an error like the following when trying to connect:

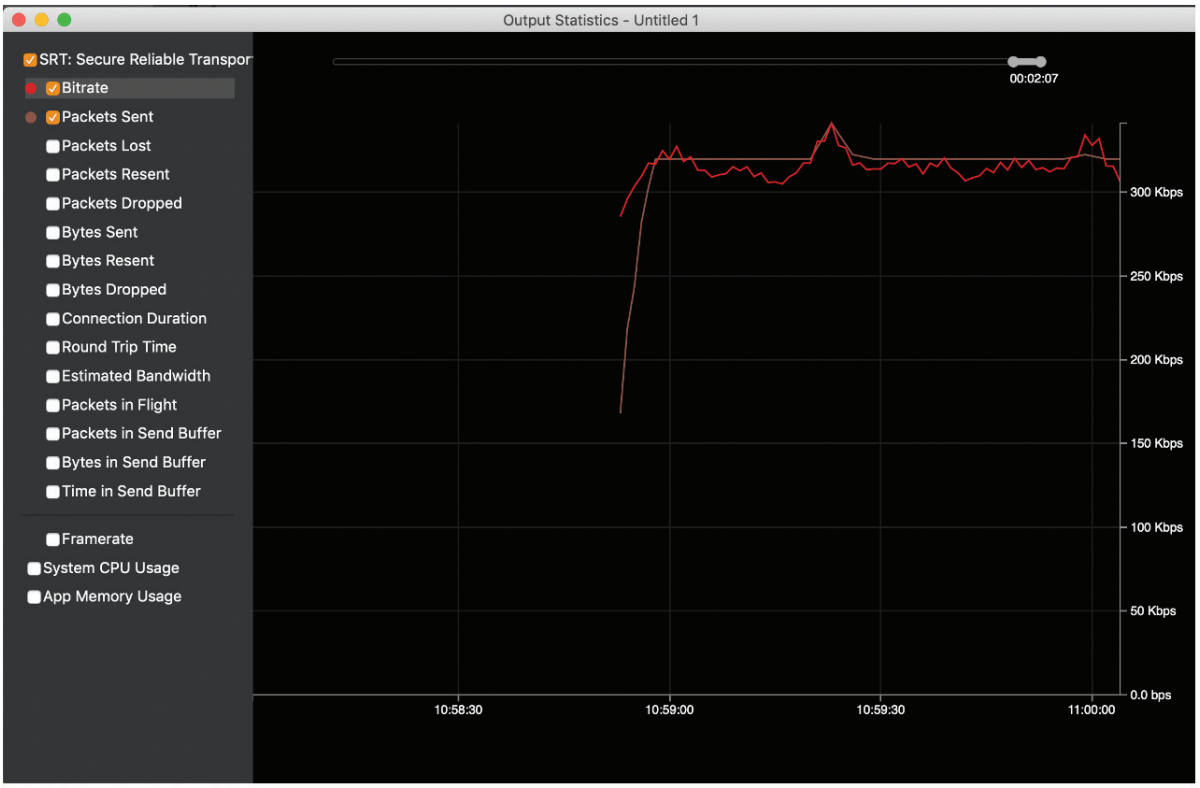
SRT: Connection to <IP>:<Port> was rejected because a password was required or unexpected.

You will also get the above error if the server requires a passphrase and you have omitted it.
If your server requires a passphrase and you have entered it incorrectly, you will see an error like this instead:

SRT: Connection to <IP>:<Port> was rejected because the password is wrong.

Location

The location field displays the complete SRT address that will be used to connect to the server. However, if there is a problem with the information entered, you may see an explanation of that problem here instead.



Output Statistics

You can access some useful statistics for monitoring connections in the Wirecast Output Statistics dialog, which is accessible by selecting “Show Statistics…” from the Output menu.

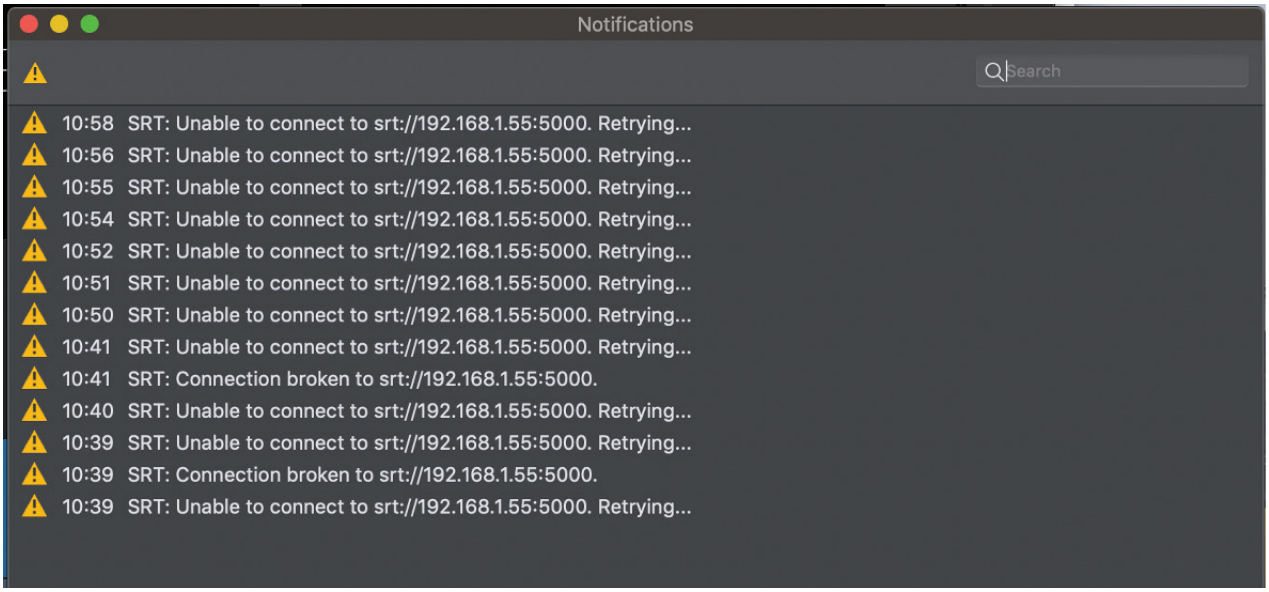
Wirecast's Output Statistics have a useful feature of being able to monitor multiple graphs simultaneously by checking multiple checkboxes in the list on the left. However, please note that the scale of the measurements (the white text along the right side of the window) is only shown for the currently selected (highlighted) statistic.

All of these statistics are generated by the SRT library. Complete documentation for them is available here.

- Bitrate This is the data transmission rate over time from Wirecast to the destination. It can be displayed in bits per second (bps), kilobits per second (Kbps), megabits per second (Mbps), gigabits per second (Gbps), or terabytes per second (Tbps). This is similar to the bitrate statistic for RTMP destinations.
- Packets Sent This is the number of sent data packets, including retransmitted packets, over time.
- Packets Lost The number of data packets considered or reported as lost at the sender side. Does not correspond to the packets detected as lost at the receiver side. A packet is considered lost in two cases:
  1. Sender receives a loss report from a receiver,
  2. Sender initiates retransmission after not receiving an ACK packet for a certain timeout.
- Packets Resent The number of retransmitted packets sent by the sender.
- Packets Dropped The number of data packets dropped by the sender because they have no chance of being delivered in time.
- Bytes Sent The number of bytes sent, including those for retransmitted packets, and including the payload and all of the headers.
- Bytes Resent The number of bytes retransmitted by the sender, including the payload and all of the headers.
- Bytes Dropped The number of bytes in the data packets dropped by the sender because they have no chance of being delivered in time, including the payload and all of the headers.
- Connection Duration The time elapsed, in milliseconds, since the current SRT socket was connected. This graph shows when disconnects and reconnects have occurred.
- Round Trip Time (RTT) This is the round trip time (RTT), in milliseconds calculated by the SRT library. The value is calculated by the receiver based on the incoming ACKACK control packets (used by sender to acknowledge ACKs from receiver). The RTT (Round-Trip time) is the sum of two STT (Single-Trip time) values, one from agent to peer, and one from peer to agent. Note that the measurement method is different than in TCP. SRT measures only the "reverse RTT", that is, the time measured at the receiver between sending a UMSG_ACK message until receiving the sender's UMSG_ACKACK response message (with the same journal). This happens to be a little different from the "forward RTT" measured in TCP, which is the time between sending a data packet of a particular sequence number and receiving UMSG_ACK with a sequence number that is later by 1. Forward RTT isn't being measured or reported in SRT, although some research works have shown that these values, even though they should be the same, happen to differ; "reverse RTT" seems to be more optimistic.
- Estimated Bandwidth Estimated bandwidth of the network link, in Mbps. The bandwidth is estimated at the receiver. The estimation is based on the time between two probing DATA packets. Every 16th data packet is sent immediately after the previous data packet. By measuring the delay between probe packets on arrival, it is possible to estimate the maximum available transmission rate, which is interpreted as the bandwidth of the link. The receiver then sends back a running average calculation to the sender with an ACK message.
- Packets in Flight The number of packets that have been sent but not yet received. This is the distance between the packet sequence number that was last reported by an ACK message and the sequence number of the latest packet sent (at the moment when the statistics are being read). NOTE: ACKs are received periodically (at least every 10 ms). This value is most accurate just after receiving an ACK and becomes a little exaggerated over time until the next ACK arrives. This is because with a new packet sent, while the ACK number stays the same for a moment, the value of the packets in flight increases. But the exact number of packets arrived since the last ACK report is unknown.

- Packets in Send Buffer The number of packets in the sender's buffer that are already scheduled for sending or even possibly sent, but not yet acknowledged. Once the receiver acknowledges the receipt of a packet, or the TL packet drop is triggered, the packet is removed from the sender's buffer. Until this happens, the packet is considered as unacknowledged. This is similar to the RTMP Queue statistic for RTMP destinations. If this value falls too low, it suggests that the computer is unable to produce the video frames quickly enough.
- Bytes in Send Buffer Instantaneous (current) value of the number of bytes in the send buffer, including payload and all headers (20 bytes IPv4 + 8 bytes UDP + 16 bytes SRT). This is similar to the RTMP Queue statistic for RTMP destinations. If this value falls too low, it suggests that the computer is unable to produce the video frames quickly enough.
- Time in Send Buffer The timespan, in milliseconds, of unacknowledged packets in the sender's buffer. This is similar to the RTMP Queue statistic for RTMP destinations. If this value falls too low, it suggests that the computer is unable to produce the video frames quickly enough.

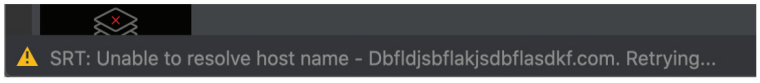
Error Messages



Wirecast's SRT implementation takes advantage of the Wirecast status bar and Notifications Window for providing errors and warnings to users. The Notifications Window can be opened by clicking on the Note Pad icon in the bottom left corner of the document window, or by selecting Notifications from the Window menu. Here are some common errors:

Unable To Resolve Domain Name

If Wirecast cannot get an IP address for the specified domain name, you will see a message like this:

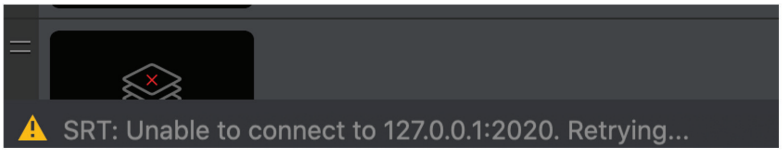


Wirecast will wait a second and try again in case the DNS server is not responding, the user changes the DNS server, or the DNS entry has been updated. You will see another entry if the next attempt fails as well. If this happens:

1. Make sure the address is entered correctly.
2. Try a different DNS server.
3. Specify the IP address instead of the domain name.

Unable To Connect

If Wirecast cannot communicate with the SRT server and the specified address and port, you will see a message like this:



Wirecast will continue trying to connect, but you will only see another error once every twenty attempts, in order to reduce clutter in the logs. If this occurs:

1. Make sure that the address and port are entered correctly.
2. Try pinging the address and port to confirm that it is reachable.
3. Try using a different SRT client. If it also fails, you have evidence that the problem is not related to Wirecast, but is either that the server is down, or it is not reachable on that network.

Connection Rejected Errors

If Wirecast is able to connect to the server, but the server rejects that connection, the SRT library will try to provide you with a reason why. Here are the possible errors for that case. These explanations come directly from the SRT documentation.

- ***SRT: Connection to <IP>:<Port> was broken due to system function error.*** One of the system functions reported a failure. Usually this means some system error or lack of system resources to complete the task.
- ***SRT: Connection to <IP>:<Port> was rejected by peer.*** The connection has been rejected by peer, but no further details are available. This usually means that the peer doesn't support rejection reason reporting.
- ***SRT: Connection to <IP>:<Port> was rejected due to internal problem with resource al-location.*** A problem with resource allocation (usually memory).
- ***SRT: Connection to <IP>:<Port> was rejected due to incorrect data in handshake.*** The data sent by one party to another cannot be properly interpreted. This should not happen during normal usage, unless it's a bug, or some weird events are happening on the network.
- ***SRT: Connection to <IP>:<Port> was rejected because the listener's backlog was exceeded.*** The listener's backlog has exceeded (there are many other callers waiting for the opportunity of being connected and wait in the queue, which has reached its limit).
- ***SRT: Connection to <IP>:<Port> was rejected due to an internal program error.*** Internal Program Error. This should not happen during normal usage and it usually means a bug in the software.
- ***SRT: Connection to <IP>:<Port> was rejected because the socket is closing.*** The listener socket was able to receive your request, but at this moment it is being closed. It's likely that your next attempt will result with timeout.
- ***SRT: Connection to <IP>:<Port> was rejected because the peer version is older than minimum.*** Any party of the connection has set up minimum version that is required for that connection, and the other party didn't satisfy this requirement.
- ***SRT: Connection to <IP>:<Port> was rejected due to a rendezvous cookie collision.*** Rendezvous cookie collision. This normally should never happen, or the probability that this will really happen is negligible. However, this can be also a result of a misconfiguration that you are trying to make a rendezvous connection where both parties try to bind to the same IP address, or both are local addresses of the same host - in which case the sent handshake packets are returning to the same host as if they were sent by the peer, who is this party itself. When this happens, this reject reason will be reported by every attempt.
- ***SRT: Connection to <IP>:<Port> was rejected because the password is wrong.*** Both parties have defined a passphrase for connection and they differ.

- ***SRT: Connection to <IP>:<Port> was rejected because a password was required or unexpected.*** Only one connection party has set up a password.
- ***SRT: Connection to <IP>:<Port> was rejected due to an API collision.*** The value for SRTO_MESSAGEAPI flag is different on both connection parties.
- ***SRT: Connection to <IP>:<Port> was rejected due to an incompatible congestion con-troller type.*** The SRTO_CONGESTION option has been set up differently on both connection parties.
- ***SRT: Connection to <IP>:<Port> was rejected due to an incompatible packet filter.*** The SRTO_PACKETFILTER option has been set differently on both connection parties.
- ***SRT: Connection to <IP>:<Port> was rejected due to an incompatible group.*** The group type or some group settings are incompatible for both connection parties. While every connection within a bonding group may have different target addresses, they should all designate the same endpoint and the same SRT application. If this condition isn't satisfied, then the peer will respond with a different peer group ID for the connection that is trying to contact a machine/application that is completely different from the existing connections in the bonding group.
- ***SRT: Connection to <IP>:<Port> was rejected due to a timeout.*** The connection wasn't rejected, but it timed out.
- ***SRT: Connection to <IP>:<Port> was rejected.*** The connection was rejected, but we do not know why.

Connection Broken

- ***SRT: Connection broken to <IP>:<Port>.*** This will occur if the SRT connection succeeded and was broadcasting, but the server has since severed the connection. This could be because the server process crashed, the server's computer has shutdown or rebooted, or the client's network connection has been severed.

Servers

Streaming via SRT requires a server that supports SRT to accept the video. Here are several options:

Wowza

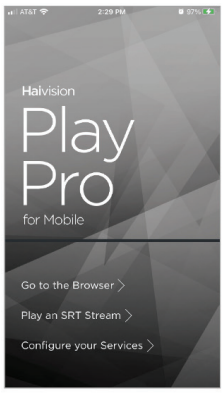
Wowza Streaming Engine supports ingesting SRT in 4.7.2 and later, but only on Linux or Windows. The Wowza Streaming Engine on Mac does not support this at this time.

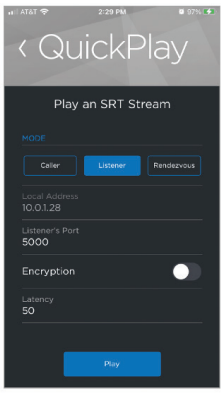
Setup

Wowza provides instructions for configuring the Wowza Streaming Engine to ingest SRT. Here is a YouTube video walkthrough. They also have instructions for distributing live streams via SRT.

Haivision's Play Pro

The fastest and easiest way to set up a server for receiving SRT is by using Haivision's Play Pro app for iOS. This can be downloaded via Apple's iOS App Store for free and run on an iPhone or iPad.





Limitations

Only runs on iOS.

Wirecast requires an IPv4 address (ex. 127.0.0.1) or domain name (ex. telestream.net) for specifying the SRT server. If your iPhone/iPad is on a cellular network, it will probably not provide an IPv4 address, and you will not be able to use that network to test SRT with Wirecast.

Wirecast does not support IPv6, so if your iPhone/iPad is on a network that hands out IPv6 addresses (ex. 2607:f0d0:1002:51::4 or 2607:f0d0:1002:0051:0000:0000:0000:0004), you will not be able to use that network to test SRT with Wirecast.

Setup

If your iPhone/iPad is connected to a WiFi network, then in just a few steps, you can have a server running and able to receive an SRT stream:

1. Launch Play Pro.
2. Tap “Play an SRT Stream.”
3. Select “Listener” and note the Local Address and Port that is displayed.
4. Tap Play button.
5. Enter the Local Address and Port into Wirecast and start streaming. You should now see the video on your iPhone/iPad.

Makito X
This is a hardware video encoder made by Haivision which supports SRT.

OBS
OBS Studio supports both ingest and output of SRT, so it can be used as either a client or a server. Here are instructions for setting it up, along with other useful information.

FAQ

What version of the SRT Protocol does Wirecast support?
Wirecast 14.1 uses version 1.4.1 of the open-source SRT library. SRT is designed to be backwards compatible, so servers that use an older version of SRT may still work. However, some features might not be available.

If you try to connect to a server that has an incompatible version of SRT, you will see an error like the following:

SRT: Connection to <IP>:<Port> was rejected because the peer version is older than minimum.

What codecs and file type are being transmitted via SRT?
The SRT destination only supports H.264 video and/or AAC audio in an MPEG-2 Transport Stream container.

What version of the Wowza Streaming Engine do you recommend for ingesting SRT?
While SRT ingest is supported in Linux and Windows versions of Wowza Streaming Engine 4.7.2 and later, we recommend Wowza Streaming Engine 4.8 or later as it includes several pertinent bug fixes.

Which H.264 codec implementations are supported?
On Mac, you can use either x264 or Apple’s VideoToolbox H.264. On Windows, only x264 is supported.
Can I use the NVIDIA NVENC H.264 encoder with SRT?
Not currently. We have received requests for this and we are investigating.

Does Wirecast allow you to configure SRT settings, such as latency?
Not at this time. However, many SRT settings can also be configured from the server side.

Can I put Wirecast into listening mode for SRT?
Not at this time. If this is important, please contact Support with a Feature Request and include a description of the workflow and related software.

Can I put Wirecast into rendezvous mode for SRT?
Not at this time. If this is important, please contact Support with a Feature Request and include a description of the workflow and related software.

How do I configure my firewall for SRT?
The SRT Alliance’s Deployment Guide has explanations of this.

Can I use an IPv6 address for SRT?
Not at this time.

telestream
www.telestream.net | info@telestream.net | tel +1 530 470 1300