



Agora Recording Server

User Guide

support@agora.io

June 2016

Contents

Introduction	3
Agora Recording Server	3
Requirements	3
<i>Hardware Requirements</i>	<i>4</i>
<i>SDK Compatibility</i>	<i>4</i>
<i>Security Requirements</i>	<i>5</i>
<i>Other Requirements</i>	<i>5</i>
Known Issues and Limitations	5
Getting Started	6
Preparation	6
<i>Deploying the ARS</i>	<i>6</i>
<i>Setting the ARS IP</i>	<i>6</i>
<i>Integrating the Agora RDK</i>	<i>7</i>
Recording	8
<i>Starting the Recording</i>	<i>8</i>
<i>Checking the Recording Status</i>	<i>8</i>
<i>Stopping the Recording</i>	<i>8</i>
Managing	9
<i>Managing the Recording Files</i>	<i>9</i>
<i>Playing the Recording Files</i>	<i>9</i>
<i>Securing the Recording File</i>	<i>10</i>

Introduction

This document describes the Agora Recording Server (ARS), how to deploy it, and how you can record video and audio on different Agora SDKs. This function is only applicable to developers who have adopted the Dynamic Key schema.

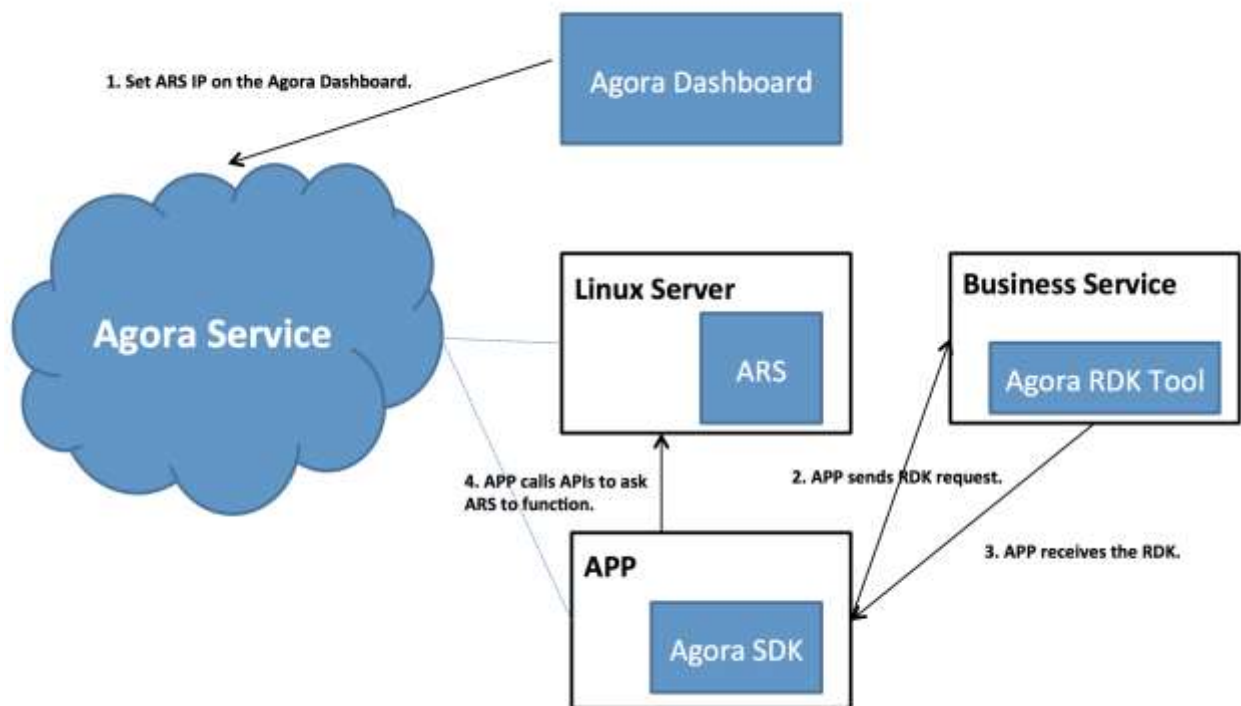
Note: Use this document together with the SDK reference manual included in the same SDK. Refer to the reference manual for any API usage.

Agora Recording Server

The Agora Recording Server (ARS) is a software suite that provides the following functions:

- Records audio and video of all participants in a channel
- Records audio and video of all participants in multiple channels simultaneously
- Records audio of all participants in a channel or in multiple channels simultaneously

The following figure depicts the ARS working process.



Requirements

Ensure that the following prerequisites are met before you begin to deploy and manage the ARS.

Hardware Requirements

The following table lists the basic hardware requirements and recommended configurations.

Hardware Requirements	
Server	Physical or Virtual. Ubuntu Linux 14.04 LTS 64-bit
Network	The Linux Server needs public IP, called ARS IP
Bandwidth	The recording is the data transferring via the network, and the bandwidth affects the number of channels that can be recorded simultaneously. Refer to the following criteria: assuming the resolution is 640x480 and one person requires 500 kbps. For a channel with two people, the bandwidth needs 1M. If you want to ensure that 100 channels are recording at the same time, then the bandwidth required is 100M.
DNS	The Server must resolve qos.agoralab.co, otherwise it cannot report the required statistics.

The following configuration is recommended:

Product		Description	Number
SUPERMICRO 6017R-TDF	SYS-6017R-TDF	1U rack-mounted Dual Intel® Xeon® E5-2600 Series Processor	1
Case		1U Rackmountable (440W high-efficiency redundant power supply w/ PMBus)	1
Processor		Intel Xeon E5-2620V2 2.1G, L3:15M, 6C (P4X-DPE52620V2-SR1AN)	2
Memory		MEM-DR380L-HL06-ER16 (8GB DDR3-1600 2Rx8 1.35v ECC REG RoHS)	1
Hard Disk		250G 3.5 SATA Enterprise (HDD-T0250-WD2503ABYZ)	2





The following assumptions are based on the recommended configuration and one channel with 2 people:

- Each channel uses 25MB memory.
- Each channel writes to disk at the speed of 60KB/s.
- The speed in each channel is 900Kbps with each person using 450Kbps, but actual speed depends on the resolution and bitrate of the user device.
- Each CPU can have 9 to 10 channels recording simultaneously. A twelve-core CPU running under a 24-thread machine can have 110 channels running simultaneously. CPU is a performance bottleneck.

SDK Compatibility

This function is applicable to the following SDKs:



- Agora Native SDK for Android v1.3 or later

-  Agora Native SDK for iOS v1.3 or later
-  Agora Native SDK for Mac v1.3 or later
-  Agora Native SDK for Windows v1.3 or later
-  Agora Native SDK for Web v1.3 or later

All platform SDKs and the related Reference Manual can all be downloaded at www.agora.io/developer.

Note: If any participant in the channel uses v1.2 SDK, the whole channel cannot record anything.





Security Requirements

-  You, as a developer, must have your own Business Server (Business Service), and the Business Service can communicate with the application.
-  Your Business Service must integrate the Recording Dynamic Key(RDK) generation module provided by Agora.io. The application needs a RDK before it calls APIs to use the recording service.

Other Requirements

The Linux server must access the Ubuntu software sources, so make sure that it can install software using apt-get install.

Known Issues and Limitations

-  When you record using an Android client, the image is turned upside-down when you switch from the front-facing camera to the rear one.
-  If no one in the channel calls the API to stop recording, then the recording file ends with 30 seconds of silence.
-  The video encoding format must be H.264. Otherwise the recording function is not supported.
-  An encrypted channel cannot use the recording function.

Getting Started

This section explains how to deploy the Agora Recording Server and how to use the ARS to manage recordings.

Preparation

Deploying the ARS

1. Download the ARS package provided by Agora.io.
ARS is available in all platform SDKs at <http://cn.agora.io/download/>.
2. Copy the ARS package provided by Agora.io to your Ubuntu server.
3. Set the server port.
The default port is 8001. If you want to change it, open the `agora_recording_service.py` file.
Change the following line of code from:
`PORT = os.environ.get('PORT', 8001)`
To:
`PORT = os.environ.get('PORT', <your port number>)`
Ensure the following UDP ports are open: 4001 to 4010, 1080, 8000, and 35000.
4. Deploy ARS on the server.
Run. `/deploy.sh --key=vendorKey --sign=signature --port=port --path=your_path`
Fill in the following information manually:

vendorKey	The Vendor Key you can fetch from PROJECTS page of https://dashboard.agora.io/ .
signature	The Sign key you can fetch from PROJECTS page of https://dashboard.agora.io/ .
port	Service Port. Ensure that the HTTP port is open.
your_path	The targeted file path to store your recording files.

If the deployment is successful, the screen displays **deploy successfully**; otherwise it reports **deploy failed**.

Setting the ARS IP




1. Log on to <https://dashboard.agora.io/>.
2. Enter the **PROJECTS** page of the dashboard.
3. Set **Recording Server IP** under the corresponding project:

1 Project(s) Total

Created time ↓

Search by project name.

TEST1



2016-06-13

API Key: a0eff2e22d044edb828144d8aeb32278

Sign Key:

.....

[Disable sign key](#)

Recording server IP:

SAVE

4. Click **SAVE**, and then Agora finds the correct ARS for you.

Integrating the Agora RDK

You should integrate the RDK generation module provided by Agora.io into your own business service. Agora.io provides source code of in C++, Java, Python and node.js, which you can use directly in your application.

For how to generate the RDK, refer to ***Using Dynamic Keys for Increased Security*** in the reference manual of the specific SDK that you have deployed.






Note:

When joining a channel or using the recording function, you will use the same Vendor Key and Sign Key, but the method to generate the Dynamic Key is different. Use `generateMediaChannelKey` to join a channel, and use `generateRecordingKey` to use the recording function.

For all the API usage, refer to the corresponding SDK reference manual.

Recording

Any participant in the same channel can call a set of APIs to manage the recordings. This section only introduces the basic processes and operations. For detailed API information and usage, refer to the following documents:

-  Agora Native SDK for Android Reference Manual v1.3 or later
-  Agora Native SDK for iOS Reference Manual v1.3 or later
-  Agora Native SDK for Mac Reference Manual v1.3 or later
-  Agora Native SDK for Windows Reference Manual v1.3 or later
-  Agora Native SDK for Web Reference Manual v1.3 or later

You can download all the platform SDKs and related documents at www.agora.io/developer.

Starting the Recording

When you, as a participant in a conference or multi-party conference, want to record the conference, click the **Record** button on the toolbar. This is the general operation, but the exact operation depends on the actual setup of your application:

1. The application sends an RDK request to the business service with two parameters: the name of the channel that needs recording and the UID.
2. The business service, after receiving the request from the application, calls the Agora RDK Tool to generate a RDK and returns the RDK to the application.
3. The application calls `startRecordingService` with the received RDK as one of the parameters. This method is called asynchronously, which means when it is called successfully, a callback method will be triggered. The callback method varies on different SDKs, refer to the Reference Manual of the specific SDK that you have deployed for details.

Checking the Recording Status


Unlike `startRecordingService` and `stopRecordingService`, calling `refreshRecordingServiceStatus` is a read-only operation, which does not require authentication and RDK. Anyone in the channel can check the recording status anytime. This method is called asynchronously, which means when it is called successfully, a callback method will be triggered. The callback method varies on different SDKs. Refer to the Reference Manual of the specific SDK that you have deployed for details.

For example, the following is a typical scenario:

1. Check whether any recording activity exists in the channel;
2. If not, call `startRecordingService` to start the recording.

Stopping the Recording


The recording stops in any of the following scenarios:

-  **Scenario 1:** Any participant in the channel calls `stopRecordingService`.

Any user in the channel can call `stopRecordingService`. The ARS stops the recording when receiving the first request and ignores the subsequent same requests. This method is called asynchronously, which means when it is called successfully, a callback method is triggered. The callback method varies on different SDKs. Refer to the reference manual of the specific SDK that you have deployed for details.

-  **Scenario 2:** The last user left the channel;



ARS knows how many users are in the channel, and the recording stops automatically when the last user left the channel, even if no one calls stopRecordingService.

-  **Scenario 3:** If the user calls stopRecordingService unsuccessfully due to an application crash, network outage or blackout, ARS follows the logic of **Scenario 2**.

Managing

Managing the Recording Files

The environment variable **AGORA_FILE_ROOT** contains the path to the top-level recording directory. The directory structure is listed as follows:

-  **yyyy_mm_dd** (Date): A new date directory is created every day if startRecordingService is called that day. All files and directories created on the corresponding date will be stored under this directory.
-  **ChannelName_HHMMSS**: Recording files are stored in the directory created on the same date as the recordings. Each recording file is with **channel name** and **hour,minute,second** timestamp.

Note: If the channel name begins with '.', Linux takes the recording folder as a hidden folder. You can use ls -A to view it.

All individual files related to the recording are under the ChannelName_HHMMSS directory:

File	Description
Metadata.txt	A text file that contains metadata about the recording. The metadata includes a number of text fields with each field separated by a Linux new line. The text lines consist of the following value pairs: <version>1.0</version> <tag>xxx</tag> ¹ <channel name>xxxxxx</channel name> <error status><status> no error status implemented yet
UID_HHMMSS.tmp	Recording files that contain video and audio data for users with UID. Each user writes its own tmp file. The file contains video and audio only related to this specific user ² .
UID_HHMMSS.mp4	After the automatic transcoding, each tmp file has one corresponding mp4 file generated under the same directory.

1. This is a tag defined by the user at the time the recording starts. It can be any text the user chooses.
2. ARS is expected to support splitting large files to smaller chunks in the future.

Playing the Recording Files

The generated recording files are in tmp format, but agora_hourly.job transcodes them to mp4 format automatically.




The task agora_hourly.job, located at /etc/crontab.hourly/ in your deployed ARS server, runs hourly to transcode the tmp files to mp4 files automatically at five times rate, for example, it takes 10 minutes to transcode a 50-minute file from tmp to mp4.

During the transcoding, the temp files are in *transcoding* format. Once the transcoding is complete for one tmp file, one corresponding *mp4* file is generated under the same directory.

After the transcoding, you can use almost any player to play the mp4 files, as shown in the table below

Operating System	Player
Windows	<ul style="list-style-type: none">• Windows Media Player• KM Player• VLC Player
Mac	<ul style="list-style-type: none">• Quick Time Player• Movist• MPlayerX• KMPlayer
iOS	<ul style="list-style-type: none">• iOS Default Player• VLC• KMPlayer
Andriod	<ul style="list-style-type: none">• Android Default Player• MXPlayer• VLC for Android• KMPlayer

Files cannot be automatically transcoded under the following circumstances:

-  The file is currently recording.
-  The file already has been transcoded.
-  The file was been created more than three days ago.

You can determine whether there is any transcoding failure by checking the log file(/var/log/agora_hourly.log), and you can also filter the errors by executing the linux command: *grep error /var/log/agora_hourly.log*. Contact support@agora.io for any transcoding failure.

Securing the Recording File

The recording file is stored only in your server, and Agora.io cannot access it. Deploy the recording services yourself or consult security experts. Secure the file just as you do the other files in your server.

Agora CaaS, Agora Global Network, Agora Native SDK and Agora Web SDK are trademarks of Agora.io. Agora Lab does business as Agora.io. Other product and company names mentioned herein are trademarks or trade names of their respective companies.