moCloud设计文档

# 历史

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Description** |
| V1.0.0 | Wujinlei | 20180211 | Create |
|  |  |  |  |

# 1.概述

moCloud定位为一个私有云系统，包含了稳定运行的云服务器，和可以正常使用的云客户端；

Server运行在树莓派上，client实现在手机上。两者可以通过局域网互联，是moCloud正常交互的前提，后续可以考虑将服务扩展至公网。

由于树莓派本身的硬件性能和存储size有限，因此设计方案是：

1. Server端存放一套数据文件；
2. 所有的客户端看到的都是这一套数据；
3. 客户端分两种权限：管理员和普通用户；普通客户端只能上传文件、下载文件、播放文件、获取文件列表等；管理员除此之外，还可以删除文件；

本文档，介绍moCloud的设计方式，实现流程，可以为后续参与该项目的人提供参考，也可以为后续维护提供依据。

# 2.应用场景

## 2.1.客户端

客户端的使用流程为：

* 启动客户端；
* 注册/登录客户端；
* 展示文件列表；
* 上传文件；下载文件；播放文件；删除文件；
* 退出登录；

### 2.1.1.启动

客户端程序运行时，首先应完成启动的动作，只有完成了启动这个步骤，我们才认为具备了与server进行交互的能力，才进行后续的注册/登录、文件操作等功能；如果启动失败，客户端只能退出登录，或调整后重新登录；

启动这个动作，主要负责：与server建立连接；与server进行密钥协商，得到本轮的密钥。过程描述如下：

1. 读取配置文件，获取客户端的ip地址：
   1. 读取配置失败，启动失败，提示用户后退出启动流程；
2. 使用随机分配的port，创建socket，连接server：
   1. 创建socket失败，启动失败，提示用户后退出启动流程；
   2. 连接server失败，启动失败，提示用户后退出启动流程；
3. 密钥协商，获取本次通信使用的密钥：
   1. 密钥协商失败，启动失败，提示用户后退出启动流程；
4. 启动心跳包流程；

**密钥协商**，负责从server端得到本次通信所使用的加解密算法，和密钥，其流程如下：

1. 构造密钥协商的request；
2. 使用本地存储的公钥，对request加密，得到request\_cipher;
3. 发送request\_cipher到server，并等待返回值：
   1. 存在一个超时时间，在该超时时间内如果没有能够得到返回值，属于协商失败的一种情况，需要提示用户后退出启动流程；
4. 得到返回值，使用公钥对返回值进行解密；
5. 将得到的密钥信息(加解密算法、密钥长度、密钥的值等)，保存在本地内存，供本次通信使用；

**心跳包流程**，指的是客户端需要间隔N秒，向服务器发送一个心跳包，告知服务器它还在进行连接，如果超过M秒没有发送，服务器会把这个客户端的连接信息删除，客户端需要重新连接才可以与server继续通信：

1. 生成一条心跳包请求request；
2. 使用密钥信息，加密得到cipherRequest；
3. 发送心跳包到server，并等待返回值；
   1. 同样有一个超时时间，如果超过了超时时间没有得到返回值，认为网络出现了异常，提示用户后，清空本轮的密钥信息等相关通信数据，尝试反复启动，直至启动成功；
4. 得到返回值cipherResponse后，执行解密；
5. 分析response，查看是否成功：
   1. 返回值可以反馈给客户端，文件列表是否发生了变化：
      1. 如果客户端已经成功登陆了，主动更新文件列表；
      2. 如果客户端没有登陆，忽略该返回值；
6. 间隔N秒后，再次执行一条心跳包请求；

### 2.1.2.注册/登录

注册和登录严格来说是两个动作，但由于体现在一起，而且注册后会有自动的登录流程，因此放在一起。这里只描述注册流程，登录流程可据此预见到：

1. 捕获注册所需要的用户名、密码信息，并据此生成一条request；
2. 使用本轮的密钥，对request加密，得到cipherRequest；
3. 发送cipherRequest，并等待返回值：
   1. 超时时间内得不到返回值，认为注册失败，提示用户后，退出注册流程，但不退出客户端，用户可以使用本次密钥协商的结果，继续注册；
4. 对返回值cipherResponse，使用密钥信息进行解密；
5. 分析解密后的response：
   1. 用户名已经被使用：必须更换一个用户名；
6. 注册成功后，自动将用户名、密码组合成一个request，用来登陆；
7. 对request加密后，发送到server，并等待返回值；
   1. 同样有超时时间，如果超过了超时时间没有返回，认为登录失败，提示登录错误，但是一定要告知注册已经成功了；
8. 获取返回值cipherResponse，解密，判断登录是否成功：
   1. 要考虑登录失败的情况，比如服务器现在接入的用户数多了，不再接入了，等；
9. 登录成功，直接进入初始化流程；

### 2.1.3.初始化

登录之后，自动执行初始化的动作，初始化主要包括两部分功能：获取文件列表并展示；继续上次未完成的上传、下载认为。

获取文件列表并展示，具体流程可以参见2.1.4详述的流程；

继续上次未完成的任务，例如上传、下载文件等，以上传文件为例，流程如下：

1. 查看是否存在unCompletedTask.log；
2. 解析文件，查看是否存在未完成的上传文件的任务；
3. 判断这些文件，是否依然存在于本地目录中，将其分类后，提示用户：有多少文件上次未完成，上次的时间是何时，现在有多少文件还存在、可以继续上传，上次上传了多少进度；
4. 请用户主动选择，是否继续上传，用户可以选择其中的一个或多个，可以选择上传了一部分的，或者完全没有上传过的；但是用户选择的，一定是当前本地存在的文件；
5. 如果选择了没有上传过的文件，直接执行2.1.7.的上传文件流程即可；
6. 如果选择了上传了一部分的文件，
   1. 首先向服务器发送请求，询问上次发送的文件是否还存在，是否还是该用户在该时间上传的：
   2. 等待返回值，支持超时时间，超时时间内没有返回，认为失败，请用户重新选择；
   3. 解密返回值，分析返回值：
      1. 如果依然是该用户上传的，并且size、offset等基本信息都正确，那么直接执行2.1.7的流程，继续上传文件；
      2. 如果不再是该用户上传的文件，或者文件在server端不存在了，或者offset等信息匹配失败，都需要告知用户该情况，并提示上传失败；

上传文件可以考虑有两个阶段开发，断点续传的功能，放到第二个阶段；

### 2.1.4.获取文件列表

文件列表展示，是moCloudClient的主界面；

获取文件列表，操作步骤如下：

1. 生成request；
2. 加密，得到cipherRequest；
3. 发送请求到server，并等待返回值：
   1. 超时时间内获取不到，提示用户，等待主动发起下一轮的获取；
4. 解密response，分析response，得到文件列表信息；
5. 将文件列表信息，展示在界面上；

### 2.1.5.下载文件

TODO

### 2.1.6.在线播放文件

TODO

### 2.1.7.上传文件

TODO

### 2.1.8.删除文件

TODO

## 2.2.服务器

TODO

注册，登录等

# 3.序列图

# 4.上下文图

# 5.模块图

# 6.内存和数据

线程，内存

# 7.类图

# 8.流程图

# 9.头文件

# 10.开发阶段

## 10.1.M1--创建模型、支持基本功能

第一阶段，主要创建客户端和服务器模型，保证正常的通信机制，和基本的业务模型，具体来说，主要是如下几个方面：

* 加解密模块：支持正常的加解密；
* 信息校验模块：支持需要的校验算法；
* 通信模块：client和server可以正常通信；
* 心跳包管理模块：客户端可以正常发送心跳包，服务器可以管理心跳包，两者都可以根据心跳包的正常与否，做出正确的反应；
* 基本功能之密钥协商：可以正常进行密钥协商；
* 基本功能之获取文件列表：client可以正确获取到server当前的所有文件的列表信息，并展示给用户；

1. MoCloudUtilsCrypt，moCloudUtilsCheck，打桩；
2. moCloudUtilsType，moCloudUtils，实现；
3. moCloudClient：
   1. Init: connectToServer; doKeyAgree;
   2. startHeartBeat : sayHiToServer;
   3. signUp : 注册;
   4. signIn : 登录；
   5. getFileList;
   6. unInit: sayByeByeToServer; freeAllResources;
4. moCloudServer：启动，接收client的connect请求，每一个client新建一个thread处理；
5. moCloudServer：文件管理模块；
6. moCloudServer：client管理模块，心跳包管理最重要；
7. moCloudServer：支持基本功能之密钥协商和获取文件列表；
8. 正式实现moCloudUtilsCrypt；
9. 正式实现moCloudUtilsCheck；

## 10.2.M2--上传、下载、续传文件

## 10.3.M3--删除文件、在线播放文件

# 11.遗留问题