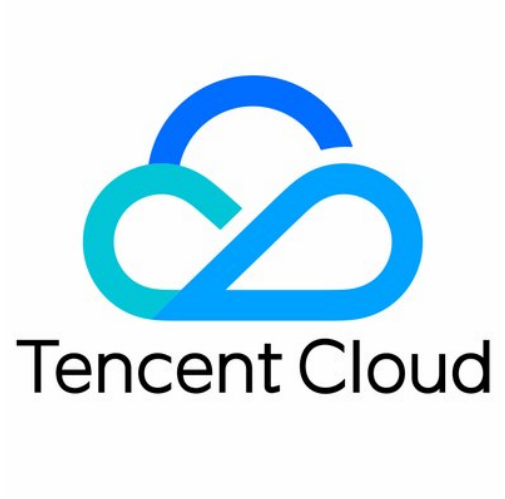
Cloud Streaming Services

Live Event Broadcasting (LEB)

Product Documentation



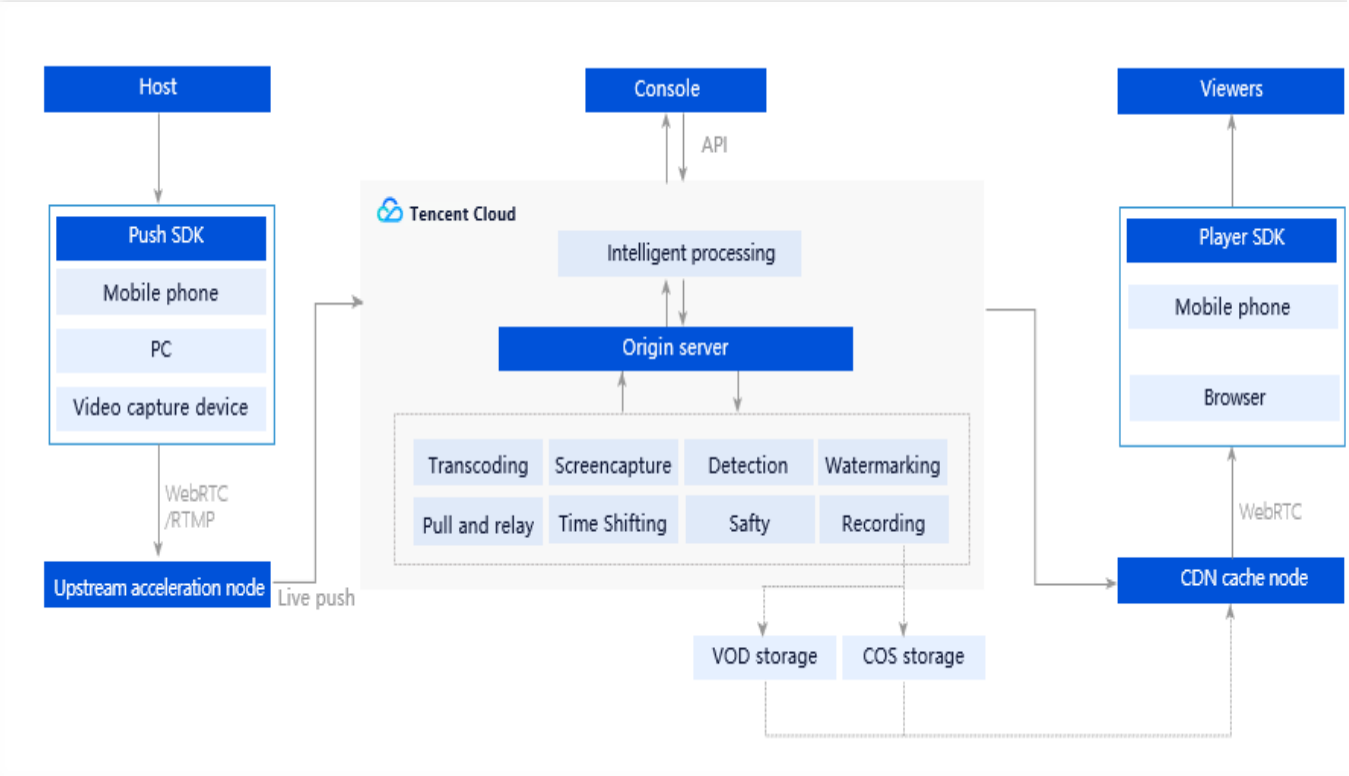
Contents

Live Event Broadcasting (LEB) Overview

Last updated：2024-11-08 16:03:46

Live Event Broadcasting (LEB) is the ultra-low-latency version of CSS. It delivers superior playback experience with millisecond latency and is suitable for scenarios with high requirements on latency, such as online education, sports streaming, and online quizzes.

Product Architecture



Features

Playback with millisecond latency

LEB uses the UDP protocol to keep the latency within 1s, much lower than 3-5s in traditional live streaming. This, along with excellent instant streaming performance and low stutter rate, guarantees a superior streaming experience.

Various features and smooth migration

LEB integrates a wide range of LVB features including live push, transcoding, recording, screencapture, porn detection, and playback. It also allows smooth migration from LVB.

User-friendly, secure, and reliable

You can easily integrate LEB as it uses standard protocols. You can use it for playback on Chrome and Safari without installing any plugins. Moreover, its protocols encrypt streams by default for improved security and reliability.

Pricing

LEB billable items include basic services and value-added services. Basic services are billed by upstream and downstream traffic/bandwidth, and value-added services such as live transcoding, recording, screencapture, and porn detection are billed by resource usage. For details, see Billing Overview.

Getting Started

For details about how to use the demo and integrate LEB, see Getting Started.

LEB Versus LVB

Last updated：2024-10-24 15:43:12

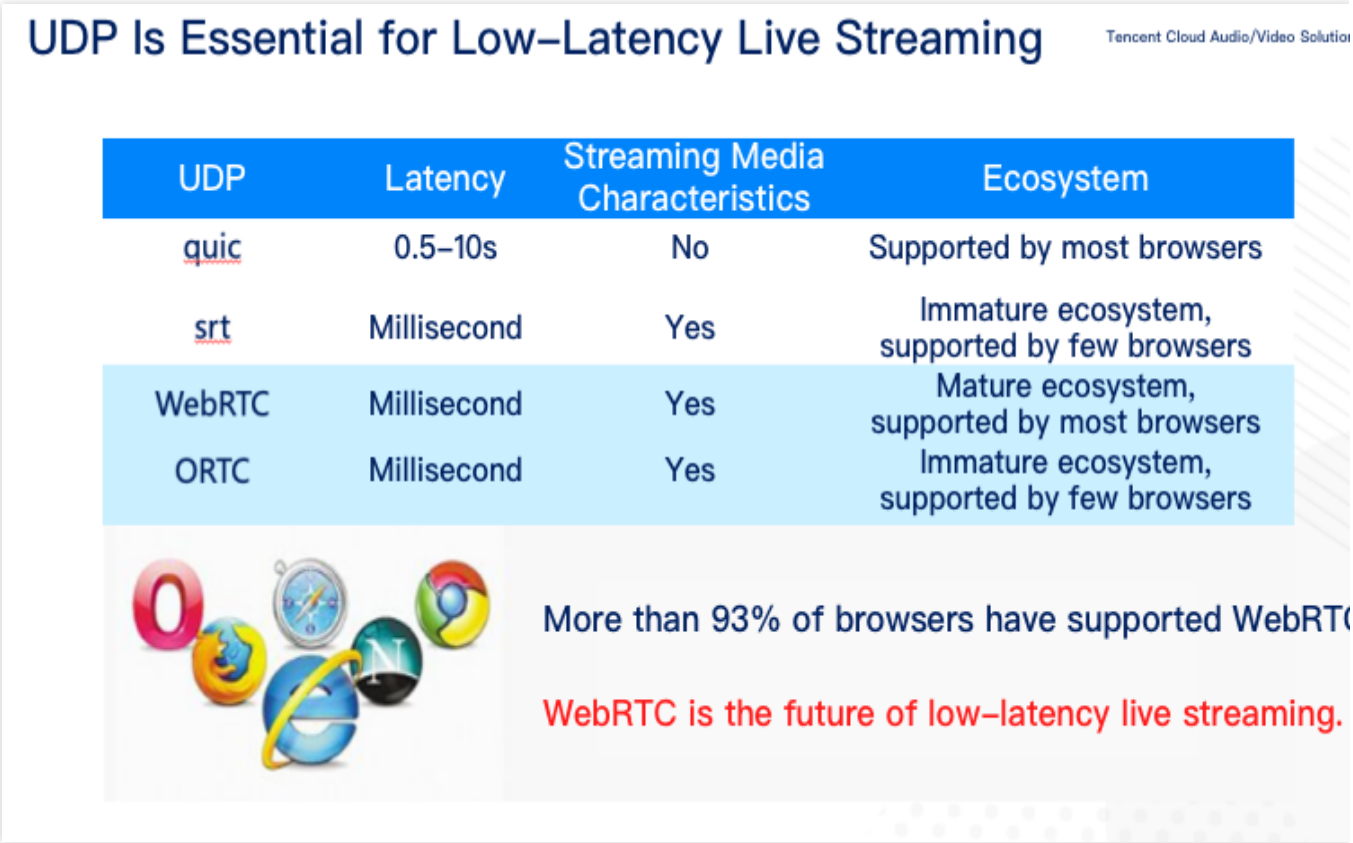
As a lower-latency version of LVB, LEB provides superb live streaming experience with millisecond playback latency, far lower than that of live stream playback using traditional protocols. LEB is designed for scenarios with high latency requirements. In addition to live shopping and online education, it is also suitable for interactive scenarios such as live sports streaming and live game streaming.

Advantages	Description
Millisecond-level ultra-low latency playback	By adopting the UDP protocol, millisecond-level latency live streaming capability is achieved in high-concurrency scenarios, improving the traditional live streaming drawback of 3-5 seconds latency. At the same time, it takes into account core indicators such as instant start and lag rate, providing users with an ultimate ultra-low latency live streaming experience.
Comprehensive features, smooth compatibility	Compatible with all standard live streaming features, including push, transcoding, recording, screenshot, content moderation, and playback, it supports customers to smoothly migrate from existing standard live streaming services.
Wide coverage of acceleration nodes and high bandwidth capacity	Currently, Live Event Broadcasting (LEB) has super acceleration nodes with global distribution and extensive coverage (supporting 2000+ nodes and 25 countries), capable of supporting 100T+ bandwidth.
Easy to use	Integration is straightforward, requiring no additional plugins. It can span multiple platforms, supporting a variety of operating systems and devices.
Excellent network resilience	In various weak network environments (such as high packet loss and high latency), Live Event Broadcasting (LEB) can still ensure high-quality video streams, providing users with a more stable live streaming experience.
Web low-latency support	Currently, CDN live streaming only supports HLS format streams on the web, but this format has a playback latency of several seconds. Live Event Broadcasting (LEB) can also support web playback with only a few hundred milliseconds of latency.
Smooth transition between multiple bitrates	Seamlessly switch between transcoded streams with different bitrates without any interruptions or jumps, ensuring a smooth transition in both visual and auditory experiences.
Adaptive Bitrate Control	Adaptively switch between different bitstreams according to network bandwidth, ensuring a smooth playback experience during varying network conditions.

Compared to Standard Live Video Broadcasting (LVB), Live Event Broadcasting (LEB) uses the WebRTC protocol during the playback process, thus it has a lower playback latency than Standard LVB. The following will specifically compare the differences between LEB.

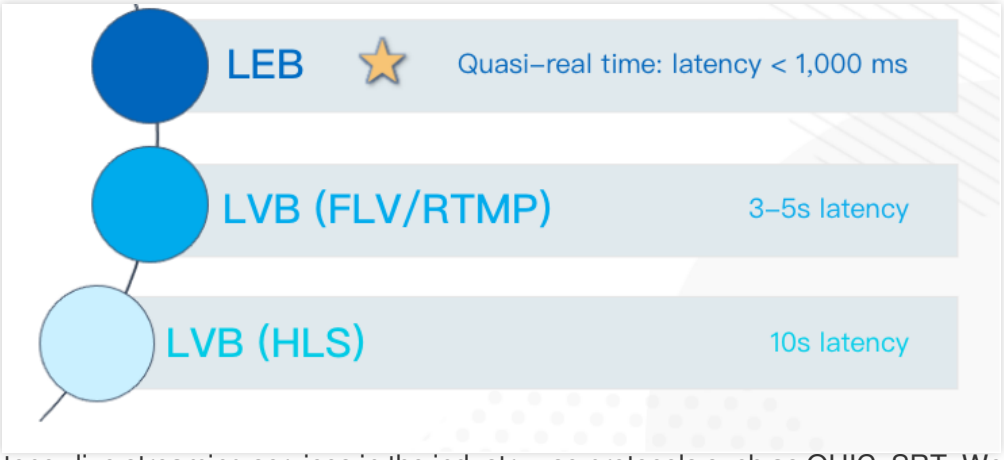
Protocol Comparison

Currently, Live Video Broadcasting (LVB) uses common playback protocols such as RTMP, FLV, and HLS. The common feature of these protocols is that they are all based on the TCP protocol. TCP has delayed acknowledgment and piggybacking, which means that it does not immediately respond with an ACK for each received data but waits for a certain amount of data before responding. This can lead to a perceived delay, and in weak network scenarios, this can even result in delays of several seconds or even tens of seconds.
In order to achieve lower latency in live streaming, faster and better-quality playback protocols are needed. Research shows that low-latency live streaming protocols in the industry include QUIC, SRT, WebRTC, and ORTC, all of which are based on the UDP protocol at the underlying level. Comparatively, QUIC has a higher latency because it does not have streaming features; SRT, WebRTC, and ORTC all have millisecond-level latency and streaming features. Among them, SRT and ORTC are less commonly used in the industry, while WebRTC is widely used and has a thriving technical ecosystem.

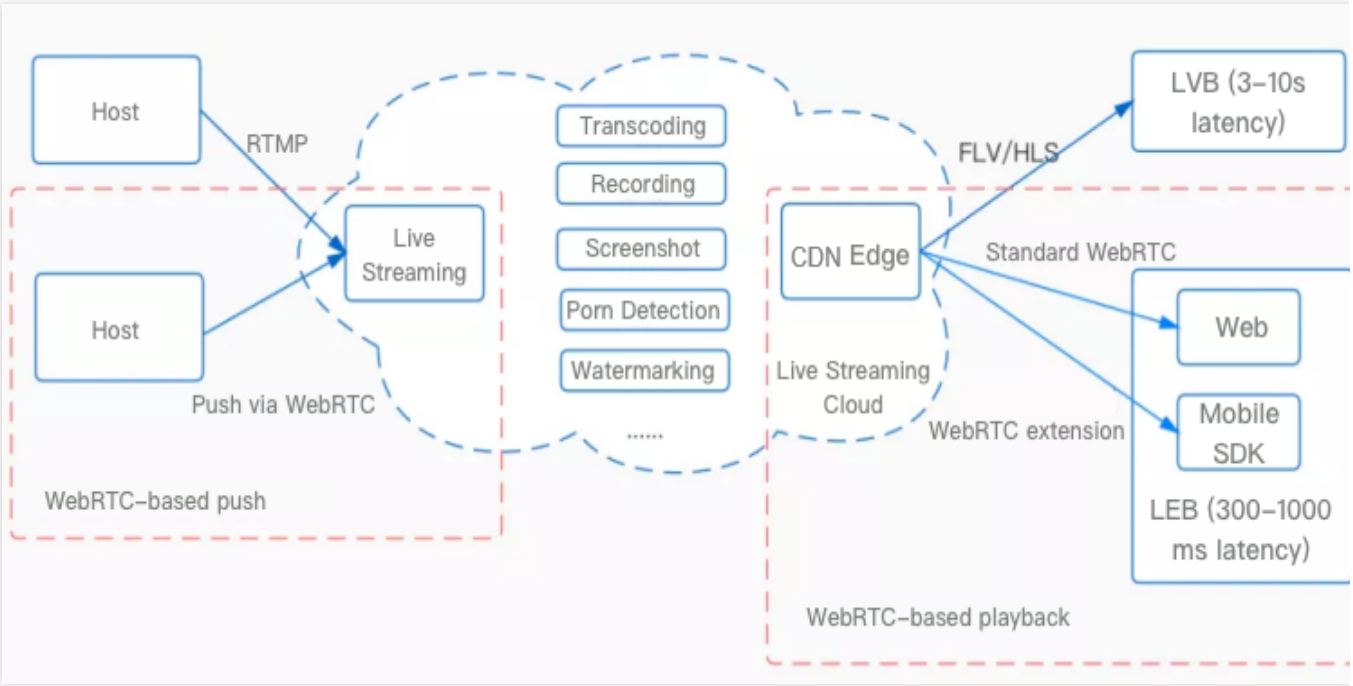


A good ecological environment is an important consideration for Live Event Broadcasting (LEB) to carry out low-latency transformation using WebRTC. Most popular browsers such as Chrome and Safari already support the

WebRTC standard, and mature open-source WebRTC SDKs.



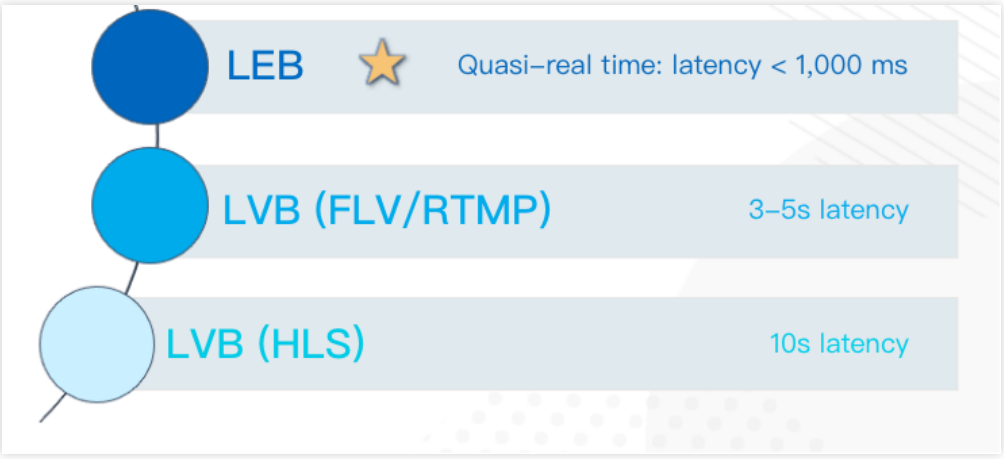
Low-latency live streaming services in the industry use protocols such as QUIC, SRT, WebRTC, and ORTC. Among these, the latency of QUIC is relatively high because it does not have the characteristics of streaming media. SRT, WebRTC, and ORTC have streaming media characteristics and can stream with millisecond latency, but SRT and ORTC are not as widely used as WebRTC. As a result, LEB uses the UDP-based WebRTC to implement low-latency live streaming.

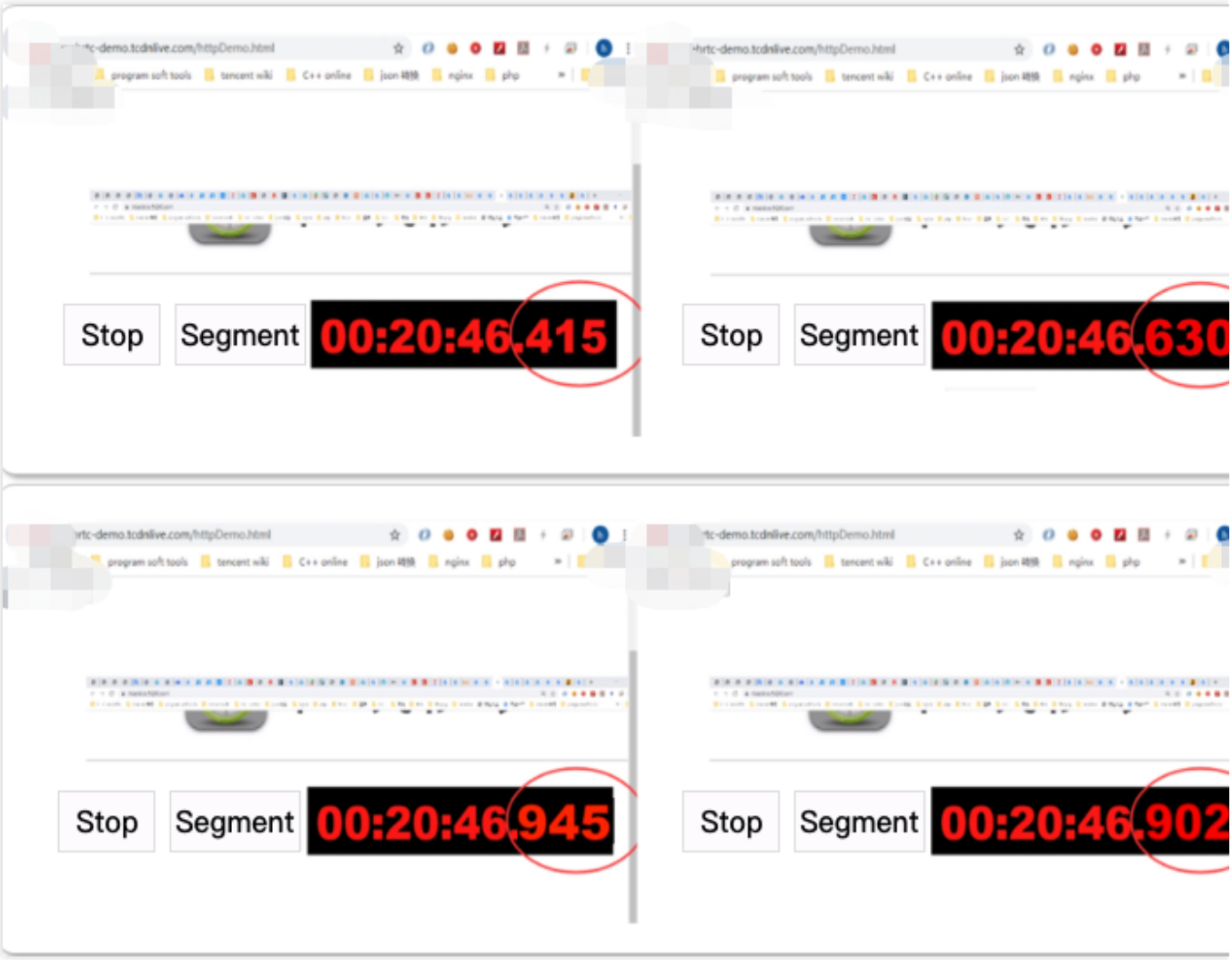


Latency Comparison

The latency of the FLV protocol in standard live streaming is generally between 2 and 10 seconds. The main factors contributing to its latency are the GOP size and TCP backlog in weak network transmission. The latency of HLS is even higher, ranging from several seconds to tens of seconds. The main factors causing HLS latency are the GOP size and TS size, making it the highest latency in standard live streaming.

In contrast, Live Event Broadcasting (LEB) adopts the WebRTC playback protocol based on the UDP protocol, which enables millisecond-level latency between nodes. The latency of LEB is typically between 300 ms and 1000 ms.





Weak Network Resilience Comparison

Live Event Broadcasting (LEB) has stronger weak network resilience compared to Standard Live Video Broadcasting (LVB), providing a more stable playback experience. To further demonstrate the weak network advantages of LEB, we conducted tests on video lag rate and other indicators for LEB and Standard LVB under normal and weak network environments. The specific test results are as follows:

Test Scenarios

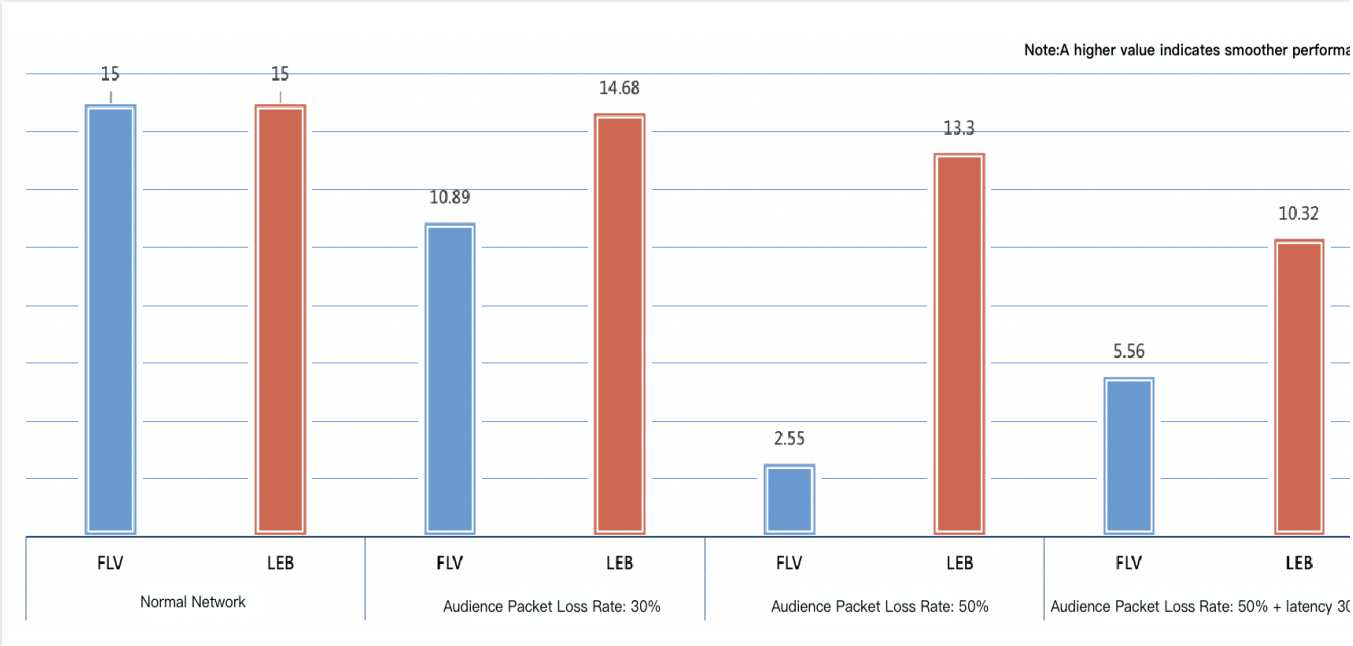
The publisher uses RTMP for streaming, and the audience plays FLV and LEB streams separately, with lag rate and other indicators recorded. The publisher has a lossless network, while the audience's network is set to different weak network conditions for testing. The main test indicators are frame rate and lag rate.

Stream Parameter Configuration

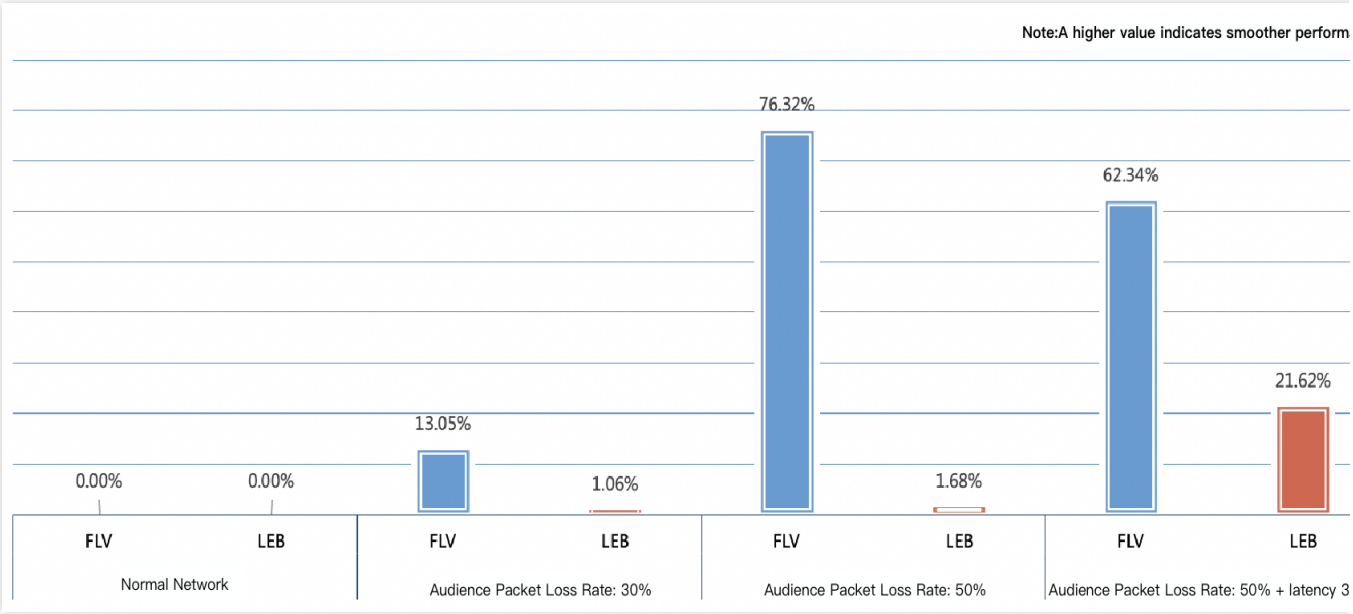
Parameter	Configuration Information
Resolution	720 × 1080
Bitrate	1800 kbps
Frame Rate	15

Comparison of Key Indicators in Several Weak Network Scenarios

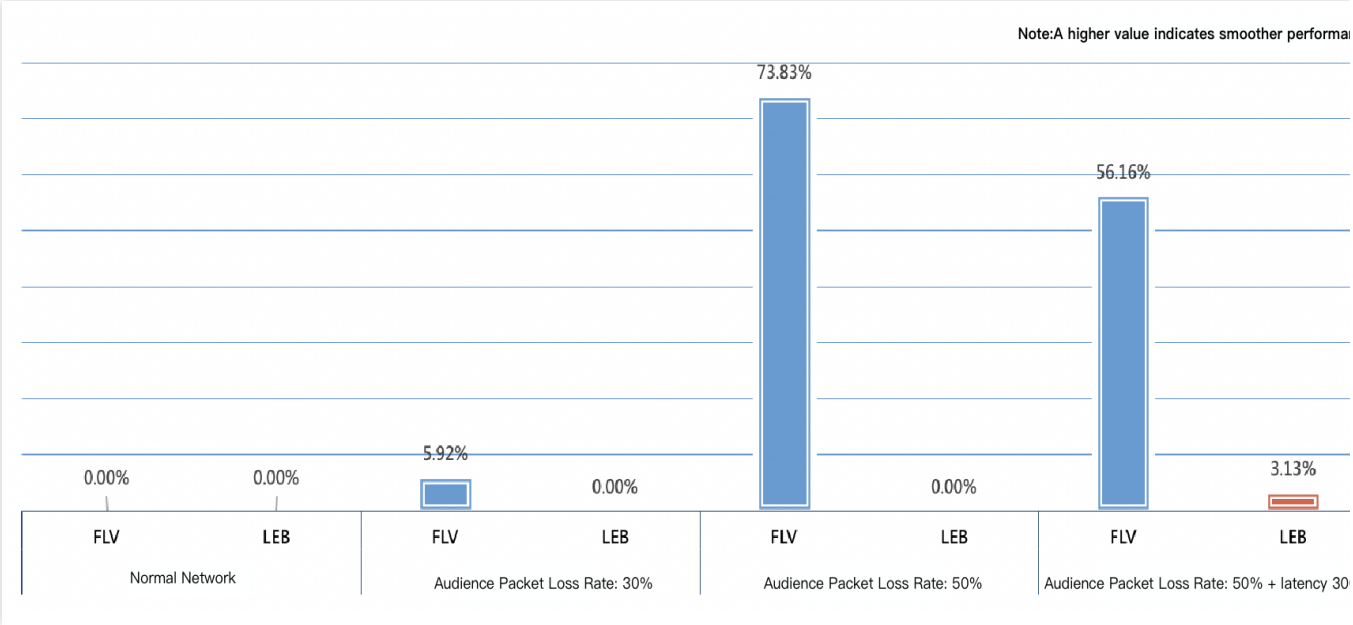
Video Frame Rate



Video Lag Rate



Audio Lag Rate



Parameter Description

Indicator	Explanation
Video Lag Rate	Video rendering intervals greater than 500 milliseconds are considered as lags. The total duration of all lags divided by the total playback duration is the lag rate.
Audio Lag Rate	Audio playback intervals greater than 200 milliseconds are considered as lags. The total duration of all lags divided by the total playback duration is the lag rate.

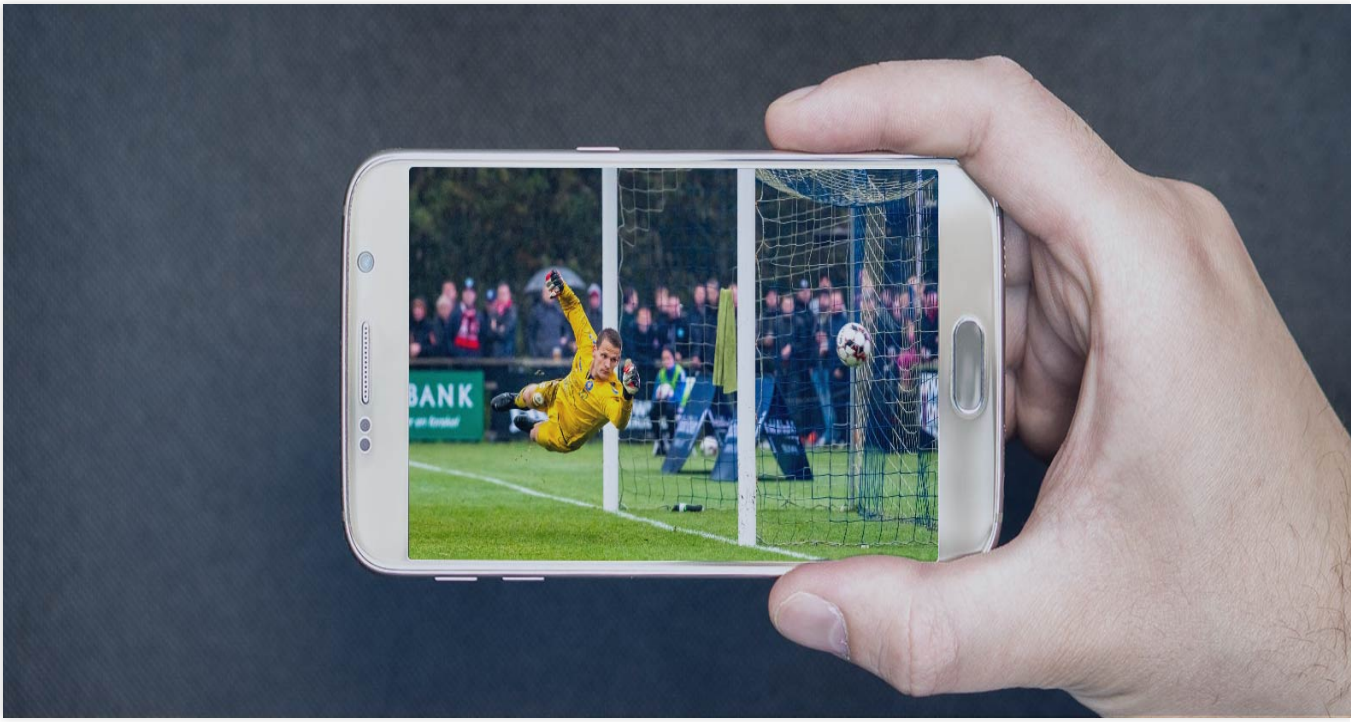
Video Frame Rate	The number of frames played per second in a video.

Use Cases

Last updated：2021-09-02 10:50:51

Sports Events

LEB offers live video streaming with ultra-low latency for sports events. The viewers can enjoy watching sports events and get the event results in real time.



Live Shopping

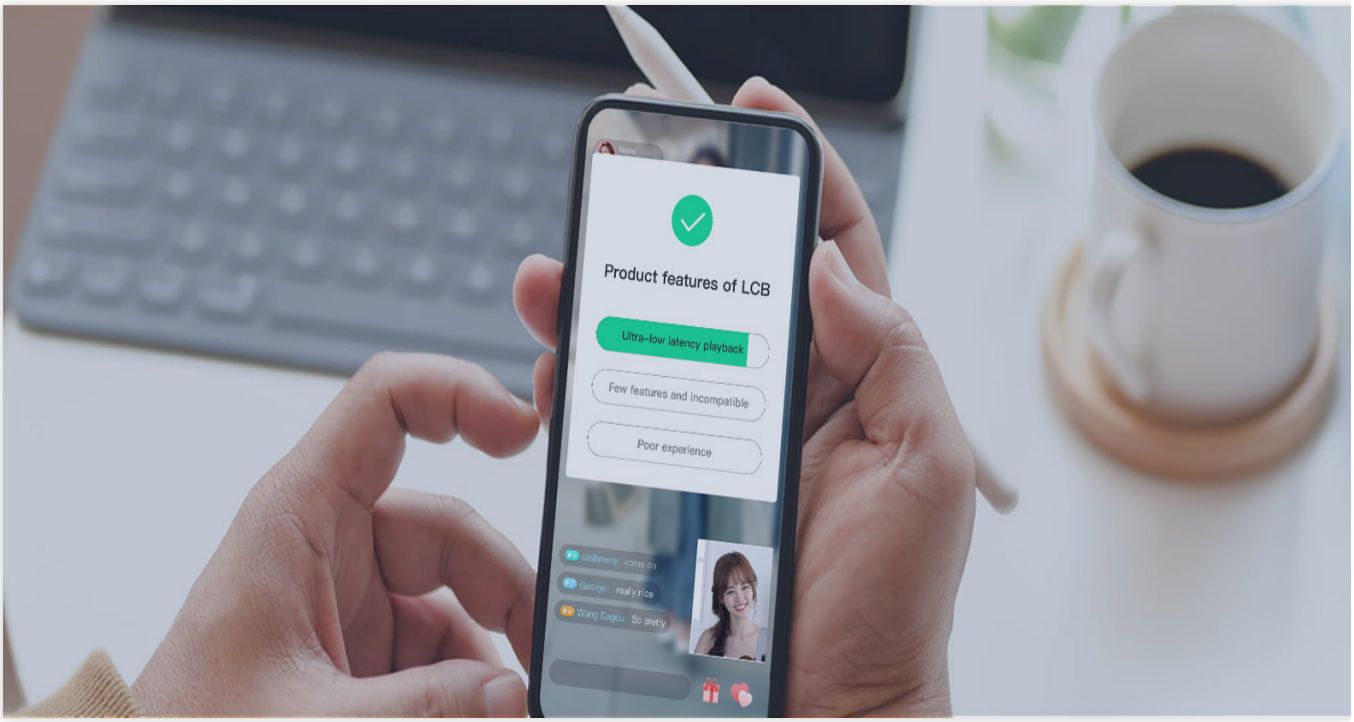
LEB delivers ultra-low latency which can well meet the requirements of live shopping scenarios such as auctions and sales promotions. LEB enables the host and viewers to get timely transaction feedback and allows viewers to easily purchase products while watching.

Online Classroom

LEB can be used to create an online classroom with ultra-low latency. LEB allows both teachers and students to broadcast live video images in real time, making online classrooms just like face-to-face learning.

Live Q&A

Due to latency in traditional live Q&A, frames need to be inserted on the viewer clients so that the host and viewers can see the same screen at the same time. Ultra-low latency LEB is a perfect solution for this problem. It ensures both sides can see the same screen in real time, thus achieving smooth live Q&A.



Interactions in Live Shows

LEB is suitable for live shows as it provides optimal user experience in giving gifts and other interactive features which require low latency.

Getting Started

Last updated：2024-09-18 21:45:51

This document shows you how to get started with LEB. Before using LEB, please read Pricing Overview to learn about its **billable items** and **prices**.

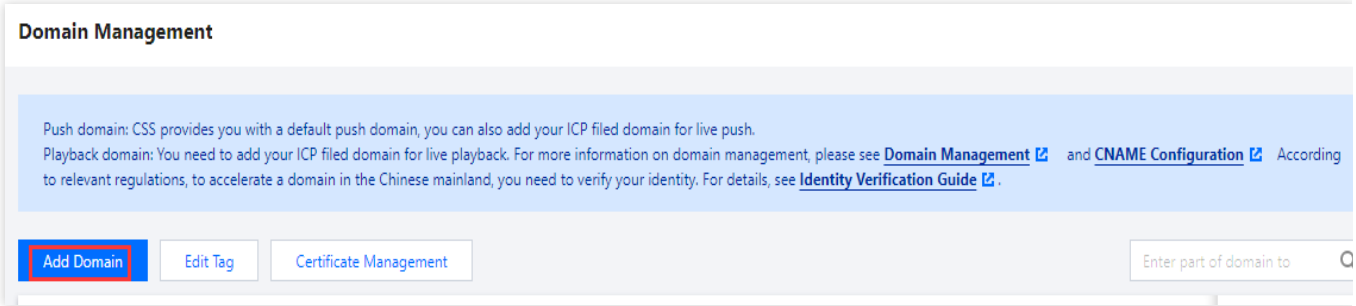
Prerequisites

1. You have signed up for a Tencent Cloud account.
2. Go to the CSS console, agree to **Tencent Cloud Service Agreement**, and click **Apply for Activation** to activate CSS.

Note:
After activating CSS, you will get 20 GB of playback traffic for the Chinese mainland for free.
The steps to configure domain names for LEB are the same as those for LVB. If you are already using LVB, you can skip to Step 4. Get a playback URL.

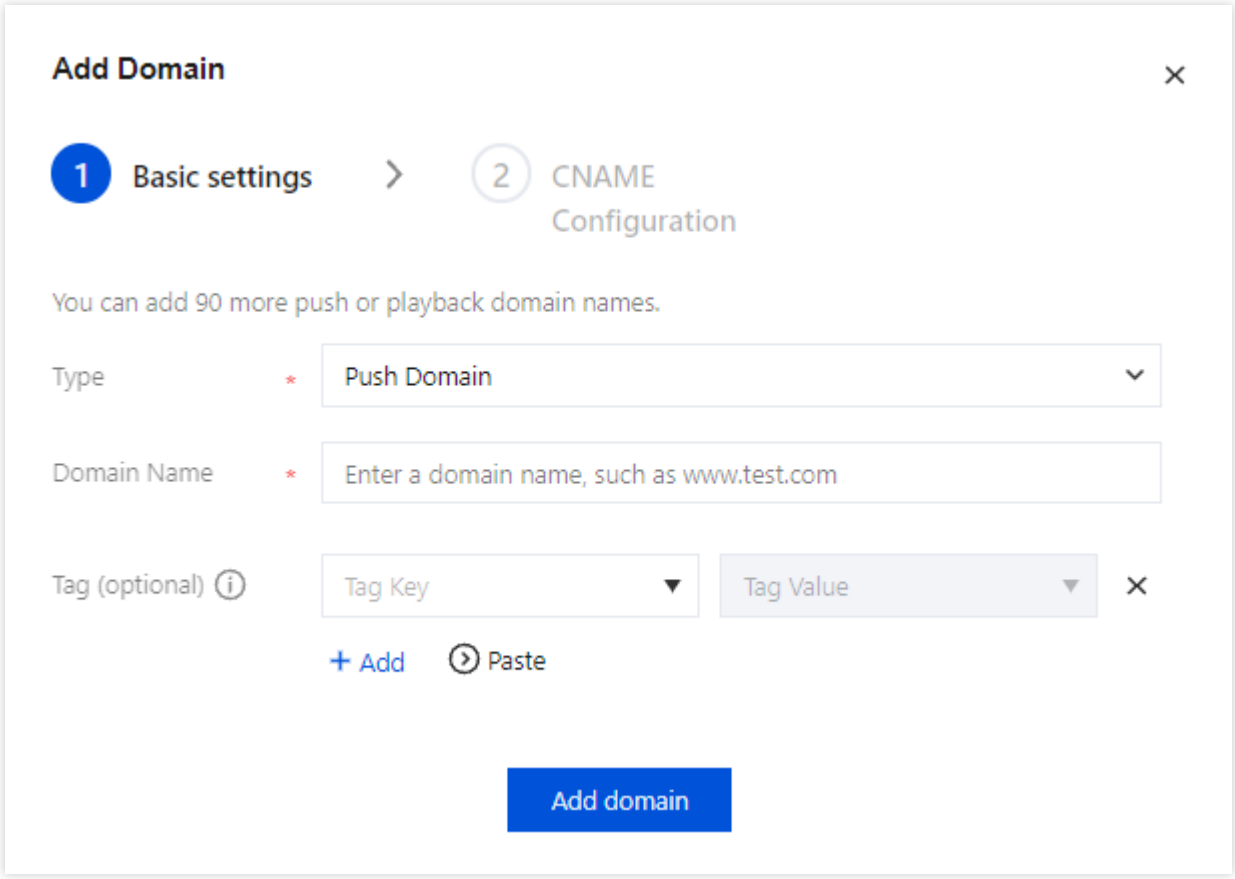
Step 1. Add domain names

To use CSS, you should have at least one **push domain name** and one **playback domain name**. You cannot use one domain name for both push and playback.
You can add your own domain names.
1. Register your domain name. ICP filing is required if you want to use the domain in the Chinese mainland.
2. Log in to the CSS Console, navigate to Domain Management, Click **Add Domain**.



3. Enter the custom domain addition page, fill in the domain name that has been completed for filing, and select **Type**.
4. Tags are used to categorize and manage resources from different dimensions. If the existing tags do not meet your requirements, you can also go to the Tag Console to manage tags uniformly.

5. Click **Add domain**.



Note:

CSS provides a test domain name `xxxx.livepush.myqcloud.com`. You can use it to test push, but we do not recommend using it for business purposes.

After the domain name is added successfully, you can view its information in the domain name list in **Domain Management**. For information on how to manage your domains, see Domain Management.

For more information on domain names for live streaming, see Live Streaming Basics.

6. Once your domain name is added, the system will assign it a canonical name (suffixed with `.txlivecdn.com` or `.tlivepush.com`), which cannot be accessed before you complete CNAME configuration at your DNS service provider. The following example shows you how to add a CNAME record if you use Tencent Cloud's DNS service:

6.1 Log in to the DNS console.

6.2 Find your domain name and click **Resolve**.

6.3 On the domain name resolution page, click **Add Record**.

6.4 Enter your domain name prefix for **Host Record**, select CNAME for **Record Type**, and enter the canonical name for **Record Value**.

6.5 Click **Save**.

Note:

It takes a while for CNAME configuration to take effect. You can use CSS only after it does.

After the CNAME configuration takes effect, you will see the

icon before the CNAME address in Domain Management.

If your CNAME configuration fails to take effect, please contact your DNS provider.

For how to add CNAME records with other DNS providers, see Configuring CNAME for Domain Name.

Step 2. Get a push URL

1. Go to **CSS Toolkit** > Address Generator.
2. Complete the following settings:

2.1 Select the address type as **Push Address**.

2.2 Select a push domain name you added in **Domain Management**.

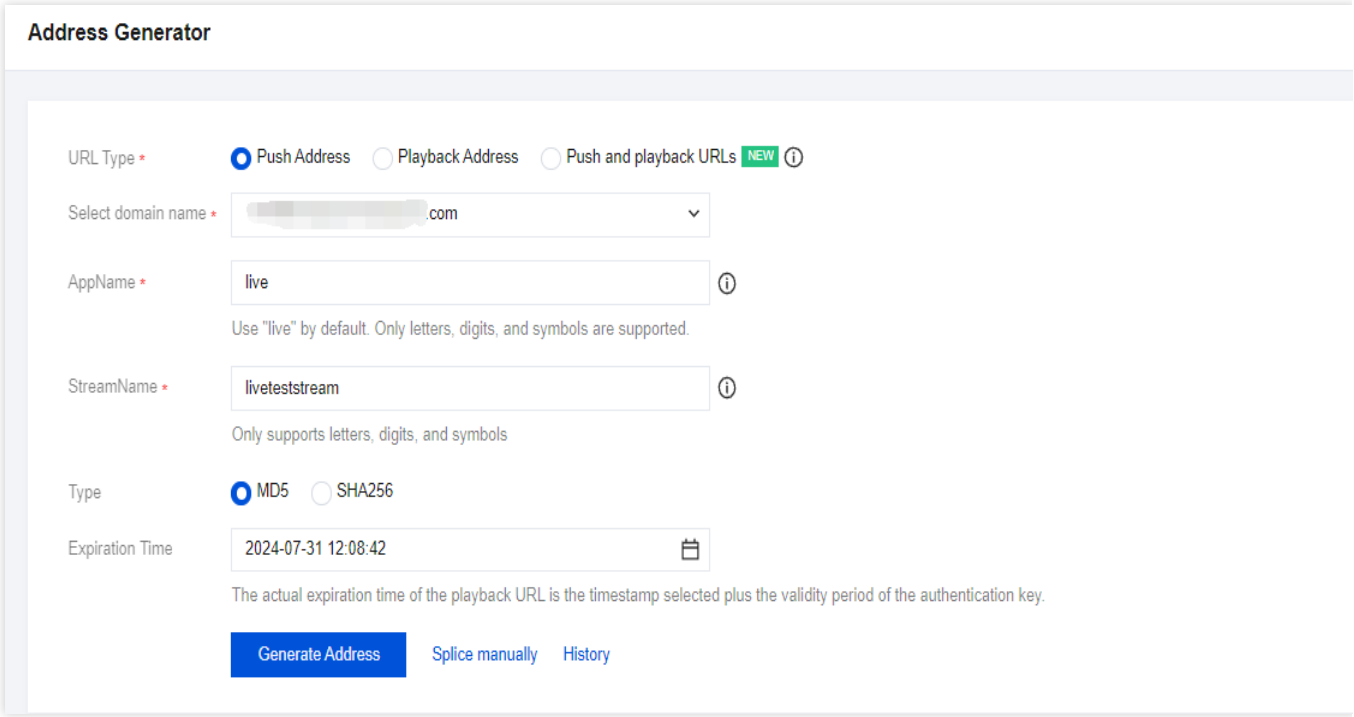
2.3 Enter an AppName. It is live by default.

2.4 Enter a custom `StreamName`, such as `liveteststream`.

2.5 You need to choose an encryption type based on your security requirements and performance considerations. The encryption type can be either **MD5** or **SHA256**, with **MD5** being the default option.

2.6 Select a URL expiration time, such as `2024-07-31 12:08:42`.

3. Click **Generate Address** to generate a push address.



Note:

The default value of `AppName` is `live` . `txSecret` is the signature for playback, and `txTime` is the URL expiration time.

Here is another way to generate a push URL: In Domain Management, find the domain name you want to use for push and click **Manage**. Under **Push Configuration**, select an expiration time for the URL, enter a custom `StreamName` , and click **Generate Push Address**.

Before generating a push URL, you can create templates and bind them to your push domain. For the prices of CSS value-added services, see Pricing Overview.

Step 3. Start pushing

To start pushing, provide the push URL generated to the software you use for push.

For PC live streaming, it is recommended to use OBS WebRTC live streaming.

For push on web, we recommend you use Web Push: Click **Generate**. In the pop-up window, select a push domain name, enter a custom `StreamName` , select a URL expiration time, and click **Confirm**. Turn the camera on, and click **Start Push**.

For push from mobile devices, download the TCToolkit app, open it, go to the live push page, and enter the push URL manually or scan the QR code generated in the previous step to auto-fill the URL. Tap **Start Push**.

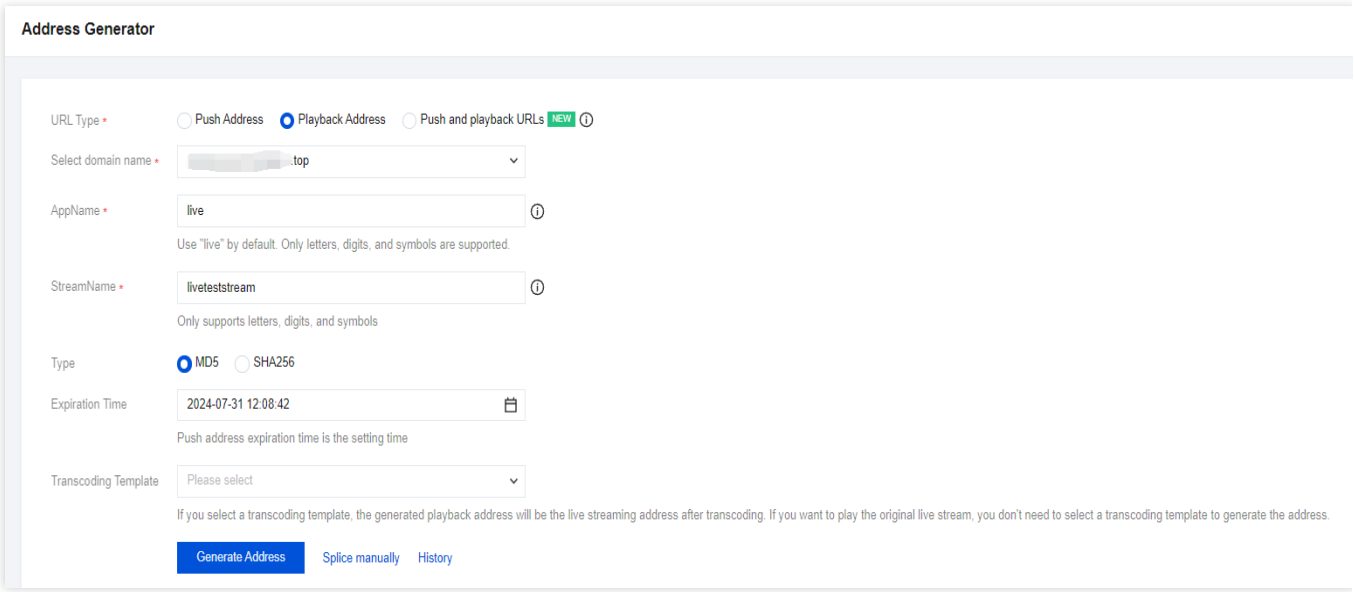
Note:

You can integrate the MLVB SDK into your app to implement the push feature.

The LEB solution for web does not support decoding or playing B-frames. For details, see B-Frames.

Step 4. Get the playback URL

1. After push succeeds, select Stream Management on the left sidebar. Under the **Live Streams** tab, you can view the status of the push URL. You can also click **Preview** to play the stream.
2. Go to **CSS Toolkit** > Address Generator and complete the following settings:
2.1 Select the address type: **Playback Address**.
2.2 Select the address type as Playback Address and choose the playback domain name that you have added to Domain Management.
2.3 Enter an AppName. It is live by default.
2.4 Enter an AppName. It is live by default.Enter the same **StreamName** as the push address. The playback address StreamName must be consistent with the push address **StreamName** to play the corresponding stream.
2.5 You need to choose an encryption type based on your security requirements and performance considerations. The encryption type can be either **MD5** or **SHA256**, with **MD5** being the default option.
2.6 Select a URL expiration time, such as `2024-07-31 12:08:42` .
2.7 Select a transcoding template if you want to transcode the stream and get a playback URL of the transcoded stream. This step is not necessary if you play the original stream.
2.8 Click **Generate Address** to generate an LEB playback URL whose format is `webrtc://domain/path/stream_id` .

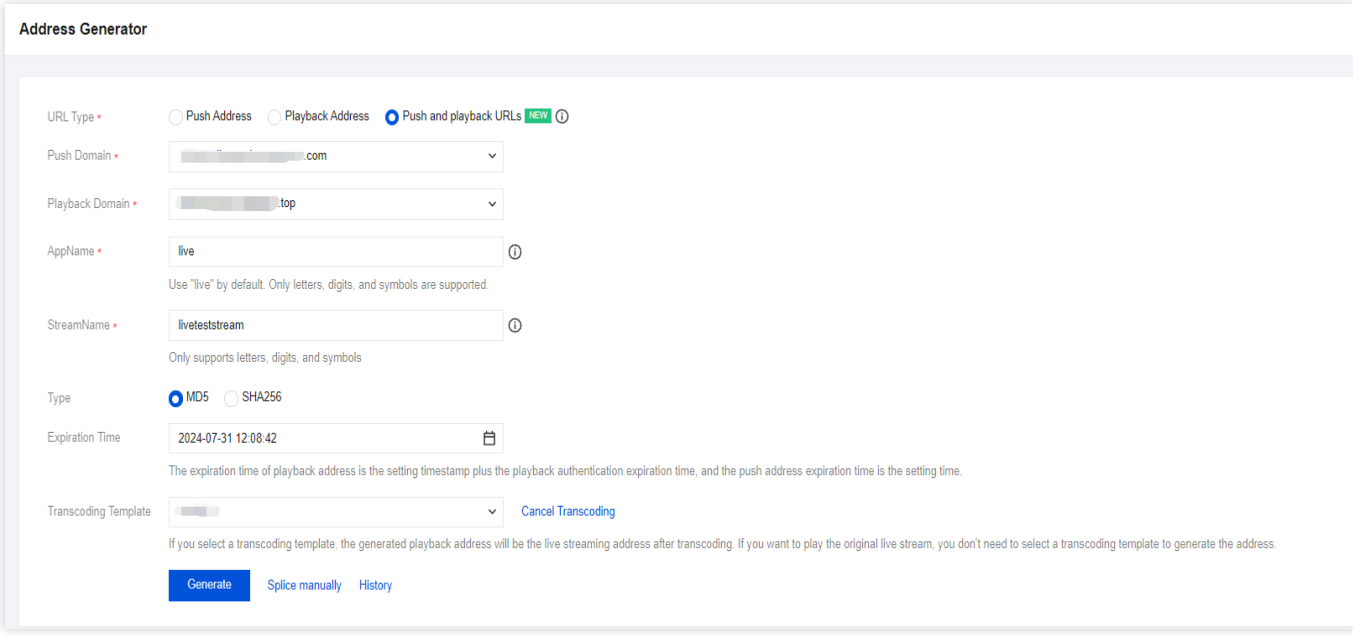


3. You can use the following methods to test playback in different scenarios:
Playback on web: We recommend you use the TCPlayer demo to test playback.
Note:
The demo supports changing video quality during playback. You can create a transcoding template to output HD and SD videos in **Feature Configuration** > Live Transcoding, enter in the demo a WebRTC URL containing the

transcoding template, and play it. If you don't need to test this feature, enter the original WebRTC URL.
For more information on live transcoding and its billing, see Live Transcoding.
Playback on mobile devices: We recommend you use the TCToolkit app to test playback on mobile devices. Open the app, select **Live broadcast > LEB Player**, enter the playback URL manually or scan the QR code generated in the previous step to auto-fill the URL, and then tap **Start Playback**.
Note:
If you want to play the stream in your app, you can integrate the MLVB SDK. If you have any questions, see FAQs.

Step 5.Generate Push and Playback Address Group

1. Go to **CSS Toolkit** > Address Generator.
2. Select the address type as **Push and playback URLs**.
3. Select the **Push Domain Name** and **Playback Domain Name** that you have added to Domain Management.
4. Enter an **AppName**. It is `live` by default.
5. Enter a StreamName, such as `liveteststream` .
6. You need to choose an encryption type based on your security requirements and performance considerations. The encryption type can be either **MD5** or **SHA256**, with **MD5** being the default option.
7. Select the URL expiration time, such as `2024-07-31 12:08:42` .
8. Select an existing transcoding template (optional).
9. Click **Generate Address.**



Step 6. Use LEB

Our LEB solution for **mobile devices** supports B-frame decoding and playback of AAC files. It has been integrated into the MLVB SDK, please refer toLEB.
Web solution: Integrated into the TCPlayer player. For integration, please refer to Web Integration.

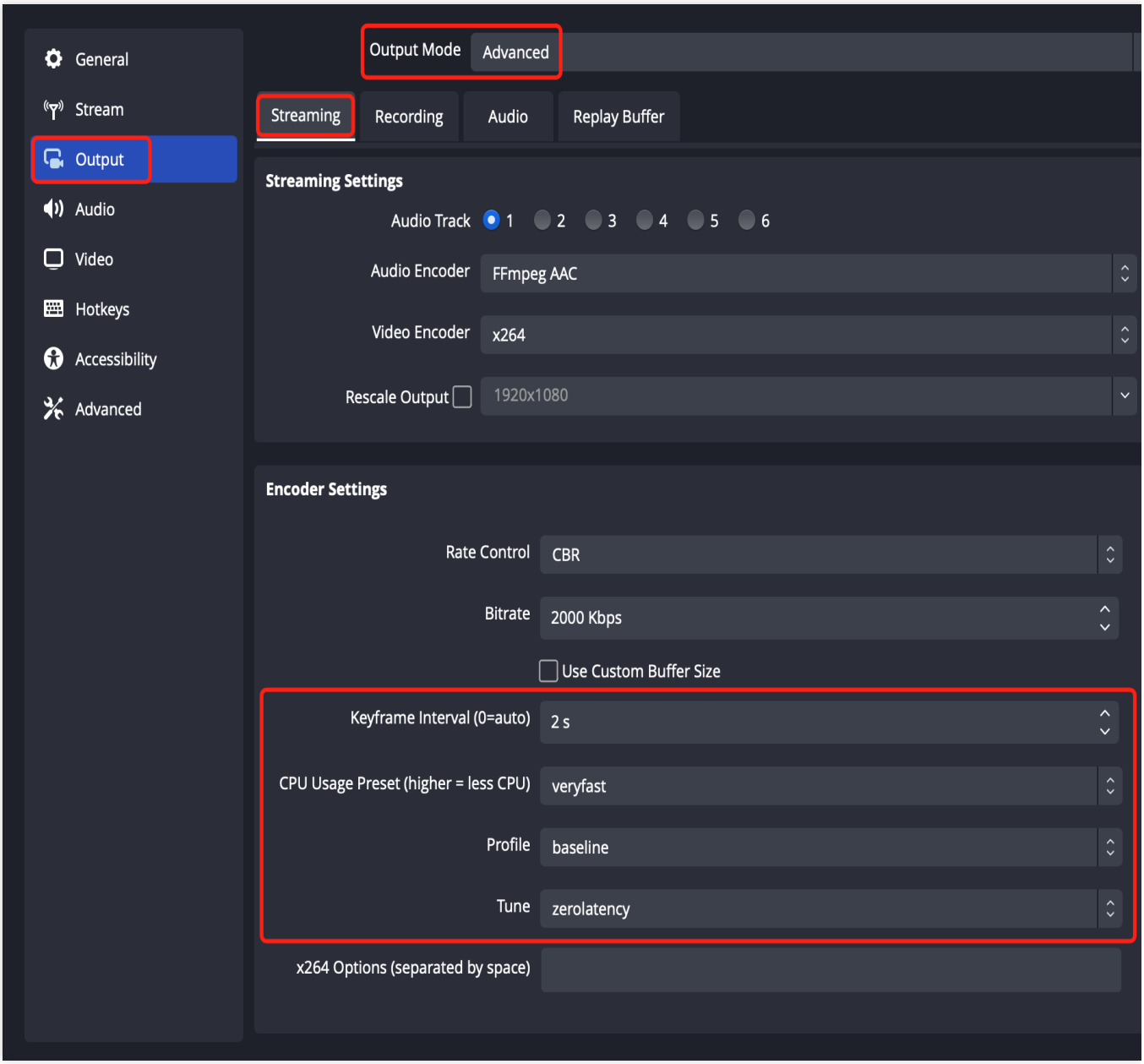
FAQs

Generating playback URLs

The format of LEB URLs is the same as that of LVB URLs except that the former start with `webrtc` while the latter start with `rtmp` .
LEB playback URL format: `webrtc://domain/path/stream_id` or `webrtc://domain/path/stream_id?txSecret=xxx&txTime=xxx` (hotlink protection enabled). For how to generate a playback URL, see Get the playback URL.
Note:
You can use URLs of transcoded streams to play at different resolutions and bitrates. For how to generate a playback URL for a transcoded stream, see Live Remuxing and Transcoding.

B-frames

Fast Live Web solution does **not support B-frame decoding and playback**. Therefore, if the original stream contains B-frames, the backend will automatically transcode the stream to remove B-frames. However, this process introduces additional transcoding latency and incurs transcoding costs.
It is recommended to avoid pushing streams containing B-frames. Users can remove B-frames by adjusting the video encoding parameters of the streaming software (such as OBS). If using OBS for live streaming, you can disable B-frames through the settings.
A large **Keyframe Interval (GOP)** may affect the Fast Live experience, and it is recommended to set the interval to 2 seconds.
As shown below:



Audio transcoding

Playback using browsers only supports the standard WebRTC protocol and does not support AAC. If a stream pushed contains audio in AAC format, the system will transcode the audio into Opus format, which will incur audio transcoding fees.

SDK Integration

Last updated：2024-09-18 21:45:51

As a lower-latency version of LVB, LEB provides superb **live streaming** experience with millisecond playback latency, far lower than that of live stream playback using traditional protocols.
Before you use LEB, please read LEB Billing Overview to learn about its billable items and pricing.

Note:

LEB uses the WebRTC protocol to ensure low latency. It adopts the Opus codec and does not support B-frames. If the original stream contains B-frames or the codec is not Opus, the CSS backend will remove the B-frames and transcode the stream to Opus format, which will incur standard transcoding fees.

Integration into Application

Directions

You can integrate the MLVB SDK into your iOS or Android application to implement the live push and playback features.
Live push: Capture from the camera or phone screen and push the stream to CSS using the RTMP protocol. For details, see Publishing (Camera) and Publishing (Screen Recording).
Live playback: Play streams using WebRTC with ultra-low latency. For details, see Playback > LEB.

Note:

The MLVB SDK leverages the capabilities of CSS, IM, TRTC, and other services to implement low-latency audio/video communication for multiple parties, allowing participants to interact with each other while others watch. For details, see Mic Connect.

Free demo

TCToolkit is an open-source and comprehensive audio/video solution developed by Tencent Cloud. You can use it to try out LEB's capability to play live streams with millisecond latency.

Platform	Demo	Push Demonstration (Android)	Play
Android			

		
iOS	Under maintenance	

Integration into Webpage

Directions

You can use the following ways to achieve live push and playback on your websites:

Live push on web: The push component is designed and packaged according to the WebRTC standard, which is supported by most browsers. You can enable the live push feature simply by importing the code we provide. For details, see WebRTC Push.

Note:

The Opus audio codec is used for the push of WebRTC streams. Therefore, if you use an LVB protocol (RTMP, HTTP-FLV, or HLS) for playback, the CSS backend will automatically convert the audio to AAC format to ensure successful playback, which will incur audio transcoding fees. For details, see Live Transcoding > Audio Transcoding.

If you publish and play streams using the LEB protocol, CSS will not transcode the audio.

With WebRTC, each push domain can be used for up to **1,000 concurrent streams** by default. If you want to push more streams, please submit a ticket.

Live playback on web: We recommend you use our web player SDK TCPlayerLite, which supports playing **WebRTC streams** on mobile and desktop browsers and delivers a superb streaming experience with millisecond

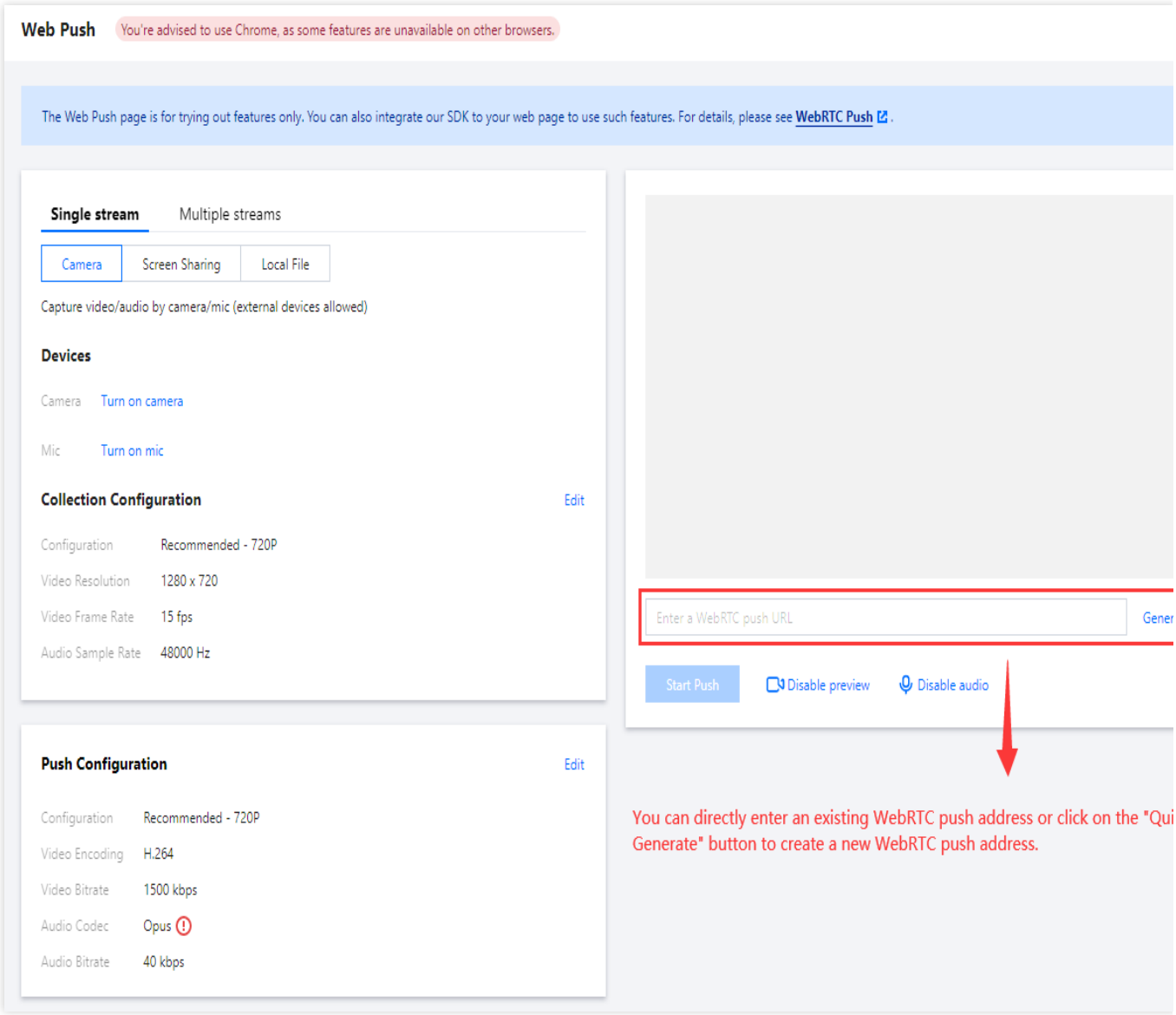
latency, far lower than that of playback using traditional live streaming protocols.

Note:

If a browser does not support WebRTC, a WebRTC URL passed into the player will be converted to better support playback. By default, WebRTC is converted to HLS on mobile browsers and HTTP-FLV on desktop browsers.

Free demo

Live push on web: You can test the web push feature in Web Push of the CSS console.



Live playback on web: You can use the TCPlayer demo to test playback on web.

Note:

Both live push and playback on web use the standard WebRTC protocol. Web push does not generate B-frames, and audio is encoded in Opus format, so there will be no costs of audio transcoding or B-frame removal.

Implementing WebRTC Push Using OBS

LEB (ultra-low-latency live streaming) uses WebRTC to push live audio/video or video files to the CSS server. The following describes how to use OBS to push WebRTC streams.
OBS supports WebRTC protocol live streaming, which means you can easily and quickly push live streams to Tencent Cloud CSS using the WebRTC protocol on PC (Windows/Mac/Ubuntu), just like using the RTMP protocol. The following content mainly introduces how to use the OBS tool to implement WebRTC protocol live streaming functionality.

Note：

There are two schemes, new and old, to choose from when integrating OBS with WebRTC protocol live streaming:

New Scheme - OBS WebRTC Live Streaming (OBS v30.0 Beta 1 or higher): This scheme does **not require a plugin**, making the integration process more convenient. For specific operation guidance, please refer to:OBS WebRTC live streaming.

Old Scheme - Using OBS Plugin for WebRTC Live Streaming: If you are using an OBS version lower than v30.0 Beta 1 and cannot directly use the WebRTC protocol for live streaming, Tencent Cloud CSS provides an integrated OBS plugin solution for WebRTC live streaming. For specific operation guidance, please refer to: OBS Plugin for WebRTC Live Streaming.

Based on your actual requirements and OBS version, you can choose the appropriate WebRTC live streaming solution. Please note that the actual streaming performance may be affected by factors such as device performance, network conditions, and player buffering. During the usage process, you can adjust the streaming parameters and tools according to your needs to optimize the live streaming experience.

LEB playback

For how to use the MLVB SDK for LEB playback, see Playback > (LEB).