标识：XXX-XX-GY-001

版本：001

XXXXX

项目技术文件

直播系统

软件概要设计说明

承 担 部 门：

完 成 日 期：

文档修改记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 修改内容描述 | 修改人 | 日期 | 备注 |
| 001 | 初次提交 | 徐祖云 | 2023-6-29 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 次

[1 引言 1](#_Toc4257)

[1.1 编写目的 1](#_Toc20166)

[1.2 背景 1](#_Toc20574)

[1.3 术语及缩略语 1](#_Toc817)

[1.4 参考资料 1](#_Toc16600)

[2 设计概述 1](#_Toc20145)

[2.1 限制和约束 1](#_Toc13612)

[2.2 设计原则和设计要求 2](#_Toc25691)

[3 软件总体设计 2](#_Toc14042)

[3.1 软件总体架构 2](#_Toc18280)

[3.2 软件技术架构 2](#_Toc17676)

[3.3 软件部署架构 2](#_Toc10213)

[4 系统流程 2](#_Toc28415)

[4.1 X1流程 3](#_Toc500)

[5 功能设计 3](#_Toc10936)

[5.1 软件功能结构图 3](#_Toc31791)

[5.2 X1模块设计 3](#_Toc4724)

[5.3 X2模块设计 3](#_Toc9692)

[6 数据设计 3](#_Toc18027)

[6.1 数据库逻辑设计 3](#_Toc31772)

[6.2 数据库物理设计 3](#_Toc24033)

[6.3 基表设计 3](#_Toc5435)

[6.4 视图设计 5](#_Toc27040)

[6.5 索引设计 6](#_Toc3013)

[6.6 触发器设计 7](#_Toc23368)

[6.7 存储过程设计 7](#_Toc20245)

[6.8 数据字典设计 7](#_Toc28)

[6.9 安全保密设计 8](#_Toc28015)

[7 接口设计 8](#_Toc29489)

[7.1 用户界面设计 8](#_Toc19435)

[7.2 内部接口设计 8](#_Toc18990)

[7.3 外部接口设计 8](#_Toc8891)

[8 安全保密设计 9](#_Toc6404)

[8.1 数据传输安全性设计 9](#_Toc2189)

[8.2 数据传输安全性设计 9](#_Toc3126)

[8.3 数据传输安全性设计 9](#_Toc28189)

[9 出错处理设计 9](#_Toc5814)

[10 系统维护设计 9](#_Toc13104)

[11 附录 9](#_Toc22372)

3引言

## 编写目的

说明编写这份文档的目的，指出预期的读者。

在需求分析阶段中，我们已经将系统用户对本系统的需求做了详细的阐述，这些用户需求已经在上一阶段的用户需求可行分析中也做了介绍，并在需求规格说明书中得到详尽的的阐明。

在分析直播系统架构图之后，北邮项目小组对该系统进行了概要设计。主要是基于以下目的编写此说明书。

1. 对系统概要设计的阶段任务成果形成文档，以便阶段验收、评审，最终的文档验收。
2. 对需求阶段的文档再次确认过程，对前一阶段需求没有做充分或错误的提出修改。
3. 明确整个系统的功能框架和数据库结构，为下一阶段的详细设计、编码、和测试提供参考依据。

4、明确编码规范和命名规范，统一程序界面。

预期读者：详细设计人员、概要设计评审小组、研究院领导。

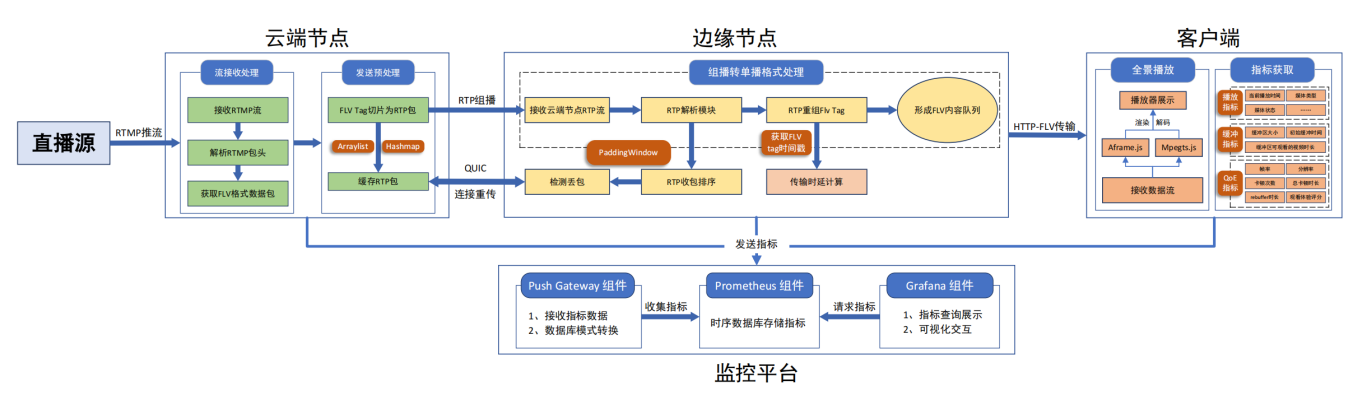
## 背景

可包括：

1. 这份文档所描述的软件系统的名称；
2. 该软件项目的任务提出者、开发者、用户（或首批用户）。
3. 该软件项目的简要描述

本项目由北京邮电大学计算机学院和中国电信广州研究院共同提出，由北京邮电大学网络体系结构研究院硕士研究生开发。

本直播系统由三大部分组成：用于接收直播源推流的云端节点，接收云端节点数据并向用户提供服务的边缘节点，用于播放全景视频并收集性能指标的播放器。有关本系统与其他系统的相关联系如下：



## 术语及缩略语

表 1‑1 本文档使用的术语或缩略语一览表

| 序号 | 术语或缩略语 | 解释 | 备注 |
| --- | --- | --- | --- |
|  | 本系统 | 基于quic重传的单组播协同直播系统 |  |

## 

## 参考资料

表 2‑1 参考资料一览表

| 序号 | 文档标识 | 文档名称 | 版本号 | 备注 |
| --- | --- | --- | --- | --- |
| 1. | 《软件工程——原理.方法与应用》-001 | 《软件工程——原理.方法与应用》 | 001 | 史济民 等 高等教育出版社2002.2 |
| 2. | Adobe’s Real Time Messaging Protocol-001 | Adobe’s Real Time Messaging Protocol | 001 | https://github.com/melpon/rfc/blob/master/rtmp.md |
| 3. | RTP: A Transport Protocol for Real-Time Applications-001 | RTP: A Transport Protocol for Real-Time Applications | 001 | https://www.rfc-editor.org/rfc/rfc3550.html |
| 4. | QUIC: A UDP-Based Multiplexed and Secure Transport-001 | QUIC: A UDP-Based Multiplexed and Secure Transport | 001 | https://datatracker.ietf.org/doc/html/rfc9000 |
| 5. | Adobe Flash Video File Format Specification  Version 10.1-001 | Adobe Flash Video File Format Specification  Version 10.1文档 | 001 | http://download.macromedia.com/f4v/video\_file\_format\_spec\_v10\_1.pdf |
| 6. | HTTP Live Streaming-001 | HTTP Live Streaming文档 | 001 | https://www.rfc-editor.org/rfc/pdfrfc/rfc8216.txt.pdf |

# 设计概述

本节描述现有开发条件和需要实现的目标，说明进行概要设计时应该遵循的设计原则和必须采用的设计方法。

## 限制和约束

**本项目存在以下约束条件：**

技术条件：主要是北邮同学进行实际开发，参考并学习现有的开源项目，电信开发人员对北邮同学进行技术指导

资金状况：

开发环境：开发环境为北邮同学的个人笔记本/台式机，windows/mac/linux操作系统，go1.18/node1.14，使用vscode、goland等开发工具

时间限制：预计在一年内完成项目交付

**应当实现的系统目标：**

将软件系统需求转换为未来系统的设计；

逐步开发强壮的系统构架，提升系统的稳定性

使设计适合于实施环境，为提高性能降低时延而进行设计；

## 设计原则和设计要求

描述对本软件系统进行概要设计的原则，通常可以考虑以下几方面的内容：

1. 命名规则；

对go的结构体、函数、变量采用驼峰命名法；struct的声明和初始化采用多行书写；单个接口/函数的命名一般以er结尾作为后缀，例如：Reader, Writer；

1. 模块独立性原则：

一个类对另一个类的依赖应该建立在最小的接口上，降低类之间的耦合度；迪米特法则（Demeter Principle）一个实体应当尽量少地与其他实体之间发生相互作用，降低类之间的耦合度，提高模块的相对独立性。

1. 边界设计原则；

对程序的数据访问、和用户的操作设置适当的边界，防止对程序的运行造成影响。

1. 必须的安全措施；

在可能影响系统正常运行的操作时拒绝执行

1. 安全性和保密原则；

对于传输的数据尽量采用加密的方式，使用https代替http，使用token代替密码。

1. 系统灵活性要求；

在不同的操作系统和版本、不同的网络、不同码率的视频下都能够正常运行

1. 系统易操作性要求；

使用配置文件代替复杂的命令行参数，减少用户需要输入的参数，提供详尽的操作文档

1. 系统可维护性要求；

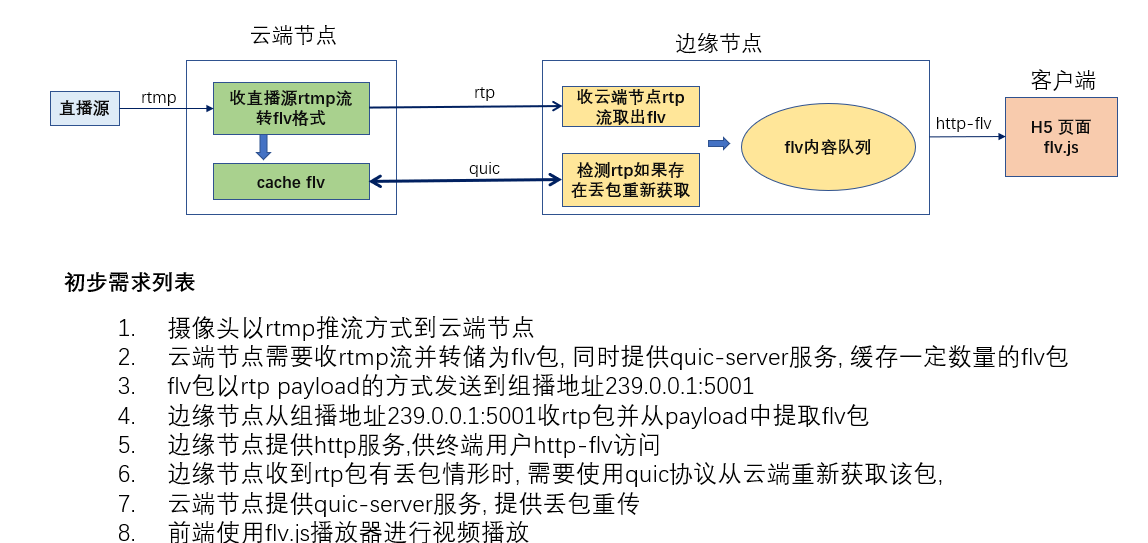
建立明确的软件质量目标和优先级；使用提高软件质量的技术和工具；进行明确的质量保证审查；选择可维护的程序设计语言；书写详尽的程序文档

# 软件总体设计

概要介绍本软件的开发目的、功能部件、主要功能等。如果说明的内容较多，本节应分小节逐一介绍。

## 软件总体架构

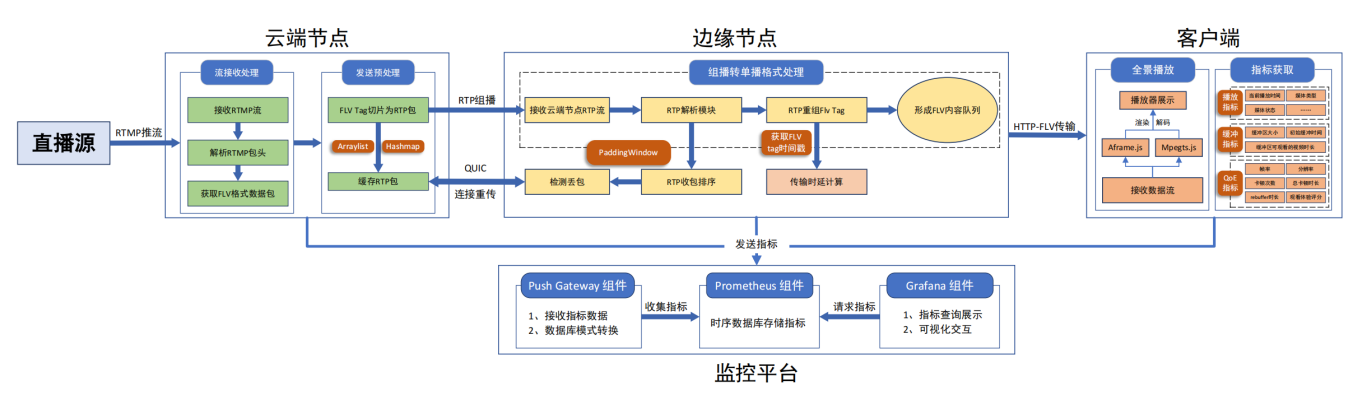
本节描述软件的总体架构设计，包括总体架构图及对架构图的说明。分层、分模块说明软件的组成方式。



系统由云端-边缘-客户端三部分组成，直播源以rtmp方式推流到云端节点，云端收到rtmp流并转储为flv格式数据，随后将flv数据以单播或组播的方式发送到发送到边缘节点，边缘节点检测出rtp丢包后通过quic重传获取该包，并提供http-flv服务，客户端使用flv.js播放器播放视频。

## 软件技术架构

本节描述软件采用的技术架构，包括技术架构图及对技术架构的说明。



在云端服务器接受直播/点播源的rtmp全景视频流，并转换为flv格式视频流，通过RTP组播切片发送至边缘服务器同时缓存一定数量的数据包；边缘节点主要由RTP缓存模块、flv解析模块和http-flv服务模块组成，对收到的Rtp序列重新整合，对于乱序的数据包重新排序，对于丢失的数据包向云端发送重传请求，然后交付给flv解析模块将Rtp数据包整合为完整的flv内容队列，最后通过http-flv向客户端提供服务。

；客户端通过就近向边缘服务器获取媒体流，减轻云端服务器的负载，通过不同的地址向边缘节点访问不同的媒体流，获得稳定的观看体验，可以通过主机Web网页直接观看，通过鼠标调整用户视野。也可以通过头戴VR设备获得沉浸式的观看体验。并配套监控页面实时监测媒体流的丢包、时延等信息。

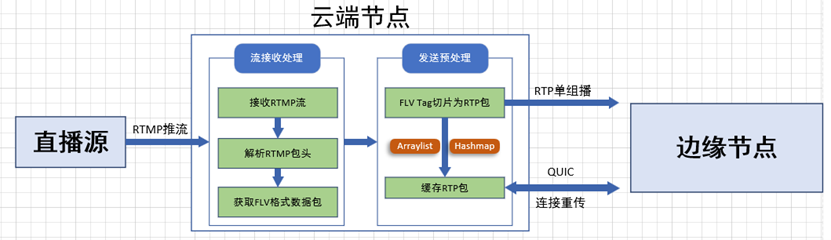
## 软件部署架构

本节描述软件的整体部署架构，包括部署架构图及对部署方式的说明。部署方式包括集中部署及多级部署。

# 系统流程

介绍系统的各类业务处理流程，本节应分小节逐一介绍。

## 云端节点流程

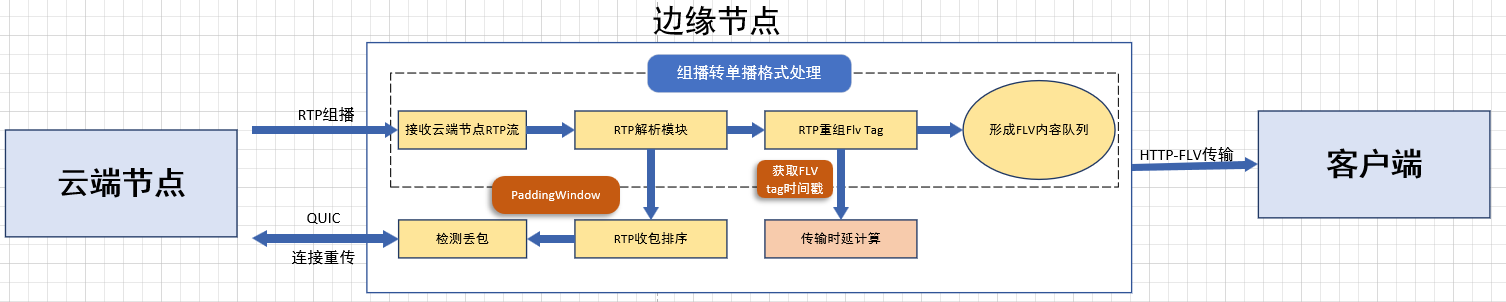


云端节点开启rtmp server服务，摄像头以rtmp推流方式到云端节点，对不同的直播流分配不同的标识。

云端节点需要收rtmp流并转储为flv包,同时提供quic-server服务,缓存一定数量的flv包。

flv包以rtp payload的方式通过单播或组播发送到边缘节点，通过SSRC确定不同的流，通过marker进行分片发送。

## 边缘节点流程

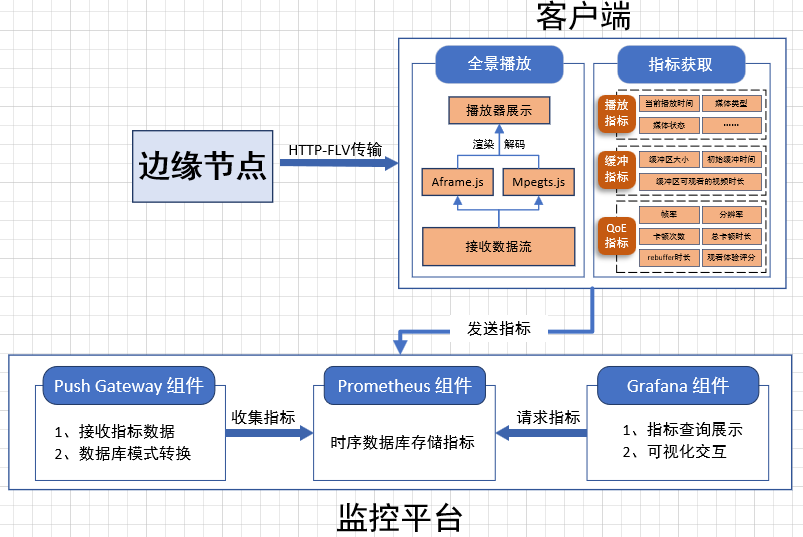


边缘节点从单播地址（127.0.0.1）或组播地址（239.0.0.1:5001）收rtp包并从payload中提取flv包。

边缘节点收到rtp包有丢包情形时,需要使用quic协议从云端重新获取该包。

形成flv内容队列，并提供http服务,供终端用户http-flv访问。

## 客户端流程



播放器通过flv.js播放视频流，并以全景方式呈现。

根据不同的边缘节点地址访问不同边缘节点上的内容。

根据不同的path访问不同的视频流。

在播放过程中监控视频流的性能，并在监控界面展示。

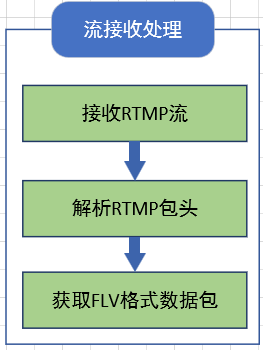
# 功能设计

分章节描述各功能模块的具体设计。

## 软件功能结构图

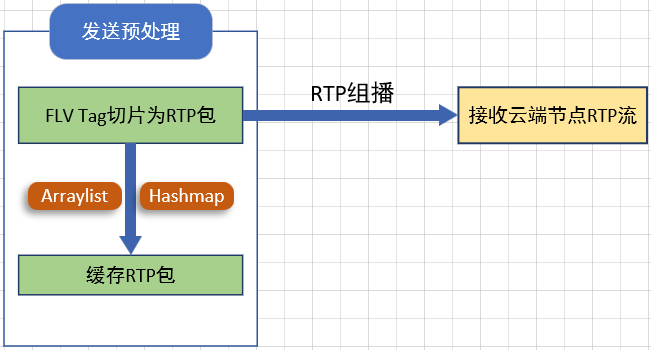
本节描述软件的功能模块构成，包括功能结构图及对各功能模块的描述。

## rtmp流转flv模块设计



对rtmp服务器，每当收到一个Rtmp Message时，解析数据包，获取包头数据和媒体数据，构造Flv相应的TagType、DataSize、TimeStamp和Data得到Flv数据包，交付Rtp发送模块。

## rtp发送模块



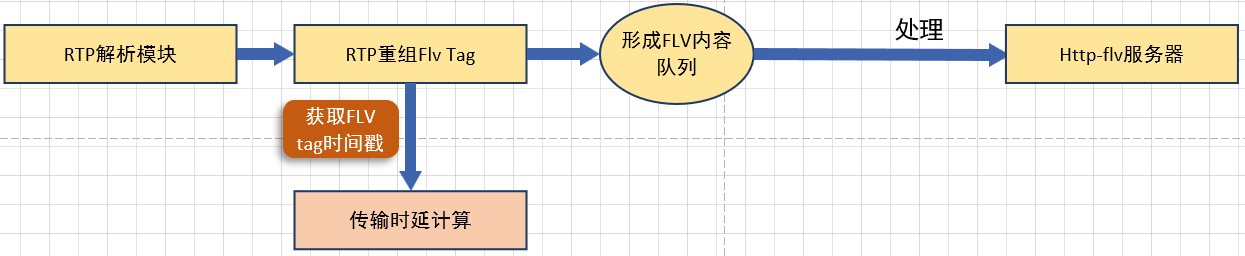
对上一模块收到的Flv数据包，切片处理，并按照规则设置marker位，得到若干Rtp包，将其通过单播/组播方式发送到边缘节点，发送前复制一份送入由Arraylist或Hashmap构成的缓存中。

## quic重传模块设计

## quic重传

云端节点开启quic服务，建立两条单向流（infoStream和dataStream），infoStream用于接收边缘节点发送的quic请求（包含ssrc和seq信息），通过请求查找缓存获得相应的rtp包，接着通过dataStream返回Rtp数据。

## http-flv模块设计



边缘节点开启http服务，收到客户端请求后解析path信息确定相应的流ssrc，找到该ssrc解析Rtp队列得到FlvTag数据，并将其以flv流的格式写入httpResponser

# 接口设计

## 用户界面设计

## 用户界面包括视频播放页面、数据展示页面和性能监控页面。在这个部分我们根据需求分析的结果，用户需要一个简洁美观的界面。在界面设计的上，应做到简单明了，易于操作，并且要注意到界面的布局，应突出的显示重要以及出错信息。外观上也要做到合理化。系统的用户界面应做到可靠性、简单性、容易学习和使用。

## 内部接口设计

在内部接口方面，各模块之间采用函数调用、参数传递、返回值的方式进行信息传递。具体参数的结构将在详细设计的内容中说明。接口传递的信息将是以数据结构封装的数据，以参数传递的方式在各模块之间传输，云端、边缘和客户端之间的信息通过Rtp、http流的方式传递。

## 外部接口设计

外部接口主要面向用户，在云端节点接收用户的推流，在客户端主要是Web网页监控用户键盘和鼠标的数据，通过键盘获取用户的播放地址，通过鼠标调整用户视野。

# 安全保密设计

## 数据传输安全性设计

由于数据的传输上需要通过网络传输，为了客户资料进行保密，需要在网络的传输过程中对数据进行加密。

这个工作主要是在准备网络包、解开网络包这两个模块完成，它们各对数据进行加密及解密还原工作。

在加密算法选择上将使用Https加密算法。

# 出错处理设计

由于边缘节点性能的限制，可能会在流数量较多或码率较大时对Udp数据包处理不过来，从而导致大面积丢包，此时会触发保护机制，主动丢包减少拥塞，并扩大udp缓存以缓存更多的udp包，还可以通过手动调整linux的socket缓存以改善。

# 系统维护设计

云端节点和边缘节点如果开启了直播录制的情况下会在录制文件夹生成所有视频流的录制文件，需要定期删除以减少磁盘的占用。

云端节点和边缘节点会在初次启动时会在当前目录下生成默认配置文件（config.yaml)，如果后期更改了配置的话需要保存好配置文件，如果删除将丢失已更改的配置。

# 附录

**技术支持信息：**

[livego](https://github.com/gwuhaolin/livego)

[GoRtp](https://github.com/wernerd/GoRTP)

[quic-go](https://github.com/quic-go/quic-go)

**相关基础知识：**

[RTMP](https://github.com/melpon/rfc/blob/master/rtmp.md)

[RTP](https://www.rfc-editor.org/rfc/rfc3550.html)

[QUIC](https://datatracker.ietf.org/doc/html/rfc9000)

[HTTP-FLV](https://ossrs.io/lts/en-us/docs/v4/doc/delivery-http-flv)

[HLS](https://www.rfc-editor.org/rfc/pdfrfc/rfc8216.txt.pdf)