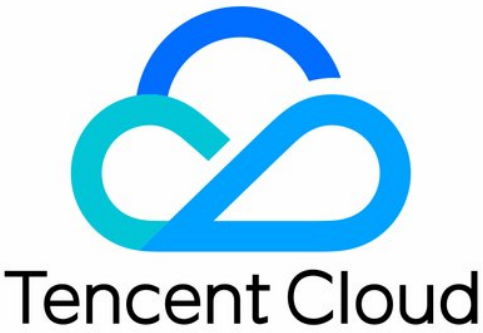
Mobile Live Video Broadcasting Technical Support Product Introduction



Contents

Technical Support
Contact Us

Last updated：2018-09-19 20:26:13

FAQ

- Basics you need to know about LVB
- How to deal with stutter?
- How to achieve instant broadcasting?
- How to reduce delay?
- Why is the Push unsuccessful
- Why did I fail to watch LVB?
- How to get a good image quality?
- How to activate cloud services?
- How to enable joint broadcasting?
- How to record and replay LVB videos?
- How does Tencent Cloud adapt to Apple's ATS?

Technical Supports

Contact us by submitting a ticket for any problems regarding MLVB. We will respond to you within 48 hours after receiving your query.

1V1 R&D Support

For any inquiries about R&D, contact Customer Service by calling 4009-100-100.

PushURL+PlayURL

Last updated：2018-02-22 19:12:31

How to Get URL Quickly?

If you want to generate a group of URLs for testing, you only need to open "LVB Console" -> "LVB Code Access" -> "Push Generator", and click **"Generate Push URL"** button to generate a push URL and three playback URLs with different playback protocols.

How to realize auto construction at the backend?

Push URL

In practical application of products, you cannot create push and playback URLs for each VJ manually. These URLs should be constructed by your server automatically. You can use any URL that conforms to the Tencent Cloud standards to push.

- **LVB Code**
  It is also called room ID, and a random number or a user ID is recommended. Note, you need to add BIZID as the prefix in a valid LVB code.

- **txTime**
  It refers to the time when the URL expires. The format is UNIX time stamp in hexadecimal notation, for example, 5867D600 means that the URL will expire at 00:00:00 AM on Jan. 1, 2017. Generally, our customers set txTime as follows: the URL will expire after 24 hours from the current time. It is not recommended to minimize the expiration time, because VJ will resume push in case of network interruption during LVB. If the expiration time is too short, VJ cannot resume push due to expiration of push URL.

- **txSecret**
  The hotlink protection signature is used to prevent attackers from forging your backend to generate push URL.

- **Sample Codes**
  Sample codes (PHP and Java) in the lower half of "LVB Console" -> "LVB Code Access" -> "Push Generator" page show how to generate hotlink protection URL.

Playback URL

Constructing a playback URL is as simple as that for constructing a push URL. You only need to change the sub-domain name from **livepush** to **liveplay**:

rtmp	rtmp://8888.liveplay.myqcloud.com/live/8888_test_123
flv	http://8888.liveplay.myqcloud.com/live/8888_test_123.flv
hls	http://8888.liveplay.myqcloud.com/live/8888_test_123.m3u8

How to Calculate Hotlink Protection?

The security hotlink protection refers to **txSecret** field in push and playback URLs. It is used to prevent attackers from forging your backend to generate push URL, or stealing your playback URL for their benefits.

Security Principle

To prevent attackers from forging your server to generate push URL, you need to configure **hotlink protection encryption key** in LVB console. Since the attackers are unable to obtain the encryption key easily, they cannot forge a valid push URL.

Calculation Procedure

- **Step1: Exchange the key**
  You need to negotiate one **encryption key** on the console at the official website. This encryption key is used to generate a hotlink protection signature on your server. Since Tencent Cloud has the same key as yours, it can decrypt and acknowledge the hotlink protection signature generated by your server.

  Encryption keys are classified into **push hotlink protection keys** and **playback hotlink protection keys**. The former are used to generate the push hotlink protection URLs and the latter are used to generate the playback hotlink protection URLs. In LVB Console, you can configure the push hotlink protection key.

Playback hotlink protection is disabled by default

Since the configuration of playback hotlink protection key needs to be synced to thousands of CDN clusters, it cannot be frequentlymodified in the debugging phase due to long synchronization period. Contact us if you need to configure the playback hotlink protection by calling our customer service. It generally takes 1 to 3 days to complete the synchronization in all of the clusters.

- **Step 2: Generate txTime**
  In the signature, the plaintext is txTime, which indicates the URL validity period. For example, if the current time is 2016-07-29 11:13:45 and the generated URL is expected to expire after 24 hours, txTime can be set to 2016-07-30 11:13:45.

  The long time string occupies large space in the URL. In actual application, 2016-07-30 11:13:45 is converted into the UNIX time stamp, i.e. 1469848425 (the various backend programming languages are directly handled by available time functions during the conversion). Then, the time stamp is converted into a hexadecimal string to further reduce the character size, that is, txTime = 1469848425 (hexadecimal) = 579C1B69 (hexadecimal).

  Generally, our customers set txTime as follows: the URL will expire after 24 hours from the current time. It is not recommended to minimize the expiration time, because VJ will resume push in case of network interruption during LVB. If the expiration time is too short, VJ cannot resume push due to expiration of push URL.

- **Step 3: Generate txSecret**
  txSecret is generated as follows: txSecret = **MD5 (KEY+ stream_id + txTime)**. The **KEY** here is the encryption key you configured in Step 1. In this example, stream_id is 8888_test001, txTime is 579C1B69 as calculated above, and **MD5** is the standard unidirectional irreversible hash algorithm.

- **Step 4: Combine to obtain the hotlink protection URL**
  After obtaining the push (or playback) URL, the txTime used to inform Tencent Cloud of the URL expiration time, and the txSecret that can be decrypted and verified only by Tencent Cloud, you can combine them to obtain a hotlink protection security URL.

Sample Codes

Sample codes (PHP and Java) in the lower half of "LVB Console" -> "LVB Code Access" -> "Push Generator" page show how to generate hotlink protection URL.

Basics of LVB

Last updated：2018-07-11 12:01:13

1. What are Push, LVB and VOD?

- **Push**: This refers to the process in which VJs push local video and audio sources to Tencent Video Cloud servers. It is also known as "RTMP Publishing" in some cases.
- **LVB**: LVB video source is generated in real time. It is only meaningful if someone pushes the live streams. Once the VJ stops broadcasting, the LVB URL becomes invalid, and since the live streams are played in real time, no progress bar is displayed on the player during the playback.
- **VOD**: VOD's video source is a file on cloud, which can be played at any time as long as it has not been deleted by the provider (such as Youku Tudou, iQIYI and Tencent Video). Since the entire video file is stored on the server, a progress bar is displayed during the playback.

2. What are the popular LVB protocols?

Three LVB protocols are commonly used: RTMP, FLV and HLS.

- **RTMP**: The RTMP protocol can be used for both push and live broadcasting. It involves breaking large video and audio frames into smaller fragments and transmitting them as small packets over the Internet. RTMP supports encryption, and thus provides a good privacy. However, the complicated fragmentation and reassembling bring about some unforeseeable stability issues to RTMP in case of a high-concurrency scenario.
- **FLV**: FLV protocol is mainly implemented by Adobe Systems. It simply places header information to the large video and audio frame headers. This simplicity makes it a sophisticated format in terms of delay control and high-concurrency performance. Its only disadvantage is the limited capability on mobile browsers. However, it's suitable for LVB on mobile Apps.
- **HLS**: HLS is a solution from Apple Inc. It splits videos into 5-10s fragments and manages them with an m3u8 index table. Since the videos downloaded on clients are complete data files with a 5-10s duration, the video smoothness can be ensured, but this leads to a great delay (typically the delay is around 10-30s when HLS is used). HLS is supported better than FLV on iPhone and most Android browsers, so it is often used for sharing URLs in QQ or WeChat's "Moments".

直播协议	优点	缺点	播放延迟
FLV	成熟度高、高并发无压力	需集成SDK才能播放	2s - 3s
RTMP	优质线路下理论延迟最低	高并发情况下表现不佳	1s - 3s
HLS(m3u8)	手机浏览器支持度高	延迟非常高	10s - 30s

3. What are the popular VOD protocols?

Three VOD formats are commonly used: MP4, HLS and FLV.

- **MP4**: MP4 is a classic format that is well supported on both mobile devices and PC browsers (The default browsers of iOS and most Android devices support MP4. On PC it can be played in a FLASH widget). However, MP4 video files are formatted in a complicated manner, which makes it time-consuming to process the files. Furthermore, the complexity of index table can cause a slow load when a long MP4 file (e.g. half an hour) is played online.
- **HLS**: HLS is implemented by Apple Inc., and is well supported on mobile browsers. But on IE, the support for HLS depends on the secondary development of FLASH. (You're recommended to use Tencent Video Cloud's FLASH player). Unlike MP4 that has a slow indexing, HLS's compact m3u8 index structure allows a fast indexing, which makes it an ideal choice for VOD.
- **FLV**: Implemented by Adobe Systems, FLV is the most popular wrapper format on live broadcasting platforms. On PC, it's well supported by FLASH. However, on mobile devices, it is only supported by the Apps which implement their players (or use this player). Most mobile browsers don't support FLV. Tencent Video Cloud uses FLV for LVB recording.

点播协议	优点	缺点
HLS(m3u8)	手机浏览器支持度高	大量小分片的文件组织形式，错误率和维护成本均高于单一文件
MP4	手机浏览器支持度高	格式过于复杂和娇贵，容错性很差，对播放器的要求很高
FLV	格式简单问题少，适合直播转录制场景	手机浏览器支持差，需集成SDK才能播放

4. What are the popular push protocols?

Although RTMP is not commonly used in live broadcasting, but it is dominant in push service (pushing data from **VJ** to **servers**). Domestic video cloud services use RTMP as the main push protocol. Tencent Video Cloud SDK's first module is VJ push, so the SDK is also called RTMP SDK.

5. Which features and protocols are supported by the Tencent RTMP SDK?

Tencent Video Cloud RTMP SDK supports Push, LVB and VOD.

- **Push**: Supports **RTMP publishing protocol**, as well as features such as hardware acceleration, beauty filter, bandwidth adaption and resolution adjustment.
- **LVB**: Supports **FLV (recommended)** and RTMP protocols, as well as instant broadcasting optimization, auto delay control and highly adaptive hardware-decoding.
- **VOD**: Supports **online** or **local** VOD services for **MP4\HLS\FLV** files. Note: The earlier versions of SDK only support FLV-based VOD.

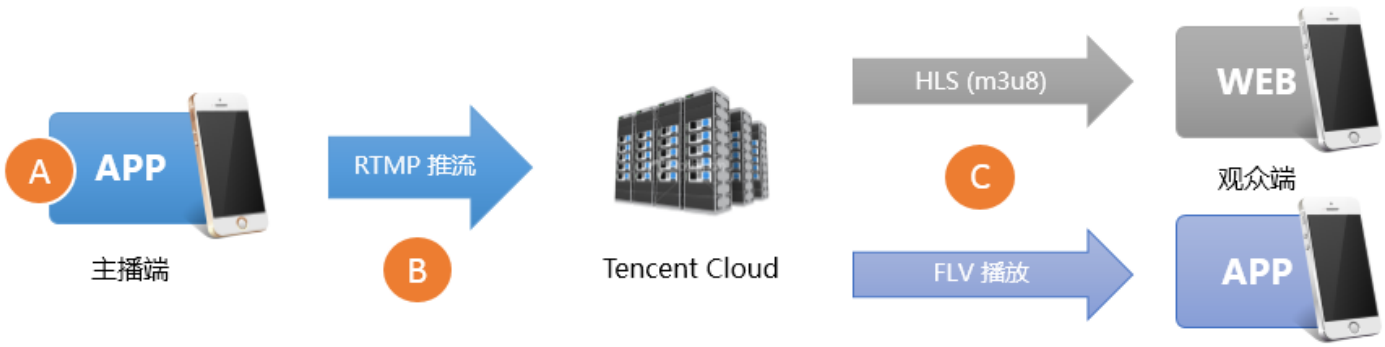
6. Is it mandatory to obtain the "License for Dissemination of Audio-Visual Programs through Information Network" for LVB?

Any applications that provide live broadcasting and other audio/video services over Internet are required to have the "License for Dissemination of Audio-Visual Programs through Information Network" issued by the relevant authority.

Video Stutter

Last updated：2018-07-11 12:01:44

1. What can cause a stutter during LVB?



Generally, there are three reasons for the stutter:

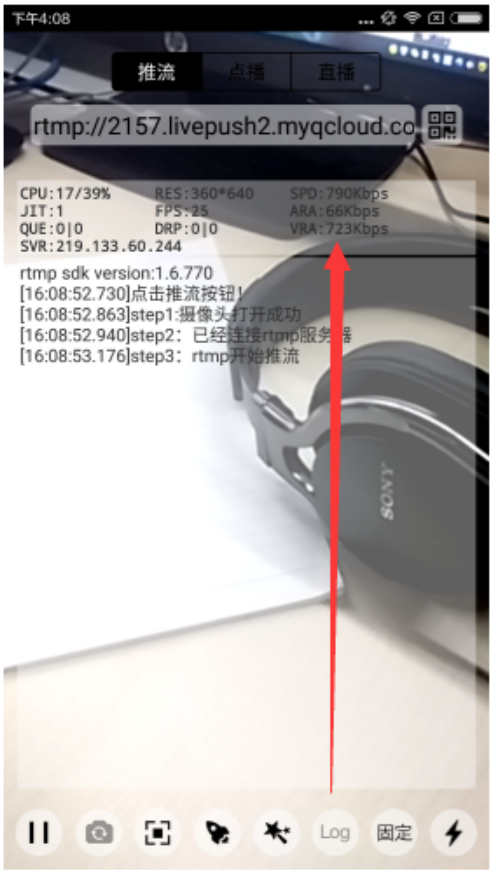
- **Reason 1: low frame rate**
  If the VJ uses a low-end phone, or there are CPU intensive applications running at the background, the frame rate of the video could be low. Typically, for an LVB to play smoothly, the frame rate of the video stream should be higher than 15 FPS. A frame rate lower than 10 FPS is **too low**, and can cause a stutter at **all the viewer ends**.
- **Reason 2: upstream clog**
  When pushing, VJs' phones generate audio and video data constantly. If the upstream bandwidth of a phone is too low, the generated audio and video data could clog the phone network and fails to be pushed, causing the stutter at **all the viewer ends**.
  Even though **domestic operators** offer broadband packages with a downstream bandwidth as fast as 10 Mbps, 20 Mbps or even 100 Mbps, the upstream bandwidth is highly limited. In many small cities, the upstream bandwidth is limited to 512 Kbps (i.e. a maximum of 64 KB data can be uploaded per second).
  Wi-Fi follows the IEEE 802.11 specification of carrier-sense multiple access and collision avoidance (CSMA/CA). To put it simply, a Wi-Fi hot spot can communicate with only one phone at one time, and other phones must verify or query if communication is possible before initiating a connection to a hot spot. Therefore, the more people using a Wi-Fi hot spot, the slower the connection is. Furthermore, Wi-Fi signal decays greatly when passing through walls or obstacles, and most of the families seldom take the Wi-Fi router position and the strength of Wi-Fi signal across rooms into consideration during the

design and decoration of their houses. Even the VJs themselves probably don't know how many walls are there between their routers and the rooms where they push video streams.

- **Reason 3: bad downstream connection**
  That is, the viewer's downstream bandwidth is insufficient or the network condition is unstable. For example, suppose the bitrate of an LVB stream is 1 Mbps (i.e. every second 1 M bits of data need to be downloaded). If the bandwidth at the viewer end is not fast enough, the viewer would experience serious stutter. Bad downstream connection only affects the viewers in the current network environment.

2. Status Monitor

The RTMP SDK provides a status feedback mechanism, by which the RTMP SDK reports various status parameters every 1-2 seconds. You can register the **TXLivePushListener** listener to obtain these status parameters.



RTMP SDK 状态参数表

推流状态	参数名称	含义说明
CPU	CPU使用率	App: 17% Sys: 39%
RES	推流分辨率	360 * 640
SPD	网络上行速度	790kbps
JIT	网络抖动	不推荐参考
FPS	视频帧率	25帧/秒
ARA	音频码率	66kbps
QUE	缓冲积压	0帧
DRP	主动丢包	推流以来尚未丢过
VRA	视频码率	723kpbs

Push Status	Description

Push Status	Description
NET_STATUS_CPU_USAGE	CPU utilization of current process and overall CPU utilization of the machine
NET_STATUS_VIDEO_FPS	Current video frame rate, that is, the number of frames produced by video encoder per second
NET_STATUS_NET_SPEED	Current transmission speed (in Kbps)
NET_STATUS_VIDEO_BITRATE	The output bitrate of the current video encoder, i.e., the number of video data bits produced by the encoder per second (in Kbps)
NET_STATUS_AUDIO_BITRATE	The output bitrate of the current audio encoder, i.e., the number of audio data bits produced by the encoder per second (in Kbps)
NET_STATUS_CACHE_SIZE	Accumulated audio/video data size. A value ≥ 10 indicates the current upstream bandwidth is not enough to consume the audio/video data produced
NET_STATUS_CODEC_DROP_CNT	The number of global packet drops. To avoid a vicious accumulation of delays, the SDK actively drops packets when the accumulated data exceeds the threshold. A higher number of packet drops means a more severe network problem.
NET_STATUS_SERVER_IP	The IP address of the connected push server. It is typically the nearest one with few hops from the client.

3. Low Frame Rate

3.1 How to verify if the frame rate is too low

We can obtain the video frame rate of the current push from the **VIDEO_FPS** status data of TXLivePushListener. Typically, for an LVB to play smoothly, the frame rate of the video stream should be higher than 15 FPS. A frame rate lower than 10 FPS could cause an obvious stutter at the viewer end.

3.2 Solutions

- **3.2.1 Observe CPU_USAGE value**
  You can obtain the **CPU utilization for the current push SDK** and **overall CPU utilization for the system** from **CPU_USAGE** status data of TXLivePushListener. If the overall CPU utilization for the system exceeds 80%, video capture and encoding may be affected; if the CPU utilization reaches 100%, the VJ end may be terribly stuck and it is impossible for the viewers to have a smooth viewing experience.

- **3.2.2 Identify the high CPU consumers**
  On an LVB App, in addition to RTMP SDK, many other features such as on-screen comments, floating stars, and interactive text messages can consume some CPU resources. To test and evaluate the CPU utilization for just the push SDK, use this simple DEMO.
- **3.2.3 Choose a reasonable resolution**
  A higher resolution doesn't always come with better video quality: firstly, a high resolution needs a higher bit-rate to work; a definition with a low bitrate and high resolution is often inferior to that with a high bitrate and a low resolution. Secondly, a high resolution such as 1280 x 720 does not have an obvious advantage on a 5'' phone screen. Only when the LVB is played full-screen on a PC can the resolution of 1280 x 720 make a significant difference from 960 x 540. However, a higher resolution can bring about a big increase of CPU utilization for SDK. Therefore, in most cases, it's recommended to simply set the video quality to **High Definition** with TXLivePush's setVideoQuality. A higher resolution isn't always the better.
- **3.3.4 Use hardware acceleration if appropriate**
  Most smart phones support hardware encoding to lower the CPU utilization for video encoding. When CPU utilization is too high for an APP, you can enable hardware encoding to lower the CPU utilization. By default, the **High Definition** video quality option of TXLivePush's setVideoQuality uses software encoding (on some Android devices, hardware encoding doesn't function well due to the severe mosaics). You can use enableHWAcceleration of TXLivePushConfig to enable hardware encoding.

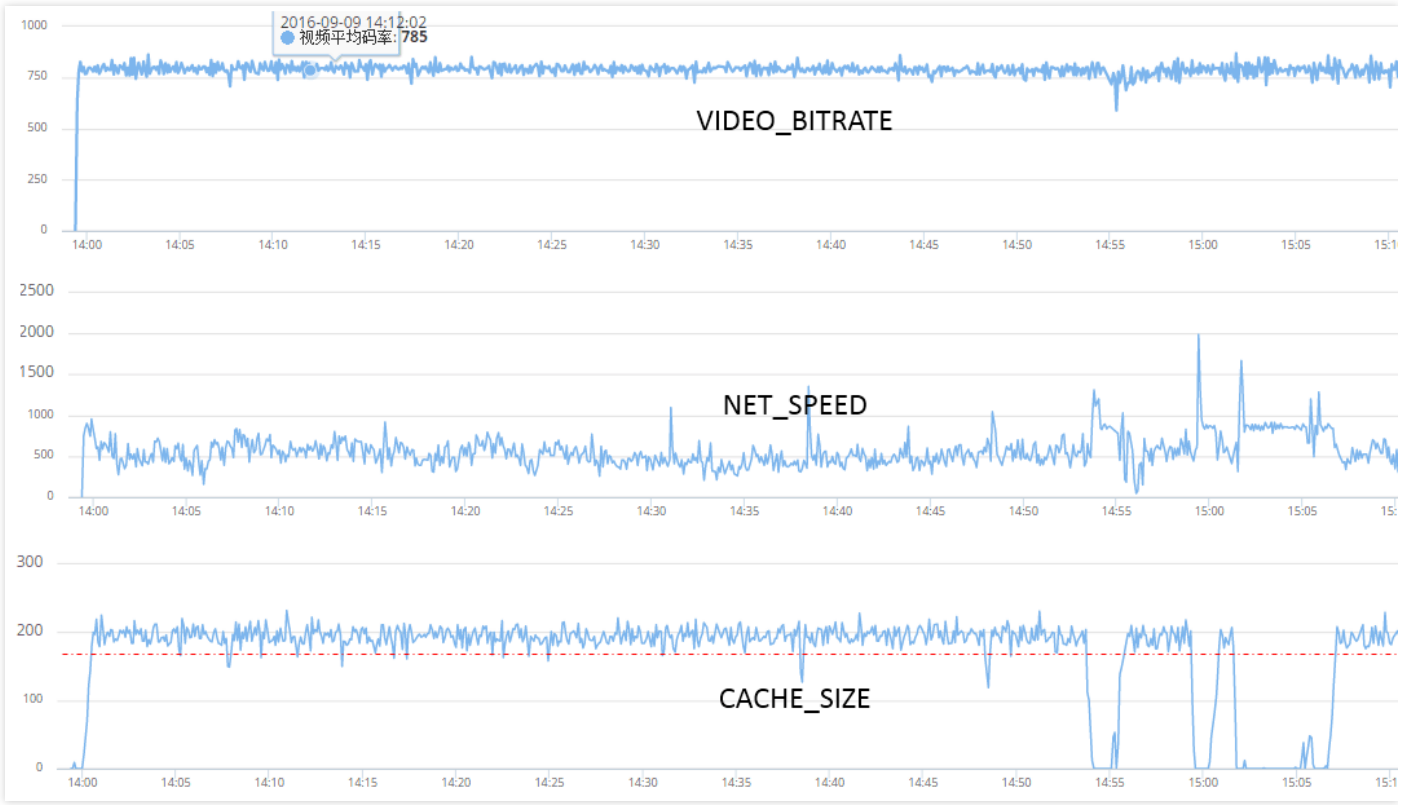
4. Upstream Clog

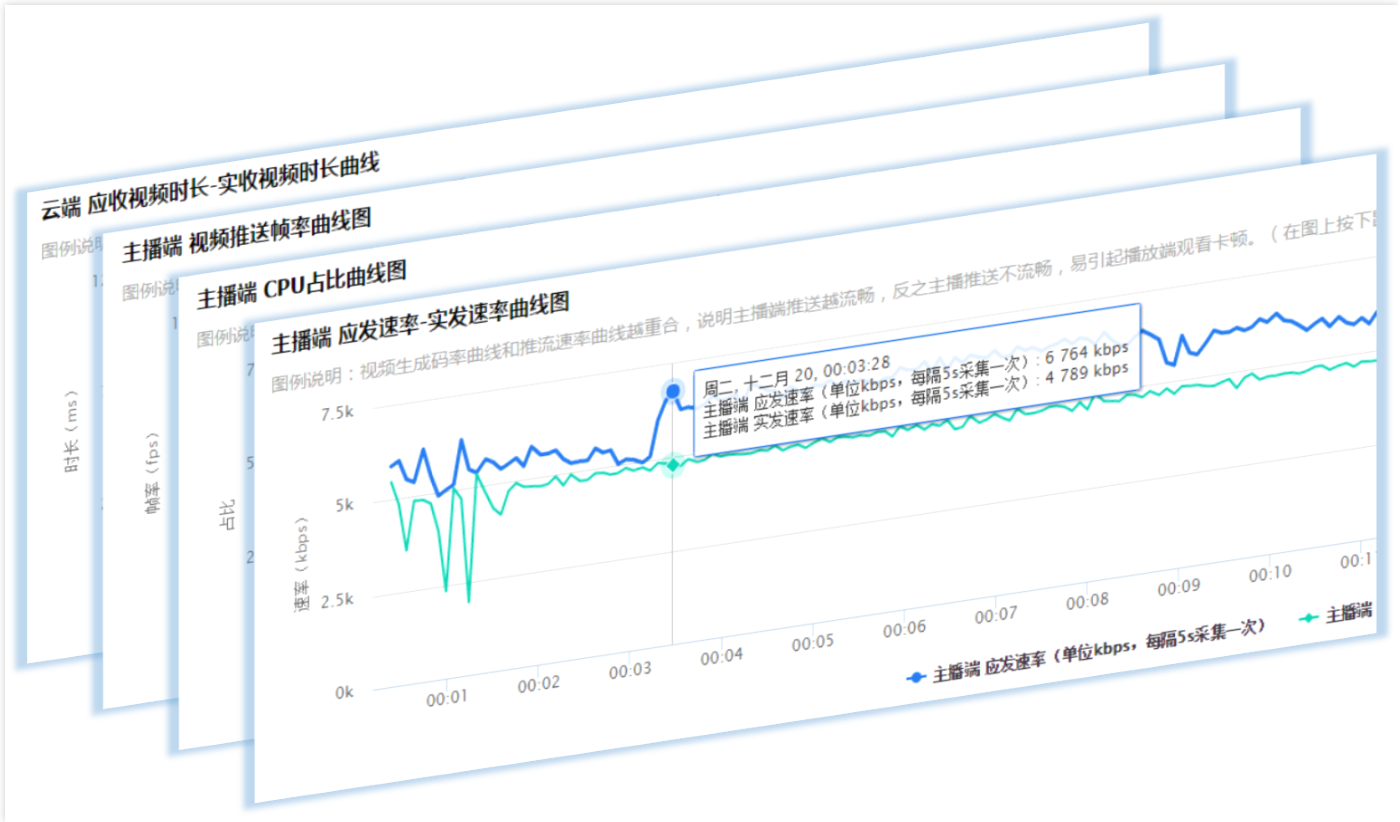
According to statistics, upstream clog at VJ end is responsible for over 80% of stutters among the video cloud's customers.

4.1 Identify upstream clog

- **4.1.1: Relation between BITRATE and NET_SPEED**
  BITRATE (= VIDEO_BITRATE + AUDIO_BITRATE) refers to the number of audio/video data bits produced by the encoder for push per second; NET_SPEED refers to the number of data bits pushed actually per second. A long duration of BITRATE == NET_SPEED means a good push quality. However, a long duration of BITRATE >= NET_SPEED indicates a bad push quality.
- **4.1.2: CACHE_SIZE and DROP_CNT values**
  Once BITRATE >= NET_SPEED, the audio/video data produced by the encoder will build up on VJ's phone, with the severity indicated by the CACHE_SIZE value. When the CACHE_SIZE value exceeds the warning level, SDK will actively drop some audio/video data, thus triggering an increment of DROP_CNT. The figure below shows a typical upstream clog, with CACHE_SIZE remaining above the **red warning level**. This means that the upstream bandwidth doesn't meet the data transfer requirements

(i.e. the upstream network is severely clogged).





Note:
The figure similar to the above can be found in **LVB Console** -> **Quality Monitor**.

4.2 Solutions

- **4.2.1 Notify VJ of the bad network condition**
  In a scenario where video quality is important, it's the best practice to notify the VJ through appropriate UI interactions, such as **"The network condition is bad now. Please move closer to your router, and make sure the signal isn't blocked by any wall or obstacle."**
  For more information on how to do this, please see the **Event Handling** section in RTMP SDK's documentation about push. VJs generally are not aware of the upstream clog until receiving a notification from the App or a viewer. Therefore, it is recommended to remind the VJ about the network condition if the App receives multiple **PUSH_WARNING_NET_BUSY** events from RTMP SDK in a short time.
- **4.2.2 Proper encoding settings**
  The following shows the recommended encoding settings (suitable for beauty show LVB. For more information, please see How to Improve Video Quality?). You can set different video quality options using API setVideoQuality of TXLivePush.

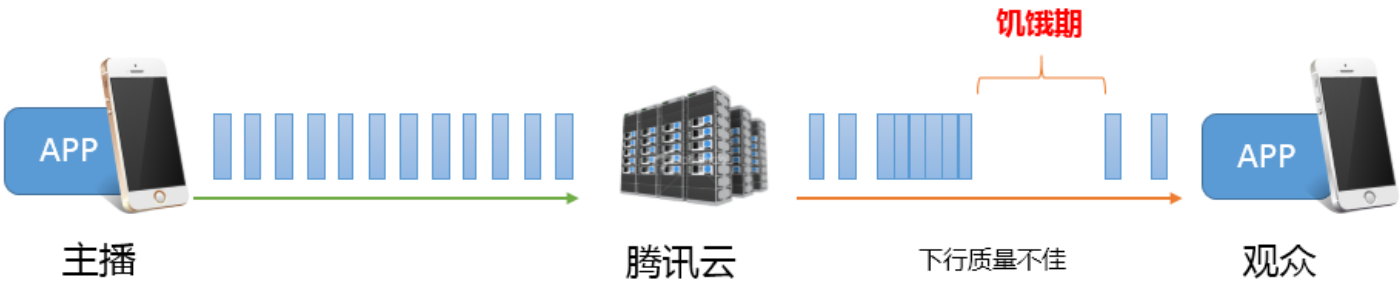
Option	Resolution	FPS	Bitrate	Scenario
Standard Definition	360*640	15	400-800 Kbps	Choose this option if you are more concerned about bandwidth cost. This option can bring a blurred video quality but reduce bandwidth cost by 60% compared to a high definition.
High Definition (recommended)	540*960	15	1200 Kbps	If you're more focused on video quality, select this option, which allows most mainstream mobile phones to present clear pictures.
Ultra High Definition	720*1280	15	1800 Kbps	This option is not recommended for scenarios where videos are mostly viewed in small screens. You can consider using this option if videos are viewed in large screens and the VJ has a great network.

- **4.2.3 Enable traffic control**
  Some clients may complain: "Any user can use our App. It's impossible to control their network conditions." If VJs' upstream bandwidth varies significantly, it's recommended to enable network adaption by referring to the documentation for iOS and Android. However, the solution described in

4.2.1 Notify VJ of network condition is the preferred one. After all, it's unrealistic to ensure the smoothness and high definition while achieving a high upstream bandwidth.

Field	Definition	Recommended Value
videoBitrateMin	The minimum video bitrate	400
videoBitrateMax	The maximum video bitrate	1,000 (the recommended value depends on the resolution)
enableAutoBitrate	Bit rate adaption	Enabled
autoAdjustStrategy	Bit rate adjustment strategy	AUTO_ADJUST_BITRATE_STRATEGY_2

5. Player Optimization



5.1 Stutter & lag

As shown in the figure above, instable downstream network or insufficient downstream bandwidth could cause **starvation periods** during playback. During these periods, the App doesn't receive enough audio and video data to play. To minimize the incidence of stutters at viewer end, you need to make the App cache as much video data as possible to survive the "starvation periods". However, caching too much audio and video data brings a new problem - **high delay**, a bad news for scenarios with a high requirement for interactions between VJ and viewers. Moreover, if the delay caused by stutter goes uncontrolled without any correction, it could **accumulate over time** (i.e. the longer the playback lasts, the higher the delay). Delay correction is a key indicator of an excellent player. Therefore, **delay and smoothness are the two ends of a balance**. Focusing too much on low delay will lead to slight network

fluctuations that produce significant stutter at the player side. Conversely, overemphasis on smoothness will cause high delay. A typical case is the HLS (m3u8) protocol, which ensures a smooth playback experience by introducing a delay of 10-30 seconds.

5.2 Solutions

To allow you to get a better playback experience without the need to have much knowledge about traffic control and processing, Tencent Cloud RTMP SDK, being optimized over several versions, features a set of automatic adjustment technologies, based on which three excellent delay control schemes are introduced:

- **Auto**: Use this mode if you are unsure about what is your main scenario.

  > You can enter the Auto mode by turning on the setAutoAdjustCache switch in TXLivePlayConfig. In this mode, the player will automatically adjust delay based on current network conditions (by default, the player will automatically adjust delay within the range of 1s-5s. You can use setMinCacheTime and setMaxCacheTime to modify the default) to minimize the delay between VJ and viewers while ensuring sufficient smoothness, and thus to deliver a good interactive experience.

- **Speedy**: Suitable for **live shows** and other scenarios with a high requirement for delay.

  > Speedy mode (set by making **SetMinCacheTime = setMaxCacheTime = 1 second**) and Auto mode only differ in MaxCacheTime value (generally, MaxCacheTime is lower in Speedy mode and is higher in Auto mode). This flexibility can be largely attributed to the automatic control technology within the SDK, which automatically adjusts delay without causing stutter. MaxCacheTime is used to indicate the adjustment speed - the higher the MaxCacheTime value is, the more conservative the adjustment speed is, and therefore the lower the probability of stutter becomes.

- **Smooth**: Suitable for **Game LVB** and other HD (high bitrate) LVB scenarios.

  > You can enter the Smooth mode by turning off setAutoAdjustCache switch in the player. In this mode, the player uses a processing strategy similar to the caching strategy of the Adobe Flash kernel. When stutter occurs in a video, the video will go into the loading status until the cache is full; then it will go into the playing status until the next network fluctuation that can't be resisted.

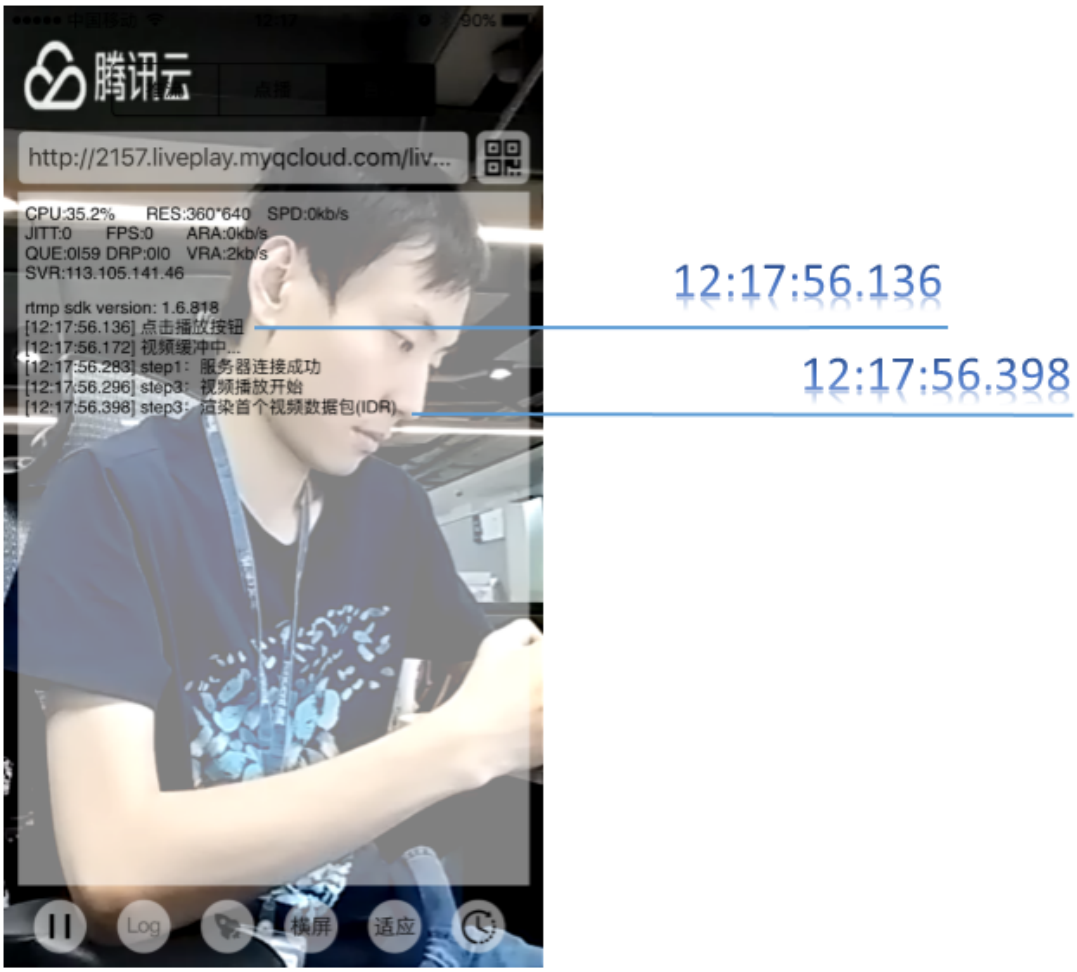
By default, the cache time is 5 seconds, which you can change using setCacheTime.
This seemingly simple mode will be more reliable in scenarios with a low requirement for delay, because the mode in essence trades off delay slightly for a reduced stutter rate.

Instant Broadcasting

Last updated：2018-07-11 12:02:24

What is "Instant Broadcasting"?

Instant Broadcasting is to make the time length between the start of video playback and the moment the first frame is displayed as short as possible (in hundreds of milliseconds) to prevent viewers from waiting long.



This depends on optimized cloud services and player. In an LVB, a combination of Tencent Cloud Audio/Video SDK and video cloud service allows you to open the first frame at a speed as high as about **200ms**, and even open it instantly if you have sufficient downstream bandwidth.

How to Achieve Instant Broadcasting?

Apps

Use Tencent Cloud Audio/Video SDK and the FLV playback protocol to achieve instant broadcasting:

- **HTTP + FLV playback protocol**
  The HTTP + FLV playback protocol is the most widely used protocol in the LVB industry. Thanks to its simple data format, it allows immediate access to Audio/Video data once the server is connected. In contrast, the RTMP protocol provides a slightly lower first frame rate than the FLV protocol due to the several negotiation handshakes required to establish a connection.
- **Tencent Cloud Audio/Video SDK**
  The way instant broadcasting works on cloud is quite simple. With a group of GOP pictures (containing at least a key frame for decoding) always cached on a server, the player can obtain a key frame (I frame) once it is connected to the server, and then proceed with decoding and playing. Such caching on cloud, however, has some side effects: the player is often suddenly stuffed with several seconds of audio/video data after it is connected to a server, which will cause a major delay on the player end. We call it "side effect of instant broadcasting".
  Besides instant broadcasting, a good player should also have excellent **delay correction** ability that automatically corrects player end delay to a proper level (such as less than 1 second) without affecting the viewing experience. Tencent Cloud Audio/Video SDK is a great way to achieve this, even allowing you to specify a delay correction mode for a player (iOS/Android).

PC browsers

The video playback kernels on PC browsers usually use the Flash control (Chrome also supports MSE, but it has no obvious advantage over Flash). Flash players adopt a rigid **forced buffering** strategy, providing little room for optimization of video loading speed, which is unlikely to be kept within 1 second. This fact can be found on many video websites and LVB platforms when you're using a PC browser.

Mobile browsers

- **iPhone**
  Safari works perfectly with HLS (m3u8) and even allows using iPhone's hardware decoding chip to facilitate video playback. Usually, you do not have to worry about the video loading speed as long as DNS caches are available, but this is limited to the iOS platform.
- **Android**
  Due to the severe fragmentation, the performance on the Android platform varies greatly with the system browsers that come along with the various OS versions and devices. The browsers within QQ and WeChat even have Tencent's X5 kernel installed, which would result in large performance variations.

Reduce Latency

Last updated：2018-07-11 12:03:43

Generally, the delay of RTMP push+FLV playback is about 2-3 seconds. An extra-long delay indicates an exception. In case of an extra-long delay, perform the troubleshooting by following steps below:

Step 1. Check the playback protocol

It is quite normal that many customers using HLS (m3u8) as the playback protocol experience a longer delay. Apple's HLS is a streaming protocol based on a total of 3-4 large-granular TS fragments, with each fragment featuring a time period of more than 5 seconds. Therefore, it is not unusual that the total delay may reach about 20-30 seconds.
To solve the problem, use the FLV protocol instead. Please note that only HLS (m3u8) playback protocol can be selected if you want to watch LVB on mobile browsers. Other LVB protocols are not supported on Apple's Safari browser.

Step 2. Check the player settings

Tencent Cloud RTMP SDK's player supports Speedy, Smooth and Auto modes. For more information about the settings, please see adjusting delay.

- **Speedy**: With a delay within 2-3 seconds in most scenarios, this mode is suitable for Beauty Show LVB.
- **Smooth**: With a delay within 5 seconds in most scenarios, this mode is suitable for scenarios that are insensitive to delay but have a high requirement for smoothness (such as Game LVB).

Step 3. Disable watermarking in background

Tencent Cloud supports watermarking in background to cater for the customers who cannot use Tencent Cloud RTMP SDK's pusher (supporting watermarking for live streaming) but have a need for watermarking. However, this solution will introduce an extra three second delay. If you are using the Tencent Cloud RTMP SDK to push streams, disable the watermarking in background and then perform watermarking on the App of VJ.

Step 4. Third-party pusher

The desired effect can be ensured only when Tencent Cloud integrated solution is used. Many third-party pushers deal with insufficient upstream bandwidth through unbounded buffer. If you're using a third-party pusher, you're recommended to use Tencent Cloud RTMP SDK's push Demo to make a comparison to eliminate the possibility that the third-party pusher causes an extra-long delay due to the encoding buffer.

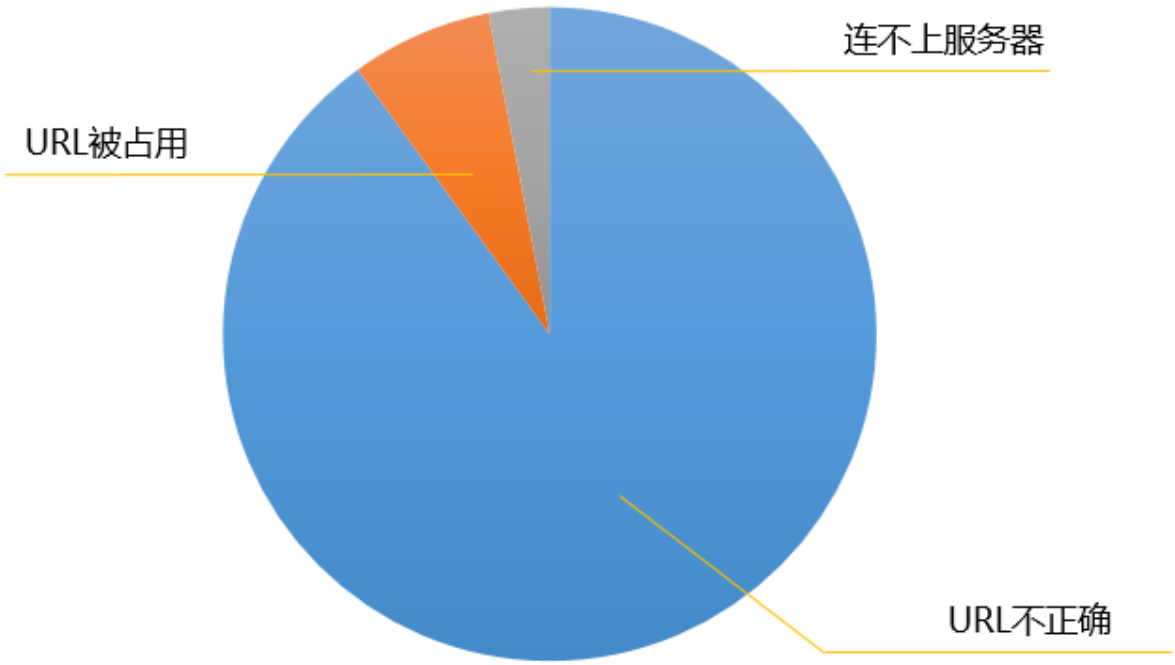
Step 5. Check OBS settings

Many customers who use OBS for push report a long delay at viewer end. It is recommended to configure parameters as described in push on PC. Be sure to set the key frame interval to 1 or 2.

Pushing Failure

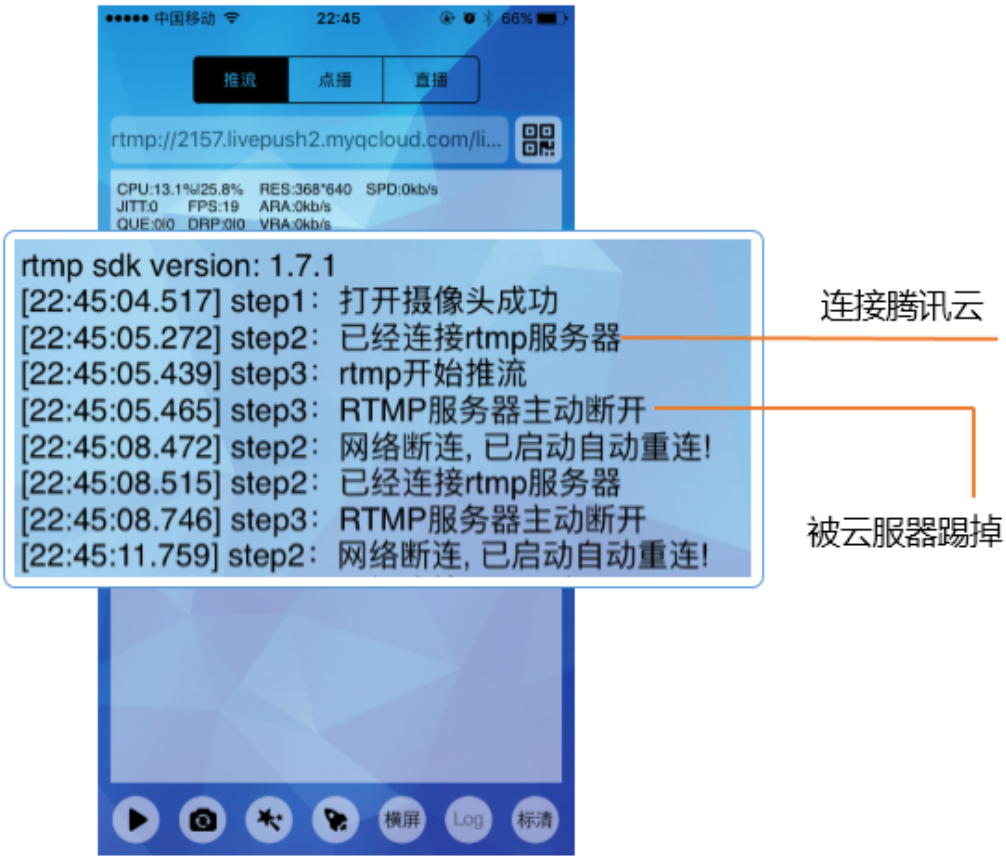
Last updated：2018-07-11 12:05:48

Tencent Cloud's customer complaints about unsuccessful push are mainly caused by the following three reasons:



Incorrect txSecret

Tencent Cloud requires adding Hotlink protection to all push URLs to ensure security. Miscalculated hotlink protection or expired push URLs will be **rejected** by Tencent Cloud. In this case, RTMP SDK will throw a **PUSH_WARNING_SERVER_DISCONNECT** event, and RTMP SDK DEMO behaves as follows:



Please see How to Get the Push URL to learn how to get a reliable push URL.

Expired txTime

Some customers who worry about their LVB traffic being hacked would set a shorter txTime, such as 5 minutes after the current time. In fact, with txSercet signature, you don't need to set such a short validity period. Furthermore, with a too-short validity period, in case of a network interruption during LVB, the VJ would not be able to resume push due to the expiration of push URL.
It's recommended to set the txTime to a value that is 12 or 24 hours after the current time, making the validity period longer than a normal LVB duration.

Push URL is in use already

A push URL can only be used by one pusher, and any other client attempting to push will be rejected by Tencent Cloud. In this case, RTMP SDK throws a **PUSH_WARNING_SERVER_DISCONNECT** event.

Unable to connect to CVM

The default port number used by RTMP push is **1935**. If the firewall of the network for your test doesn't open the port 1935 to the Internet, you may be unable to connect to the CVM. In this case, you can verify whether the problem is caused by this reason by changing network (for example, use 4G instead).

Push URL for Mini LVB

The push URL for Mini LVB can be obtained through debugging. You can search for the keyword **startPush** in the global search, then set a debugging breakpoint, where the RTMP SDK is called by Mini LVB. The parameter startPush is the push URL.

Watching LVB Failure

Last updated：2018-07-11 12:06:03

If you're unable to watch the LVB and have no idea what goes wrong with it, you can identify the cause of the problem in a short time by following the steps below:

Step 1. Check the playback URL

First of all, check whether the playback URL is correct. An incorrect URL is the most likely cause of most problems. Tencent Cloud's LVB URLs include push URL and playback URL. You need to first verify whether **the push URL is accidentally used as the playback URL**.

RTMP推流地址	rtmp://6666.livepush.myqcloud.com/live/6666_xxxxxxxxxxx?bizid=6666
RTMP播放地址	rtmp://6666.liveplay.myqcloud.com/live/6666_xxxxxxxxxxx
FLV播放地址（荐）	http://6666.liveplay.myqcloud.com/live/6666_xxxxxxxxxxx.flv
HLS播放地址	http://6666.liveplay.myqcloud.com/6666_xxxxxxxxxxx.m3u8

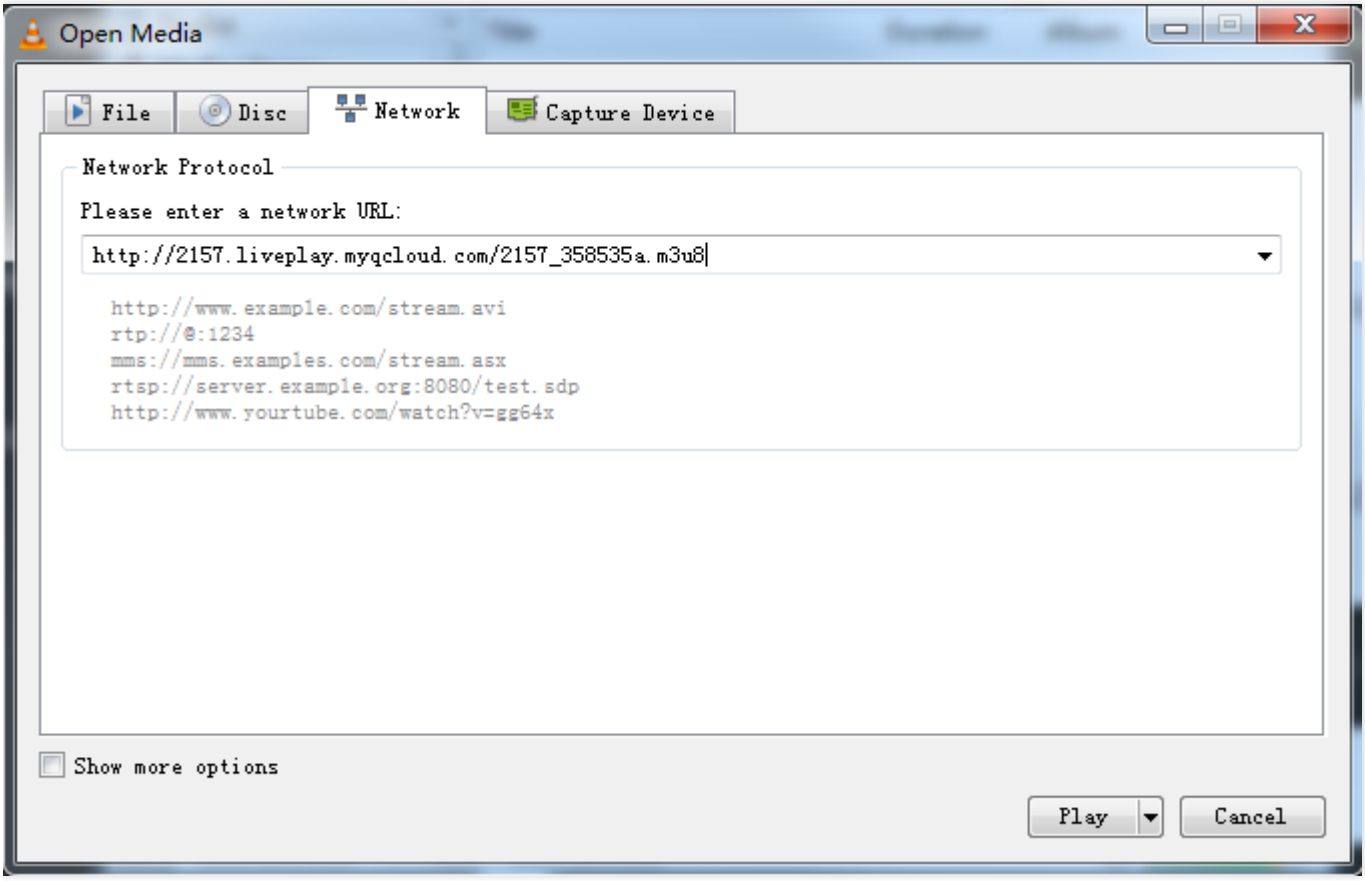
Playback URL for Mini LVB:
The playback URL for Mini LVB can be obtained through debugging. You can search for the keyword **startPlay** in the global search, then set a debugging breakpoint, where the RTMP SDK is called by Mini LVB. The parameter startPlay is the playback URL.

Step 2. Check the video stream

A correct playback URL does not always mean a normal playback. Next, you need to check whether the video stream is normal:

- In **LVB**, the LVB URL becomes unavailable once the VJ stops the push.
- In **VOD**, if the video files have been removed, watching videos is also impossible.

A frequently used solution is making a check using VLC, an open-source player on PC that supports many protocols.



Step 3. Check the player

If there's no problem with the video stream, then you need to check whether the player is normal on a case-by-case basis:

3.1 Web browser (A)

- **Format**: Mobile browsers only support playback URLs in **HLS (m3u8) and MP4** formats.
- **HLS (m3u8)**: Tencent Cloud HLS protocol is based on "Lazy Start". In short, Tencent Cloud only starts the transcoding for HLS format when a viewer requests a playback URL in an HLS format. The purpose is to prevent waste of resources. But it also creates a problem: **The playback URL in an HLS format cannot be played until 30 seconds after the first user in the world initiates a request**.
- **Tencent Cloud Web player:** Supports playback URLs based on multiple protocols, and adopts the optimal playback policy based on the current platform (PC/Android/iOS). The internal selective retry logic can also deal with the Lazy Start of HLS (m3u8).

3.2 RTMP SDK (B)

If RTMP SDK DEMO works normally for playback, it's recommended to check whether the interfacing logic is incorrect by referring to the RTMP SDK playback document (iOS & Android).

Step 4. Check for firewall blocking (C)

It is common that the corporate network environments of many customers restrict video playback through firewalls that detect whether the resources requested by HTTP are streaming media resources (After all, no boss wants his employees to watch videos during working hours). The fact that you can watch the LVB normally over 4G network but cannot watch it over your company's Wi-Fi network indicates your company has imposed restrictions on the network policies. In this case, contact the administrator for a special treatment of your IP.

Step 5. Check the pusher (D)

If the LVB URL does not work and there is no possibility of firewall blocking described in Step 4, it is likely that the push is unsuccessful. Go to Why the Push is Unsuccessful for a further troubleshooting.

Increase Definition

Last updated：2018-07-11 12:06:18

Scenario 1: Beauty LVB

Step 1: Update SDK to the latest version

The beauty filter effects are optimized with each release of new version of SDK. For example:

- In 1.9.1, the beauty filter engine was updated, and many improvements were made to the foreground focus, algorithm, exposure and performance.
- In 1.9.2, the noise reduction effect was optimized to greatly reduce the noises at night and improve the clarity of person images in videos.
- In 2.0.0, more filters were provided for iOS to improve the visual effect of yellowish skin tone.
- ... ...

Step 2: Configure image quality

The video image quality displayed to VJs is different from that displayed to viewers:

- **VJ vs Viewer**:
  The video images seen by a VJ are produced from the captured videos that are directly rendered on the phone screen and thus have the best clarity. The images need to go through **Video Encoding** > **Network Transmission** > **Video Decoding** before being rendered on the phone screen of viewers. The video encoding can affect the image quality, so the images displayed to viewers are not as clear as the ones displayed to the VJ. Inappropriately configured parameters can greatly reduce the image quality. A typical example is "high resolution plus low bitrate", which can cause blurred display and serious mosaics.
- **setVideoQuality**
  In 1.9.1, the function setVideoQuality was added in TXLivePusher with a range of levels available. You can have the best image quality in beauty LVB by simply choosing **HD** mode. For more information, please see iOS Platform and Android Platform.

Step 3: Add manual exposure on Android

The same beauty filter algorithm may yield different effects on different Android phones. This is because the difference between various devices in terms of exposure performance leads to different visual effects. On iOS devices, auto-exposure is adopted. But Android devices are significantly different from iOS ones, and some low-end Android phones have an unsatisfactory auto-exposure effect. Therefore, it is recommended to provide a slider on the page for VJs to adjust exposure value manually as needed.

The API TXLivePush::**setExposureCompensation** in the Android RTMP SDK can be used to adjust exposure, with the parameter value being a float value between -1 and 1: 0 - no adjustment; -1 - minimum exposure; 1 - maximum exposure.

Step 4: Add color filters

Filters are also important because different color filters create different atmosphere. A VJ can choose a filter to match his/her clothes or room light to create a better visual effect.

Color filters have been supported as of RTMP SDK 1.9.1. The setFilter added in TXLivePusher can be used to set filter effect. Eight sets of color filter materials are made available in the Demo. You can use them on a royalty-free basis.

Step 5: Serious mosaics on Android

Some customers find that the images pushed by the Android RTMP SDK have serious mosaics, especially for dynamic images. This is a common problem created by Android hardware encoding, and you can solve it in two ways:

- **For a lower battery drain**
  If you care more about the App's battery drain, increase the bitrate of the pushed streams or use **HD** in setVideoQuality (if you set a low bitrate, the Android's hardware encoding module ensures a consistent bitrate by greatly reducing the image quality).
- **For a lower bandwidth cost**
  If you care more about the bandwidth cost, increasing bitrate may not be a good solution. Instead, you can solve this problem by disabling hardware acceleration. For more information, please see setHardwareAcceleration.

Step 6: Disable network adaption

The **AutoAdjustBitrate** in **TXLivePushConfig** is used to enable/disable network adaption. If it is enabled, the image quality is reduced to ensure smoothness in case of a poor network of VJ. But this feature is **not suitable** for beauty shows. Network adaption is suitable for game LVB scenarios, where viewers attach more importance to smoothness than image quality. If the VJ's network becomes unstable during a battle, it's ok to have a degraded image quality but stutters are totally unacceptable, so it is necessary to trade

off image quality for smoothness (frame rate). However, image quality is more important in beauty show scenarios. Many customers report that the image quality varies greatly with different rooms. This is likely caused by the enabling of network adaption.
We recommend that you disable network adaption and deal with network fluctuations by using system alerts to solve the problem more fundamentally.

Scenario 2: Game LVB

Option 1: Simple approach

Provide three definition options on the LVB starting page - SD, HD and UHD - **for VJs to select from**. A VJ in game LVB can find out which option is suitable for the game he/she is playing. The configurations for the three options are as follows:

Option	Resolution	FPS	Bitrate

Option	Resolution	FPS	Bitrate
SD	VIDEO_RESOLUTION_TYPE_360_640	20	800 kbps
HD	VIDEO_RESOLUTION_TYPE_540_960	20	1000 kbps
UHD	VIDEO_RESOLUTION_TYPE_720_1280	20	1800 kbps

Note:
The minimum FPS in game LVB scenarios is 20. Serious stutters can occur at viewer end in case of an FPS less than 20.

Option 2: Professional approach

Configure different resolutions and bitrates for different games. For example:

- **Clash Royale** - For such a game that features less dynamic images, the combination of a resolution of **960 * 540** and a bitrate between 800 kbps and 1000 kbps can produce a good effect.
- **Fishlord** - For such a game that features more dynamic images, the combination of a resolution of **960 * 540** and a higher bitrate between 1200 kbps and 1500 kbps is recommended.
- **Temple Run** - For such a game that features highly dynamic images, it is recommended to choose a resolution of **640 * 360** and a very high bitrate, for example, 2000 kbps, to avoid serious mosaics.

Tips on Audio/Video

1: A resolution of 720p does not necessarily mean a higher clarity

For a given bitrate, for example, 800 kbps, **a higher resolution will make it harder for an encoder to deliver a good image quality**. The encoder can support sufficient pixels only by decreasing color elements or introducing mosaics. For the same movie file sized at 2 GB, a 1080p resolution may render less clear images than a 720p resolution.
If the viewers watch videos on small phone screens, they won't see much difference between **960 * 540 1000 kbps** and **1280 * 720 1800 kbps**. For instance, the two images below are captured from screencap

LVB on iOS using the airplay technology:

960 * 540 1000kbps

1280 * 720 1800kbps

Note:
You will see the difference if the images are displayed in full-screen mode on a 32-inch LCD screen.

2: Keep the FPS within 24

For a given bitrate, for example, 800 kbps, a higher FPS makes it necessary for the encoder to increase the compression ratio for each frame, which means reducing the image quality to support enough frames. If the video source is camera, then 24 FPS is the maximum frame rate for naked eyes. Therefore, an FPS of 20 is already enough to offer a good user experience. Some 3D game players may ask: "Does a higher frame rate, such as 60 or 120 FPS, mean a higher smoothness? "
It depends on the scenarios: In a game scenario, a higher **rendering frame rate** is recommended to make the motion effects rendered with 3D models more similar to the motion trajectories in the real world. But a high frame rate is not needed for capturing. For example, what are captured by a phone camera are the

objects in reality, which are in motion continuously and are not simulated through refreshes of images, so 20 FPS is enough.
For game LVB, an FPS of 24 is ideal, but you also need to consider such factors as system encoding cost, phone temperature, and CPU utilization.

Enable Cloud Services

Last updated：2018-07-11 12:02:54

1. Live Video Broadcasting (LVB)

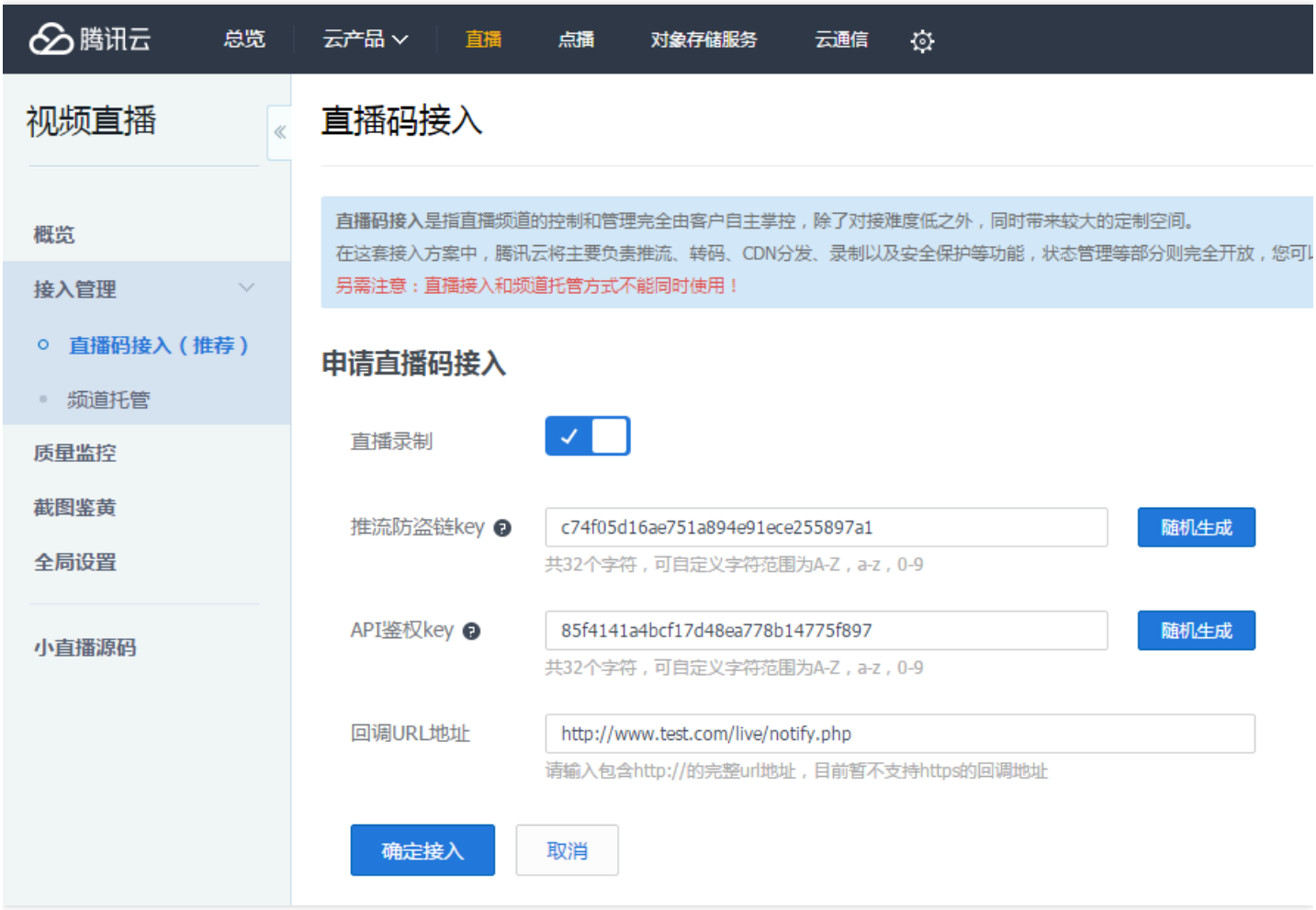
1.1 Activating LVB service

Log in to the LVB Console. If you have not activated the service, the following page appears:

Click **Apply**, and then go to the application approval step. The service is activated upon the approval of Tencent Cloud.

1.2 Enabling LVB Code Access

After activating the LVB service, go to **LVB Console -> LVB Code Access -> Access Configuration** to complete the configurations, and then activate the LVB Code service:



Configuration Item	Value Range	Description
LVB Recording	Enable or Disable	If this is enabled, all LVB videos are recorded in background by default. DOC
Push Hotlink Protection Key	A 32-character lowercase string	To ensure security, the push URL needs to be bound to hotlink protection to avoid being hacked by others. The key is used to calculate the hotlink protection signature. DOC
API Authentication Key	A 32-character lowercase string	Your server needs to provide authentication information when calling the Tencent Cloud backend APIs. The key is used by Tencent Cloud to verify the validity of your server's identity. DOC

Configuration Item	Value Range	Description
Callback URL	HTTP protocol-based URL	Tencent Cloud sends the notifications of push, push interruption and other events to you via this URL. HTTPS protocol is not supported. DOC

Click **OK** to switch your Tencent Cloud LVB service to the LVB Code mode.

1.3 LVB APPID

Each Tencent Cloud account for which the LVB service has been activated is assigned an LVB APPID, which is displayed at the top of LVB Console, as shown below:

1.4 LVB BIZID

Each Tencent Cloud account for which the LVB service has been activated is assigned an LVB BIZID. An LVB BIZID is used to construct the push and playback URLs in the LVB Code. After the LVB Code mode is

enabled, the BIZID appears at the top of LVB Console, as shown below:

1.5 Push hotlink protection key

Push hotlink protection key is a security protection which ensures that only your App users can push streams. The key can be specified when LVB Code is enabled, and can be modified as needed in the LVB Console:

1.6 API authentication key

An API access authentication key is required when your backend server calls the Cloud API related to Tencent Cloud LVB Code, and allows Tencent Cloud to verify the validity of the call. The key can be

specified when LVB Code is enabled, and can be modified as needed in the LVB Console:

1.7 Event notification URL

An event callback URL is an address from your backend server. When Tencent Cloud needs to notify your backend server of some LVB-related Events, it sends the notifications to this address in the form of HTTP POST. The event callback URL can be specified when LVB Code is enabled, and can be modified as needed in the LVB Console.

2. Video on Demand (VOD)

2.1 Activating VOD service

Each new Tencent Cloud account that has completed identity verification is eligible for a seven-day free trial. Upon the expiration of the trial period, you can purchase the package as needed on the VOD Console.

2.2 VOD APPID

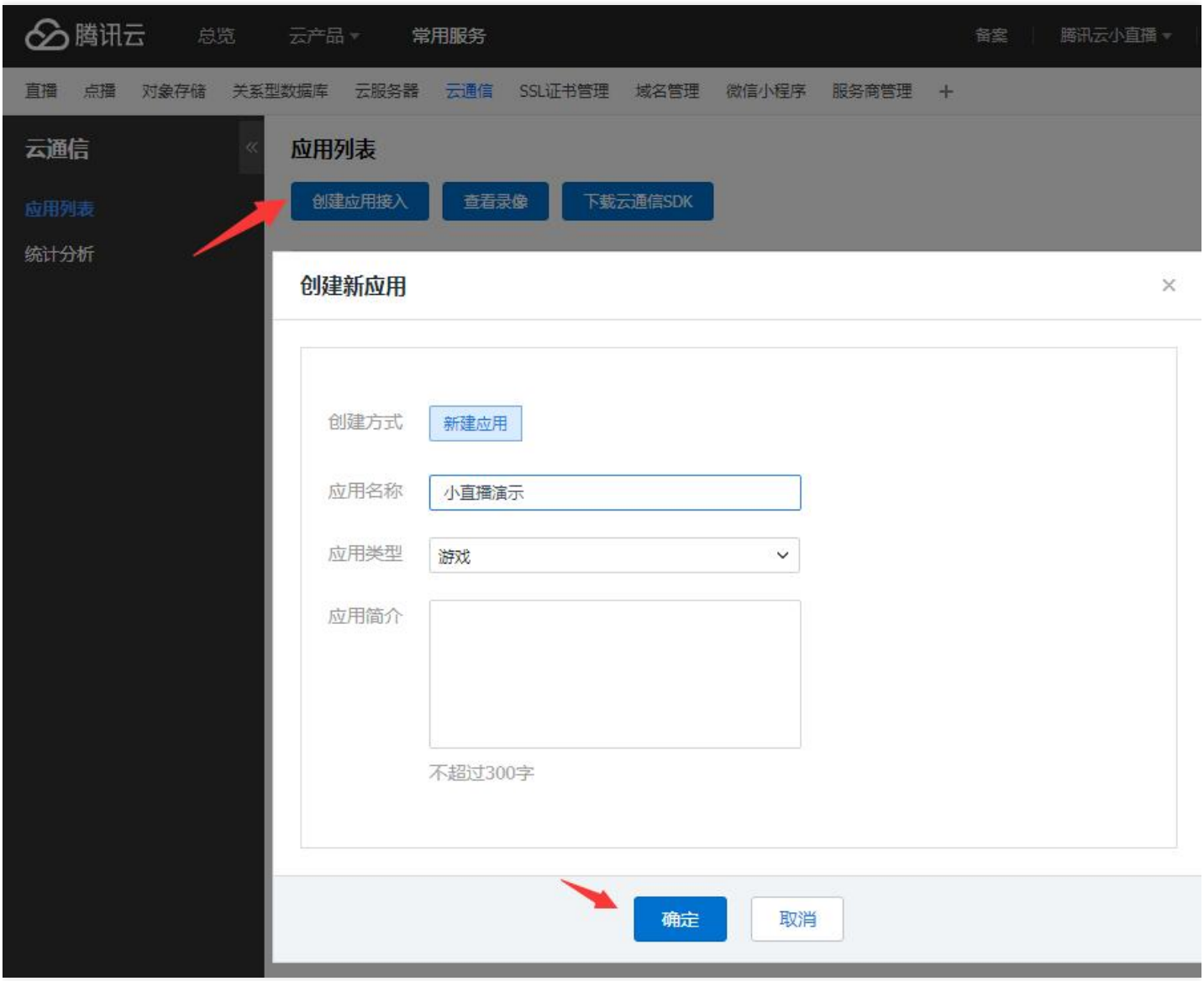
Each Tencent Cloud account has a unique VOD APPID, which is located in an unobvious place on the VOD Console. It is only displayed when you have at least one uploaded or recorded video file under your account, as shown below:

3. Instant Messaging (IM) Service

3.1 Activating IM service

Log in to the IM Console, and then click **Activate IM**.

The IM application list is empty under a new Tencent Cloud account. Click the **New Application** button to create an application:



3.2 SDK APPID

The ID shown in the figure below is an sdkappid, which identifies a product under your Tencent Cloud account. If there are multiple products under your account, multiple sdkappids are provided.

3.3 AccountType

Click **Application Configuration** on the right of the list to proceed with the configuration, as shown below:

Account Name allows letters and numbers. **Account Administrator** is used only when REST API of IM is called.

Click **Save** to complete the IM access configuration for the App. At the same time, AccountType is generated, as shown below:

3.4 Administrator

IM provides a set of REST APIs for your backend server to call IM service directly, for example, to create a group, send system messages and remove a user from the group. But the IM REST API can only be called by an administrator, that is, you need an administrator username and password (UserSig). For more information, please see DOC.

3.5 PrivateKey & PublicKey

IM SDK can be considered as the QQ without user interaction page. Integrating the IM SDK into your App is like integrating a QQ message kernel.

As we all know, QQ can receive and send messages of private chats and group chats, but you need to log in to it before you can use it. Logging in to QQ requires a QQ account and password. Similarly, logging in to IM SDK needs a user-specified username (userid) and password (usersig).

You can specify any username you like, but Tencent Cloud will verifies its validity with asymmetric encryption technology. The encryption key and the decryption key used in asymmetric encryption are different. The private key can be kept on your server to asymmetrically encrypt the userid and appid to generate the usersig. At the same time, Tencent Cloud keeps the corresponding public key to verify whether the usersig is valid and is signed by your server.

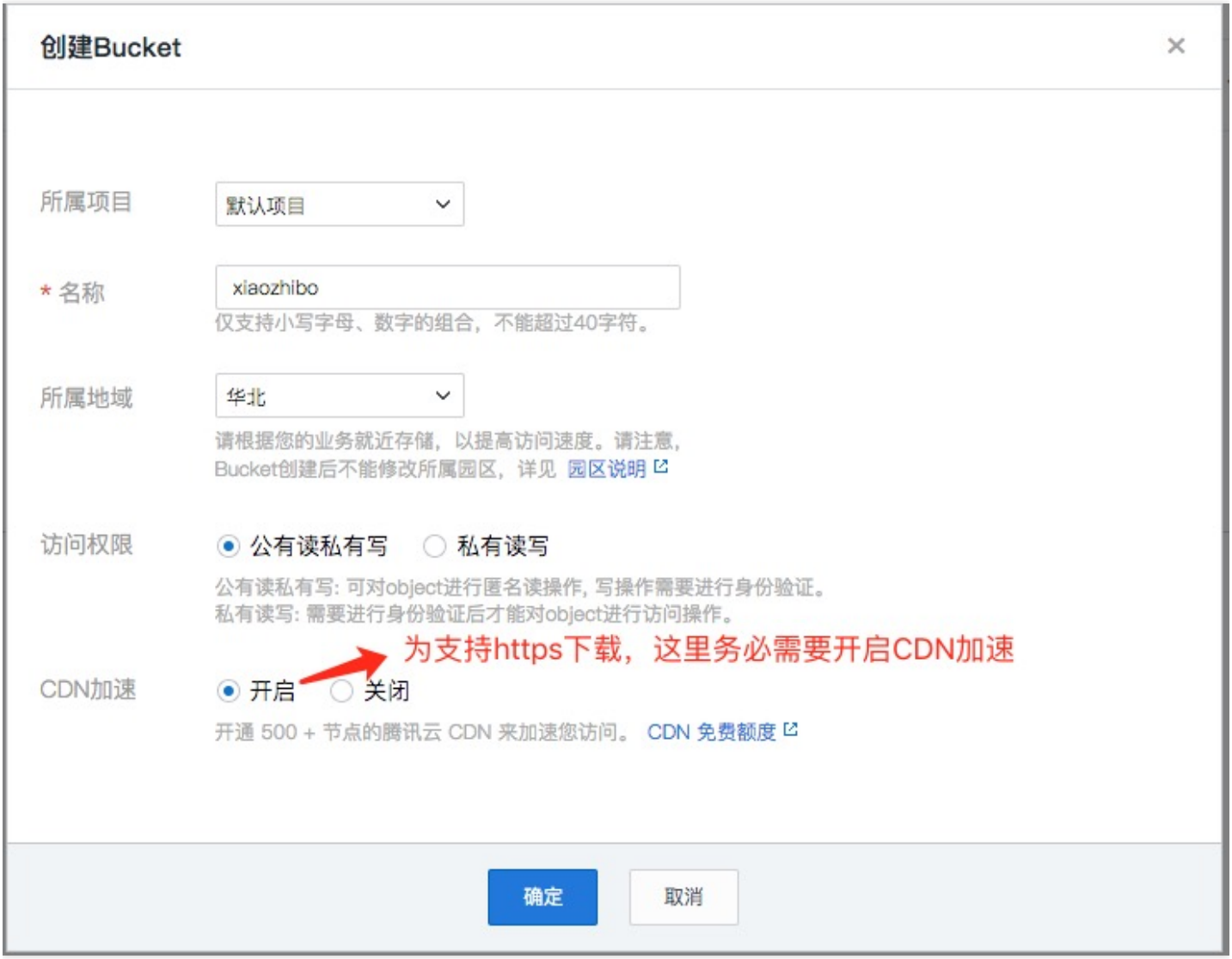
The public and private keys used to calculate the userig signature can be downloaded from the following location:

4. Cloud Object Storage (COS)

4.1 Activating COS service

Cloud Object Storage (COS) is available for any new Tencent Cloud account which has completed identity verification. You can enable it by going to COS Console to create a Bucket. **Note: Enable CDN**

acceleration to support HTTPS download (adapt to Apple's ATS).



4.2 What is Bucket?

Bucket is a technical term that can be simply understood as **disk partition**. For example, the COS service purchased from Tencent Cloud can be compared to a new disk purchased from JD.COM. Generally, you partition and format the disk before storing data on it. A partition you create on the disk is similar to a bucket you create on Tencent Cloud COS.

4.3 Bucket name

The name you specified during bucket creation is a bucket name. For example, in Example 4.1, xiaozhibo is a bucket name.

4.4 COS APPID

Click the Key Management tab on the COS console to get the COS APPID, which must be bound with one pair of APIs.

4.5 COS SecretId and SecretKey

Click the Key Management tab on the COS console to get the SecretId and SecretKey bound with your COS APPID. They are used by APIs for accessing COS. Since COS is a cloud service that has a high requirement for security, if an API fails to transfer a correct key, Tencent Cloud will reject the API request.

5. Cloud Virtual Machine (CVM) (optional)

You can use your own server as the business server for deploying backend script. But you're recommended to use Tencent Cloud's CVM for a higher reliability. In addition, if you select Tencent Cloud's Cloud Database to use as a distributed database, you must use Tencent Cloud CVM to access the database.
Log in to the CVM Console, and then click **Purchase CVM** to go to the CVM purchase page:

Click **Next** to go to the image selection page. You are recommended to select a Linux image with Nginx+PHP+MySQL from the marketplace.

Complete the subsequent steps as instructed by the system. The CVM becomes available when the image is installed.

6. Cloud Database (optional)

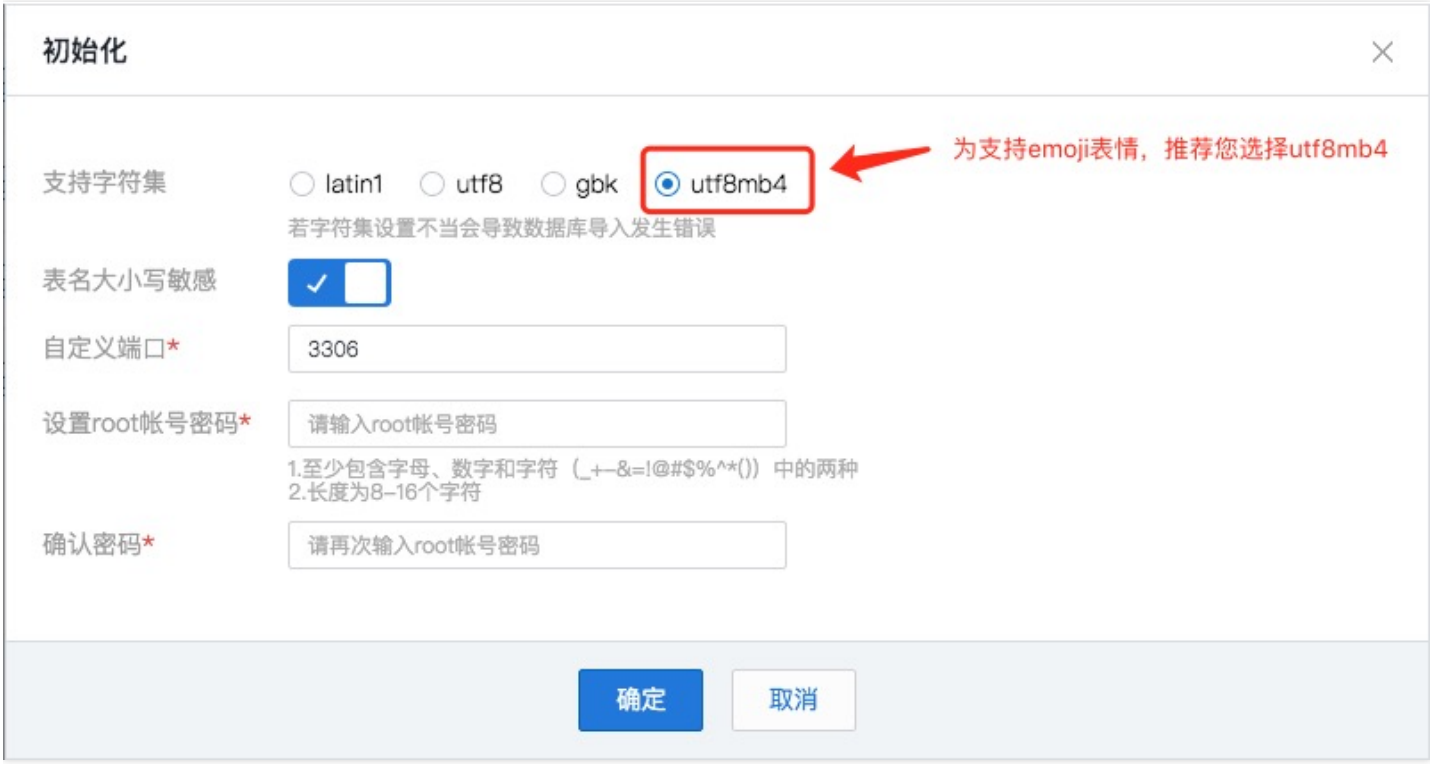
6.1 Activating Cloud Database

Log in to the Cloud Database Console. If you have not activated the CDB (MYSQL) service, click **New**.

Note: The Tencent Cloud account used to activate Cloud Database service should be same as the one used to activate CVM, and the region selected for Cloud Database should be same as the one selected

for CVM.

After the purchase, the instance is displayed in the **Instance List**. Click **Initialize** on the right to set a character set and password for the cloud database.



6.2 Using the cloud database

After initialization of the instance, the private IP of the instance can be found in the **Instance List**.

You can remotely connect to the database instance to perform operations on it using the mysql command on the CVM, or click **Manage** in the **Instance List** to go to the management page to work with the database:

Record and Replay

Last updated：2018-07-11 11:54:43

Feature Description

Recording & Replay means recording a user's entire LVB and replaying it as VOD.

At the early stage after an APP is launched, there are a small number of VJs, so adding recording & replay feature in the LVB list can enrich the APP resources to some degree for viewers.

Even if the number of VJs grows significantly with the increasing popularity of an App, the accumulation of good LVB content is still necessary. In addition to name, photos and other personal information of a VJ, replays of historical LVBs is also an indispensable part of a VJ's profile.

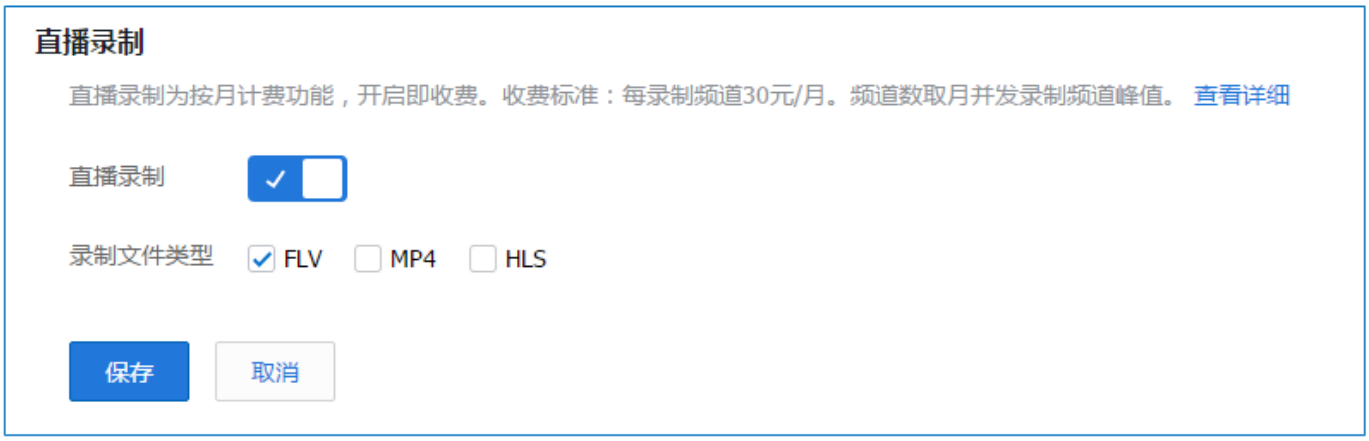
Enabling Recording

Recording & Replay is built on Tencent Cloud's **VOD service**. To use this feature, you need to Activate VOD Service on Tencent Cloud console. After the service is enabled, you can find the new recorded files in Video Management on VOD console.

How to enable recording? Two methods are available:

1. Enable recording globally

You can enable or disable recording for all LVB streams in **LVB Console -> Access Management -> LVB Code Access -> Access Configuration**, as shown below:



Note: The length of a video fragment for global recording defaults to 30 minutes. Submit a ticket if you need to configure the length of a video fragment for global recording.

2. Specify a room for recording

When the recording for all LVB streams is disabled, you can enable recording for important video streams individually by appending **&record=mp4**, **&record=hls** or **&record=mp4|hls** to the push URL.

```
rtmp://2121.livepush.myqcloud.com/live/2121_15919131751?txSecret=aaa&txTime=bbb&record=mp4&record_interval=5400
```

Notes:

- The video wrapper formats supported by the recording feature are MP4, HLS and FLV. For more information about the VOD formats, please see DOC.

- The delimiter-separated format record=mp4|hls|flv is used to specify multiple formats in which the video is recorded at a time (mobile browsers only support playback of MP4 and HLS videos).
- Changing resolution or switching between landscape/portrait modes during LVB is not supported for MP4 videos.
- If the specified recording format is flv or mp4, you can specify the recording length of a single video fragment using the parameter record_interval (in sec). The maximum length is 120 minutes (i.e. 7,200 seconds). If no value is specified, the default length is 30 minutes (i.e. 1,800 seconds).
- HLS (m3u8) file is on a fragmentation basis in essence, so you can always get a single m3u8 file as long as no push interruption occurs during LVB. But in case of a push interruption during LVB, fragmentation will occur in the process of recording (you will get multiple m3u8 files). One of the common problems would be switching of App to background. To solve this problem, you're recommended to use background push solution.
- You can use the parameter record_type=audio to specify recording audio only.

Getting Files

When a new recorded video file is generated, a playback URL will be generated. You can process it based on your business needs. In Mini LVB, we directly put together the recording file URL and the room list to fill in the gap of insufficient online VJs.

You can implement many extensions based on your business scenarios. For example, you can add the URL to a VJ's profile as a historical program for that VJ, or put it in the replay list to recommend a high-quality video to your App users after manual filtering.

You can get the URL of the recorded file by the following two ways:

1. Listen to notification passively

You can use Tencent Cloud **Event Notification Service**: Register a **callback URL** for your server on Tencent Cloud which will notify you of the generation of a new recorded file using this URL.

The following is a typical notification message, which indicates: a new FLV recorded file with ID 9192487266581821586 has been generated, and the playback URL is: `http://200025724.vod.myqcloud.com/200025724_ac92b781a22c4a3e937c9e61c2624af7.f0.flv` .

```
{
"channel_id": "2121_15919131751",
"end_time": 1473125627,
"event_type": 100,
"file_format": "flv",
"file_id": "9192487266581821586",
"file_size": 9749353,
"sign": "fef79a097458ed80b5f5574cbc13e1fd",
"start_time": 1473135647,
"stream_id": "2121_15919131751",
"t": 1473126233,
"video_id": "200025724_ac92b781a22c4a3e937c9e61c2624af7",
"video_url": "http://200025724.vod.myqcloud.com/200025724_ac92b781a22c4a3e937c9e61c2624af7.f0.flv"
}
```

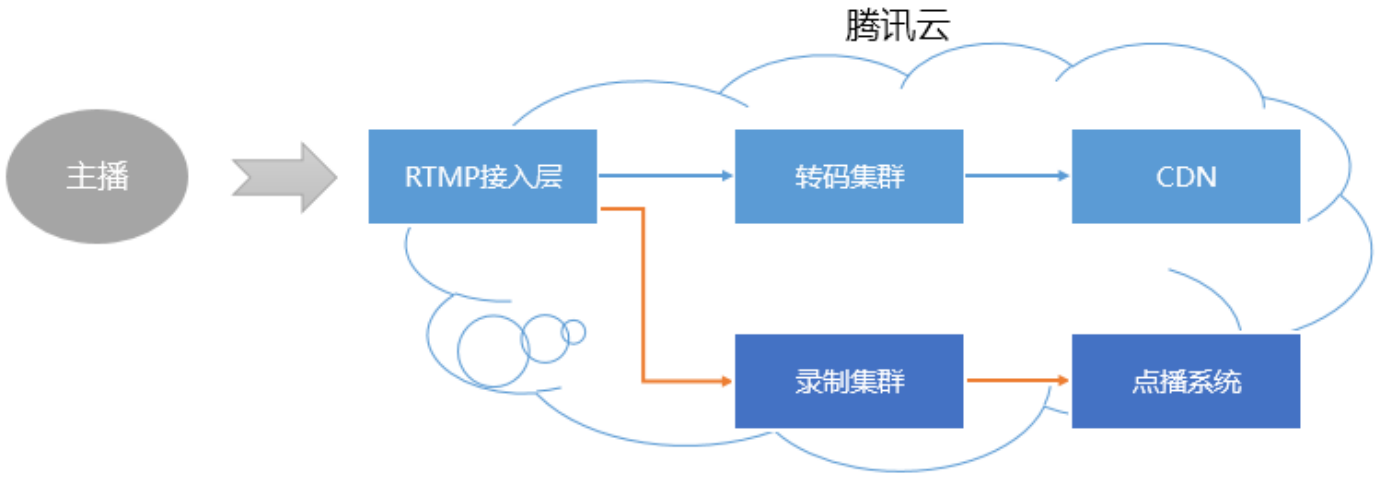
2. Query files actively

You can check if any new recorded file is generated on a regular basis using Tencent Cloud's query API (**Live_Tape_GetFilelist**). However, this method is not recommended for frequent use due to its

unsatisfactory real-timeness and reliability since it has a slow response in case of a query for a large number of channels and cannot be called at a high frequency (only suitable for the channels that have just finished).

FAQ

1. How does LVB recording work?



When you enable the recording for an LVB stream, the audio/video data is bypassed to the recording system. Every frame pushed from VJ's mobile phone is written into the recorded file by the recording system.

If an LVB push is interrupted, the access layer will immediately notify the recording server to record the file being written, store it into the VOD system and generate an index. Then you can find the new recorded file in the VOD system. If you have configured recording event notification on a server, the recording system will send the **index ID** and **online playback URL** to the server.

However, an error will occur in the processes of transferring and processing of a large file on the cloud. To ensure success, the maximum recording length of a file is 90 minutes, and you can specify a shorter fragment using parameter record_interval.

2. How many recorded files are generated in an LVB process?

- If the duration of an LVB is too short (for example, shorter than 1 second), no recorded file is generated.

- If the duration of an LVB is not long (shorter than record_interval), and the push is not interrupted during LVB, only one recorded file is generated.

- If the duration of an LVB is very long (longer than record_interval), the video will be fragmented based on the length of time specified in record_interval, to avoid the time uncertainty of the flow of the file with a longer duration in a distributed system.

- If the push is interrupted during an LVB (SDK will re-push later), a new fragment is generated every time the interruption occurs.

3. How do I know which files belong to a certain LVB?

To be precise, as PAAS, Tencent Cloud does not know how to define your LVB. If one of your LVBs lasted for 20 minutes, during which a push interruption caused by network switching occurred, and the LVB was stopped and restarted once manually. Should we calculate the number of LVBs as one or three?

For normal MLVB scenarios, the period between the following two interfaces is defined as one LVB session:

Therefore, the time information sent from APP client is very important. If you wish to define that the files recorded during this period belong to this LVB, you just need to retrieve received recording notification using LVB code and time information (each recording notification event will come with information such as stream_id, start_time and end_time).

4. How do I stitch fragments?

Tencent Cloud supports stitching video fragments with cloud API. For more information on how to use this API, please see Video Stitching.

Compatibility with Apple ATS

Last updated：2018-07-11 12:06:40

Apple Inc. announced in WWDC 2016 that by default all new Apps submitted as of January 1, 2017 will not be allowed to use `NSAllowsArbitraryLoads=YES` to bypass ATS restriction. Tencent Cloud will officially support HTTPS as of December 12th. By then, you just need to use the new SDK version (API remains the same) and change the video URLs' prefix from `http://` to `https://`. The new SDK can be automatically adapted to the change.

Please note that compared with HTTP, HTTPS brings a higher security, which is not absolutely necessary for videos, while leading to a reduced connection speed and a higher CPU utilization. If your App still needs to use HTTP under Apple's new policy, you need to modify Info.plist by adding `myqcloud.com` to `NSExceptionDomains`, as shown below.

▼ App Transport Security Settings ⌃⌄	Dictionary	(1 item)
▼ Exception Domains ⌃⌄	Dictionary	(3 items)
▼ myqcloud.com	Dictionary	(3 items)
NSExceptionRequiresForwardSecrecy	Boolean	NO
NSExceptionAllowsInsecureHTTPLoads	Boolean	YES
NSIncludesSubdomains	Boolean	YES

Disabling ATS for specific domain names can be approved by Apple's audit team, but you may need to specify that `myqcloud.com` is a domain name for video playback.

Event Code

Last updated：2018-07-10 14:28:54

PushEvent

Normal events

A notification event is prompted after each successful push. For example, receiving 1003 means that the system will start rendering the camera pictures.

code	Event	Description
1001	PUSH_EVT_CONNECT_SUCC	Connected to the CVM
1002	PUSH_EVT_PUSH_BEGIN	Handshake with the server completed; everything is OK; ready to start upstream push
1003	PUSH_EVT_OPEN_CAMERA_SUCC	Camera enabled. Cannot enable the camera if the camera has been occupied or you have limited permission to the camera.
1004	PUSH_EVT_SCREEN_CAPTURE_SUCC	Screencap started successfully (for Android SDK)
1005	PUSH_EVT_CHANGE_RESOLUTION	Dynamic push resolution change
1006	PUSH_EVT_CHANGE_BITRATE	Dynamic push bitrate change
1007	PUSH_EVT_FIRST_FRAME_AVAILABLE	The first frame is captured
1008	PUSH_EVT_START_VIDEO_ENCODER	Encoder is started
1009	PUSH_EVT_CAMERA_REMOVED	Camera has been removed (for Windows SDK)
1010	PUSH_EVT_CAMERA_AVAILABLE	Camera is available again (for Windows SDK)
1011	PUSH_EVT_CAMERA_CLOSED	Camera is disabled (for Windows SDK)
1012	PUSH_EVT_SNAPSHOT_RESULT	Error code of screencap snapshot (for Windows SDK)
1018	PUSH_EVT_ROOM_IN	Already in the webrtc room and will be notified after successful entry into the room

code	Event	Description
1019	PUSH_EVT_ROOM_OUT	Not in the webrtc room and will be notified after failure to enter the room or in the process of exiting the room
1020	PUSH_EVT_ROOM_USERLIST	Issue the webrtc room member list (excluding the user him/herself)
1021	PUSH_EVT_ROOM_NEED_REENTER	Switching Wi-Fi to 4G will trigger disconnection and reconnection, which requires you to re-enter the webrtc room (pull the optimal server address)

Critical errors

The push cannot continue as the SDK detected critical problems. For example, the user disabled camera permission for the App so the camera cannot be started.

> Note: A video encoding failure does not affect push process directly. The SDK will handle it automatically to ensure success of the subsequent video encoding.

code	Event	Description
-1301	PUSH_ERR_OPEN_CAMERA_FAIL	Failed to enable the camera
-1302	PUSH_ERR_OPEN_MIC_FAIL	Failed to enable the microphone
-1303	PUSH_ERR_VIDEO_ENCODE_FAIL	Video encoding failed
-1304	PUSH_ERR_AUDIO_ENCODE_FAIL	Audio encoding failed
-1305	PUSH_ERR_UNSUPPORTED_RESOLUTION	Unsupported video resolution
-1306	PUSH_ERR_UNSUPPORTED_SAMPLERATE	Unsupported audio sampling rate
-1307	PUSH_ERR_NET_DISCONNECT	The network is disconnected and fails to be reconnected after three attempts, so the push needs to be restarted

code	Event	Description
-1308	PUSH_ERR_CAMERA_OCCUPY PUSH_ERR_AUDIO_SYSTEM_NOT_WORK PUSH_ERR_SCREEN_CAPTURE_START_FAILED	Camera is in use and you may try enabling a different camera (Windows) System exception and recording failed (iOS) Failed to start screencap, which is possibly rejected by the user (Android)
-1309	PUSH_ERR_SCREEN_CAPTURE_UNSURPORT	Screencap failed due to unsupported Android OS version; 5.0 or above is required (for Android SDK)

Warning events

Most warning events will trigger protection logics or recovery logics that involve retrying, and in most cases the problems can be recovered by the SDK itself. Some warning events need to be handled by developer.

- **WARNING_NET_BUSY:**If the VJ has poor network connections, the user can be notified through UI.
- **WARNING_SERVER_DISCONNECT:** The push request is rejected by the backend. This is usually caused by miscalculated txSecret in the push address, or because the push address is occupied by others (a push URL can only have one pushing end at a time).

code	Event	Description
1101	PUSH_WARNING_NET_BUSY	The upstream network is poor. It is recommended to remind users of improving the current network environment.
1102	PUSH_WARNING_RECONNECT	Network disconnected and auto reconnection has started (auto reconnection will be stopped after three failed attempts)
1103	PUSH_WARNING_HW_ACCELERATION_FAIL	Failed to start hardware encoding. Use software encoding instead.

code	Event	Description
1104	PUSH_WARNING_VIDEO_ENCODE_FAIL	Video encoding failed. Non-fatal error. Encoder will be restarted internally.
1107	PUSH_WARNING_VIDEO_ENCODE_SW_SWITCH_HW	Automatically switch to hardware encoding due to machine performance issues (for Android SDK)
3001	PUSH_WARNING_DNS_FAIL	DNS resolution failed and trigger retry process.
3002	PUSH_WARNING_SEVER_CONN_FAIL	Failed to connect to the server and trigger retry process.
3003	PUSH_WARNING_SHAKE_FAIL	Server handshake failed and trigger retry process.
3004	PUSH_WARNING_SERVER_DISCONNECT	The RTMP server is actively disconnected. Check the validity of the push address or the validity period of hotlink protection
3005	PUSH_WARNING_READ_WRITE_FAIL	RTMP failed to read/write and will be disconnected

PlayEvent

Key events

code	Event	Description
2001	PLAY_EVT_CONNECT_SUCC	Connected to the server
2002	PLAY_EVT_RTMP_STREAM_BEGIN	Connected to the server and started to pull streams
2003	PLAY_EVT_RCV_FIRST_I_FRAME	Render the first video packet (IDR)
2004	PLAY_EVT_PLAY_BEGIN	Video playback starts
2005	PLAY_EVT_PLAY_PROGRESS	Video playback progress (for VOD)
2006	PLAY_EVT_PLAY_END	Video playback ends

code	Event	Description
2007	PLAY_EVT_PLAY_LOADING	Video playback loading
2008	PLAY_EVT_START_VIDEO_DECODER	Decoder starts
2009	PLAY_EVT_CHANGE_RESOLUTION	Video resolution changes
2010	PLAY_EVT_GET_PLAYINFO_SUCC PLAY_EVT_SNAPSHOT_RESULT	VOD file information obtained successfully (Android iOS) The error code of screencap snapshot (Windows)
2011	PLAY_EVT_CHANGE_ROATION	MP4 video rotation angle (Android, iOS)
2012	PLAY_EVT_GET_MESSAGE	Used to receive messages inserted into the audio/video stream. For details, please see iOS Message Reception and Android Message Reception
2013	PLAY_EVT_PREPARED	Video loaded (Android, iOS)
2014	PLAY_EVT_VOD_LOADING_END	Loading ends (Android, iOS)
-2301	PLAY_ERR_NET_DISCONNECT	The network is disconnected and fails to be reconnected after three attempts, so the push needs to be restarted.
-2302	PLAY_ERR_GET_RTMP_ACC_URL_FAIL	Failed to get the accelerated pull address
-2303	PLAY_ERR_FILE_NOT_FOUND	No playback file exists (Android, iOS)
-2304	PLAY_ERR_HEVC_DECODE_FAIL	H265 decoding failed (Android, iOS)
-2305	PLAY_ERR_HLS_KEY	Failed to get the HLS decoding key (Android, iOS)
-2306	PLAY_ERR_GET_PLAYINFO_FAIL	Failed to get VOD file information (Android, iOS)

Note:

- **Determine whether the LVB is over:** Because of the varying implementation principles of different standards, many LVB streams usually don't throw end events (2006) and it is expected that when the VJ stops pushing stream, the SDK will soon find that data stream pull fails (WARNING_RECONNECT) and attempt to retry until the PLAY_ERR_NET_DISCONNECT event is

thrown after three failed attempts. Therefore, judgment needs to be made for both 2006 and -2301, which are used as the events to determine the end of LVB.

- **Do not hide the playback view after receiving PLAY_LOADING**, because the time length between PLAY_LOADING and PLAY_BEGIN can be different (sometimes 5 seconds, sometimes 5 milliseconds). Some customers consider hiding the view upon LOADING and displaying the view upon BEGIN, which will cause serious flickering (especially in LVB scenarios). It is recommended to place a translucent loading animation on top of the video view.

Warning events

Errors in internal warnings are recoverable. The SDKs initiate appropriate recovery measures. The purpose of the warning is mainly to prompt developers or end users of the error.

code	Event	Description
2101	PLAY_WARNING_VIDEO_DECODE_FAIL	Failed to decode the current video frame
2102	PLAY_WARNING_AUDIO_DECODE_FAIL	Failed to decode the current audio frame
2103	PLAY_WARNING_RECONNECT	Network disconnected and auto reconnection has started (the PLAY_ERR_NET_DISCONNECT event will be thrown after three failed attempts)
2104	PLAY_WARNING_RECV_DATA_LAG	Video stream is not stable. This may be caused by bad network connection.
2105	PLAY_WARNING_VIDEO_PLAY_LAG	Stutter occurred during the current video playback
2106	PLAY_WARNING_HW_ACCELERATION_FAIL	Failed to start hardware decoding. Use software decoding instead.
2107	PLAY_WARNING_VIDEO_DISCONTINUITY	Current video frames are discontinuous. This may be caused by frame loss. Blurred screen may occur.
2108	PLAY_WARNING_FIRST_IDR_HW_DECODE_FAIL	Hard decoding of the first I-frame of current stream failed. Switched to soft decoding automatically
3001	PLAY_WARNING_DNS_FAIL	DNS resolution failed (thrown only if the playback URL starts with "RTMP://")

code	Event	Description
3002	PLAY_WARNING_SEVER_CONN_FAIL	Server connection failed (thrown only if the playback URL starts with "RTMP://")
3003	PLAY_WARNING_SHAKE_FAIL	Server handshake failed (thrown only if the playback URL starts with "RTMP://")
3004	PLAY_WARNING_SERVER_DISCONNECT	The RTMP server is actively disconnected