SHadowsocks设计分析文档

2017/12/27

目录

[项目背景和原理 1](#_Toc499830161)

[项目方法 2](#_Toc499830162)

[OOA模型 2](#_Toc499830163)

[1. 需求模型 2](#_Toc499830164)

[系统边界 2](#_Toc499830165)

[2. 基本模型 3](#_Toc499830166)

[3. 辅助模型 4](#_Toc499830167)

[OOD模型 4](#_Toc499830168)

[1. 问题域部分 4](#_Toc499830169)

[2. 数据接口部分 4](#_Toc499830170)

[3. 控制驱动部分 4](#_Toc499830171)

[4. 人机交互部分 4](#_Toc499830172)

[总结 4](#_Toc499830173)

SHadowsocks设计分析文档

2017/12/27

# 项目背景和原理

Shadowsocks是一款比较知名和应用比较广泛的翻墙软件。Shadowsocks的代码质量很高，并且有python编程语言编写的版本，符合面向对象编程的要求。

Shadowsocks的翻墙原理如下列示意图所示：



图1. Shadowsocks原理示意图

Shadowsocks基于Sock5协议完成了一个墙内墙外的通信过程：

1. SS Local把PC的数据包进行加密
2. SS Local转发加密后的数据包到墙外的服务器SS Server
3. SS Server把解密后的数据包转发到目标地址
4. SS Server把返回的数据包进行加密
5. SS Server把加密后的数据包转发到墙内的SS Local
6. SS Local把解密后的数据包返回到PC

上述就是Shadowsocks的基本原理。该软件实现过程涉及到网络通信、命令行交互、加密解密、IO复用等方面，是一个有一定复杂度的软件。

# 项目方法

本次项目主要针对shadoesocks的2.8.2 python版本进行建模分析。在这个项目里面我们主要采用的建模方法是面向对象建模。原因主要有以下几点：

1. 本次分析的软件是用python语言进行编写的。而python是一个面向对象的编程语言，与面向对象建模及其吻合
2. 面向对象建模是目前比较流行的一种建模方法。掌握好这一方法有利于提高以后的编程能力。

因此在采用面向对象建模方法以后，主要流程包括OOA和OOD分析两个阶段。这两个阶段建立的模型分别为OOA模型和OOD模型。

# OOA模型

OOA阶段主要发生在面向对象建模的分析阶段。OOA模型框架的示意图如下：



图2. OOA模型框架

本小节主要按照需求模型、基本模型和辅助模型的顺序一一阐述。

## 需求模型

### 系统边界

在面向对象建模的方法中，第一步需要进行的就是需求分析建模。而需求模型主要是由用况图(use case diagram)和对应的文档构成。在构建需求模型之前，我们需要确定好系统边界。

在系统边界的表示上，我们首先把整个软件系统看成一个单独的系统，把软件之外的参与者先确定出来。

首先对整个系统进行边界确定。其示意图如图3所示。首先整个系统的参与者有以下几个：

1. 客户端：即墙内的PC，利用本系统与墙外的某些服务器通信
2. 服务器端：即墙外的服务，利用本系统与墙内的PC通信
3. DNS服务器：主要是客户端再必要的时候，通过DNS服务器查询服务器的ip地址。



图3. 整体系统的边界

根据以上信息建立需求模型如图4所示下：



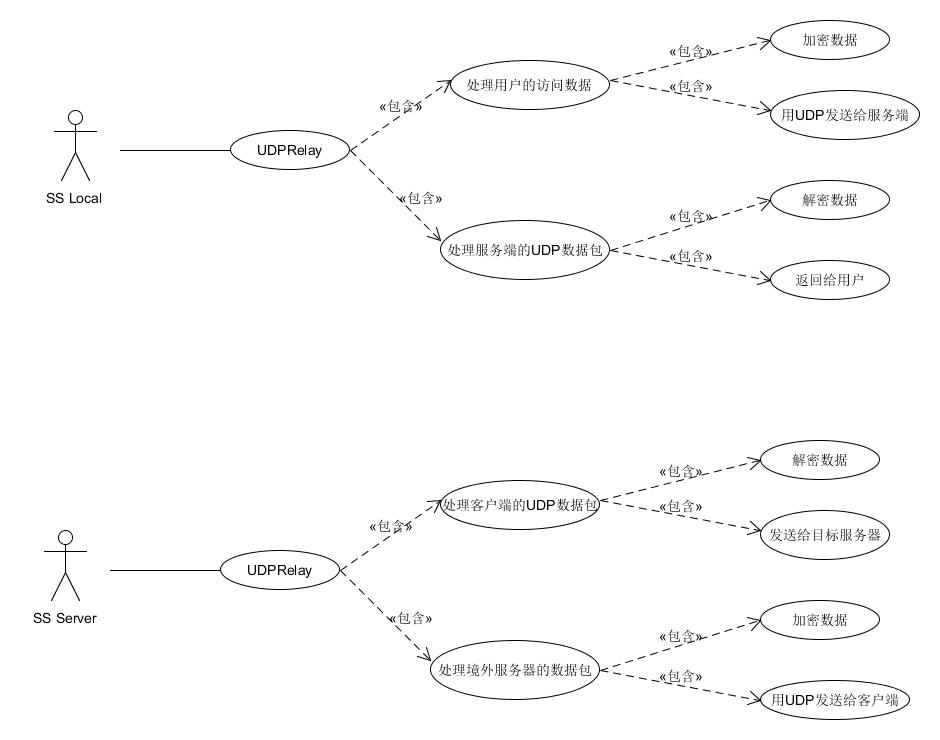
图4 需求模型用况图

其中每个用况的如下：

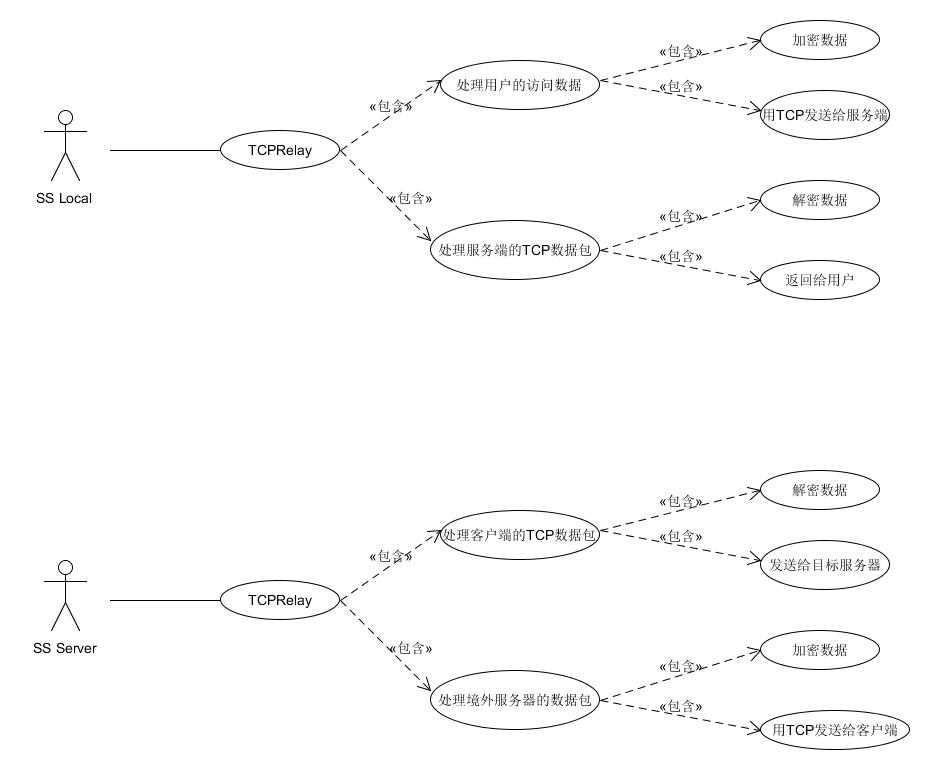
|  |  |
| --- | --- |
| 用况名称 | SS Local |
| 用况编号 | 1 |
| 行为陈述 | SS Local 获取客户端配置文件 启动客户端进程 while(1)  if DNS解析请求 then  call DNS解析  if tcp转发 then  call tcp转发  if udp转发 then  call udp转发  if 收到退出请求 then  关闭tcp、udp连接  if 收到终止请求 then  关闭客户端进程  break |

|  |  |
| --- | --- |
| 用况名称 | SS Local |
| 用况编号 | 1 |
| 行为陈述 | SS Local 获取客户端配置文件 启动客户端进程 while(1)  if DNS解析请求 then  call DNS解析  if tcp转发 then  call tcp转发  if udp转发 then  call udp转发  if 收到退出请求 then  关闭tcp、udp连接  if 收到终止请求 then  关闭客户端进程  break |

|  |  |
| --- | --- |
| 用况名称 | DNS解析 |
| 用况编号 | 1 |
| 行为陈述 | DNS 解析 if 域名为空 then  退出并提示域名为空 elif 域名已经是ip地址 then  直接返回结果 elif 域名的解析结果已经存在操作系统了 then  直接返回结果 elif 域名的解析结果已经缓存了 then  直接返回结果 else  if 域名的拼写无效 then  直接退出并提示域名的拼写错误  end if  向DNS服务器发送DNS解析请求  等待DNS解析结果返回  返回DNS解析结果  关闭DNS解析请求 end if |



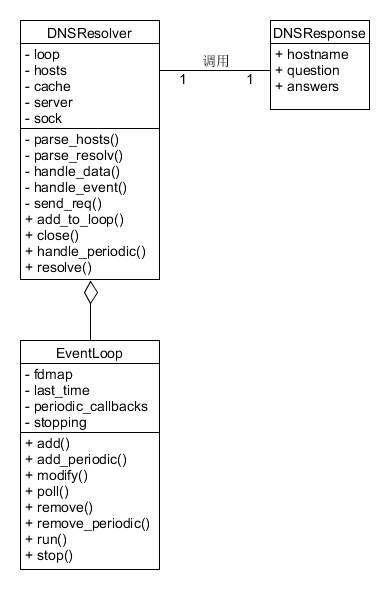
|  |  |
| --- | --- |
| 用况名称 | UPDRelay |
| 用况编号 | 1 |
| 行为陈述 | UDPRelay  While（1）  if 运行在服务器端 then  if 收到来自客户端的UDP数据包 then  解密包  将数据发送到目标服务器  end if  if 收到来自境外服务器的数据包 then  将数据加密  将加密后的数据包使用UDP发送到目标客户端  end if  end if  if 运行在客户端 then  if 收到用户的访问数据 then  加密数据  将加密的数据使用UDP发送到服务端  end if  if 收到来自服务端的UDP数据包 then  加密数据包  将加密后的数据返回给用户  end if  end if |



|  |  |
| --- | --- |
| 用况名称 | TCPRelay |
| 用况编号 | 2 |
| 行为陈述 | TCPRelay  While（1）  if 运行在服务器端 then  if 收到来自客户端的TCP数据包 then  解密包  将数据发送到目标服务器  end if  if 收到来自境外服务器的数据包 then  将数据加密  将加密后的数据包使用TCP发送到目标客户端  end if  end if  if 运行在客户端 then  if 收到用户的访问数据 then  加密数据  将加密的数据使用TCP发送到服务端  end if  if 收到来自服务端的TCP数据包 then  加密数据包  将加密后的数据返回给用户  end if  end if |

## 基本模型

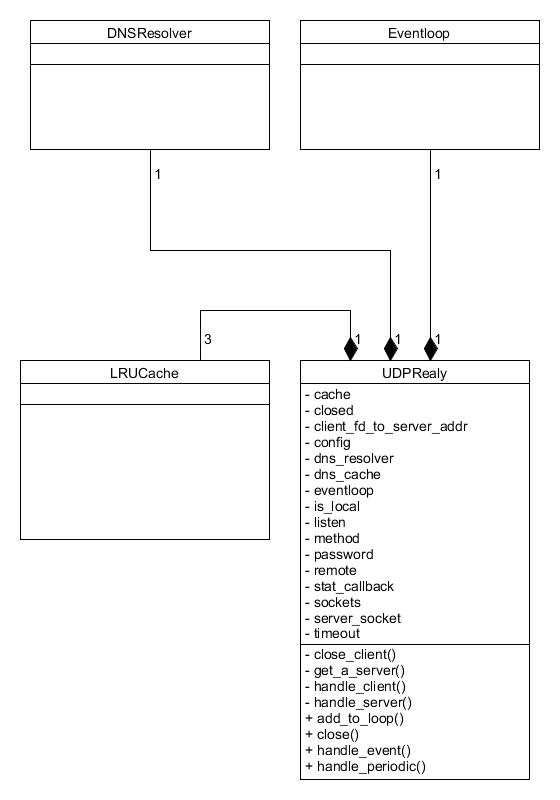
基本模型是类图



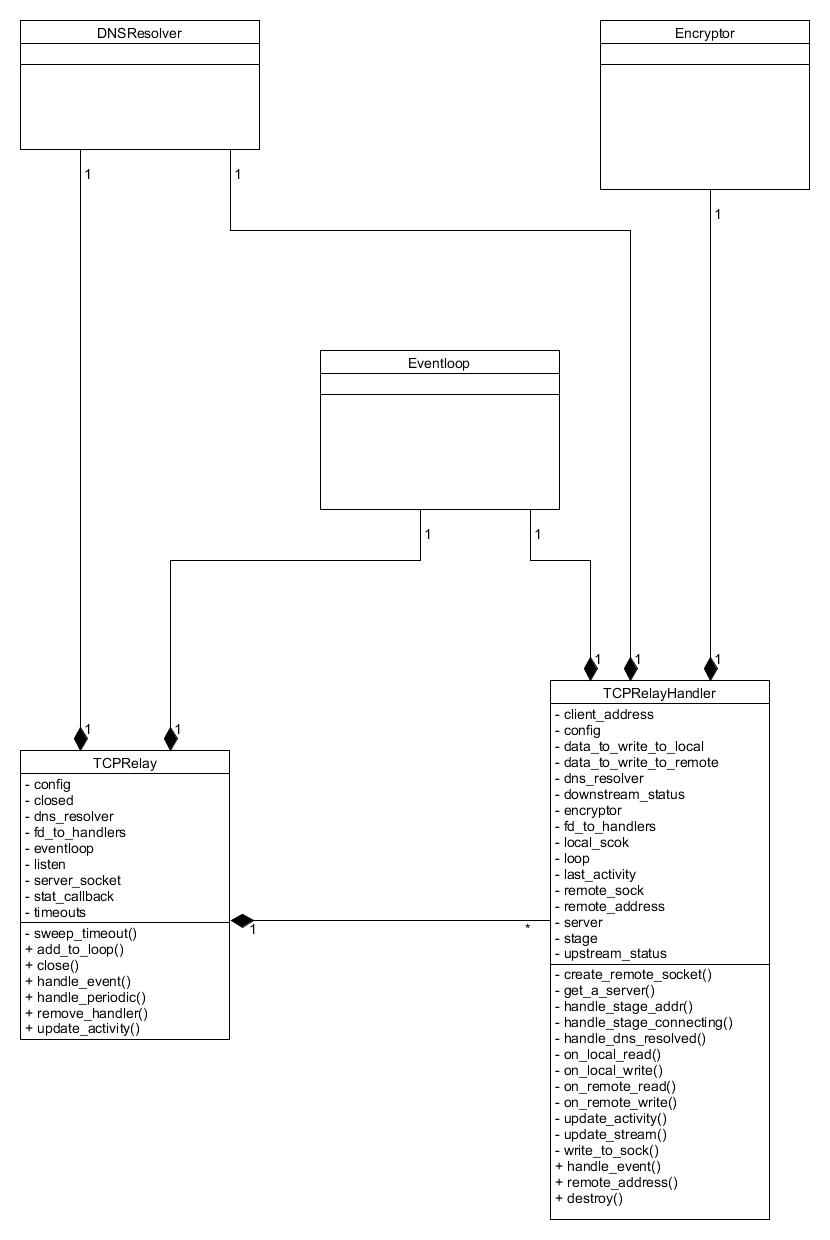
|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名：<名字> | 给出类的名称 |
|  | 解释：[<文字描述>] | 对这个类做必要的文字解释 |
|  | 一般类：[<类名>]{, 类名} | 列出被这个类继承的每个一般类 |
|  | 主动性：Yes | No | 指出这个类是不是主动类 |
|  | 持久性：Yes | No | 指出类的对象是不是持久对象 |
|  | 辅助模型：{<访问路径和名字>} | 若有针对本类的状态机图、定时图或组合  结构图等辅助模型，则指出 |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { | 针对每一个属性说明如下内容 |
|  | 名称与数据类型：<属性名>：<类型> | 给出属性的名称及数据类型 |
|  | 属性解释：[<文字描述>] | 做必要的文字解释 |
|  | 多态性：[\*|x] | 如果该属性是多态的则做标记 |
|  | 关联、聚合或组合：[关联|聚合|组合][<文字描述>] | 指出用该属性实现的关系，并作必要的说明 |
|  | 其他： |  |
|  | } |  |
| 操作说明 |  |  |
|  | { | 针对每一个操作说明如下内容 |
|  | 特征标记：<操作名>([<参数>:<类型>]{,<参数>:<类型>}[:<返回类型>]) | 给出操作名称、输入参数及类型，必要  时指明返回类型 |
|  | 操作解释：[<文字描述>] | 对操作功能做必要的解释 |
|  | 主动性：主动[进程|线程]|被动 | 指出该操作是不是主动的，若是，可指  出是进程还是线程 |
|  | 多态性：[\*|x] | 如果该操作是多态的则做标记 |
|  | 消息发送：[<类名><操作名>]{,<类名>·<操作名>} | 对由该操作发送的每一种消息，指出接收者 |
|  | 操作流程：[<访问路径和名字>] | 如有描述本操作的活动图或流程图，则  指出 |
|  | 其他： |  |
|  | } |  |
| 对象实例说明 | { | 对每种需要该类对象的处理机说明如下  内容 |
|  | 处理机：<结点名>{,<结点名>} | 指出这种处理机的每一个实际的节点 |
|  | 内存对象：{<名称>[(n元数组)][<文字描述>]} | 指出用该类创建所有内存对象 |
|  | 外存对象：{<名称>}[<文字描述>] | 指出为这个类保存的外存对象 |
|  | } |  |

|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名 | DNSResolver |
|  | 解释 | 这个类的主要功能就是处理DNS解析请求。 |
|  | 一般类 | None |
|  | 主动性 | No |
|  | 持久性 | No |
|  | 辅助模型 | 若有针对本类的状态机图、定时图或组合  结构图等辅助模型，则指出 |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { |  |
|  | 名称与数据类型 | loop : EventLoop |
|  | 属性解释 | 这个是一个Eventloop的实例，主要是为了利用IO复用机制来获得DNS解析请求返回的  结果。 |
|  | 多态性 | x |
|  | 组合 | DNSResolver必须得依靠Eventloop才能利用IO复用机制。因此两者之前有一个紧密、牢  固的关系，适用于组合范畴。 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | hosts : dict |
|  | 属性解释 | hosts保存了操作系统中从域名到ip地址的映射。如果操作系统也没有缓存到从域名到ip  地址的映射，则再调用其它方法获取映射。 |
|  | 多态性 | No |
|  | 聚合 | DNSResolver可以从hosts中来获取域名到ip地址的映射，也可以从其它方法获取映射。  因此hosts和DNSResolver之间是一种比较松散的聚合关系 |
|  | 其它 |  |
|  | } |  |
| 操作说明 |  |  |
|  | { |  |
|  | 特征标记 | parse\_hosts() |
|  | 操作解释 | 该操作主要是解析本地操作系统的hosts文件配置，并将其中的映射信息提取到hosts属性  中。 |
|  | 主动性方法 | 被动 |
|  | 多态性 | x |
|  | 消息发送 | None |
|  | 操作流程 | 如有描述本操作的活动图或流程图，则  指出 |
|  | 其他 |  |
|  | } |  |
| 对象实例说明 | { | 对每种需要该类对象的处理机说明如下  内容 |
|  | 处理机 | 指出这种处理机的每一个实际的节点 |
|  | 内存对象 | 指出用该类创建所有内存对象 |
|  | 外存对象 | 指出为这个类保存的外存对象 |
|  | } |  |

|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名 |  |
|  | 解释 |  |
|  | 一般类 |  |
|  | 主动性 |  |
|  | 持久性 |  |
|  | 辅助模型 |  |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { |  |
|  | 名称与数据类型 |  |
|  | 属性解释 |  |
|  | 多态性 |  |
|  | 组合 |  |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 |  |
|  | 属性解释 |  |
|  | 多态性 |  |
|  | 聚合 |  |
|  | 其它 |  |
|  | } |  |
| 操作说明 |  |  |
|  | { |  |
|  | 特征标记 |  |
|  | 操作解释 |  |
|  | 主动性方法 |  |
|  | 多态性 |  |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
| 对象实例说明 | { |  |
|  | 处理机 |  |
|  | 内存对象 |  |
|  | 外存对象 |  |
|  | } |  |



|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名 | UDPRelay |
|  | 解释 | 这个类负责处理客户端跟服务端的所有通过UDP协议交换数据的事件 |
|  | 一般类 | None |
|  | 主动性 | No |
|  | 持久性 | No |
|  | 辅助模型 |  |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { |  |
|  | 名称与数据类型 | Cache：LRUCache |
|  | 属性解释 | 缓存对象，用于缓存每一个client的信息，可以在该缓存找到具体的client |
|  | 多态性 | No |
|  | 组合 | Cache使用了LRUCache’类的方法存储数据，组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | closed：bool |
|  | 属性解释 | 标记UDPRelay是否关闭，如果标记为关闭，清空所有缓存，断开相关的所有udp会话 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Client\_fd\_to\_server\_addr:LRUCache |
|  | 属性解释 | 这是存储client socket号，通过该fd号能够找到具体的client |
|  | 多态性 | No |
|  | 组合 | Cache使用了LRUCache’类的方法存储数据，组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | config：json |
|  | 属性解释 | 这是配置文件的信息 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | dns\_resolver: DNSResolver |
|  | 属性解释 | 这是用于将域名转换成IP地址的实例，负责处理有关域名解析的工作，在udp连接时  有可能需要进行域名解析工作 |
|  | 多态性 | No |
|  | 组合 | 这是专门设计的处理域名解析的类，属于组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | dns\_cache: LRUCache |
|  | 属性解释 | 这是域名解析的缓存，用于将解析过的域名存储，下次需要域名解析时先查找该缓存 |
|  | 多态性 | No |
|  | 组合 | Cache使用了LRUCache’类的方法存储数据，组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | eventloop: EventLoop |
|  | 属性解释 | 事件循环，通过IO复用将事件传入该类进行处理 |
|  | 多态性 | No |
|  | 组合 | 跟Eventloop是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | listen：string |
|  | 属性解释 | 这个是udp监听的ip地址和端口 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | method: string |
|  | 属性解释 | 只是使用SOCK5的方法，通过方法和密码指定客户端与服务端之间信息交换的方法 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Password: string |
|  | 属性解释 | 这是加密信息时使用的密码，这是用户自行设定的，在本地端和服务端使用密码加密后  再传输信息 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Remote: string |
|  | 属性解释 | 远端服务端的ip地址和端口号，用于连接 |
|  | 多态性 |  |
|  | 聚合 |  |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | stat\_callback：bool |
|  | 属性解释 | 回调标记位，如果有udp连接有新的信息，标记位变为真，可以通过检查该标记为确定  有没有新的信息 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Sockets：socket[] |
|  | 属性解释 | 存储所有client的socket实例 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Server\_socket: socket |
|  | 属性解释 | 这是服务端的socket实例 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Timeout：int |
|  | 属性解释 | 时间阈值，如果计时超过阈值还没有新的信息到来，将该client标记为断开 |
|  | 多态性 |  |
|  | 组合 |  |
|  | 其它 |  |
|  | } |  |
| 操作说明 |  |  |
|  | { |  |
|  | 特征标记 | Close\_client() |
|  | 操作解释 | 该操作会关闭一个client的udp连接，并把对应的cache信息删除 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 | No |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | Get\_a\_server() |
|  | 操作解释 | 从配置信息中选取一个可用的服务端ip地址和端口并返回用于连接 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_client(sock: socket) |
|  | 操作解释 | 接收服务器回复给客户端的数据，并将其转发给客户端 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_server() |
|  | 操作解释 | 服务端处理数据，如果是远端服务端则解密转发数据，如果是本地服务端则加密传送到远端服务端 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | add\_to\_loop(loop: Eventloop) |
|  | 操作解释 | 将UDP会话加入到事件循环中 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | close() |
|  | 操作解释 | 关闭UDP转接，将所有udp连接断开 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_event(sock: socket, fd: int, event: event) |
|  | 操作解释 | 判断新事件是来自哪个socket，若是server的就新建一个UDP用户会话，若是UDP用户的，则转发 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_periodic() |
|  | 操作解释 | 判断UDPRealy是否关闭，若是，关闭与之相关的socket连接，清除缓存 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
| 对象实例说明 | { |  |
|  | 处理机 |  |
|  | 内存对象 |  |
|  | 外存对象 |  |
|  | } |  |



|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名 | TCPRelay |
|  | 解释 | 对所有的TCP连接进行管理，然后根据类型来新建TCP连接或者分发任务事件 |
|  | 一般类 | None |
|  | 主动性 | No |
|  | 持久性 | No |
|  | 辅助模型 |  |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { |  |
|  | 名称与数据类型 | Config: json |
|  | 属性解释 | 配置文件的信息 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Closed：bool |
|  | 属性解释 | 标记该tcp转接是否关闭了 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Dns\_resolver:: DNSResolver |
|  | 属性解释 | 用于tcp连接时将域名解析为ip地址 |
|  | 多态性 | No |
|  | 组合 | 使用专门定义的域名解析的DNSResolver类，是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Fd\_to\_handlers: int |
|  | 属性解释 | 这是每个tcphandler的标识号，根据标识号可以拿到具体的tcphandler |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | eventloop: EventLoop |
|  | 属性解释 | 事件循环，通过IO复用将事件传入该类进行处理 |
|  | 多态性 | No |
|  | 组合 | 跟Eventloop是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Listen:string |
|  | 属性解释 | 这是tcprelay监听的ip地址和端口 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Server\_socket: socket |
|  | 属性解释 | 这是服务端的socket连接 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | stat\_callback: bool |
|  | 属性解释 | 回调标记位，标记有没有新的信息 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | 名称与数据类型 | Timeouts: int |
|  | 属性解释 | 时间阈值，如果handler超过时间阈值还没有新的消息，则删除该handler |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
| 操作说明 |  |  |
|  | { |  |
|  | 特征标记 | sweep\_timeout() |
|  | 操作解释 | 当handler关闭时，需要删除对应的计时 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 | No |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | add\_to\_loop(loop: Eventloop) |
|  | 操作解释 | 将TCP会话加入到事件循环中 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | Close() |
|  | 操作解释 | 关闭tcp转接 进程 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_event(sock: socket, fd: int, event: event) |
|  | 操作解释 | 判断事件是新建handler还是已有连接的事件并分发事件 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_periodic() |
|  | 操作解释 | 检查一个TCPRealy是否关闭，如果关闭了，就断开与其相关的TCP连接，并将其从事件循环中删除，调用 \_sweep\_timeout 定期清理一段时间内不活跃的 TCPRelayHandler |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | remove\_handler(handler: TCPRelayHandler) |
|  | 操作解释 | 删除一个TCPRealy Handler |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | update\_activity(handler: TCPRelayHandler, data\_len: int) |
|  | 操作解释 | 更新一个handler的激活状态，将该handler的计时刷新，更新超时队列 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  |  |  |
| 对象实例说明 | { |  |
|  | 处理机 |  |
|  | 内存对象 |  |
|  | 外存对象 |  |
|  | } |  |

|  |  |  |
| --- | --- | --- |
| 类的总体说明 |  |  |
|  | 类名 | TCPRelayHandler |
|  | 解释 | 这个是用来处理单个TCP连接的事件的handler，每一个handler负责一个tcp连接的事件 |
|  | 一般类 | None |
|  | 主动性 | No |
|  | 持久性 | No |
|  | 辅助模型 |  |
|  | 其他 |  |
| 属性说明 |  |  |
|  | { |  |
|  | 名称与数据类型 | Client\_address: string |
|  | 属性解释 | TCP连接中客户端的ip地址 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Config: json |
|  | 属性解释 | 配置文件的信息 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | data\_to\_write\_to\_local：bytes |
|  | 属性解释 | 需要发送给本地端的数据 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | data\_to\_write\_to\_remote：bytes |
|  | 属性解释 | 需要发送给服务端的数据 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Dns\_resolver:: DNSResolver |
|  | 属性解释 | 用于tcp连接时将域名解析为ip地址 |
|  | 多态性 | No |
|  | 组合 | 使用专门定义的域名解析的DNSResolver类，是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | downstream\_status：int |
|  | 属性解释 | 标记输入数据流是属于什么数据流 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | encryptor：Encryptor |
|  | 属性解释 | 加密类实例，用于传输数据时的加密 |
|  | 多态性 | No |
|  | 组合 | 与Encryptor类是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | fd\_to\_handlers：int |
|  | 属性解释 | 属于该handler的fd标识号，专属于该handler的标识号 |
|  | 多态性 |  |
|  | 聚合 |  |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | local\_scok：socket |
|  | 属性解释 | 与本地tco连接的socket实例 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | loop：Eventloop |
|  | 属性解释 | 事件循环，通过IO复用将事件传入该类进行处理 |
|  | 多态性 | No |
|  | 聚合 | 跟Eventloop是组合关系 |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | last\_activity：int |
|  | 属性解释 | 记录上次信息到来的时刻 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | remote\_sock：socket |
|  | 属性解释 | 这是跟服务端的socket实例 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | remote\_address：string |
|  | 属性解释 | 这是跟服务端tcp连接时服务端的ip地址 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Server：socket |
|  | 属性解释 | 这是与服务端的连接socket |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | Stage：int |
|  | 属性解释 | 标识当前handler的状态 |
|  | 多态性 | No |
|  | 组合 | No |
|  | 其它 |  |
|  | } |  |
|  | { |  |
|  | 名称与数据类型 | upstream\_status：int |
|  | 属性解释 | 标识输出数据流属于什么数据流 |
|  | 多态性 | No |
|  | 聚合 | No |
|  | 其它 |  |
|  | } |  |
| 操作说明 |  |  |
|  | { |  |
|  | 特征标记 | create\_remote\_socket（ip: string, port: int） |
|  | 操作解释 | 建立与远端服务器的TCP连接，并返回连接的socket |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | get\_a\_server（） |
|  | 操作解释 | 从可选服务器中选出一个端口建立TCP连接 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_stage\_addr(data: bytes) |
|  | 操作解释 | 建立SOCK5的连接，握手阶段 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_stage\_connecting(data: bytes) |
|  | 操作解释 | 建立SOCK5的连接，连接阶段 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_dns\_resolved(result: string, error:string) |
|  | 操作解释 | 在建立SOCK5时，通过域名解析将服务器转换成IP地址，然后根据IP地址和端口建立连接 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | on\_local\_read() |
|  | 操作解释 | 本地端向服务端读取信息 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | on\_local\_write() |
|  | 操作解释 | 本地端向服务端发送信息 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | on\_remote\_read() |
|  | 操作解释 | 服务端从本地端读取信息 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | on\_remote\_write() |
|  | 操作解释 | 服务端向本地端发送信息 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | update\_activity(data\_len: int) |
|  | 操作解释 | 更新TCPhandler的状态 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | update\_stream(stream: int, status: int) |
|  | 操作解释 | 更新当前socket监听的事件，会在status发生变化时更新 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | write\_to\_sock(data:bytes, sock: socket) |
|  | 操作解释 | 向socket传输数据 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | handle\_event(sock: socket, event:event) |
|  | 操作解释 | 处理TCPRelay分发过来的事件 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | remote\_address() |
|  | 操作解释 | 获取远端服务端地址 |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
|  | { |  |
|  | 特征标记 | destroy() |
|  | 操作解释 | 删除TCPRelayHandler |
|  | 主动性方法 | 被动 |
|  | 多态性 | No |
|  | 消息发送 |  |
|  | 操作流程 |  |
|  | 其他 |  |
|  | } |  |
| 对象实例说明 | { |  |
|  | 处理机 |  |
|  | 内存对象 |  |
|  | 外存对象 |  |
|  | } |  |

## 辅助模型

本项目主要用到的辅助模型包括

# OOD模型

OOD阶段是建立在OOA阶段的基础上，对OOA模型作必要的修改和调整或补充某些细节。其模型框架如下图所示：



本小节主要是按照问题域部分、数据接口部分、控制驱动部分和人机交互部分去阐述。

## 问题域部分

问题域部分主要是对

## 数据接口部分

数据接口部分包括

## 控制驱动部分

控制驱动部分包括

## 人机交互部分

人机交互部分包括

# 总结

总结