数据加密用户指南

support@agora.io

目录

简介

Agora SDK 允许您的应用程序通过以下任意方式对语音和视频数据进行加密：

- 使用自定义的数据加密算法
- 使用 Agora SDK 的内置加密方案

注：本文提到的所有 API 详细介绍，请参考各平台参考手册。

使用自定义的数据加密算法

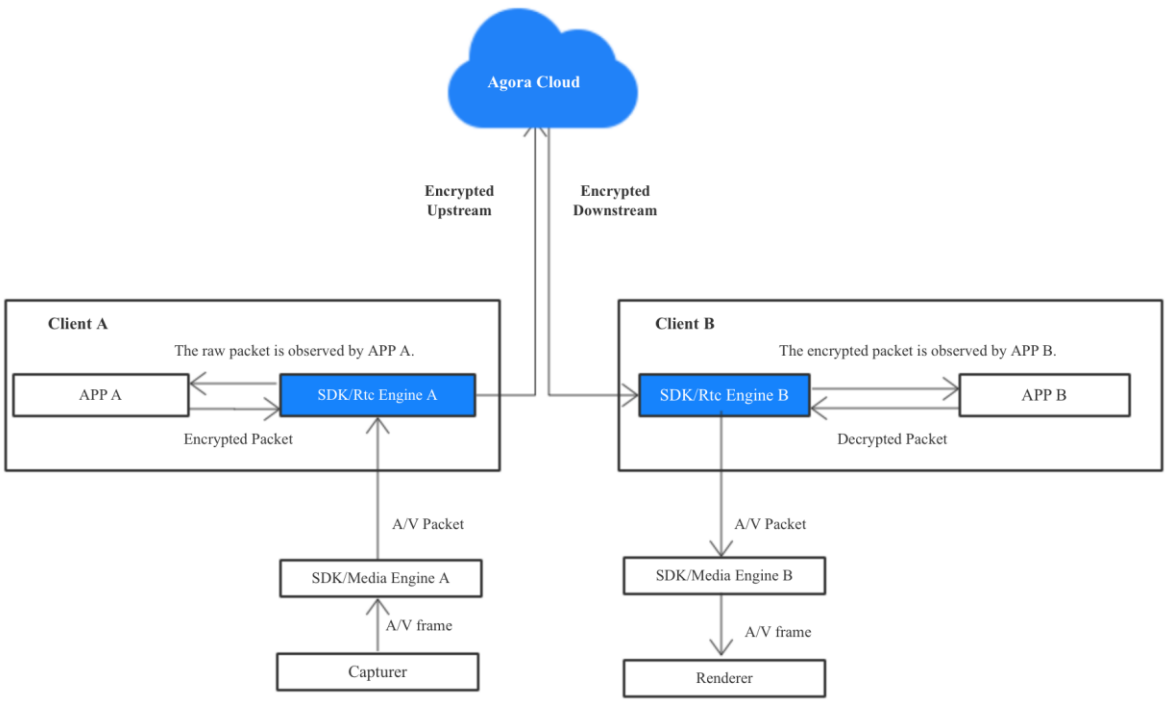
Agora SDK 允许您的应用程序使用您定制的数据加密算法对语音和视频数据进行加密。

支持以下 SDK：

- Agora Native SDK for Windows
- Agora Native SDK for Mac
- Agora Native SDK for iOS
- Agora Native SDK for Android

数据加密流程

下图描述了数据加解密流程：



实现数据加密

根据以下步骤实现数据加密：

1. 注册数据包观测器(Packet Observer)
2. 使用您定制的数据加密算法
3. 注册实例

注册数据包观测器(Packet Observer)

Agora SDK 允许您的应用程序注册数据包观测器(Packet Observer),用于在语音或视频数据包传输时接收事件。

在您的应用程序上，调用以下 API 注册数据包观测器：

virtual int registerPacketObserver(IPacketObserver* observer);

观测器必须继承 agora::rtc::IPacketObserver 类，且用 C++语言实现。以下为 IPacketObserver 类的定义:

```
class IPacketObserver
{
public:

    struct Packet
    {
        const unsigned char* buffer;
        unsigned int size;
    };
    /**
    * called by sdk before the audio packet is sent to other participants
    * @param [in,out] packet:
    *      buffer *buffer points the data to be sent
    *      size of buffer data to be sent
    * @return returns true to send out the packet, returns false to discard the packet
    */
    virtual bool onSendAudioPacket(Packet& packet) = 0;
    /**
    * called by sdk before the video packet is sent to other participants
    * @param [in,out] packet:
    *      buffer *buffer points the data to be sent
    *      size of buffer data to be sent
    * @return returns true to send out the packet, returns false to discard the packet
    */
    virtual bool onSendVideoPacket(Packet& packet) = 0;
    /**
    * called by sdk when the audio packet is received from other participants
    * @param [in,out] packet
    *      buffer *buffer points the data to be sent
    *      size of buffer data to be sent
    * @return returns true to process the packet, returns false to discard the packet
    */
    virtual bool onReceiveAudioPacket(Packet& packet) = 0;
    /**
    * called by sdk when the video packet is received from other participants
    * @param [in,out] packet
    *      buffer *buffer points the data to be sent
    *      size of buffer data to be sent
    * @return returns true to process the packet, returns false to discard the packet
    */
    virtual bool onReceiveVideoPacket(Packet& packet) = 0;
        };
```

使用您定制的数据加密算法

继承 agora::rtc::IPacketObserver，在您的应用程序上使用您自定义的数据加密算法。

以下例子对数据进行异或处理。对于 Agora SDK，发送和接收音频、视频数据包由不同线程处理，因此加密和解密可在不同缓冲区中进行。

```
class AgoraPacketObserver : public agora::rtc::IPacketObserver
{
public:
  AgoraPacketObserver()
  {
    m_txAudioBuffer.resize(2048);
    m_rxAudioBuffer.resize(2048);
    m_txVideoBuffer.resize(2048);
    m_rxVideoBuffer.resize(2048);
  }
  virtual bool onSendAudioPacket(Packet& packet)
  {
    int i;
    //encrypt the packet
    const unsigned char* p = packet.buffer;
    const unsigned char* pe = packet.buffer+packet.size;
    for (i = 0; p < pe && i < m_txAudioBuffer.size(); ++p, ++i)
    {
      m_txAudioBuffer[i] = *p ^ 0x55;
    }
    //assign new buffer and the length back to SDK
    packet.buffer = &m_txAudioBuffer[0];
    packet.size = i;
    return true;
  }

  virtual bool onSendVideoPacket(Packet& packet)
  {
    int i;
    //encrypt the packet
    const unsigned char* p = packet.buffer;
    const unsigned char* pe = packet.buffer+packet.size;
    for (i = 0; p < pe && i < m_txVideoBuffer.size(); ++p, ++i)
    {
      m_txVideoBuffer[i] = *p ^ 0x55;
    }
    //assign new buffer and the length back to SDK
    packet.buffer = &m_txVideoBuffer[0];
    packet.size = i;
    return true;
  }

  virtual bool onReceiveAudioPacket(Packet& packet)
  {
    int i = 0;
    //decrypt the packet
    const unsigned char* p = packet.buffer;
    const unsigned char* pe = packet.buffer+packet.size;
    for (i = 0; p < pe && i < m_rxAudioBuffer.size(); ++p, ++i)
    {
      m_rxAudioBuffer[i] = *p ^ 0x55;
    }
    //assign new buffer and the length back to SDK
    packet.buffer = &m_rxAudioBuffer[0];
    packet.size = i;
    return true;
```

```
  }

  virtual bool onReceiveVideoPacket(Packet& packet)
  {
    int i = 0;
    //decrypt the packet
    const unsigned char* p = packet.buffer;
    const unsigned char* pe = packet.buffer+packet.size;
    for (i = 0; p < pe && i < m_rxVideoBuffer.size(); ++p, ++i)
    {
      m_rxVideoBuffer[i] = *p ^ 0x55;
    }
    //assign new buffer and the length back to SDK
    packet.buffer = &m_rxVideoBuffer[0];
    packet.size = i;
    return true;
  }

private:
  std::vector<unsigned char> m_txAudioBuffer; //buffer for sending audio data
  std::vector<unsigned char> m_txVideoBuffer; //buffer for sending video data

  std::vector<unsigned char> m_rxAudioBuffer; //buffer for receiving audio data
  std::vector<unsigned char> m_rxVideoBuffer; //buffer for receiving video data
        };
```

注册实例

Windows

调用 **registerPacketObserver** 为您应用程序使用的 agora::rtc::IPacketObserver 类注册一个实例。

Android

1. 实现一个 java 包装程序。

   例如：

```
    JNIEXPORT jint JNICALL
Java_io_agora_video_demo_RtcEngineEncryption_enableEncryption(JNIEnv *env, jclass clazz,
jlong engineHandle)

    {

      typedef jint (*PFN_registerAgoraPacketObserver)(void* engine,
agora::rtc::IPacketObserver* observer);


      void* handle = dlopen("libagora-rtc-sdk-jni.so", RTLD_LAZY);

      if (!handle)

      {

        __android_log_print(ANDROID_LOG_ERROR, "aogra encrypt demo", "cannot find
libagora-rtc-sdk-jni.so");

        return -1;
```

```
        }
        PFN_registerAgoraPacketObserver pfn =
(PFN_registerAgoraPacketObserver)dlsym(handle, "registerAgoraPacketObserver");
        if (!pfn)
        {
          __android_log_print(ANDROID_LOG_ERROR, "aogra encrypt demo", "cannot find
registerAgoraPacketObserver");
          return -2;
        }
        return pfn((void*)engineHandle, &s_packetObserver);
      }

      Java wrapper:
      public class RtcEngineEncryption {
        static {
          System.loadLibrary("agora-encrypt-demo-jni");
        }
        public static native int enableEncryption(long rtcEngineHandle);
      }
```

2. 调用 **registerAgoraPacketObserver** 为您应用程序使用的 agora::rtc::IPacketObserver 类注册一个实例。

iOS/Mac

调用**registerAgoraPacketObserver**为您应用程序使用的agora::rtc::IPacketObserver类注册一个实例。

使用 Agora SDK 的内置加密方案

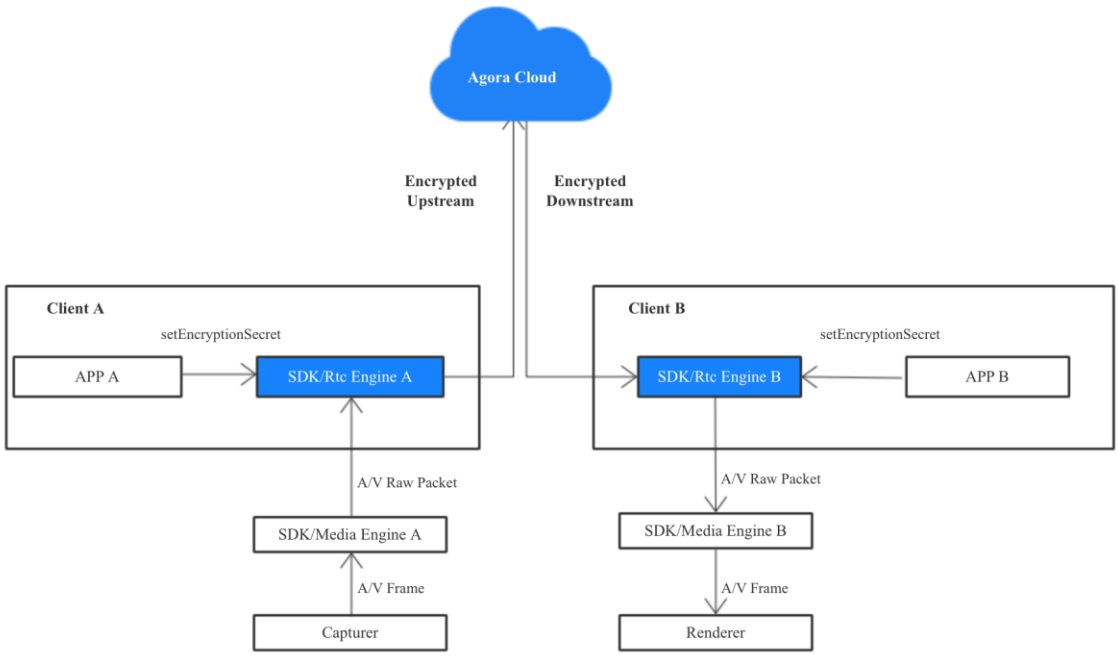
Agora SDK 使用 AES-128 加密算法实现内置加密。

支持以下 SDK：

- Agora Native SDK for Windows
- Agora Native SDK for Mac
- Agora Native SDK for iOS
- Agora Native SDK for Android
- Agora Native SDK for Web

数据加密流程

下图描述了数据加解密流程：



实现数据加密

调用 setEncryptionSecret 启用和设置加密密钥。Agora SDK 根据 secret 字符串和频道名称生成一条 128 位的加密密钥用于通话。由于加密密钥是根据频道名和 secret 字符串生成的，即使使用相同的 secret，只要频道名不同，生成的密钥也会不同。

加密密钥是根据您应用程序所指定的 secret 动态生成的，因此同一频道内的所有用户应设置相同的 secret。Agora SDK 仅在客户端本地使用 secret，不会将 secret 传送给服务器端。