Seafile

Sharing & Collaboration

目錄

简介

Seafile 是一个开源的文件云存储平台，更注重于隐私保护和对团队文件协作的支持。

Seafile 通过“资料库”来分类管理文件，每个资料库可单独同步，用户可加密资料库， 且密码不会保存在服务器端，所以即使是服务器管理员也无权访问你的文件。

Seafile 允许用户创建“群组”，在群组内同步文件、创建维基、发起讨论等，方便团队内协同工作。

软件许可协议

Seafile 及其桌面、移动客户端遵循 GPLv3。

Seahub（Seafile 服务器的 web 端）遵循 Apache License。

关于

本手册“源码”托管在Github https://github.com/haiwen/seafile-docs-cn

联系方式

- Twitter: @seafile https://twitter.com/seafile
- 微博： @seafile

更多内容

- 更多内容请见 Seafile Wiki

概览

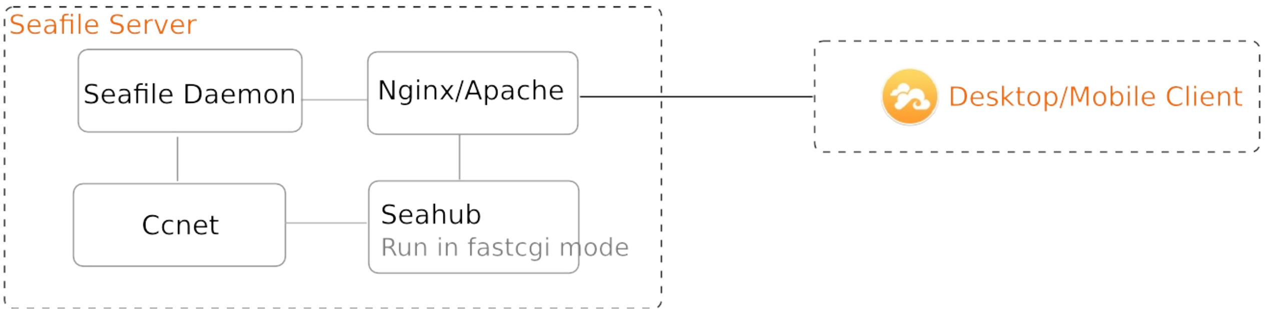
- Seafile 组件
- FAQ
- Changelog
- Roadmap
- Contribution

Seafile服务器组件

Seafile 包含以下系统组件：

- Seahub：网站界面，供用户管理自己在服务器上的数据和账户信息。Seafile服务器通过"gunicorn"（一个轻量级的Python HTTP服务器）来提供网站支持。Seahub作为gunicorn的一个应用程序来运行。
- **Seafile server** (`seaf-server`)： 数据服务进程, 处理原始文件的上传/下载/同步。
- **Ccnet server** (`ccnet-server`)： 内部 RPC 服务进程，连接多个组件。
- **Controller**: 监控 ccnet 和 seafile 进程，必要时会重启进程。

下面这张图显示了将 Seafile 部署在 Nginx/Apache 后的架构。客户端需要在选项界面中开启 "sync over HTTP/HTTPS"。



Roadmap

3.1 (To be released before 2014.7.30)

Version 3.1 will focus on improving the file syncing algorithm. Including

- Improve performance: easily syncing 10k+ files in a library.
- Don't need to download files if they are moved to another directory.

Platform

- Add OAuth support for easy integrating into other systems
- Enable creating sub-libraries from an encrypted library
- Rename FileServer to FileServer to remove confusing.

Web

- Enable deleting of personal messages
- Improved notification
- Upgrade pdf.js
- Password protection for sharing links

More details

- Seafile roadmap for 3-x
- Seafile roadmap for 2-x

FAQ

When downloading a library, the client hangs at "connecting server"

First, you can check the ccnet.log in client (`~/.ccnet/logs/ccnet.log` for Linux, `C:/users/your_name/ccnet/logs/ccnet.log` for Windows) to see what's wrong.

Possible reasons:

- Miss config of `SERVICE_URL`: Check whether the value of is set correctly in server's `ccnet.conf`.
- Firewall: Ensure the firewall is configured properly. See [[Firewall Settings for Seafile Server]]

Trouble shooting:

- Manually telnet to see if you can connect: `telnet your-server-IP-or-domain 10001`

Failed to upload/download file online

- Make sure you firewall for seafile fileserver is opened.
- Using chrome/firefox debug mode to find which link is given when click download button and what's wrong with this link

Does Seafile server support Python 3?

No, You must have python 2.6 or 2.7 installed on your server.

Can Seafile server run on FreeBSD?

There was an unconfirmed solution on the internet, which later has vanished. (Please explain the installation routine if you successfully set up Seafile in FreeBSD.)

Website displays "Page unavailable", what can I do?

- You can check the back trace in seahub log files('''installation folder/logs/seahub_django_request.log''')

- You can also turn on debug by adding `DEBUG = True` to seahub_settings.py and restart seahub by `./seahub.sh restart`, then refresh that page, all the debug infomations will be displayed. Make sure ./seahub.sh was started as: ./seahub.sh start-fastcgi

Avatar pictures vanished after upgrading server, what can I do?

- You need to check whether the "avatars" symbolic link under seahub/media/ is correctly link to ../../../seahub-data/avatars. If not, you need to correct the link according to the "minor upgrade" section in [[Upgrading-Seafile-Server]]

- If your avatars link is correctly linked, and avatars are still broken, you may refresh seahub cache by `rm -rf /tmp/seahub_cache/*`

How to change seafile-data location after setup?

Modify file seafile.ini under ccnet. This file contains the location of seafile-data. Move seafile-data to another place, like `/opt/new/seafile-data`. Then modify the file accordingly.

Failed to send email, what can I do?

Please check logs/seahub.log.

There are some common mistakes:

1. Check whether there are some typos in the config, e.g., forget single quote, EMAIL_HOST_USER = XXX, which should be EMAIL_HOST_USER = 'XXX'
2. Your mail server is not available.

Changelog

v3.0.4 Server

- [API] Add replace if exist into upload-api
- Show detailed error message when Gunicorn failed to start
- Improve object and block writting performance
- Add retry when failed getting database connection

Previous changelog

- changelog of server
- changelog of client
- changelog of 2-x

Contribution

Licensing

Seafile and its desktop and mobile clients are published under the GPLv3.

The Seafile server's web end, i.e. Seahub, is published under the Apache License.

Discussion

Google Group: https://groups.google.com/forum/?fromgroups#!forum/seafile

IRC: #seafile on freenode

Follow us @seafile https://twitter.com/seafile

Reporting a Bug?

- Find the existing issues that fit your situation if any, or create a new issue. We are using github as our issue tracer https://github.com/haiwen/seafile/issues?state=open
- Join us on Google Group: https://groups.google.com/forum/?fromgroups#!forum/seafile

Code Style

The source code of seafile is ISO/IEC 9899:1999 (E) (a.k.a. C99) compatible. Take a look at code standard.

Linux 下部署 Seafile 服务器

此文档用来说明如何使用预编译安装包来部署 Seafile 服务器.

使用安装脚本在 Ubuntu 14.04 上快速安装。

我们准备了一个安装脚本帮助您在 Ubuntu 14.04 快速的安装部署 Seafile 服务(配置好 MariaDB, Memcached, WebDAV, Ngnix 和开机自动启动脚本)： https://github.com/haiwen/seafile-server-installer-cn

在此基础上，您可以继续根据下面的文档来配置 AD，邮箱发送等服务。

家庭/个人 环境下部署 Seafile 服务器

- 部署 Seafile 服务器（使用 SQLite）

生产/企业 环境下部署 Seafile 服务器

企业环境下我们建议使用 MySQL 数据库，并将 Seafile 部署在 Nginx 或者 Apache 上，如果对于 Nginx 和 Apache 都不是很熟悉的话，我们建议使用 Nginx，相对于 Apache 来说，Nginx 使用起来比较简单。

基础功能:

- 部署 Seafile 服务器（使用 MySQL）
- Nginx 下配置 Seahub
- Nginx 下启用 Https
- Apache 下配置 Seahub
- Apache 下启用 Https
- 使用 Memcached

高级功能:

- Seafile LDAP 配置
- 开机启动 Seafile
- 防火墙设置
- Logrotate 管理系统日志

其他部署事项

- 使用 NAT
- 非根域名下部署 Seahub
- 从 SQLite 迁移至 MySQL

更多配置选项（比如开启用户注册功能），请查看 服务器个性化配置。

注意 如果在部署 Seafile 服务器是遇到困难

1. 阅读 Seafile 组件 以了解 Seafile 的运行原理。

2. 安装常见问题。
3. 通过 论坛 或者 Seafile 服务器管理员 QQ 交流群: 315229116 寻求帮助。

升级 Seafile 服务器

- 升级

个人打包 Seafile 服务器

如果想要自己打包 Seafile 服务器（安装在自己喜欢的 Linux 系统中），请务必使用 tags:

- 当 Seafile 客户端发行新版本时，比如 `v3.0.1`， 我们会将 `v3.0.1` 标签打在 **ccnet**, **seafile** 和 **seafile-client** 上。
- 同样，当 Seafile 服务器发行新版本时，比如 `v3.0.1`， 我们会将 `v3.0.1` 标签打在 **ccnet**, **seafile** 和 **seahub** 上。
- 对于 **libsearpc**，会一直使用 `v3.0-latest` 标签。

注意: 每个项目的版本号和 tag 名，不存在必然联系。

部署 Seafile 服务器（使用 SQLite）

这份文档用来说明如何使用预编译好的文件包来安装和运行 Seafile 服务器。

下载

到下载页面下载最新的服务器安装包.

部署和目录结构

注意:如果你把 Seafile文件放在一个外部存储的目录里（比如NFS，CIFS）,你应该使用 MySQL 而不是SQLite来作为数据库。请参考下载和安装Seafile服务器（使用MySQL）。

假设你公司的名称为"haiwen",你也已经下载 seafile-server_1.4.0_* 到你的home 目录下。 我们建议这样的目录结构:

```
mkdir haiwen
mv seafile-server_* haiwen
cd haiwen
#将 seafile-server_* 移动到 haiwen 目录下后
tar -xzf seafile-server_*
mkdir installed
mv seafile-server_* installed
```

现在，你的目录看起来应该像这样：

```
# tree . -L 2
.
├── installed
│   └── seafile-server_1.4.0_x86-64.tar.gz
└── seafile-server-1.4.0
    ├── reset-admin.sh
    ├── runtime
    ├── seafile
    ├── seafile.sh
    ├── seahub
    ├── seahub.sh
    ├── setup-seafile.sh
    └── upgrade
```

这样设计目录的好处在于

- 和 seafile 相关的配置文件都可以放在 haiwen 目录下，便于集中管理.
- 后续升级时,你只需要解压最新的安装包到"haiwen"目录下.

这样你可以重用*"haiwen"*目录下已经存在的配置文件，而不用重新配置.

安装 **Seafile** 服务器

安装前的准备工作

安装 Seafile 服务器之前，请确认已安装以下软件

- python 2.7
- python-setuptools
- python-imaging
- sqlite3

```
#Debian系统下
apt-get update
apt-get install python2.7 python-setuptools python-imaging sqlite3
```

安装

```
cd seafile-server-*
./setup-seafile.sh  #运行安装脚本并回答预设问题
```

如果你的系统中没有安装上面的某个软件，那么 Seafile 初始化脚本会提醒你安装相应的软件包. 该脚本会依次询问你一些问题，从而一步步引导你配置 Seafile 的各项参数

参数	作用	说明
seafile server name	seafile 服务器的名字，将来在客户端会显示为这个名字	3 ~ 15 个字符，可以用英文字母，数字，下划线
seafile server ip or domain	seafile 服务器的 IP 地址或者域名	客户端将通过这个 IP 或者地址来访问你的 Seafile 服务
seafile data dir	seafile 数据存放的目录，用上面的例子，默认将是 /data/haiwen/seafile-data	seafile 数据将随着使用而逐渐增加，请把它放在一个有足够大空闲空间的分区上
seafile fileserver port	seafile fileserver 使用的 TCP 端口	一般使用默认的 8082 端口，如果已经被占用，可以设置为其他的端口

如果安装正确完成，会打印成功消息

现在你的目录结构将会是如下:

```
#tree haiwen -L 2
haiwen
├── ccnet               # configuration files
│   ├── ccnet.conf
│   ├── mykey.peer
│   ├── PeerMgr
│   └── seafile.ini
```

```
├── installed
│   └── seafile-server_1.4.0_x86-64.tar.gz
├── seafile-data
│   └── seafile.conf
├── seafile-server-1.4.0  # active version
│   ├── reset-admin.sh
│   ├── runtime
│   ├── seafile
│   ├── seafile.sh
│   ├── seahub
│   ├── seahub.sh
│   ├── setup-seafile.sh
│   └── upgrade
├── seafile-server-latest  # symbolic link to seafile-server-1.4.0
├── seahub-data
│   └── avatars
├── seahub.db
├── seahub_settings.py   # optional config file
└── seahub_settings.pyc
```

`seafile-server-latest` 文件夹是当前 Seafile 服务器文件夹的符号链接.将来你升级到新版本后, 升级脚本会自动更新使其始终指向最新的 Seafile 服务器文件夹.

启动运行 Seafile 服务器

启动之前

因为 Seafile 在客户端和服务器之间使用持续连接，如果你的客户端数量巨大, 你应该在启动 Seafile 之前修改你的 Linux 文件最大打开数，如下:

```
ulimit -n 30000
```

启动 Seafile 服务器和 Seahub 网站

在 seafile-server-1.4.0 目录下，运行如下命令：

- 启动 Seafile:

```
./seafile.sh start # 启动 Seafile 服务
```

- 启动 Seahub

```
./seahub.sh start <port> # 启动 Seahub 网站 （默认运行在8000端口上）
```

小贴士: 你第一次启动 seahub 时， `seahub.sh` 脚本会提示你创建一个 seafile 管理员帐号。

服务启动后, 打开浏览器并输入以下地址

```
http://192.168.1.111:8000/
```

你会被重定向到登陆页面. 输入你在安装 Seafile 时提供的用户名和密码后，你会进入 Myhome 页面，新建资料库.

恭喜**!** 现在你已经成功的安装了 Seafile 服务器.

在另一端口上运行 **Seahub**

如果你不想在默认的 8000 端口上运行 Seahub, 而是想自定义端口（比如8001）中运行，请按以下步骤操作:

- 关闭 Seafile 服务器

```
./seahub.sh stop # 停止 Seafile 进程
./seafile.sh stop # 停止 Seahub
```

- 更改`haiwen/ccnet/ccnet.conf`文件中`SERVICE_URL` 的值(假设你的 ip 或者域名时`192.168.1.100`), 如下:

```
SERVICE_URL = http://192.168.1.100:8001
```

- 重启 Seafile 服务器

```
./seafile.sh start # 启动 Seafile 服务
./seahub.sh start 8001 # 启动 Seahub 网站 （运行在8001端口上）
```

`ccnet.conf` 更多细节请看[[Seafile服务器配置选项]].

关闭**/**重启 **Seafile** 和 **Seahub**

关闭

```
./seahub.sh stop # 停止 Seahub
./seafile.sh stop # 停止 Seafile 进程
```

重启

```
./seafile.sh restart # 停止当前的 Seafile 进程，然后重启 Seafile
./seahub.sh restart  # 停止当前的 Seahub 进程，并在 8000 端口重新启动 Seahub
```

如果停止**/**重启的脚本运行失败

大多数情况下 seafile.sh seahub.sh 脚本可以正常工作。如果遇到问题：

- 使用**pgrep**命令检查 seafile/seahub 进程是否还在运行中

```
pgrep -f seafile-controller # 查看 Seafile 进程
pgrep -f "manage.py run_gunicorn" # 查看 Seahub 进程
```

- 使用**pkill**命令杀掉相关进程

```
pkill -f seafile-controller # 结束 Seafile 进程
pkill -f "manage.py run_gunicorn" # 结束 Seafile 进程
```

OK!

查看 Seafile 更多信息请移至..

- 管理员手册

部署 Seafile 服务器（使用 MySQL）

此文档用来说明通过预编译好的安装包，来安装并运行基于 MySQL 的 Seafile 服务器.

下载

到下载页面下载最新的服务器安装包.

部署和目录设计

假设你公司的名称为 **haiwen**,你也已经下载 seafile-server_1.4.0_* 到你的 **home** 目录下。 我们建议这样的目录结构:

```
mkdir haiwen
mv seafile-server_* haiwen
cd haiwen
#将 seafile-server_* 移动到 haiwen 目录下后
tar -xzf seafile-server_*
mkdir installed
mv seafile-server_* installed
```

现在，你的目录看起来应该像这样：

```
#tree haiwen -L 2
haiwen
├── installed
│   └── seafile-server_1.8.2_x86-64.tar.gz
└── seafile-server-1.8.2
    ├── reset-admin.sh
    ├── runtime
    ├── seafile
    ├── seafile.sh
    ├── seahub
    ├── seahub.sh
    ├── setup-seafile.sh
    └── upgrade
```

这样设计目录的好处在于

- 和 seafile 相关的配置文件都可以放在 **haiwen** 目录下，便于集中管理.
- 后续升级时,你只需要解压最新的安装包到 **haiwen** 目录下.

安装 Seafile 服务器

安装前的准备工作

安装 Seafile 服务器之前，请确认已安装以下软件

- python 2.7
- python-setuptools
- python-imaging
- python-mysqldb

```
#在Debian/Ubuntu系统下
apt-get update
apt-get install python2.7 python-setuptools python-imaging python-mysqldb
```

安装

```
cd seafile-server-*
./setup-seafile-mysql.sh  #运行安装脚本并回答预设问题
```

如果你的系统中没有安装上面的某个软件，那么 Seafile初始化脚本会提醒你安装相应的软件包.

该脚本会依次询问你一些问题，从而一步步引导你配置 Seafile 的各项参数:

参数	作用	说明
seafile server name	seafile 服务器的名字，目前只在客户端的日志文件中使用	3 ~ 15 个字符，可以用英文字母，数字，下划线
seafile server ip or domain	seafile 服务器的 IP 地址或者域名	客户端将通过这个 IP 或者地址来访问你的 Seafile 服务
seafile data dir	seafile 数据存放的目录，用上面的例子，默认将是 /data/haiwen/seafile-data	seafile 数据将随着使用而逐渐增加，请把它放在一个有足够大空闲空间的分区上
seafile fileserver port	seafile fileserver 使用的 TCP 端口	该端口用于文件同步，请使用默认的 8082，不能更改。

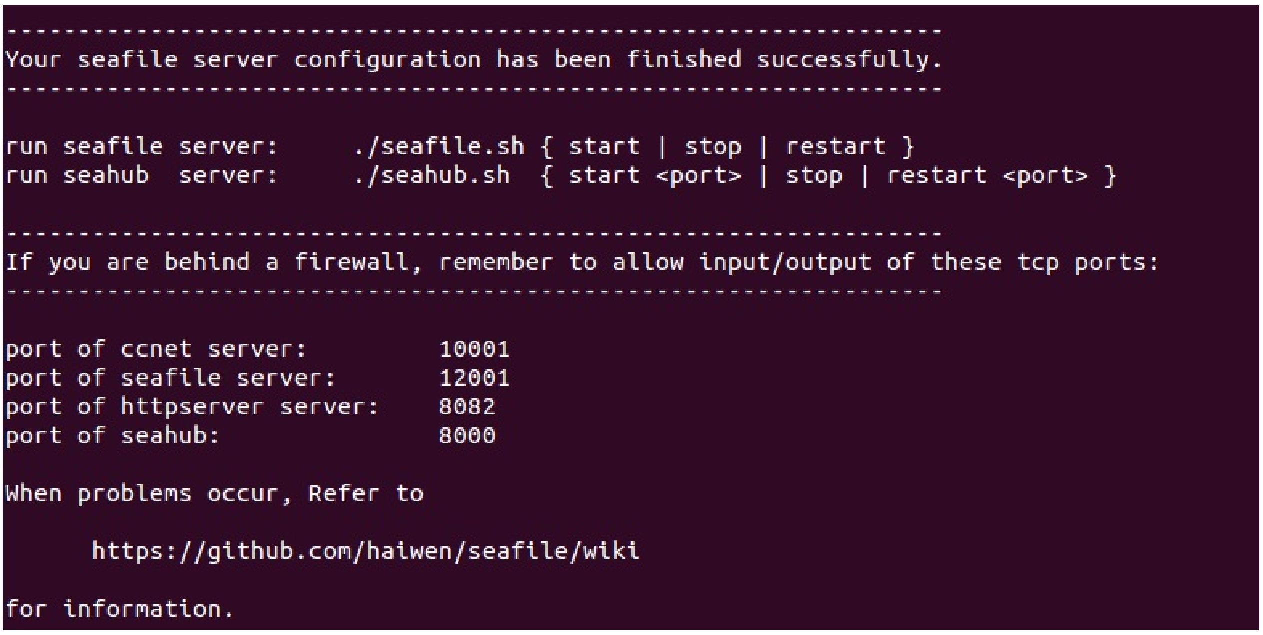
在这里, 你会被要求选择一种创建 Seafile 数据库的方式:

```
-------------------------------------------------------
Please choose a way to initialize seafile databases:
-------------------------------------------------------

[1] Create new ccnet/seafile/seahub databases
[2] Use existing ccnet/seafile/seahub databases
```

- 如果选择 `1` , 你需要提供根密码. 脚本程序会创建数据库和用户。
- 如果选择 `2` , ccnet/seafile/seahub 数据库应该已经被你（或者其他人）提前创建。

如果安装正确完成，你会看到下面这样的输出



```
-----------------------------------------------------------------
Your seafile server configuration has been finished successfully.
-----------------------------------------------------------------

run seafile server:     ./seafile.sh { start | stop | restart }
run seahub  server:     ./seahub.sh  { start <port> | stop | restart <port> }

-----------------------------------------------------------------
If you are behind a firewall, remember to allow input/output of these tcp ports:
-----------------------------------------------------------------

port of ccnet server:         10001
port of seafile server:       12001
port of httpserver server:    8082
port of seahub:               8000

When problems occur, Refer to

      https://github.com/haiwen/seafile/wiki

for information.
```

现在你的目录结构看起来应该是这样:

```
#tree haiwen -L 2
haiwen
├── ccnet               # configuration files
│   ├── ccnet.conf
│   ├── mykey.peer
│   ├── PeerMgr
│   └── seafile.ini
├── installed
│   └── seafile-server_1.8.2_x86-64.tar.gz
├── seafile-data
│   └── seafile.conf
├── seafile-server-1.8.2  # active version
│   ├── reset-admin.sh
│   ├── runtime
│   ├── seafile
│   ├── seafile.sh
│   ├── seahub
│   ├── seahub.sh
│   ├── setup-seafile.sh
│   └── upgrade
├── seafile-server-latest  # symbolic link to seafile-server-1.8.2
├── seahub-data
│   └── avatars
├── seahub_settings.py   # optional config file
└── seahub_settings.pyc
```

`seafile-server-latest` 文件夹为指向当前 Seafile 服务器文件夹的符号链接. 将来你升级到新版本后, 升级脚本会自动更新使其始终指向最新的 Seafile 服务器文件夹.

启动 Seafile 服务器

启动之前

因为 Seafile 在客户端和服务器之间使用持续连接，如果你的客户端数量巨大, 你应该在启动 Seafile 之前修改你的 Linux 文件最大打开数，如下:

```
ulimit -n 30000
```

启动 **Seafile** 服务器和 **Seahub** 网站

在 seafile-server-1.8.2 目录下，运行如下命令

- 启动 Seafile:

```
./seafile.sh start # 启动 Seafile 服务
```

- 启动 Seahub

```
./seahub.sh start <port>  # 启动 Seahub 网站 （默认运行在8000端口上）
```

小贴士**:** 你第一次启动 seahub 时，`seahub.sh` 脚本会提示你创建一个 seafile 管理员帐号。

服务启动后, 打开浏览器并输入以下地址

```
http://192.168.1.111:8000/
```

你会被重定向到登陆页面. 输入你在安装 Seafile 时提供的用户名和密码后，你会进入 Myhome 页面，新建资料库.

恭喜**!** 现在你已经成功的安装了 Seafile 服务器.

在另一端口上运行 **Seahub**

如果你不想在默认的 8000 端口上运行 Seahub, 而是想自定义端口（比如8001）中运行，请按以下步骤操作:

- 关闭 Seafile 服务器

```
./seahub.sh stop # 停止 Seafile 进程
./seafile.sh stop # 停止 Seahub
```

- 更改 `haiwen/ccnet/ccnet.conf` 文件中 `SERVICE_URL` 的值(假设你的 ip 或者域名时 `192.168.1.100`), 如下:

```
SERVICE_URL = http://192.168.1.100:8001
```

- 重启 Seafile 服务器

```
./seafile.sh start # 启动 Seafile 服务
./seahub.sh start 8001 # 启动 Seahub 网站 （运行在8001端口上）
```

关闭/重启 Seafile 和 Seahub

关闭

```
./seahub.sh stop # 停止 Seahub
./seafile.sh stop # 停止 Seafile 进程
```

重启

```
./seafile.sh restart # 停止当前的 Seafile 进程，然后重启 Seafile
./seahub.sh restart  # 停止当前的 Seahub 进程，并在 8000 端口重新启动 Seahub
```

如果停止/重启的脚本运行失败

大多数情况下 seafile.sh seahub.sh 脚本可以正常工作。如果遇到问题：

- 使用**pgrep**命令检查 seafile/seahub 进程是否还在运行中

```
pgrep -f seafile-controller # 查看 Seafile 进程
pgrep -f "manage.py run_gunicorn" # 查看 Seahub 进程
```

- 使用**pkill**命令杀掉相关进程

```
pkill -f seafile-controller # 结束 Seafile 进程
pkill -f "manage.py run_gunicorn" # 结束 Seafile 进程
```

OK!

查看seafile更多信息请访问:

- Nginx 下配置 Seahub / Apache 下配置 Seahub
- Nginx 下启用 Https / Apache 下启用 Https
- Seafile LDAP配置
- 管理员手册

Nginx 下配置 Seahub

准备工作

Ubuntu 下安装 `python-flup` 库:

```
sudo apt-get install python-flup
```

Nginx 环境下部署 Seahub/SeafServer

Seahub 是 Seafile 服务器的网站界面. SeafServer 用来处理浏览器端文件的上传与下载. 默认情况下, 它在 8082 端口上监听 HTTP 请求.

这里我们通过 fastcgi 部署 Seahub, 通过反向代理（Reverse Proxy）部署 SeafServer. 我们假设你已经将 Seahub 绑定了域名"www.myseafile.com".

这是一个 Nginx 配置文件的例子 (你可以创建文件 /etc/nginx/conf.d/seafile.conf， 并拷贝以下内容, 如果你用 gedit 编辑， 别忘了删除 seafile.conf~ 这个临时文件).

```
server {
    listen 80;
    server_name www.myseafile.com;

    proxy_set_header X-Forwarded-For $remote_addr;

    location / {
        fastcgi_pass    127.0.0.1:8000;
        fastcgi_param   SCRIPT_FILENAME     $document_root$fastcgi_script_name;
        fastcgi_param   PATH_INFO           $fastcgi_script_name;

        fastcgi_param    SERVER_PROTOCOL        $server_protocol;
        fastcgi_param   QUERY_STRING        $query_string;
        fastcgi_param   REQUEST_METHOD      $request_method;
        fastcgi_param   CONTENT_TYPE        $content_type;
        fastcgi_param   CONTENT_LENGTH      $content_length;
        fastcgi_param    SERVER_ADDR         $server_addr;
        fastcgi_param    SERVER_PORT         $server_port;
        fastcgi_param    SERVER_NAME         $server_name;
        fastcgi_param   REMOTE_ADDR         $remote_addr;

        access_log      /var/log/nginx/seahub.access.log;
        error_log       /var/log/nginx/seahub.error.log;
    }

    location /seafhttp {
        rewrite ^/seafhttp(.*)$ $1 break;
        proxy_pass http://127.0.0.1:8082;
        client_max_body_size 0;
        proxy_connect_timeout  36000s;
        proxy_read_timeout  36000s;
    }

    location /media {
        root /home/user/haiwen/seafile-server-latest/seahub;
    }
}
```

Nginx 默认设置 "client_max_body_size" 为 1M。如果上传文件大于这个值的话，会报错，相关 HTTP 状态码为 423 ("Request Entity Too Large"). 你可以将值设为 `0` 以禁用此功能.

修改 ccnet.conf 和 seahub_setting.py

修改 ccnet.conf

你需要在 `/data/haiwen/ccnet/ccnet.conf` 的 `SERVICE_URL` 字段中自定义域名。

```
SERVICE_URL = http://www.myseafile.com
```

注意:如果你改变了 Seahub的域名,也需要同步更改 `SERVICE_URL` .

修改 seahub_settings.py

请在 `seahub_settings.py` 新增一行，设定 `FILE_SERVER_ROOT` 的值

```
FILE_SERVER_ROOT = 'http://www.myseafile.com/seafhttp'
```

启动 Seafile 和 Seahub

```
./seafile.sh start
./seahub.sh start-fastcgi
```

Nginx 下启用 Https

在 Seahub 端启用 https

免费 Self-Signed SSL 数字认证用户请看. 如果你是 SSL 付费认证用户可跳过此步.

通过 OpenSS L生成 SSL 数字认证

```
openssl genrsa -out privkey.pem 2048
openssl req -new -x509 -key privkey.pem -out cacert.pem -days 1095
```

修改 Nginx 配置文件

假设你已经按照[[在Nginx环境下部署Seafile]]对 nginx 进行了相关设置. 请修改 nginx 配置文件以使用 HTTPS.

```
server {
  listen       80;
  server_name  www.yourdoamin.com;
  rewrite ^ https://$http_host$request_uri? permanent;    #强制将http重定向到https
}

server {
  listen 443;
  ssl on;
  ssl_certificate /etc/ssl/cacert.pem;    #cacert.pem 文件路径
  ssl_certificate_key /etc/ssl/privkey.pem;    #privkey.pem 文件路径
  server_name www.yourdoamin.com;
  # ......
  fastcgi_param   HTTPS               on;
  fastcgi_param   HTTP_SCHEME         https;
}
```

配置文件示例

这里是配置文件示例:

```
server {
  listen       80;
  server_name  www.yourdoamin.com;
  rewrite ^ https://$http_host$request_uri? permanent;    #强制将http重定向到https
}
server {
  listen 443;
  ssl on;
  ssl_certificate /etc/ssl/cacert.pem;            #cacert.pem 文件路径
  ssl_certificate_key /etc/ssl/privkey.pem;    #privkey.pem 文件路径
```

```
    server_name www.yourdoamin.com;
    proxy_set_header X-Forwarded-For $remote_addr;
    location / {
        fastcgi_pass    127.0.0.1:8000;
        fastcgi_param   SCRIPT_FILENAME     $document_root$fastcgi_script_name;
        fastcgi_param   PATH_INFO           $fastcgi_script_name;

        fastcgi_param   SERVER_PROTOCOL    $server_protocol;
        fastcgi_param   QUERY_STRING        $query_string;
        fastcgi_param   REQUEST_METHOD      $request_method;
        fastcgi_param   CONTENT_TYPE        $content_type;
        fastcgi_param   CONTENT_LENGTH      $content_length;
        fastcgi_param   SERVER_ADDR         $server_addr;
        fastcgi_param   SERVER_PORT         $server_port;
        fastcgi_param   SERVER_NAME         $server_name;
        fastcgi_param   HTTPS               on;
        fastcgi_param   HTTP_SCHEME         https;

        access_log      /var/log/nginx/seahub.access.log;
        error_log       /var/log/nginx/seahub.error.log;
    }
    location /seafhttp {
        rewrite ^/seafhttp(.*)$ $1 break;
        proxy_pass http://127.0.0.1:8082;
        client_max_body_size 0;
        proxy_connect_timeout  36000s;
        proxy_read_timeout  36000s;
    }
    location /media {
        root /home/user/haiwen/seafile-server-latest/seahub;
    }
}
```

重新加载 Nginx

```
nginx -s reload
```

修改相关配置以使用 https

ccnet 配置

因为你想使用 https 而非 http, 你需要修改`ccnet/ccnet.conf`中**SERVICE_URL**字段的值:

```
SERVICE_URL = https://www.yourdomain.com
```

seahub_settings.py 配置

```
FILE_SERVER_ROOT = 'https://www.yourdomain.com/seafhttp'
```

启动 Seafile 和 Seahub

```
./seafile.sh start
./seahub.sh start-fastcgi
```

Apache 下配置 Seahub

准备工作

1. Ubuntu 下安装 `python-flup` 库:

   ```
   sudo apt-get install python-flup
   ```

2. Ubuntu 下安装和启用 mod_fastcgi 和 mod_rewrite :

   ```
   sudo apt-get install libApache2-mod-fastcgi
   sudo a2enmod rewrite
   sudo a2enmod fastcgi
   ```

   CentOS/Redhat 下需要从源码编译安装 mod_fastcgi, 请参考这里。

3. 启用 Apache proxy

   ```
   sudo a2enmod proxy_http
   ```

Apache 环境下部署 Seahub/FileServer

Seahub 是 Seafile 服务器的网站界面. FileServer 用来处理浏览器端文件的上传与下载. 默认情况下, 它在 8082 端口上监听 HTTP 请求.

这里我们通过 fastcgi 部署 Seahub, 通过反向代理（Reverse Proxy）部署 FileServer. 我们假设你已经将 Seahub 绑定了域名"www.myseafile.com".

首先编辑你的 Apache 配置文件.根据你的 Linux 版本, 你需要在文件末尾增加以下语句:

`Apache2.conf` , for ubuntu/debian:

```
FastCGIExternalServer /var/www/seahub.fcgi -host 127.0.0.1:8000
```

`httpd.conf` , for centos/fedora:

```
FastCGIExternalServer /var/www/html/seahub.fcgi -host 127.0.0.1:8000
```

注意, `seahub.fcgi` 只是一个位置标识符, 你并不需要在你的系统中新建这个文件夹.

二, 修改 Apache 配置文件: (`sites-enabled/000-default`) for ubuntu/debian (`vhost.conf`) for centos/fedora

```
<VirtualHost *:80>
  ServerName www.myseafile.com
  DocumentRoot /var/www
  Alias /media  /home/user/haiwen/seafile-server-latest/seahub/media

  RewriteEngine On

  <Location /media>
    Require all granted
  </Location>

  #
  # seafile fileserver
  #
  ProxyPass /seafhttp http://127.0.0.1:8082
  ProxyPassReverse /seafhttp http://127.0.0.1:8082
  RewriteRule ^/seafhttp - [QSA,L]

  #
  # seahub
  #
  RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.*)$ /seahub.fcgi$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
</VirtualHost>
```

修改 ccnet.conf 和 seahub_setting.py

修改 ccnet.conf

你需要在 `/data/haiwen/ccnet/ccnet.conf` 的 `SERVICE_URL` 字段中自定义域名。

```
SERVICE_URL = http://www.myseafile.com
```

注意:如果你改变了 Seahub 的域名,也需要同步更改 `SERVICE_URL` .

修改 seahub_settings.py

请在 `seahub_settings.py` 新增一行，设定 `FILE_SERVER_ROOT` 的值

```
FILE_SERVER_ROOT = 'http://www.myseafile.com/seafhttp'
```

启动 Seafile 和 Seahub

```
sudo service Apache2 restart
./seafile.sh start
./seahub.sh start-fastcgi
```

其他说明

阅读Seafile 组件会帮你更好的理解 Seafile

在 Seafile 服务器端有两个组件：Seahub 和 FileServer。 FileServer 通过监听 8082 端口处理文件的上传与下载. Seahub 通过监听 8000 端口负责其他的WEB页面。在 https 下, Seahub 应该通过 fastcgi 模式监听 8000 端口 (运行./seahub.sh start-fastcgi). 而且在 fastcgi 模式下, 如果直接访问 `http://domain:8000` 时, 会返回错误页面.

当一个用户访问 `https://domain.com/home/my/` 时, Apache 接受到访问请求后，通过 fastcgi 将其转发至 Seahub。 可通过以下配置来实现:

```
#
# seahub
#
RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^/(seahub.*)$ /seahub.fcgi/$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorizatio
```

和

```
FastCGIExternalServer /var/www/seahub.fcgi -host 127.0.0.1:8000
```

当一个用户在 Seahub 中点击文件下载链接时， Seahub 读取 `FILE_SERVER_ROOT` 的值，并将其用户重定向到 `https://domain.com/seafhttp/xxxxx/` 。当 Apache 在 `https://domain.com/seafhttp/xxxxx/` 接收到访问请求后, 它把请求发送到正在监听 127.0.0.1:8082 的 FileServer 组件, 可通过以下配置来实现:

```
ProxyPass /seafhttp http://127.0.0.1:8082
ProxyPassReverse /seafhttp http://127.0.0.1:8082
RewriteRule ^/seafhttp - [QSA,L]
```

Apache 下启用 Https

通过 OpenSSL 生成 SSL 数字认证

免费 Self-Signed SSL 数字认证用户请看. 如果你是 SSL 付费认证用户可跳过此步.

```
openssl genrsa -out privkey.pem 2048
openssl req -new -x509 -key privkey.pem -out cacert.pem -days 1095
```

在 Seahub 端启用 https

假设你已经按照Apache 下配置 Seahub对 Apache 进行了相关设置.请启用 mod_ssl

```
[sudo] a2enmod ssl
```

Windows 下, 你需要在 httpd.conf 中增加 SSL 模块

```
LoadModule ssl_module modules/mod_ssl.so
```

接下来修改你的Apache配置文件，这是示例:

```
<VirtualHost *:443>
  ServerName www.myseafile.com
  DocumentRoot /var/www
  Alias /media  /home/user/haiwen/seafile-server-latest/seahub/media

  SSLEngine On
  SSLCertificateFile /path/to/cacert.pem
  SSLCertificateKeyFile /path/to/privkey.pem

  RewriteEngine On

  #
  # seafile fileserver
  #
  ProxyPass /seafhttp http://127.0.0.1:8082
  ProxyPassReverse /seafhttp http://127.0.0.1:8082
  RewriteRule ^/seafhttp - [QSA,L]

  #
  # seahub
  #
  RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.*)$ /seahub.fcgi/$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
</VirtualHost>
```

修改相关配置以使用 https

ccnet 配置

因为你想使用 https 而非 http, 你需要修改 `ccnet/ccnet.conf` 中 `SERVICE_URL` 字段的值:

```
SERVICE_URL = https://www.myseafile.com
```

seahub_settings.py 配置

```
FILE_SERVER_ROOT = 'https://www.myseafile.com/seafhttp'
```

启动Seafile和Seahub

```
./seafile.sh start
./seahub.sh start-fastcgi
```

其他说明

阅读Seafile 组件会帮你更好的理解 Seafile.

在 Seafile 服务器端的两个组件：Seahub 和 FileServer. FileServer 通过监听 8082 端口处理文件的上传与下载. Seahub 通过监听 8000 端口负责其他的WEB页面. 但是在 https 下, Seahub 应该通过 fastcgi 模式监听8000端口 (运行 `./seahub.sh start-fastcgi`). 而且在 fastcgi 模式下, 如果直接访问 `http://domain:8000` ， 会返回错误页面.

当一个用户访问 `https://domain.com/home/my/` 时, Apache 接受到访问请求后，通过 fastcgi 将其转发至 Seahub. 可通过以下配置来实现:

```
#
# seahub
#
RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^/(seahub.*)$ /seahub.fcgi/$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorizatio
```

and

```
FastCGIExternalServer /var/www/seahub.fcgi -host 127.0.0.1:8000
```

当一个用户在 Seahub 中点击文件下载链接时， Seahub 读取`FILE_SERVER_ROOT`的值，并将其用户重定向到`https://domain.com/seafhttp/xxxxx/`. `https://domain.com/seafhttp`是`FILE_SERVER_ROOT`的值. 这里,`FILE_SERVER`表示是 Seafile 中只负责文件上传与下载的的 FileServer 组件.

当 Apache 在`https://domain.com/seafhttp/xxxxx/`接收到访问请求后,它把请求发送到正在监听`127.0.0.1:8082`的 FileServer 组件,可通过以下配置来实现:

```
ProxyPass /seafhttp http://127.0.0.1:8082
ProxyPassReverse /seafhttp http://127.0.0.1:8082
RewriteRule ^/seafhttp - [QSA,L]
```

Seafile LDAP 和 Active Directory 配置

LDAP (Light-weight Directory Access Protocol) 是企业广泛部署的用户信息管理服务器，微软的活动目录服务器（Active Directory）完全兼容 LDAP。这个文档假定您已经了解了 LDAP 相关的知识和术语。

在目前的 Seahub，只支持 email 格式的用户名登陆，所以，使用 UNIX 和 Windows Domain 用户名并不能登录到 Seahub，后续版本中会对此进行改进.

Seafile 会通过数据库和 LDAP 来搜寻用户. 默认首先搜寻 LDAP. （请注意，在安装时设置的 Seafile 管理员账户，会始终保存在 SQLite/MySQL 数据库中。）

LDAP 用户管理

Seafile 如下管理来自 LDAP 中的用户：

- 当一个 LDAP 用户第一次登录的时候，他会被自动导入到 Seafile 的数据库中（ccnet 数据库的 LDAPUser 表）。
- 系统管理员可以对导入的 LDAP 用户进行管理，包括禁用、删除、设置为系统管理员等。
- 对于企业版，系统只会把已经导入系统的 LDAP 用户计算到当前用户数量中。因此企业版客户可以购买比他们 LDAP/AD 服务器中用户数量少的 license。

在 Seafile 企业版 4.2 及以上的版本，我们加入了从 LDAP/AD 服务器定期同步用户到 Seafile 数据库的功能。这项功能有以下好处：

- 除了定期把用户从 LDAP 中导入到数据库，还能把诸如用户的全名、部门名称等额外信息导入。
- 自动检测用户已经从 LDAP 服务器上删除。从 LDAP 服务器删除的用户会在 Seafile 中禁用。

关于这项功能的更详细信息请参考 这个文档。

LINUX 下连接LDAP/AD

要通过 LDAP 来认证用户，您需要把下面的配置加入 ccnet.conf。需要注意的是下面的配置只是例子，您需要根据自己的实际情况来进行修改。

```
[LDAP]
HOST = ldap://ldap.example.com
BASE = ou=users,dc=example,dc=com
USER_DN = cn=seafileadmin,dc=example,dc=com
PASSWORD = secret
LOGIN_ATTR = mail
```

各个配置选项的含义如下：

- HOST: LDAP 服务器的地址 URL。如果您的 LDAP 服务器监听在非标准端口上，您也可以在 URL 里面包含端口号，如 ldap://ldap.example.com:389。
- BASE: 在 LDAP 服务器的组织架构中，用于查询用户的根节点的唯一名称（Distingushed Name，简

称 DN）。这个节点下面的所有用户都可以访问 Seafile。
- USER_DN: 用于查询 LDAP 服务器中信息的用户的 DN。这个用户应该有足够的权限访问 BASE 以下的所有信息。通常建议使用 LDAP/AD 的管理员用户。
- PASSWORD: USER_DN 对应的用户的密码。
- LOGIN_ATTR: 用作 Seafile 中用户登录 ID 的 LDAP 属性。在 LDAP 里面，每个用户都关联了很多属性。默认我们使用 'mail' 作为登录 ID。

如果您使用的是 AD，以下信息将有助于您配置：

- 要确定您的 BASE 属性，您首先需要打开域管理器的图形界面，并浏览您的组织架构。
    - 如果您想要让系统中所有用户都能够访问 Seafile，您可以用 'cn=users,dc=yourdomain,dc=com' 作为 BASE 选项（需要替换成你们的域名）。
    - 如果您只想要某个部门的人能访问，您可以把范围限定在某个 OU （Organization Unit）中。您可以使用 `dsquery` 命令行工具来查找相应 OU 的 DN。比如，如果 OU 的名字是 'staffs'，您可以运行 `dsquery ou -name staff`。更多的信息可以参考这里。
- AD 支持使用 'user@domain.com' 格式的用户名作为 `USER_DN`。比如您可以使用 administrator@example.com 作为 `USER_DN`。有些时候 AD 不能正确识别这种格式。此时您可以使用 `dsquery` 来查找用户的 DN。比如，假设用户名是 'seafileuser'，运行 `dsquery user -name seafileuser` 来找到该用户的 DN。更多的信息可以参考这里。

针对 AD 的配置例子：

```
[LDAP]
HOST = ldap://192.168.1.123/
BASE = cn=users,dc=example,dc=com
USER_DN = administrator@example.local
PASSWORD = secret
LOGIN_ATTR = mail
```

针对 OpenLDAP 或者其他 LDAP 服务器的配置例子：

```
[LDAP]
HOST = ldap://192.168.1.123/
BASE = ou=users,dc=example,dc=com
USER_DN = cn=admin,dc=example,dc=com
PASSWORD = secret
LOGIN_ATTR = mail
```

如果您使用 AD 但是没有给用户设置邮件地址（mail 属性），您也可以使用以下的配置：

```
[LDAP]
HOST = ldap://192.168.1.123/
BASE = cn=users,dc=example,dc=com
USER_DN = administrator@example.local
PASSWORD = secret
LOGIN_ATTR = userPrincipalName
```

`userPrincipalName` 是一个 AD 支持的特殊属性。它具有 `username@domain-name` 的格式，其中

`username` 是 Windows 用户的登录名。使用上述配置之后，用户可以用 `username@domain-name` 作为用户名登录 Seafile。注意这个登录名并不是真实的邮件地址，因此 Seafile 的邮件通知功能可能不能工作。

注意：

1. 如果配置项包含中文，需要确保配置文件使用 UTF8 编码保存。
2. 这个文档中描述的配置方法经过很多人在不同的 LDAP/AD 环境下试过，肯定是能工作的。

通过 Windows 服务器连接到 LDAP/AD

Windows 下的配置语法与 Linux 下的有些不同. 你不需要增加 **`ldap:// prefix to the HOST field`.**

要通过 LDAP 来认证用户，您需要把下面的配置加入 ccnet.conf。需要注意的是下面的配置只是例子，您需要根据自己的实际情况来进行修改。

```
[LDAP]
HOST = ldap.example.com
BASE = ou=users,dc=example,dc=com
USER_DN = cn=seafileadmin,dc=example,dc=com
PASSWORD = secret
LOGIN_ATTR = mail
```

各个配置选项的含义如下：

- HOST: LDAP 服务器的地址 URL。如果您的 LDAP 服务器监听在非标准端口上，您也可以在 URL 里面包含端口号，如 ldap://ldap.example.com:389。
- BASE: 在 LDAP 服务器的组织架构中，用于查询用户的根节点的唯一名称（Distingushed Name，简称 DN）。这个节点下面的所有用户都可以访问 Seafile。
- USER_DN: 用于查询 LDAP 服务器中信息的用户的 DN。这个用户应该有足够的权限访问 BASE 以下的所有信息。通常建议使用 LDAP/AD 的管理员用户。
- PASSWORD: USER_DN 对应的用户的密码。
- LOGIN_ATTR: 用作 Seafile 中用户登录 ID 的 LDAP 属性。在 LDAP 里面，每个用户都关联了很多属性。默认我们使用 'mail' 作为登录 ID。

如果您使用的是 AD，以下信息将有助于您配置：

- 要确定您的 BASE 属性，您首先需要打开域管理器的图形界面，并浏览您的组织架构。
  - 如果您想要让系统中所有用户都能够访问 Seafile，您可以用 'cn=users,dc=yourdomain,dc=com' 作为 BASE 选项（需要替换成你们的域名）。
  - 如果您只想要某个部门的人能访问，您可以把范围限定在某个 OU （Organization Unit）中。您可以使用 `dsquery` 命令行工具来查找相应 OU 的 DN。比如，如果 OU 的名字是 'staffs'，您可以运行 `dsquery ou -name staff`。更多的信息可以参考这里。
- AD 支持使用 'user@domain.com' 格式的用户名作为 `USER_DN`。比如您可以使用 administrator@example.com 作为 `USER_DN`。有些时候 AD 不能正确识别这种格式。此时您可以使用 `dsquery` 来查找用户的 DN。比如，假设用户名是 'seafileuser'，运行 `dsquery user -name seafileuser` 来找到该用户的 DN。更多的信息可以参考这里。

针对 AD 的配置例子：

```
[LDAP]
HOST = 192.168.1.123
BASE = cn=users,dc=example,dc=com
USER_DN = administrator@example.local
PASSWORD = secret
LOGIN_ATTR = mail
```

针对 OpenLDAP 或者其他 LDAP 服务器的配置例子：

```
[LDAP]
HOST = 192.168.1.123
BASE = ou=users,dc=example,dc=com
USER_DN = cn=admin,dc=example,dc=com
PASSWORD = secret
LOGIN_ATTR = mail
```

如果您使用 AD 但是没有给用户设置邮件地址（mail 属性），您也可以使用以下的配置：

```
[LDAP]
HOST = 192.168.1.123
BASE = cn=users,dc=example,dc=com
USER_DN = administrator@example.local
PASSWORD = secret
LOGIN_ATTR = userPrincipalName
```

`userPrincipalName` 是一个 AD 支持的特殊属性。它具有 `username@domain-name` 的格式，其中 `username` 是 Windows 用户的登录名。使用上述配置之后，用户可以用 `username@domain-name` 作为用户名登录 Seafile。注意这个登录名并不是真实的邮件地址，因此 Seafile 的邮件通知功能可能不能工作。

使用多个 BASE DN 以及用户过滤选项

当您想把公司中多个 OU 加入 Seafile 中时，您可以使用在配置中指定多个 BASE DN。您可以在"BASE"配置中指定一个 DN 的列表，标识名由";"分开, 比如：`cn=developers,dc=example,dc=com;cn=marketing,dc=example,dc=com`

当你的公司组织庞大，但是只有一小部分人使用 Seafile 的时候，搜索过滤器（Search filter）会很有用处. 过滤器可以通过修改"FILTER"配置来实现，例如，在 LDAP 配置中增加以下语句:

```
FILTER = memberOf=CN=group,CN=developers,DC=example,DC=com
```

请注意上面的示例只是象征性的简介. `memberOf` 只有在活动目录(Active Directory)中才适用.

这里是另一个示例:

```
FILTER = &(!(UserAccountControl:1.2.840.113556.1.4.803:=2))
```

把 Seafile 用户限定在 AD 的一个组中

您可以利用用户过滤器选项来只允许 AD 某个组中的用户使用 Seafile。

1. 首先，您需要找到这个组的 DN。我们再次使用 `dsquery` 命令。比如，如果组的名字是 'seafilegroup'，那么您可以运行 `dsquery group -name seafilegroup`。
2. 然后您可以把一下配置加入 ccnet.conf 的 LDAP 配置中：

```
FILTER = memberOf={dsquery 命令的输出}
```

企业版提供的高级 LDAP 功能

使用结果分页扩展（paged results extension）

LDAP 协议 v3 支持一个称为 "paged results" 的扩展功能。当您在 LDAP 中有大量用户的时候，这个选项能够大大提高列出用户的速度。而且，AD 限制了单次请求中返回的用户条目数量，您需要启用这个选项才能避免查询错误。

在 Seafile 企业版中，在 LDAP 配置中加入以下设置：

```
USE_PAGED_RESULT = true
```

从 LDAP 中导入群组

请参考这个文档。

开机启动 Seafile

Ubuntu 系统

使用 /etc/init.d/ 来配置 Seafile/Seahub 开机启动.

创建/etc/init.d/seafile-server脚本

```
sudo vim /etc/init.d/seafile-server
```

脚本内容为: (同时需要修改相应的 `user` 和 `script\_path` 字段的值)

```
#!/bin/sh

# 请将 user 改为你的Linux用户名
user=haiwen

# 请将 script_dir 改为你的 Seafile 文件安装路径
seafile_dir=/data/haiwen
script_path=${seafile_dir}/seafile-server-latest
seafile_init_log=${seafile_dir}/logs/seafile.init.log
seahub_init_log=${seafile_dir}/logs/seahub.init.log

# 若使用 fastcgi, 请将其设置为true
fastcgi=false
# fastcgi 端口, 默认为 8000.
fastcgi_port=8000

case "$1" in
        start)
                sudo -u ${user} ${script_path}/seafile.sh start >> ${seafile_init_log}
                if [  $fastcgi = true ];
                then
                        sudo -u ${user} ${script_path}/seahub.sh start-fastcgi ${fastcgi_
                else
                        sudo -u ${user} ${script_path}/seahub.sh start >> ${seahub_init_l
                fi
        ;;
        restart)
                sudo -u ${user} ${script_path}/seafile.sh restart >> ${seafile_init_log}
                if [  $fastcgi = true ];
                then
                        sudo -u ${user} ${script_path}/seahub.sh restart-fastcgi ${fastcg
                else
                        sudo -u ${user} ${script_path}/seahub.sh restart >> ${seahub_init
                fi
        ;;
        stop)
                sudo -u ${user} ${script_path}/seafile.sh $1 >> ${seafile_init_log}
                sudo -u ${user} ${script_path}/seahub.sh $1 >> ${seahub_init_log}
        ;;
        *)
                echo "Usage: /etc/init.d/seafile {start|stop|restart}"
                exit 1
        ;;
esac
```

注意: 如果你想在 fastcgi 下运行 Seahub,请设置`fastcgi`变量为`true`

为日志文件创建目录

```
 mkdir /path/to/seafile/dir/logs
```

创建/etc/init/seafile-server.conf文件

非使用 MySQL

```
start on (runlevel [2345])
stop on (runlevel [016])

pre-start script
/etc/init.d/seafile-server start
end script

post-stop script
/etc/init.d/seafile-server stop
end script
```

使用 MySQL

```
start on (started mysql
and runlevel [2345])
stop on (runlevel [016])

pre-start script
/etc/init.d/seafile-server start
end script

post-stop script
/etc/init.d/seafile-server stop
end script
```

设置 seafile-sever 脚本为可执行文件

```
sudo chmod +x /etc/init.d/seafile-server
```

完成

其他 Debian 系的 Linux 下

创建脚本/etc/init.d/seafile-server

```
sudo vim /etc/init.d/seafile-server
```

脚本内容为: (同时需要修改相应的`user`和`script\_path`字段的值)

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:          seafile-server
# Required-Start:    $local_fs $remote_fs $network
# Required-Stop:     $local_fs
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Starts Seafile Server
# Description:       starts Seafile Server
### END INIT INFO

# 请将 user 改为你的Linux用户名
user=haiwen

# 请将 script_path 改为你的 Seafile 文件安装路径
seafile_dir=/data/haiwen
script_path=${seafile_dir}/seafile-server-latest
seafile_init_log=${seafile_dir}/logs/seafile.init.log
seahub_init_log=${seafile_dir}/logs/seahub.init.log

# 若使用 fastcgi, 请将其设置为true
fastcgi=false
# fastcgi 端口, 默认为 8000.
fastcgi_port=8000

case "$1" in
        start)
                sudo -u ${user} ${script_path}/seafile.sh start >> ${seafile_init_log}
                if [  $fastcgi = true ];
                then
                        sudo -u ${user} ${script_path}/seahub.sh start-fastcgi ${fastcgi_
                else
                        sudo -u ${user} ${script_path}/seahub.sh start >> ${seahub_init_l
                fi
        ;;
        restart)
                sudo -u ${user} ${script_path}/seafile.sh restart >> ${seafile_init_log}
                if [  $fastcgi = true ];
                then
                        sudo -u ${user} ${script_path}/seahub.sh restart-fastcgi ${fastcg
                else
                        sudo -u ${user} ${script_path}/seahub.sh restart >> ${seahub_init
                fi
        ;;
        stop)
```

```
                sudo -u ${user} ${script_path}/seafile.sh $1 >> ${seafile_init_log}
                sudo -u ${user} ${script_path}/seahub.sh $1 >> ${seahub_init_log}
        ;;
        *)
                echo "Usage: /etc/init.d/seafile {start|stop|restart}"
                exit 1
        ;;
esac
```

注意: 如果你想在 fastcgi 下运行 Seahub,请设置 `fastcgi` 变量为 `true`

为日志文件创建目录

```
mkdir /path/to/seafile/dir/logs
```

设置 **seafile-sever** 脚本为可执行文件

```
sudo chmod +x /etc/init.d/seafile-server
```

在 **rc.d** 中新增 **seafile-server**

```
sudo update-rc.d seafile-server defaults
```

完成

RHEL/CentOS 系统统方法 1

RHEL/CentOS 下,/etc/rc.local 脚本会随系统开机自动执行,所以我们在这个脚本中设置启动 Seafile/Seahub.

- 定位 python(python 2.6 or 2.7)

```
which python2.6 # or "which python2.7"
```

- 在 /etc/rc.local 脚本中, 将 python2.6(2.7)路径加入到**PATH**字段中, 并增加 Seafile/Seahub 启动命令

```
`
# 假设 python 2.6(2.7) 可执行文件在 /usr/local/bin 目录下
PATH=$PATH:/usr/local/bin/

# 请将 user 改为你的Linux用户名
user=haiwen
```

```
# 请将 script_path 改为你的 Seafile 文件安装路径
seafile_dir=/data/haiwen
script_path=${seafile_dir}/seafile-server-latest

sudo -u ${user} ${script_path}/seafile.sh start > /tmp/seafile.init.log 2>&1
sudo -u ${user} ${script_path}/seahub.sh start > /tmp/seahub.init.log 2>&1
```

注意: 如果你想在fastcgi下启动Seahub,只需将上文中最后一行**"seahub.sh start"**改为**"seahub.sh start-fastcgi"**

RHEL/CentOS 系统方法 2

RHEL/CentOS 下 , 我们通过 /etc/init.d/ 脚本将 Seafile/Seahub作为服务程序随开机启动.

创建**/etc/sysconfig/seafile**文件

```
# 请将 user 改为你的Linux用户名
user=haiwen

# 请将 script_path 改为你的 Seafile 文件安装路径
seafile_dir=/home/haiwen
script_path=${seafile_dir}/seafile-server-latest
seafile_init_log=${seafile_dir}/logs/seafile.init.log
seahub_init_log=${seafile_dir}/logs/seahub.init.log

# 若使用 fastcgi, 请将其设置true
fastcgi=false

# fastcgi 端口, 默认为 8000.
fastcgi_port=8000
```

创建**/etc/init.d/seafile**文件

```
#!/bin/bash
#
# seafile

#
# chkconfig: - 68 32
# description: seafile

# Source function library.
. /etc/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

if [ -f /etc/sysconfig/seafile ];then
        . /etc/sysconfig/seafile
        else
            echo "Config file /etc/sysconfig/seafile not found! Bye."
            exit 200
        fi

RETVAL=0

start() {
        # Start daemons.
        echo -n $"Starting seafile: "
        ulimit -n 30000
        su - ${user} -c"${script_path}/seafile.sh start >> ${seafile_init_log} 2>&1"
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/seafile
        return $RETVAL
}

stop() {
        echo -n $"Shutting down seafile: "
        su - ${user} -c"${script_path}/seafile.sh stop >> ${seafile_init_log} 2>&1"
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/seafile
        return $RETVAL
}

# See how we were called.
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart|reload)
        stop
        start
        RETVAL=$?
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart}"
        RETVAL=3
esac

exit $RETVAL
```

创建**/etc/init.d/seahub**脚本

```
#!/bin/bash
#
# seahub

#
# chkconfig: - 69 31
# description: seahub

# Source function library.
. /etc/init.d/functions
```

```
# Source networking configuration.
. /etc/sysconfig/network

if [ -f /etc/sysconfig/seafile ];then
        . /etc/sysconfig/seafile
        else
            echo "Config file /etc/sysconfig/seafile not found! Bye."
            exit 200
        fi

RETVAL=0

start() {
        # Start daemons.
        echo -n $"Starting seahub: "
        ulimit -n 30000
        if [  $fastcgi = true ];
                then
                su - ${user} -c"${script_path}/seahub.sh start-fastcgi ${fastcgi_port} >>
                else
                su - ${user} -c"${script_path}/seahub.sh start >> ${seahub_init_log} 2>&1
                fi
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/seahub
        return $RETVAL
}

stop() {
        echo -n $"Shutting down seafile: "
        su - ${user} -c"${script_path}/seahub.sh stop >> ${seahub_init_log} 2>&1"
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/seahub
        return $RETVAL
}

# See how we were called.
case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart|reload)
        stop
        start
        RETVAL=$?
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart}"
        RETVAL=3
esac

exit $RETVAL
```

接下来启动服务程序:

```
chmod 550 /etc/init.d/seafile
chmod 550 /etc/init.d/seahub
chkconfig --add seafile
chkconfig --add seahub
chkconfig seahub on
chkconfig seafile on
```

执行:

```
service seafile start
service seahub start
```

完成

默认情况下，Seafile 开启了以下两个端口：

Seahub	8000
FileServer	8082

如果你在 Nginx/Apache 下运行 Seafile，并且使用了 HTTPS, 开启 443 端口即可。

在服务器端设置 logrotate

工作原理

自 3.1 版本以后，seaf-server 和 ccnet-server 支持通过接收 SIGUR1 信号来管理日志文件。

这个功能在你需要剪切日志文件但是不想关闭服务器的时候非常有用。

> 注意: 此功能在 Windows 下并不适用

logrotate 默认配置

对于 Debian, logrotate 默认存储在 `/etc/logrotate.d/`

配置示例

假设你的 ccnet-server 的日志文件是 `/home/haiwen/logs/ccnet.log`, ccnet-server 进程的 pidfile 是 `/home/haiwen/pids/ccnet.pid`. seaf-server's 的日志文件是 `/home/haiwen/logs/seaf-server.log`, seaf-server 进程的 pidfile 是 `/home/haiwen/pids/seaf-server.pid`:

则请按如下配置 logroate:

```
/home/haiwen/logs/seaf-server.log
{
        daily
        missingok
        rotate 52
        compress
        delaycompress
        notifempty
        sharedscripts
        postrotate
                [ ! -f /home/haiwen/pids/seaf-server.pid ] || kill -USR1 `cat /home/haiwe
        endscript
}

/home/haiwen/logs/ccnet.log
{
        daily
        missingok
        rotate 52
        compress
        delaycompress
        notifempty
        sharedscripts
        postrotate
                [ ! -f /home/haiwen/pids/ccnet.pid ] || kill -USR1 `cat /home/haiwen/pids
        endscript
}
```

对于 Debian 用户, 可以将以上配置文件存储在 `/etc/logrotate.d/seafile`

使用 memcached

安装 Memcached 能够显著提高系统性能，首先安装 memcached 和相应的库：

- memcached
- python memcached module (python-memcache or python-memcached)

然后在 **seahub_settings.py** 中加入以下配置信息.

```
CACHES = {
    'default': {
        'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
        'LOCATION': '127.0.0.1:11211',
    }
}
```

最后重启 Seahub 以使更改生效：

```
./seahub.sh restart
```

如果更改没有生效，请删除`seahub_setting.pyc`缓存文件.

防火墙 / NAT 设置

通过广域网(WAN)访问部署在局域网(LAN)的 Seafile 服务器,需要:

- 一台支持端口转发的路由器
- 使用动态域名解析服务
- 配置 Seafile 服务器

安装 Seafile 服务器

首先,按照部署 Seafile 服务器（使用 SQLite）安装 Seafile 服务器。

并确保以下功能正常运行:

- 正常访问 Seahub 网站
- 在 Seafile 客户端可以下载/同步一个资料库

在路由器中设置端口转发

确保路由器支持端口转发功能

首先, 确保你的路由器支持端口转发功能：

- 根据路由器管理手册操作说明(或网络搜索), 进入路由器的管理用户界面。

- 找到包含 "转发" 或者 "高级" 等关键词的页面, 说明此路由器支持端口转发功能。

设置路由转发规则

Seafile 服务器包含很多组件， 请根据以下规则为所有组件设置端口转发。

组件	默认端口
ccnet	10001
seaf-server	12001
fileserver	8082
seahub	8000

- 如果是在 Apache/Nginx 环境下部署的 Seafile, 则不需要打开 8000 和 8082 端口。
- 以上是默认端口设置，具体配置可自行更改.

端口转发测试

设置端口转发后，可按以下步骤测试是否成功:

- 打开一个命令行终端

- 访问 `http://who.is` 得到本机的IP
- 通过以下命令连接 Seahub

  ```
  telnet  8000
  ```

如果端口转发配置成功，命令行会提示连接成功。否则, 会显示 *connection refused* 或者 *connection timeout*， 提示连接不成功。

若未成功，原因可能如下:

- 端口转发配置错误
- 需要重启路由器
- 网络不可用

设置 SERVICE_URL

`ccnet.conf` 中的 "SERVICE_URL" 字段，是用来在在线访问文件时，生成上传/下载链接的，更改此字段的值为你的IP。

```
SERVICE_URL = http://<Your WAN IP>:8000
```

大部分路由器都支持 NAT loopback. 当你通过内网访问 Seafile 时, 即时你的外部 IP 被占用，文件上传/下载仍然会工作。

使用域名解析服务

为什么使用动态域名解析服务?

完成以上端口转发配置工作后，就可以通过外网 IP 访问部署在局域网内的 Seafile 服务器了。但是对于大多数人来说， 外网 IP 会被 ISP (互联网服务提供商)定期更改, 这就使得，需要不断的进行重新配置.

可以使用动态域名解析服务来解决这个问题。通过使用域名解析服务，你可以通过域名（而不是 IP）来访问 Seahub，即使 IP 会不断变化，但是域名始终会指向当前 IP。

互联网上提供域名解析服务的有很多，我们推荐 www.noip.com。

怎样使用域名解析服务，不在本手册说明范围之内，但是基本上，你需要遵循以下步骤:

1. 选择一个域名解析服务提供商。
2. 注册成为此服务商的一个用户。

更改 Seafile 配置

当你配置好域名解析服务之后，需要对 `ccnet.conf` 进行更改:

```
SERVICE_URL = http://<你的域名>:8000
```

然后重新 Seafile 服务.

网络设置

默认情况下，你需要打开以下四个端口.

```
|
| Seahub
   | 8000
   |-
   | FileServer
   | 8082
   |-
   | Ccnet Daemon
   | 10001
   |-
   | Seafile Daemon
   | 12001
   |
```

如果你的 Seafile 服务器是运行在 Nginx/Apache 环境下，并且开启了 HTTPS, 则需要开启以下端口：

```
|
| HTTPS
   | 443
   |-
   | Ccnet Daemon
   | 10001
   |-
   | Seafile Daemon
   | 12001
   |
```

在非根域名下部署 Seahub

这份文档将说明如何在网站的非根文件夹下通过 Apache/Nginx 部署 Seafile。

注意: 请先阅读 Nginx 下配置 Seahub 或者 Apache 下配置 Seahub.

Nginx 下部署

首先更改 `seahub_settings.py` 中一些变量的值:

```
SERVE_STATIC = False
MEDIA_URL = '/seafmedia/'
SITE_ROOT = '/seafile/'
COMPRESS_URL = MEDIA_URL
STATIC_URL = MEDIA_URL + 'assets/'
```

我们将使用 Nginx 来管理静态文件(js, css, etc), 所以将 `SERVE_STATIC` 设置为 `False` 。

可以自定义 `MEDIA_URL` 的值，但是确保结尾包含斜线。

因为要在 `/seafile/` 目录下而不是根目录下部署 Seafile, 所以设置 `SITE_ROOT` 的值为 `/seafile/` 。

接下来，配置 Nginx 如下:

```
server {
    listen 80;
    server_name www.example.com;
    proxy_set_header X-Forwarded-For $remote_addr;
    location /seafile {
        fastcgi_pass    127.0.0.1:8000;
        fastcgi_param   SCRIPT_FILENAME     $document_root$fastcgi_script_name;
        fastcgi_param   PATH_INFO           $fastcgi_script_name;

        fastcgi_param    SERVER_PROTOCOL        $server_protocol;
        fastcgi_param   QUERY_STRING        $query_string;
        fastcgi_param   REQUEST_METHOD      $request_method;
        fastcgi_param   CONTENT_TYPE        $content_type;
        fastcgi_param   CONTENT_LENGTH      $content_length;
        fastcgi_param    SERVER_ADDR         $server_addr;
        fastcgi_param    SERVER_PORT         $server_port;
        fastcgi_param    SERVER_NAME         $server_name;
#       fastcgi_param   HTTPS               on; # 如果使用 https，请取消掉这行的注释。
        access_log      /var/log/nginx/seahub.access.log;
        error_log       /var/log/nginx/seahub.error.log;
    }

    location /seafhttp {
        rewrite ^/seafhttp(.*)$ $1 break;
        proxy_pass http://127.0.0.1:8082;
        client_max_body_size 0;
    }
    location /seafmedia {
        rewrite ^/seafmedia(.*)$ /media$1 break;
        root /home/user/haiwen/seafile-server-latest/seahub;
    }
}
```

接下来设置 `SERVICE_URL` 和 `FILE_SERVER_ROOT` 的值。

Apache 下部署

首先更改 `seahub_settings.py` 中一些变量的值:

```
SERVE_STATIC = False
MEDIA_URL = '/seafmedia/'
SITE_ROOT = '/seafile/'
```

在 `httpd.conf` 文件中加入以下语句:

```
  FastCGIExternalServer /var/www/seahub.fcgi -host 127.0.0.1:8000
```

接下来配置 Apache，示例如下:

```

  ServerName www.example.com
  DocumentRoot /var/www
  Alias /seafmedia  /home/user/haiwen/seafile-server-2.0.2/seahub/media

  RewriteEngine On

  #
  # seafile fileserver
  #
  ProxyPass /seafhttp http://127.0.0.1:8082
  ProxyPassReverse /seafhttp http://127.0.0.1:8082
  RewriteRule ^/seafhttp - [QSA,L]

  #
  # seahub
  #
  RewriteRule ^/(seafmedia.*)$ /$1 [QSA,L,PT]
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^/(seafile.*)$ /seahub.fcgi/$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authoriza
```

请注意更改 Alias 的值为自己的文件路径。

更改 ccnet.conf 和 seahub_setting.py

更改 ccnet.conf

为使 Seafile 知道你所使用的域名，请更改 `/data/haiwen/ccnet/ccnet.conf` 中 `SERVICE_URL` 变量的值。

```
SERVICE_URL = http://www.myseafile.com/seafile
```

注意: 如果以后域名有所变动，请记得更改 `SERVICE_URL` .

更改 seahub_settings.py

更改 `seahub_settings.py` 中 `FILE_SERVER_ROOT` 的值

```
FILE_SERVER_ROOT = 'http://www.myseafile.com/seafhttp'
```

启动 Seafile 和 Seahub

```
./seafile.sh start
./seahub.sh start-fastcgi
```

从 SQLite 迁移到 MySQL

首先请确认 MySQL 的 Python 模块已经安装. Ubuntu 下，安装命令为 `apt-get install python-mysqldb` .

请按以下步骤操作:

1. 停止 Seafile 和 Seahub
2. 下载 sqlite2mysql.sh 和 sqlite2mysql.py 到 Seafile 的安装根目录（ `/data/haiwen` ）里.
3. 运行 sqlite2mysql.sh 脚本

   ```
   chmod +x sqlite2mysql.sh
   ./sqlite2mysql.sh
   ```

   这个脚本将生成三个文件： `ccnet-db.sql` , `seafile-db.sql` , `seahub-db.sql` 。
4. 新建3个数据库，分别命名为 `ccnet-db` , `seafile-db` , `seahub-db` .

   ```
   create database `ccnet-db` character set = 'utf8';
   create database `seafile-db` character set = 'utf8';
   create database `seahub-db` character set = 'utf8';
   ```
5. 修改 /etc/my.conf, 添加下列语句，并重启 mysql (sudo service mysql restart)，这个语句主要是确保数据库使用 UTF8 编码

   ```
    [mysqld]
    collation-server = utf8_unicode_ci
    init-connect='SET NAMES utf8'
    character-set-server = utf8
   ```
6. 运行 sql 文件:

   ```
   mysql> use `ccnet-db`
   mysql> source ccnet-db.sql
   mysql> use `seafile-db`
   mysql> source seafile-db.sql
   mysql> use `seahub-db`
   mysql> source seahub-db.sql
   ```
7. 更改配置

   在 `ccnet/ccnet.conf` 中增加以下语句:

```
[Database]
ENGINE=mysql
HOST=127.0.0.1
USER=root
PASSWD=root
DB=ccnet-db
CONNECTION_CHARSET=utf8
```

注意: 使用 `127.0.0.1`, 不要使用 `localhost`.

将 `seafile-data/seafile.conf` 中的数据库配置信息更改文以下语句:

```
[database]
type=mysql
host=127.0.0.1
user=root
password=root
db_name=seafile-db
CONNECTION_CHARSET=utf8
```

在 `seahub_settings.py` 中增加以下语句:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'USER' : 'root',
        'PASSWORD' : 'root',
        'NAME' : 'seahub-db',
        'HOST' : '127.0.0.1',
        'OPTIONS': {
            "init_command": "SET storage_engine=INNODB",
        }
    }
}
```

8. 重启 Seafile and Seahub

安装常见问题

无法上传/下载

- 检查 ccnet.conf 中 `SERVICE_URL` 的配置，检查 seahub_settings.py 中 `FILE_SERVER_ROOT` 的配置
- 确认防火墙没有禁用 seafile fileserver
- 使用 chrome/firefox 调试模式,找到点击下载按钮时使用的链接并查看错误信息。

Apache 日志文件报错**: "File does not exist: /var/www/seahub.fcgi"**

请确定在 httpd.conf 或者 apache2.conf 中使用了 `FastCGIExternalServer /var/www/seahub.fcgi -host 127.0.0.1:8000` 尤其是 `/var/www/seahub.fcgi` 部分.

Apache/HTTPS 下，Seafile 只显示文本文件(没有 CSS 样式和图片显示)

多媒体文件访问权限错误 (Alias location identified in /etc/apache2/sites-enabled/000-default (Ubuntu)

解决方法:

1. 切换到非根（non-root）用户重新运行安装脚本
2. 复制 `/media` 文件夹到 `var/www/` 下，并在 `/etc/apache2/sites-enabled/000-default` 中重新编辑文件路径。

初始化 Seafile 时，显示 "Error when calling the metaclass bases" 错误

Seafile 使用 Django 1.5, 所需 Python 版本为 2.6.5+ 。

Seafile

升级指南

使用预编译 Seafile 服务器安装包的用户请看.

- 如果你是 源码编译安装 Seafile 的，请参考那篇文档中的 升级 部分。
- 升级之后, 如果没有正常运行，请清空 Seahub 缓存。

主版本升级 (比如从 2.x 升级到 3.y)

假设你现在使用 2.1.0 版本，想要升级到 3.1.0 版本，下载、解压新版本安装包后，得到目录结构如下：

```
haiwen
   -- seafile-server-2.1.0
   -- seafile-server-3.1.0
   -- ccnet
   -- seafile-data
```

升级到 3.1.0：

1. 关闭 Seafile 服务（如果正在运行）：

   ```
   cd haiwen/seafile-server-2.1.0
   ./seahub.sh stop
   ./seafile.sh stop
   ```

2. 查看 *seafile-server-3.1.0* 目录下的升级脚本：

   ```
   cd haiwen/seafile-server-3.1.0
   ls upgrade/upgrade_*
   ```

   可以看到升级脚本文件如下:

   ```
   ...
   upgrade/upgrade_2.0_2.1.sh
   upgrade/upgrade_2.1_2.2.sh
   upgrade/upgrade_2.2_3.0.sh
   upgrade/upgrade_3.0_3.1.sh
   ```

3. 从当前版本（2.1.0）开始，按顺序运行以下脚本：

   ```
   upgrade/upgrade_2.1_2.2.sh
   upgrade/upgrade_2.2_3.0.sh
   ```

```
upgrade/upgrade_3.0_3.1.sh
```

4. 启动新版本 Seafile 服务器，完成升级：

```
cd haiwen/seafile-server-3.1.0/
./seafile.sh start
./seahub.sh start
```

小版本升级 (比如从 3.0.x 升级到 3.2.y)

假设你现在使用 3.0.0 版本，想要升级到 3.2.2 版本，下载、解压新版本安装包，得到目录结构如下：

```
haiwen
   -- seafile-server-3.0.0
   -- seafile-server-3.2.2
   -- ccnet
   -- seafile-data
```

升级到 3.2.2：

1. 关闭 Seafile 服务（如果正在运行）：

```
cd haiwen/seafile-server-3.0.0
./seahub.sh stop
./seafile.sh stop
```

2. 查看 *seafile-server-3.2.2* 目录下的升级脚本：

```
cd haiwen/seafile-server-3.2.2
ls upgrade/upgrade_*
```

可以看到升级脚本文件如下:

```
...
upgrade/upgrade_2.2_3.0.sh
upgrade/upgrade_3.0_3.1.sh
upgrade/upgrade_3.1_3.2.sh
```

3. 从当前版本（3.0.0）开始，按顺序运行以下脚本：

```
upgrade/upgrade_3.0_3.1.sh
upgrade/upgrade_3.1_3.2.sh
```

4. 启动新版本 Seafile 服务器，完成升级：

```
cd haiwen/seafile-server-3.2.2/
./seafile.sh start
./seahub.sh start
```

维护版本升级 (比如从 3.1.0 升级到 3.1.2)

类似从 3.1.0 升级到 3.1.2，为维护版本升级。

1. 关闭 Seafile 服务（如果正在运行）；
2. 对于此类升级，只需更新头像链接，直接运行升级脚本即可(因为历史原因，此升级脚本命名为 `minor-upgrade.sh`):

```
cd seafile-server-3.1.2
upgrade/minor-upgrade.sh
```

3. 运行升级脚本之后，启动新版本 Seafile 服务器，完成升级；

4. 如果新版本运行正常，可以删除旧版本 Seafile 文件。

```
rm -rf seafile-server-3.1.0
```

安装与升级

我们测试用的系统是 Windows 2008 server R2 SP1。

- 下载安装 Windows 版 Seafile 服务器
- 安装 Seafile 为 Windows 服务
- 所用端口说明
- 升级

注意：默认情况下，Seafile 需要用到 8000, 8082, 10001, 12001 四个端口。

服务器管理

- 垃圾回收不再需要的数据块

常见问题

如果您安装 Seafile 服务器失败， 请首先查看 `seafserv-applet.log` 文件。

安装完后，本地网页无法打开

确保您使用的是 Python 2.7.4 32位版本。

"ERROR: D:/seafile-server\seahub.db not found"

此文件是在 Seafile 初始化过程中创建的。请执行下面两步:

1. 检查您的 Python 以及 Python 环境变量是否设置正确。
2. 将您的 Seafile 服务器包放在一个简短的路径下， 比如 `C:\seafile-packages` 。

创建 `seahub.db` 文件失败

请使用 Python 2.7.4 32 位版本， 不要使用 Python 3.0 及以上版本。

不能通过 Web 端上传或下载文件

请先确保您已经更改了 `ccnet.conf` 配置文件中的 `SERVICE_URL` ， 且更改正确。

浏览器不能获得 css 和 javascript 文件

1. 使用 python 2.7.4 32 位版本。如果您已经安装了 python 的其他版本，请先卸载然后安装 python 2.7.4 版本。重启 Seafile 服务器确认此问题是否依然存在。
2. 将注册表路径 `HKEY_CLASSES_ROOT\MIME\Database\Content Type` 下的非 ASCII 键删除，然后重试。

如何移动 seafile-server 文件夹

假设你的 Seafile 服务器程序位置为 `C:/SeafileProgram` , 数据文件夹位置为 `D:/seafile-server` 。现在你希望把数据文件夹从 `D:/seafile-server` 移动到 `E:/seafile-server`

1. 先在托盘菜单里选 **"停止并退出 seafile 服务器"**
2. 把数据文件夹 `D:/seafile-server` 移动到新位置 `E:/seafile-server`
3. 打开 `C:/SeafileProgram` 文件夹下的 `seafserv.ini` 这个文件。这个文件记录了数据文件夹的路径。把这个文件的内容改为 `E:/seafile-server` 。 注意： 如果你的新位置的路径包含非英文字符，那么请用支持 UTF8 格式的文本编辑器来编辑 `seafserv.ini` 文件，并保存为 UTF8 格式。否则 Seafile 服务器程序可能无法正确读取这个文件的内容。
4. 重新启动 Seafile 服务器。

更多信息

想了解更多关于 Seafile 服务器的信息， 请访问 https://github.com/haiwen/seafile/wiki

下载安装 Windows 版 Seafile 服务器

安装 Python 2.7.4 32 位版本

- 下载并安装 python 2.7.4 32 位版本
- 将 python2.7 的安装路径添加到系统的环境变量中 (PATH 变量)。比如：如果您将 python 2.7.4 安装在 `C:\Python27` 路径下，那么就将 `C:\Python27` 添加到环境变量中。

下载并解压 Seafile 服务器

- 获取 Seafile 服务器的最新版本。
- 为 Seafile 服务器程序创建一个新的文件夹，比如 `C:\SeafileProgram\` 。请记住此文件夹的位置，我们将在以后用到它。
- 将**seafile-server_1.7.0_win32.tar.gz**解压到 `C:\SeafileProgram\` 目录下。

现在，您的目录结构应该像如下这样：

```
C:\SeafileProgram
         |__ seafile-server-1.7.0
```

启动与初始化

启动 Seafile 服务器

在 `C:\SeafileProgram\seafile-server-1.7.0\` 文件夹下，找到**run.bat**文件并双击，以启动 Seafile 服务器。此时，您应该注意到 Seafile 服务器的图标已经出现在您的系统托盘中。

选择一个磁盘作为 Seafile 服务器数据的存储位置

现在，您可以在弹出的对话框中选择一个磁盘，以便存储 Seafile 服务器的数据：

- 请确保选择的磁盘拥有足够的剩余空间
- 点击确认按钮后， Seafile 将会在您选择的磁盘下为您创建一个名为 `seafile-server` 的文件夹。这个文件夹就是 Seafile 服务器的数据文件夹。如果您选择*D*盘，那么数据文件夹为 `D:\seafile-server`

添加管理员帐号

右击 Seafile 服务器的系统托盘图标， 选择"添加管理员帐号"选项。在弹出的对话框中输入您的管理员用户名和密码。

如果操作成功， Seafile 服务器托盘图标处会弹出一个气泡提示您"添加 Seahub 管理员账户成功"

配置 Seafile 服务器

初始化服务器之后，还需配置以下选项:

- 右击 Seafile 托盘图标，选择"打开 **seafile-server** 文件夹"选项。您的 seafile-server 数据文件夹将会打开。
- 编辑*ccnet/ccnet.conf*文件。在*ccnet.conf*文件中更改以下两行：

```
NAME = XXXXX
SERVICE_URL = XXX
```

- 将**NAME**的值配置成您的 Seafile 服务器的名字，比如`NAME = my-company-seafile`。这个名字将会在您的 Seafile 客户端上显示。

- 将**SERVICE_URL**的值配置成`http://<您的 IP 地址>:8000`。比如您的 Windows 服务器地址为*192.168.1.100*， 那么配置成`SERVICE_URL = http://192.168.1.100:8000`

编辑完成后，右击 Seafile 服务器托盘图标，选择"重启 **Seafile Server**"选项以重启 Seafile 服务器。

访问Seahub

打开您的浏览器，访问 *http://127.0.0.1:8000* 网址。用您的管理员账户登录， 如果成功登录，那么说明您的 Seafile 服务器初始化成功。

配置完成

Seafile 服务器的配置到此已经完成。如果您想了解如何使用 Seafile 客户端，请参考 Seafile 客户端手册

您可能还会想要了解以下信息：

- Seafile LDAP配置
- 安装 Seafile 为 Windows 服务
- 所用端口说明
- 升级
- 个性化配置

安装 Seafile 为 Windows 服务

将 Seafile 服务器作为 Windows 服务安装的好处

- 在您的所有用户注销后 Seafile 服务器能够继续保持运行
- 系统启动时，即使没有用户登录， Seafile 服务器也会开始运行

如何作为 Windows 服务安装

- 右击 Seafile 服务器托盘图标，选择"安装为 Windows 服务"选项
- 在弹出的对话框中，点击是按钮

如果操作成功，将会弹出一个对话框提示您"已经成功安装 Seafile 服务"。

确认 Seafile 服务器已经开始作为 Windows 服务运行

- 注销当前用户
- 在另一台电脑上访问 Seahub 。如果 Seahub 网站仍然可以访问，那么说明 Seafile 服务器已经开始作为 Windows 服务运行

安装为 Windows 服务后如何启动托盘图标

如果您已经将 Seafile 服务器安装为 Windows 服务，那么在您下次系统启动时， Seafile 服务将会在后台自动运行。这样，当用户登录时， Seafile 服务器托盘图标就不会自动出现。

启动托盘图标，只需双击 `C:\SeafileProgram\seafile-server-1.7.0` 文件夹下的 `run.bat` 文件。

卸载 Seafile 服务器的 Windows 服务

如果您想卸载 Seafile 服务器的 Windows 服务，请执行以下两步：

- 右击托盘图标，选择"卸载 Windows 服务"选项
- 在弹出的确认对话框中点击"是"按钮

所用端口说明

Seafile 服务器由两个组件组成，默认情况下用到 8000, 8082 两个端口号 (TCP)。

配置文件

所有端口的相关配置都记录在`ccnet.conf`文件和`seafile.conf`文件中

如何打开 **ccnet.conf** 文件

- 右击 Seafile 服务器托盘图标，选择"打开 seafile-server 文件夹"选项
- 打开`seafile-server`目录下的`ccnet`文件夹。`ccnet.conf`文件就在此文件夹下。

如何打开 **seafile.conf** 文件

- 右击 Seafile 服务器托盘图标，选择"打开 seafile-server 文件夹"选项
- 打开`seafile-server`目录下的`seafile-data`文件夹。`seafile.conf`文件就在此文件夹下。

在接下来的部分，我们分别列举了 Seafile 服务器各个组件用到的TCP端口以及如何改变它们（比如，一些端口很有可能已经被其他应用程序占用）。

注意：如果您改变了以下任何端口，请重启 Seafile 服务器。

seahub

`seahub` 是 Seafile 服务器的 Web 端。

注意：如果您改变了 Seahub 的端口号，`ccnet.conf`文件中的`SERVICE_URL`也应该随之改变。

- 默认端口： 8000
- 如何设置端口号： 编辑`seafile.conf`文件。 设置在`seahub`段下`port`的值.

```
[seahub]
port=8000
```

- 编辑`ccnet.conf`文件，改变`SERVICE_URL`的值。比如， 如果您将端口号重新设置为 8001， 那么更改`SERVICE_URL`的值如下：

  ```
  [General]
  SERVICE_URL = <您的 IP 或者域名>:8001
  ```

seafile fileserver

`seafile fileserver` 负责为 Seahub 处理文件的上传和下载

- 默认端口： 8082
- 如何设置端口号： 桌面客户端会连接这个端口来同步文件，所以不要修改这个端口。

升级

- 小版本升级
- 大版本升级
- 升级 Windows 服务

注意：升级之前，你需要先停止 Seafile 服务器

解压新版本服务器

假设升级之前，你的目录结构是：

```
C:/SeafileProgram
             |______ seafile-server-1.7.0/
```

那么，升级的第一步是下载新版本的程序包，并解压到文件夹 ` C:/SeafileProgram` 下面。

```
C:/SeafileProgram
             |______ seafile-server-1.7.0/
             |______ seafile-server-1.8.0/
```

小版本升级 (如从 **1.7.0** 版本升级到 **1.7.1** 版本)

现在假定您要将 Seafile 服务器的 Windows 服务从 1.7.0 版本升级到 1.7.1 版本

迁移 **avatars** 文件夹的内容

找到**seafile-server-1.7.0/seahub/media/avatars**目录

在**avatars/**文件夹中包含着所有Seafile用户的头像。

如果您有一个用户名为 `foo@foo.com` 的用户，那么在**avatars/**文件夹中，您会发现一个叫作 `foo@foo.com` 的子文件夹。这个子文件夹包含着用户 `foo@foo.com` 的头像图片。

将所有像 `foo@foo.com` 的这种子文件夹拷贝到 `seafile-server-1.7.1/seahub/media/avatars` 目录下。这样，当您启动新的 1.7.1 版本的 Seafile 服务器时，这些头像可以正确加载。

大版本升级 (如从 **1.7.0** 版本升级到 **1.8.0** 版本)

现在假定您要将 Seafile 服务器的 Windows 服务从 1.7.x 版本升级到 1.8.y 版本

运行数据库升级脚本

- 找到seafile-server-1.8.y/upgrade目录

- 在这个目录下，右击`upgrade_1.7_1.8.bat`文件
- 选择**"**以管理员身份运行**"**

拷贝头像

将在**seafile-server-1.7.0/seahub/media/avatars**目录下的所有子文件夹拷贝到**seafile-server-1.8.0/seahub/media/avatars**目录下

升级 Windows 服务

如果您已经将 Seafile 服务器作为 Windows 服务安装，您需要做以下几步：

- 运行老版本的 Seafile Windows 服务器，右击托盘图标，在菜单中选择卸载 **Windows** 服务
- 退出老版本的 Seafile Windows 服务器
- 启动新版本的 Seafile Windows 服务器，右击托盘图标，在菜单中选择安装为 **Windows** 服务

服务器从 **Windows** 迁移到 **Linux**

假设你当前已经在使用 Windows 服务器(使用 SQLite 数据库)，现在希望把服务器迁移到 Linux 下。

1. 安装 **Linux** 服务器

第一步你需要安装全新一个 Linux 服务器。同样使用 SQLite 数据库。下面假设你把 Seafile 服务器默认安装在 `/home/haiwen/` 目录下。

2. 替换数据和配置文件

删除 **Linux** 的配置文件和数据

```
rm /home/haiwen/seahub_settings.py
rm /home/haiwen/seahub.db
rm -r /home/haiwen/seafile-data
cp /home/haiwen/ccnet/seafile.ini /home/haiwen/seafile.ini
rm -r /home/haiwen/ccnet
```

其中 seafile.ini 指向 seafile-data 目录所在位置，等会需要用到，这里先拷贝出来。

拷贝配置文件和数据

- 将 Windows 中 **seafile-server** 文件夹下的 `seahub_settings.py` 文件，拷贝到 linux `/home/haiwen/` 目录下
- 将 Windows 中 **seafile-server** 文件夹下的 `seahub.db` 文件，拷贝到 linux `/home/haiwen/` 目录下；
- 将 Windows 中 **seafile-server** 的子文件夹 `seafile-data`，拷贝到 linux `/home/haiwen/` 目录下；
- 将 Windows 中 **seafile-server** 的子文件夹 `ccnet`，拷贝到 linux `/home/haiwen/` 目录下；
- 将 `/home/haiwen/seafile.ini` 拷贝到新 **ccnet** 目录中

垃圾回收

- 右击 Seafile 托盘图标，选择 退出并停止 *Seafile Server*
- 打开文件浏览器，找到 Seafile 安装目录 `seafile-server-3.x.x`
- 右击 **gc.bat**, 并选择 以管理员身份运行

垃圾回收程序会运行并删除所有未用的数据块。

部署 Seafile 专业版

安装

- 下载与安装 Seafile 专业版服务器
- 从 Seafile 社区版服务器迁移到专业版服务器
- 升级 Seafile 专业版服务器

S3/Swift/Ceph

- 安装 Seafile 专业版服务器并使用亚马逊 S3
- 安装 Seafile 专业版服务器并使用 OpenStackSwift
- 安装 Seafile 专业版服务器并使用 Ceph

配置选项

- Seafile 专业版服务器可配置的选项

搜索功能

- 关于文件搜索的一些细节

FAQ

- Seafile 专业版服务器的 FAQ

变更记录

许可证书

- Seafile 专业版软件许可协议

准备工作

安装依赖库。

Ubuntu 14.04，可用以下命令安装全部依赖。

```
sudo apt-get install openjdk-7-jre poppler-utils  libpython2.7 python-pip mysql-server \
  python-setuptools python-imaging python-mysqldb python-memcache

sudo pip install boto
```

CentOS 6.6 下:

```
wget https://bootstrap.pypa.io/get-pip.py
sudo python get-pip.py
sudo yum install java-1.7.0-openjdk poppler-utils libreoffice libreoffice-headless \
  libreoffice-pyuno python-dev python-setuptools python-imaging MySQL-python mysql-server
sudo pip install boto
sudo /etc/init.d/mysqld start
```

补充说明： 关于 **Java**

注意：Seafile 专业版需要 java 1.7 以上版本, 请用 `java -version` 命令查看您系统中的默认 java 版本. 如果不是 java 7, 那么, 请 更新默认 java 版本.

下载与安装 **Seafile** 专业版服务器

获得许可证书

将您得到的许可证书放在顶层目录下。比如，在这篇文档页面里，我们把 `/data/haiwen/` 作为顶层目录。

下载与解压 **Seafile** 专业版服务器

```
tar xf seafile-pro-server_2.1.5_x86-64.tar.gz
```

现在您的目录结构应该像如下这样：

```
haiwen
├── seafile-license.txt
```

```
└── seafile-pro-server-2.1.5/
```

安装

Seafile 专业版服务器的安装步骤与Seafile 社区版服务器安装步骤相同。

1. 下载与安装 Seafile 服务器并使用 MySQL 数据库
2. 使用 Nginx 为 Web 服务器
3. 配置和使用 Memcached (可选，建议用户数超过 50 人的时候配置)
4. 配置和使用 HTTPS (可选)
5. 配置开机启动脚本 (可选)

在您成功安装 Seafile 专业版服务器之后，您的目录结构应该像如下这样：

```
#tree haiwen -L 2
haiwen
├── seafile-license.txt # license file
├── ccnet               # configuration files
│   ├── ccnet.conf
│   ├── mykey.peer
│   ├── PeerMgr
│   └── seafile.ini
├── pro-data            # data specific for professional version
│   └── seafevents.conf
├── seafile-data
│   └── seafile.conf
├── seafile-pro-server-2.1.5
│   ├── reset-admin.sh
│   ├── runtime
│   ├── seafile
│   ├── seafile.sh
│   ├── seahub
│   ├── seahub-extra
│   ├── seahub.sh
│   ├── setup-seafile.sh
│   ├── setup-seafile-mysql.py
│   ├── setup-seafile-mysql.sh
│   └── upgrade
├── seahub-data
│   └── avatars         # for user avatars
├── seahub_settings.py   # seahub config file
```

限制条件

您可能已经部署过 Seafile 社区版服务器，并想要切换到专业版，或者反过来从专业版迁移到社区版。但是有一些限制条件需要您注意：

- 您只能在相同大版本的社区版服务器和专业版服务器之间进行切换。

这意味着，如果您正在使用 2.0 版本的社区版服务器， 并且想要切换到 2.1 版本的专业版服务器，您必须先将您的社区版服务器升级到 2.1 版本， 然后按照以下指南切换到 2.1 版本的专业版服务器。(版本号 2.1.x 中的最后一位没有关系)

准备工作

安装 Java 运行时环境 (JRE)

如果您的系统环境是 Ubuntu 或者 Debian，执行以下命令：

```
sudo apt-get install openjdk-7-jre
```

如果您的系统环境是 CentOS 或者 Red Hat，执行以下命令：

```
sudo yum install java-1.7.0-openjdk
```

注意：您也可以使用 Oracle JRE.

注意：Seafile 专业版需要 java 1.7 以上版本, 请用 `java -version` 命令查看您系统中的默认 java 版本. 如果不是 java 7, 那么, 请 更新默认 java 版本.

安装 poppler-utils

poppler-utils 提供对 pdf 文件的全文检索功能。

如果您的系统环境是 Ubuntu 或者 Debian，执行以下命令：

```
sudo apt-get install poppler-utils
```

如果您的系统环境是 CentOS 或者 Red Hat，执行以下命令：

```
sudo yum install poppler-utils
```

安装 Libreoffice 和 UNO 库

Libreoffice 和 Python-uno 库提供对办公文件的在线预览功能。如果它们没有安装，办公文件就不能在线预览。

如果您的系统环境是 Ubuntu 或者 Debian，执行以下命令：

```
sudo apt-get install libreoffice python-uno
```

如果您的系统环境是 CentOS 或者 RHEL，执行以下命令：

```
sudo yum install libreoffice libreoffice-headless libreoffice-pyuno
```

对于其他的 Linux 发行版您可以参考：Linux 下 LibreOffice 的安装

一般地，您还需要为您的使用语言安装字体，特别是在亚洲地区，否则 office 文件和 pdf 文件不能正确地显示。

比如， 中国的用户可能希望安装文泉驿系列的 TrueType 字体：

```
# 如果您的系统环境是 Ubuntu 或者 Debian，执行以下命令：
sudo apt-get install ttf-wqy-microhei ttf-wqy-zenhei xfonts-wqy
```

迁移

我们假定您已经在 `/data/haiwen/seafile-server-2.1.0` 目录下部署了 Seafile 社区版服务器的 2.1.0 版本。

获得许可证书

将您获得的许可证书放在 Seafile 安装位置的顶层目录下。在我们的例子中，顶层目录是 `/data/haiwen/` 。

下载与解压 Seafile 专业版服务器

- 32 位
- 64 位

您应该将压缩包解压到您的 Seafile 安装位置的顶层目录，在我们的例子中，顶层目录是 `/data/haiwen` 。

```
tar xf seafile-pro-server_2.1.0_x86-64.tar.gz
```

现在您的目录结构像如下这样：

```
haiwen
├── seafile-license.txt
├── seafile-pro-server-2.1.0/
├── seafile-server-2.1.0/
├── ccnet/
├── seafile-data/
├── seahub-data/
├── seahub.db
└── seahub_settings.py
```

您应该已经注意到社区版服务器和专业版服务器名字的不同。以 64 位的 2.1.0 版本为例：

- Seafile 社区版服务器压缩包叫作 `seafile-server_2.1.0_x86-86.tar.gz` ；解压后，文件夹名叫作 `seafile-server-2.1.0`
- Seafile 专业版服务器压缩包叫作 `seafile-pro-server_2.1.0_x86-86.tar.gz` ；解压后，文件夹名叫作 `seafile-pro-server-2.1.0`

迁移

- 如果 Seafile 社区版服务器正在运行，请先停止它：

  ```
  cd haiwen/seafile-server-2.1.0
  ./seafile.sh stop
  ./seahub.sh stop
  ```

- 运行迁移脚本

  ```
  cd haiwen/seafile-pro-server-2.1.0/
  ./pro/pro.py setup --migrate
  ```

迁移脚本将会为您做以下的工作：

- 确保您满足所有的先决条件
- 创建必要的额外配置选项
- 更新 avatar 目录
- 创建额外的数据库表

现在您的目录结构像如下这样：

> haiwen
> ├── seafile-license.txt
> ├── seafile-pro-server-2.1.0/
> ├── seafile-server-2.1.0/
> ├── ccnet/
> ├── seafile-data/

> ├── seahub-data/
> ├── seahub.db
> ├── seahub_settings.py
> └── **pro-data/**

启用 **Seafile** 专业版服务器

```
cd haiwen/seafile-pro-server-2.1.0
./seafile.sh start
./seahub.sh start
```

切换回社区版服务器

如果 Seafile 专业版服务器正在运行，请先停止它：

```
cd haiwen/seafile-pro-server-2.1.0/
./seafile.sh stop
./seahub.sh stop
```

更新 avatar 目录的链接，参考小版本升级

```
cd haiwen/seafile-server-2.1.0/
./upgrade/minor-upgrade.sh
```

启用 Seafile 社区版服务器

```
cd haiwen/seafile-server-2.1.0/
./seafile.sh start
./seahub.sh start
```

升级

请参考 升级

Amazom S3 下安装

为了安装 Seafile 专业版服务器并使用亚马逊 S3，您需要：

- 按照 下载安装 Seafile 专业版服务器 指南安装基本的 Seafile 专业版服务器。
- 安装 python 的 `boto` 库。它可以用来访问 S3 服务。

```
sudo easy_install boto
```

- 安装和使用 Memcached.
- 编辑 `/data/haiwen/seafile-data/seafile.conf` 文件，添加下面几行：

```
[commit_object_backend]
name = s3
# bucket 的名字只能使用小写字母，数字，点号，短划线
bucket = my.commit-objects
key_id = your-key-id
key = your-secret-key
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[fs_object_backend]
name = s3
# bucket 的名字只能使用小写字母，数字，点号，短划线
bucket = my.fs-objects
key_id = your-key-id
key = your-secret-key
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[block_backend]
name = s3
# bucket 的名字只能使用小写字母，数字，点号，短划线
bucket = my.block-objects
key_id = your-key-id
key = your-secret-key
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100
```

建议您为 commit，fs 和 block objects 分别创建 buckets。 key_id 和 key 用来提供 S3 的身份认证。您可以在您的 AWS 账户页面的"安全证书"段找到 key_id 和 key。

当您在 S3 上创建 buckets 时，请先阅读 S3 bucket 命名规则。注意， 尤其不要在 bucket 的名字中使用大写字母（不要使用骆驼式命名法，比如 MyCommitOjbects）。

为了获得最佳性能，强烈建议您安装 memcached 并且为 objects 启用 memcache。

OpenStackSwift 下安装

从专业版服务器的 2.0.5 版本开始，Seafile 可以使用兼容 S3 的云存储（比如 OpenStack/Swift）作为后端。这篇文档将以使用 Swift 为例。

准备工作

首先您需要为 Swift 启用 S3 的模拟中间件。有关说明可以参考以下链接：

- http://www.buildcloudstorage.com/2011/11/s3-apis-on-openstack-swift.html
- http://docs.openstack.org/grizzly/openstack-compute/admin/content/configuring-swift-with-s3-emulation-to-use-keystone.html

成功安装设置 S3 中间件后，您就可以用任何 S3 的客户端访问它了。在 Swift 中，访问的 key id 作为用户名 secret key 作为用户的密码。下一步您要做的就是为 Seafile 创建 buckets。使用 Python 的 boto 库您可以这样做：

```
import boto
import boto.s3.connection

connection = boto.connect_s3(
    aws_access_key_id='swifttest:testuser',
    aws_secret_access_key='testing',
    port=8080,
    host='swift-proxy-server-address',
    is_secure=False,
    calling_format=boto.s3.connection.OrdinaryCallingFormat())
connection.create_bucket('seafile-commits')
connection.create_bucket('seafile-fs')
connection.create_bucket('seafile-blocks')
```

修改 seafile.conf 文件

将下面几行追加到 `seafile-data/seafile.conf` 文件中

```
[block_backend]
name = s3
bucket = seafile-blocks
key_id = swifttest:testuser
key = testing
host = <swift-proxy-server-address>:8080
path_style_request = true
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[commit_object_backend]
name = s3
bucket = seafile-commits
key_id = swifttest:testuser
key = testing
```

```
host = <swift-proxy-server-address>:8080
path_style_request = true
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[fs_object_backend]
name = s3
bucket = seafile-fs
key_id = swifttest:testuser
key = testing
host = <swift-proxy-server-address>:8080
path_style_request = true
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100
```

为了提高性能，Seafile 会在 memcached 中缓存小的 objects。所以强烈建议您安装并且运行 memcached。最好是为 Seafile 提供一个专门的 memcached 实例并为其分配 128MB 的内存空间。

Ceph 下安装

Ceph 是一种可扩展的分布式存储系统。Seafile 可以使用 Ceph 的 RADOS 对象存储层作为存储后端。

拷贝 ceph 的配置文件和客户端的密钥环

Seafile 可以看作 Ceph/RADOS 的客户端，所以它需要访问 ceph 集群的配置文件和密钥环。您必须将 ceph 管理员节点 /etc/ceph 目录下的文件拷贝到 seafile 的机器上。

```
seafile-machine# sudo scp user@ceph-admin-node:/etc/ceph/ /etc
```

编辑 Seafile 配置文件

编辑 `seafile-data/seafile.conf` 文件，添加以下几行：

```
[block_backend]
name = ceph
ceph_config = /etc/ceph/ceph.conf
pool = seafile-blocks
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[commit_object_backend]
name = ceph
ceph_config = /etc/ceph/ceph.conf
pool = seafile-commits
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[fs_object_backend]
name = ceph
ceph_config = /etc/ceph/ceph.conf
pool = seafile-fs
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100
```

建议您为 commit， fs 和 block objects 分别创建连接池：

```
ceph-admin-node# rados mkpool seafile-blocks
ceph-admin-node# rados mkpool seafile-commits
ceph-admin-node# rados mkpool seafile-fs
```

安装并启用 memcached

为了最佳性能，强烈建议您安装 memcached 并为 objects 启用 memcache。

我们建议您为 memcached 分配 128MB 的内存空间。编辑 /etc/memcached.conf 文件如下：

```
# Start with a cap of 64 megs of memory. It's reasonable, and the daemon default
# Note that the daemon will grow to this size, but does not start out holding this much
# memory
# -m 64
-m 128
```

使用阿里云开放存储（OSS）作为后端存储

准备工作

为了安装 Seafile 专业版服务器并使用阿里云OSS，您需要：

- 按照 下载安装 Seafile 专业版服务器 指南安装基本的 Seafile 专业版服务器。
- 按照 这个文档 安装 OSS Python SDK。
- 安装和使用 Memcached。Seafile 会将部分对象缓存在 memcached 中，以提高性能。建议至少给 memcached 分配 128MB 内存。请修改 memcached 的配置文件（Ubuntu 上在 /etc/memcached.conf）：

```
# Start with a cap of 64 megs of memory. It's reasonable, and the daemon default
# Note that the daemon will grow to this size, but does not start out holding this much
# memory
# -m 64
-m 128
```

修改 seafile.conf

编辑 `/data/haiwen/seafile-data/seafile.conf` 文件，添加下面几行：

```
[commit_object_backend]
name = oss
bucket = <your-seafile-commits-bucket>
key_id = <your-key-id>
key = <your-key>
region = beijing
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[fs_object_backend]
name = oss
bucket = <your-seafile-fs-bucket>
key_id = <your-key-id>
key = <your-key>
region = beijing
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100

[block_backend]
name = oss
bucket = <your-seafile-blocks-bucket>
key_id = <your-key-id>
key = <your-key>
region = beijing
memcached_options = --SERVER=localhost --POOL-MIN=10 --POOL-MAX=100
```

关于上面配置的几点说明：

- 建议您为 commit，fs 和 block objects 分别创建 buckets。为了性能和节省网络流量成本，您应该在 seafile 服务器运行的区域内创建 bucket。
- key_id 和 key 用来提供 OSS 的身份认证。您可以在 OSS 管理界面上找到它们。
- region 是您创建的 bucket 所在的区域，比如 beijing, hangzhou, shenzhen 等。

配置选项

在 `/data/haiwen/pro-data/seafevents.conf` 配置文件中：

```
[Audit]
## 审计日志默认是关闭的
enable = true

[INDEX FILES]
# 要启用搜索，必须设置为 "true"
enabled = true

# 搜索索引更新的时间间隔。可以是 s（秒）， m（分）， h（小时）， d（天）
interval=10m

# 如果设置为 "true"，搜索索引更新时也会索引办公文件和 pdf 文件中的内容
# 注意： 如果您将此选项从 "false" 设置为 "true"， 那么您需要清空搜索索引然后再次更新索引。更多详情请参
index_office_pdf=false

[OFFICE CONVERTER]

# 要启用办公文件和 pdf 文件的在线预览功能必须设置为 "true"
enabled = true

# 能够并发运行的 libreoffice 工作进程数
workers = 1

# 转换后的办公文件和 pdf 文件的存储位置
outputdir = /tmp/

# 允许的最大在线预览页数。默认为 50 页
max-pages = 50

# 允许的最大预览文件的大小，单位是 MB。默认为 2 MB
max-size = 2

[SEAHUB EMAIL]

# 要启用用户邮件通知，必须设置为 "true"
enabled = true

# 发送 seahub 邮件的时间间隔。可以是 s（秒）， m（分）， h（小时）， d（天）
interval = 30m
```

您可能想要更改的配置选项

以上小节已经列出了 `/data/haiwen/pro-data/seafevents.conf` 配置文件中的所有配置选项。大多数情况下，使用默认配置就足够了。但是为了更好地满足自身需求，您可能想要更改其中的某些选项。

我们将这些配置选项列出在下面的表中，以及我们选择默认设置的原因。

		默认	

		值	
INDEX FILES	index_office_pdf	false	默认情况下， office 文档和 pdf 文档的全文搜索功能是不开启的。这是因为它会占用相当大的搜索索引空间。要开启它，将它的值设置为 "true" 然后重新创建搜索索引。更多详情请参考 [[Seafile 专业版服务器的 FAQ]]。
OFFICE CONVERTER	max-size	2MB	允许的最大在线预览文件的大小是 2MB。在线预览会把 office 和 pdf 文档转换成 HTML 然后在浏览器中显示。如果文件太大，转换会花费很长时间且占用很多空间。
OFFICE CONVERTER	max-pages	50	当在线预览一个 office 或者 pdf 文档时，文档的前 50 页将会首先被显示。如果此值太大，转换会花费很长时间且占用很多空间。

Seafile 专业版服务器支持在线预览 office 文件，配置方法如下。

安装 Libreoffice/UNO

Office 预览依赖于 Libreoffice 4.1+ 和 Python-uno 库。

Ubuntu/Debian:

```
sudo apt-get install libreoffice libreoffice-script-provider-python
```

> For older version of Ubuntu: `sudo apt-get install libreoffice python-uno`

Centos/RHEL:

```
sudo yum install libreoffice libreoffice-headless libreoffice-pyuno
```

其他 Linux 发行版可以参考: Installation of LibreOffice on Linux

你还需要安装字体文件:

```
# For ubuntu/debian
sudo apt-get install ttf-wqy-microhei ttf-wqy-zenhei xfonts-wqy
```

开启配置项

1. 打开 `pro-data/seafevents.conf`, 添加:

   ```
   [OFFICE CONVERTER]
   enabled = true
   ```

2. 保存后 `seafevents.conf` 重启 Seafile 服务 `./seafile.sh restart`

其他配置选项

```
[OFFICE CONVERTER]

## How many libreoffice worker processes to run concurrenlty
workers = 1

## where to store the converted office/pdf files. Deafult is /tmp/.
outputdir = /tmp/

## how many pages are allowed to be previewed online. Default is 50 pages
max-pages = 50

## the max size of documents to allow to be previewed online, in MB. Default is 2 MB
## Preview a large file (for example >30M) online will freeze the browser.
```

```
max-size = 2
```

FAQ

Office 预览不能工作，日志文件在哪?

你可以查看 logs/seafevents.log

Office 预览不能工作的一个可能原因是你的机器上的 libreoffice 版本太低，可以用以下方法修复

删除安装的 libreoffice:

```
sudo apt-get remove libreoffice* python-uno python3-uno
```

从官网下载最新版 libreoffice official site，并安装

```
tar xf LibreOffice_4.1.6_Linux_x86-64_deb.tar.gz
cd LibreOffice_4.1.6.2_Linux_x86-64_deb
cd DEBS
sudo dpkg -i *.deb
```

重启 Seafile 服务

```
./seafile.sh restart
```

常见问题

我不能搜索 office/PDF 文档

首先，确认在 **seafevents.conf** 文件中，已经设置 `index_office_pdf = true` ：

```
[INDEX FILES]
...
index_office_pdf = true
```

然后，检查是否安装 **pdftotext**

如果 **pdftotext** 没有安装，就不能从 PDF 文件中提取文本。

```
which pdftotext
```

执行上面的命令，如果没有输出，那么您需要安装它：

```
sudo apt-get install poppler-utils
```

pdftotext 安装完成后，您需要重新构建您的搜索索引。

```
./pro/pro.py search --clear
./pro/pro.py search --update
```

当我搜索一个关键词时，没有返回结果

默认情况下，搜索索引每 10 分钟更新一次。所以，在第一次索引更新前，无论您搜索什么都不会返回结果。

为了使搜索能够立即生效，您可以手动更新搜索索引：

- 确保您已经启动了 Seafile 服务器
- 手动更新搜索索引：

  ```
   cd haiwen/seafile-pro-server-2.0.4
  ./pro/pro.py search --update
  ```

不能搜索加密的文件

这是因为服务器不能索引加密的文件，因为它们被加密了。

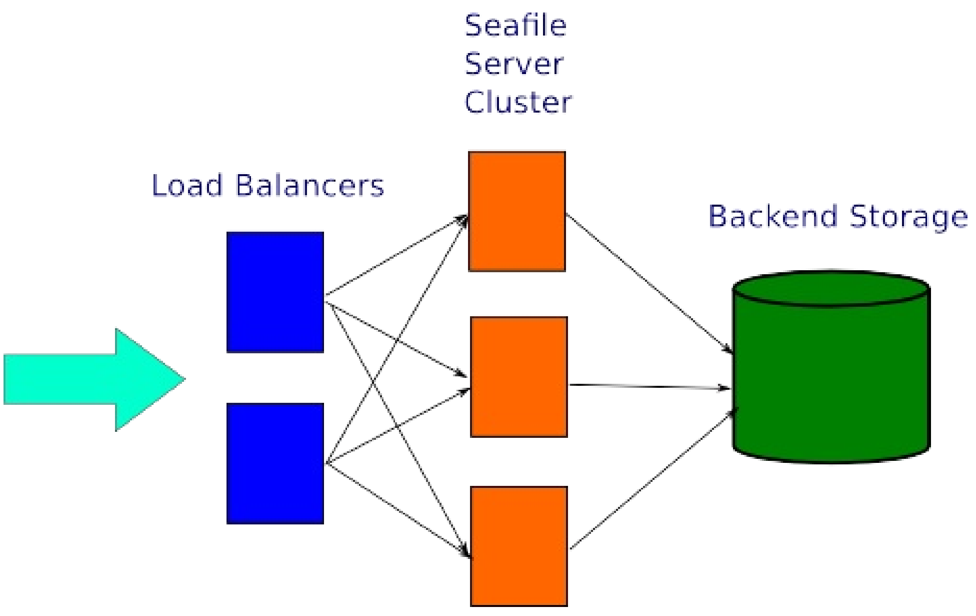
Deploy in a cluster

Architecture

The Seafile cluster solution employs a 3-tier architecture:

- Load balancer tier: Distribute incoming traffic to Seafile servers. HA can be achieved by deploying multiple load balancer instances.
- Seafile server cluster: a cluster of Seafile server instances. If one instance fails, the load balancer will stop handing traffic to it. So HA is achieved.
- Backend storage: Distributed storage cluster, such as S3, Openstack Swift, Ceph.

This architecture scales horizontally. That is, you can handle more traffic by adding more machines. The architecture is presented in the following picture.



Preparation

Hardware

At least 2 Linux server with at least 2GB RAM.

Install Python libraries

On each mode, you need to install some python libraries.

First make sure your have installed python 2.6 or 2.7, then:

```
sudo easy_install pip
sudo pip install boto
```

If you receive an error about "Wheel installs require setuptools >= ...", run this between the pip and boto lines above

```
sudo pip install setuptools --no-use-wheel --upgrade
```

Setup Memcached

All seafile server instances will share the same memcached servers. Let's assume that the address of memcached server is 192.168.1.134, listening on port 11211 (the default).

By default, memcached only listen on 127.0.0.1. So you should modify memcached.conf and restart memcached.

```
# Specify which IP address to listen on. The default is to listen on all IP addresses
# This parameter is one of the only security measures that memcached has, so make sure
# it's listening on a firewalled interface.
-l 0.0.0.0
```

It's also recommended to set a higher limit for memcached's memory, such as 256MB.

```
# Start with a cap of 64 megs of memory. It's reasonable, and the daemon default
# Note that the daemon will grow to this size, but does not start out holding this much
# memory
-m 256
```

Seafile servers share session information within memcached. If you set up a memcached cluster, please make sure all the seafile server nodes connects to all the memcached nodes.

Configure a Single Node

You should make sure the config files on every Seafile server are consistent. **It's critical that you don't set up seafile server on each machine separately. You should set up seafile server on one machine then copy the config directory to the other machines.**

Get the license

Put the license you get under the top level diretory. In our wiki, we use the diretory `/data/haiwen/` as the top level directory.

Download/Uncompress Seafile Professional Server

```
tar xf seafile-pro-server_2.1.3_x86-64.tar.gz
```

Now you have:

```
haiwen
├── seafile-license.txt
└── seafile-pro-server-2.1.3/
```

Setup Seafile Config

The setup process of Seafile Professional Server is the same as the Seafile Community Server. See Download and Setup Seafile Server With MySQL in the community wiki.

Note: **Use the load balancer's address or domain name for the server address. Don't use the local IP address of each Seafile server machine. This assures the user will always access your service via the load balancers.**

After the setup process is done, you still have to do a few manually changes to the config files.

seafile-data/seafile.conf

You have to add the following configuration to `seafile-data/seafile.conf`

```
[cluster]
enabled = true
memcached_options = --SERVER=192.168.1.134 --POOL-MIN=10 --POOL-MAX=100
```

If you have a memcached cluster, you need to specify all the memcached server addresses in seafile.conf. The format is

```
[cluster]
enabled = true
memcached_options = --SERVER=192.168.1.134 --SERVER=192.168.1.135 --SERVER=192.168.1.136
```

(Optional) The Seafile server also opens a port for the load balancers to run health checks. Seafile by default use port 11001. You can change this by adding the following config to seafile-data/seafile.conf

```
[cluster]
health_check_port = 12345
```

seahub_settings.py

Add following configuration to `seahub_settings.py`. These settings tell Seahub to store avatar in database and cache avatar in memcached, and store css CACHE to locale memory of every node.

```
CACHES = {
    'default': {
        'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
        'LOCATION': '192.168.1.134:11211',
    }
}

AVATAR_FILE_STORAGE = 'seahub.base.database_storage.DatabaseStorage'

COMPRESS_CACHE_BACKEND = 'locmem://'
```

If you enable thumbnail feature, you'd better set thumbnail storage path to a **Shared Folder**, so that every node will create/get thumbnail through the same **Shared Folder** instead respectively.

```
THUMBNAIL_ROOT = 'path/to/shared/folder/'
```

pro-data/seafevents.conf

Add following to `pro-data/seafevents.conf` to disable file indexing service:

```
[INDEX FILES]
external_es_server = true
```

Update Seahub Database

In cluster environment, we have to store avatars in the database instead of in a local disk.

```
CREATE TABLE `avatar_uploaded` (`filename` TEXT NOT NULL, `filename_md5` CHAR(32) NOT NUL
```

Backend Storage Settings

You also need to add the settings for backend cloud storage systems to the config files.

- For NFS: Setup Seafile cluster with NFS
- For S3: Setup With Amazon S3
- For OpenStack Swift: Setup With OpenStackSwift
- For Ceph: Setup With Ceph

Run and Test the Single Node

Once you have finished configuring this single node, start it to test if it can run correctly:

```
cd /data/haiwen/seafile-server-latest
./seafile.sh start
./seahub.sh start
```

Note: The first time you start seahub, the script would prompt you to create an admin account for your seafile server.

Open your browser and visit http://ip-address-of-this-node:8000, login with the admin account.

Configure other nodes

Now you have one node working fine, let's continue to configure other nodes.

Copy the config to all Seafile servers

Supposed your seafile installation directory is `/data/haiwen`, compress this whole directory into a tar ball and copy the tar ball to all other Seafile server machines. You can simply uncompress the tar ball and use it.

On each node, run `./seafile.sh` and `./seahub.sh` to start seafile server.

Setup Nginx/Apache and Https

You'll usually want to use Nginx/Apache and https for web access. You need to set it up on each machine running Seafile server. **Make sure the certificate on all the servers are the same.**

- For Nginx:
  - Config Seahub with Nginx
  - Enabling Https with Nginx
- For Apache:
  - Config Seahub with Apache
  - Enabling Https with Apache

Firewall Settings

Beside standard ports of a seafile server, there are 2 firewall rule changes for Seafile cluster:

- On each Seafile server machine, you should open the health check port (default 11001);
- On the memcached server, you should open the port 11211. For security, only the Seafile servers should be allow to access this port.

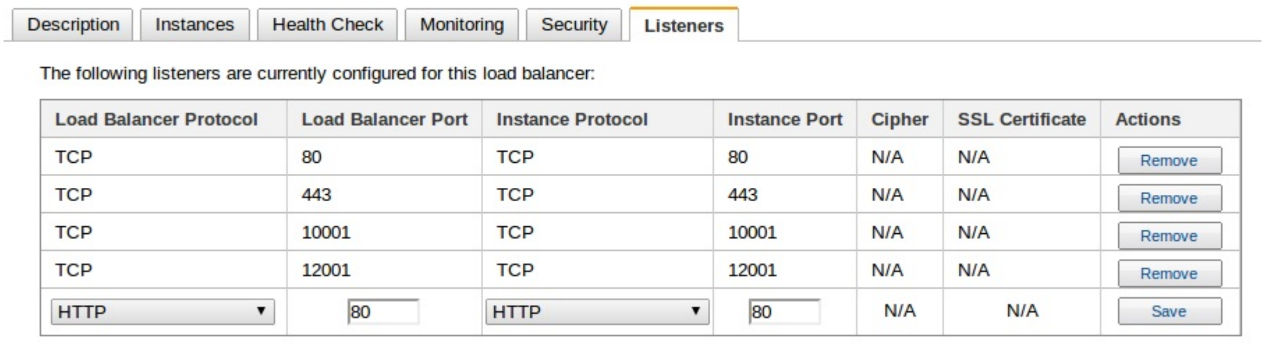
Load Balancer Setting

Now that your cluster is already running, fire up the load balancer and welcome your users.

AWS Elastic Load Balancer (ELB)

In the AWS ELB management console, after you've added the Seafile server instances to the instance list, you should do two more configurations.

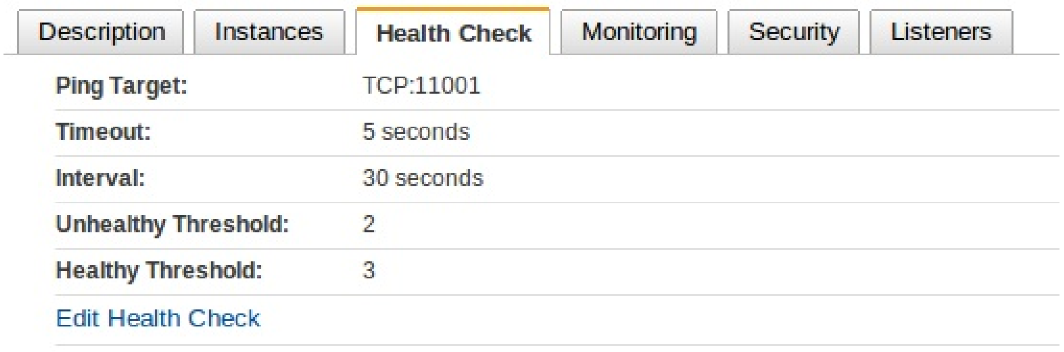
First you should setup TCP listeners



The following listeners are currently configured for this load balancer:

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port	Cipher	SSL Certificate	Actions
TCP	80	TCP	80	N/A	N/A	Remove
TCP	443	TCP	443	N/A	N/A	Remove
TCP	10001	TCP	10001	N/A	N/A	Remove
TCP	12001	TCP	12001	N/A	N/A	Remove
HTTP	80	HTTP	80	N/A	N/A	Save

Then you setup health check



Ping Target:	TCP:11001
Timeout:	5 seconds
Interval:	30 seconds
Unhealthy Threshold:	2
Healthy Threshold:	3

Edit Health Check

HAProxy

This is a sample `/etc/haproxy/haproxy.cfg` :

(Assume your health check port is `12345`)

```
global
    log 127.0.0.1 local1 notice
    maxconn 4096
    user haproxy
    group haproxy

defaults
    log global
    mode http
    retries 3
    maxconn 2000
    contimeout 5000
    clitimeout 50000
    srvtimeout 50000

listen seahub 0.0.0.0:80
    mode http
    option httplog
    option dontlognull
    option forwardfor
    server seahubserver01 192.168.1.165:80 check port 11001
    server seahubserver02 192.168.1.200:80 check port 11001

listen seahub-https 0.0.0.0:443
    mode tcp
    option tcplog
    option dontlognull
    server seahubserver01 192.168.1.165:443 check port 11001
    server seahubserver02 192.168.1.200:443 check port 11001

listen ccnetserver :10001
    mode tcp
    option tcplog
    balance leastconn
    server seafserver01 192.168.1.165:10001 check port 11001
    server seafserver02 192.168.1.200:10001 check port 11001

listen seafserver :12001
    mode tcp
    option tcplog
    balance leastconn
    server seafserver01 192.168.1.165:12001 check port 11001
    server seafserver02 192.168.1.200:12001 check port 11001
```

See how it runs

Now you should be able to test your cluster. Open https://seafile.example.com in your browser and enjoy. You can also sync file with Seafile clients.

If the above works, the next step would be Enable search and background tasks in a cluster.

Enable search and background tasks in a cluster

In the seafile cluster, only one server should run the background tasks, including:

- indexing files for search
- email notification

You need to choose one node to run the background tasks.

Let's assume you have three nodes in your cluster, namely A, B, and C, and you decide that:

- Node A would run the background tasks
- Node B and Node C are normal nodes

Configuring Node A (the background-tasks node)

On this node, you need:

Install Java

On Ubuntu/Debian:

```
sudo apt-get install openjdk-7-jre
```

On CentOS/Red Hat:

```
sudo yum install java-1.7.0-openjdk
```

提示：您也可以使用 Oracle JRE.

注意：Seafile 专业版需要 java 1.7 以上版本, 请用 `java -version` 命令查看您系统中的默认 java 版本. 如果不是 java 7, 那么, 请 更新默认 java 版本.

Edit pro-data/seafevents.conf

REMOVE the line:

```
external_es_server = true
```

Edit the firewall rules

In your firewall rules for node A, you should open the port 9500 (for search requests).

Configure Other Nodes

On nodes B and C, you need to:

- Edit pro-data/seafevents.conf, add the following lines:

  ```
  [INDEX FILES]
  external_es_server = true
  es_host = <ip of node A>
  es_port = 9500
  ```

Start the background tasks

On node A (the background tasks node), you can star/stop background tasks by:

```
./seafile-background-tasks.sh { start | stop | restart }
```

Setup Seafile cluster with NFS

In a Seafile cluster, one common way to share data among the Seafile server instances is to us NFS. You should only share the files objects on NFS. Here we'll provide a tutorial about how and what to share.

How to setup nfs server and client is beyond the scope of this wiki. Here are few references:

- Ubuntu: https://help.ubuntu.com/community/SettingUpNFSHowTo
- CentOS: http://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-nfs.html

Supposed your seafile server installation directory is `/data/haiwen` , after you run the setup script there should be a `seafile-data` directory in it. And supposed you mount the NFS drive on `/seafile-nfs` , you should follow a few steps:

- Move the `seafile-data` folder to `/seafile-nfs` :

```
mv /data/haiwen/seafile-data /seafile-nfs/
```

- On every node in the cluster, make a symbolic link to the shared seafile-data folder

```
cd /data/haiwen
ln -s /seafile-nfs/seafile-data /data/haiwen/seafile-data
```

This way the instances will share the same `seafile-data` folder. All other config files and log files will remain independent.

- 关于搜索功能的 FAQ
- Office/PDF 文档预览 FAQ

关于搜索功能的 FAQ

- 无论我怎样尝试，加密资料库中的文件都不会出现在搜索结果中

  这是因为服务器不能索引加密文件，因为它们是加密的。

- 我从社区版服务器切换到专业版服务器后，无论我搜索什么，都不会得到结果

默认情况下，搜索索引每 10 分钟更新一次。所以，在第一次索引更新前，无论您搜索什么都不会返回结果。

为了使搜索能够立即生效，您可以手动更新索引：

- 确保您已经启动了 Seafile 服务器
- 手动更新搜索索引：

  ```
   cd haiwen/seafile-pro-server-2.1.5
  ./pro/pro.py search --update
  ```

  如果您的文件很多，这个过程会花费很长一段时间。

- 我想启用对 office/pdf 文档的全文搜索功能，所以我在配置文件中将 `index_office_pdf` 的值设置为 `true`，但它没起作用。

  在这种情况下，您需要做以下几步：

  1. 编辑 `/data/haiwen/pro-data/seafevents.conf` 文件，将 `index_office_pdf` 的值设置为 `true`
  2. 重启 Seafile 服务器：

     ```
     cd /data/haiwen/seafile-pro-server-2.1.5
     ./seafile.sh restart
     ```

  3. 删除已经存在的搜索索引：

     ```
     ./pro/pro.py search --clear
     ```

  4. 创建并且再次更新搜索索引：

     ```
     ./pro/pro.py search --update
     ```

Office/PDF 文档预览 FAQ

怎么修改可预览最大文件大小和页面数?

在 `/data/haiwen/pro-data/seafevents.conf` 中的 `OFFICE CONVERTER` 配置部分添加配置选项

```
# the max size of documents to allow to be previewed online, in MB. Default is 2 MB
max-size = 2
# how many pages are allowed to be previewed online. Default is 50 pages
max-pages = 50
```

然后重启 Seafile 服务

```
cd /data/haiwen/seafile-pro-server-1.7.0/
./seafile.sh restart
./seahub.sh restart
```

```
Seafile Professional Edition
SOFTWARE LICENSE AGREEMENT

NOTICE: READ THE FOLLOWING TERMS AND CONDITIONS CAREFULLY BEFORE YOU DOWNLOAD, INSTALL OR

1. DEFINITIONS
"Seafile, Inc" means Seafile, Inc

"You and Your" means the party licensing the Software hereunder.

"Software" means the computer programs provided under the terms of this license by Seafil

2. GRANT OF RIGHTS
2.1 General
The License granted for Software under this Agreement authorizes You on a non-exclusive b

2.2 License Provisions
Subject to the receipt by Seafile, Inc of the applicable license fees, You have the right

* You may use and install the Software on an unlimited number of computers that are owned
* Nothing in this Agreement shall permit you, or any third party to disclose or otherwise
* You agree to indemnify, hold harmless and defend Seafile, Inc from and against any clai
* You do not permit further redistribution of the Software by Your end-user customers

5. NO DERIVATIVE WORKS
The inclusion of source code with the License is explicitly not for your use to customize

6. OWNERSHIP
You acknowledge that all copies of the Software in any form are the sole property of Seaf

7. CONFIDENTIALITY
You hereby acknowledge and agreed that the Software constitute and contain valuable propr

8. DISCLAIMER OF WARRANTIES
EXCEPT AS OTHERWISE SET FORTH IN THIS AGREEMENT THE SOFTWARE IS PROVIDED TO YOU "AS IS",

9. LIMITATION OF LIABILITY
YOU ACKNOWLEDGE AND AGREE THAT THE CONSIDERATION WHICH Seafile, Inc IS CHARGING HEREUNDER

10. INDEMNIFICATION
You agree to defend, indemnify and hold Seafile, Inc and its employees, agents, represent

11. TERMINATION
Your license is effective until terminated. You may terminate it at any time by destroyin

12. UPDATES AND SUPPORT
Seafile, Inc has the right, but no obligation, to periodically update the Software, at it

13. TAXES AND OTHER CHARGES
You are responsible for paying all sales, use, excise valuated or other taxes or governme

YOU HEREBY ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT AND AGREE TO BE B
```

服务器个性化配置

配置文件

本章节用来说明如果个性化配置服务器。

在开源版中由以下三个配置文件:

- ccnet/ccnet.conf: 用来配置网络
- seafile-data/seafile.conf: 用来配置 Seafile 与 FileServer.
- seahub_settings.py: 用来配置 Seahub

专业版中还包含以下一个配置文件:

- `pro-data/seafevents.conf` : 包含搜索与文件预览的配置信息

配置

邮件:

- 发送邮件提醒
- 个性化邮件提醒

用户管理

- 用户管理

用户存储容量和上传/下载文件大小限制

- 存储容量与文件上传/下载大小限制

自定义 Web

- 自定义 Web

ccnet.conf 配置

通过修改`ccnet/ccnet.conf`文件，可以对 Seafile 的网络选项进行设置，示例如下:

```
[General]

# 该设置不再使用
USER_NAME=example

# 请不要改变这个 ID.
ID=eb812fd276432eff33bcdde7506f896eb4769da0

# Seafile 服务器名称, 目前只出现在客户端日志中。
NAME=example

# Seahub(Seafile Web) 外部 URL，如果改值没有设对，会影响文件的上传下载。
# 注意: 外部 URL 意味着"如果你使用 Nginx, 请使用 Nginx 对外的 URL"
SERVICE_URL=http://www.example.com:8000


[Network]

# 该设置不再使用
PORT=10001

[Client]
# 该设置不再使用
PORT=13419
```

注意: 为使更改生效，请重启 Seafile.

```
cd seafile-server
./seafile.sh restart
```

seafile.conf 配置

存储空间容量设置(seafile.conf)

如果你想向所有用户分配存储空间(e.g. 2GB)时 . 你可以在 `seafile-data/seafile.conf` 文件中增加以下语句

```
[quota]
# 单位为 Gb, 请使用数字
default = 2
```

这个设置对所有用户生效. 如果你想对某一特定用户进行容量分配, 请以管理员身份登陆 Seahub 网站, 在**System Admin**页面中进行设置.

默认历史记录设置(seafile.conf)

如果你不想存储所有的文件修改历史, 可以对所有的资料库，设置一个默认的文件修改历史记录。

```
[history]
keep_days = days of history to keep
```

Seafile fileserver 配置(seafile.conf)

可通过 `seafile-data/seafile.conf` 的 `[fileserver]` (3.1 版之前用 `[httpserver]`) 部分对 Seafile fileserver 进行配置

```
[fileserver]
# fileserver 的 tcp 端口
port = 8082
```

更改上传/下载设置.

```
[fileserver]
# 上传文件最大为200M.
max_upload_size=200

# 最大下载目录限制为200M.
max_download_dir_size=200
```

注意: 请重启 Seafile和 Seahub以使修改生效.

```
./seahub.sh restart
```

```
./seafile.sh restart
```

Seahub 配置

Seahub 下发送邮件提醒

邮件提醒会使某些功能有更好的用户体验, 比如发送邮件提醒用户新消息到达. 请在 `seahub_settings.py` 中加入以下语句以安装邮件提醒功能 (同时需要对你的邮箱进行设置).

```
EMAIL_USE_TLS = False
EMAIL_HOST = 'smtp.domain.com'        # smpt 服务器
EMAIL_HOST_USER = 'username@domain.com'    # 用户名和域名
EMAIL_HOST_PASSWORD = 'password'    # 密码
EMAIL_PORT = '25'
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER
SERVER_EMAIL = EMAIL_HOST_USER
```

Gmail 用户请加入以下语句:

```
EMAIL_USE_TLS = True
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_HOST_USER = 'username@gmail.com'
EMAIL_HOST_PASSWORD = 'password'
EMAIL_PORT = 587
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER
SERVER_EMAIL = EMAIL_HOST_USER
```

注意**1**:如果邮件功能不能正常使用，请在 `logs/seahub.log` 日志文件中查看问题原因. 更多信息请见 Email notification list.

注意**2**: 如果你想在非用户验证情况下使用邮件服务，请将 `EMAIL_HOST_PASSWORD` 置为**blank** (`''`).

缓存

Seahub 在默认文件系统(/tmp/seahub_cache/)中缓存文件(avatars, profiles, etc) . 你可以通过 Memcached 进行缓存操作 (前提是你已经安装了 `python-memcache` 模块).

```
CACHES = {
    'default': {
        'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
        'LOCATION': '127.0.0.1:11211',
    }
}
```

Seahub 设置

通过修改 `seahub_settings.py` 文件，可以对 Seahub 网站进行更改.

```
# 时区设置, 更多请见:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# 部分操作系统下有效.
# 如果是 Windows 用户, 请设置为你的系统时区.
TIME_ZONE = 'UTC'

# 语言选项, 系统默认语言以及发邮件默认语言。
# 参考：http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = 'en'

# Seahub 网站 URL. 邮件提醒中会包含此地址
SITE_BASE = 'http://www.example.com/'

# 网站名称. 邮件提醒中会包含此名称.
SITE_NAME = 'example.com'

# 网站标题
SITE_TITLE = 'Seafile'

# 若果不想再根路径上运行 Seahub 网站, 请更改此设置.
# e.g. setting it to '/seahub/' would run seahub on http://example.com/seahub/.
SITE_ROOT = '/'

# 是否使用 pdf.js 在线查看 pdf 文件. 默认为 `True`.
# NOTE: 1.4版本后可用.
USE_PDFJS = True

# 是否在网站页面显示注册按钮, 默认为 `False`.
# NOTE: 1.4版本后可用.
ENABLE_SIGNUP = False

# 用户注册后是否立刻激活, 默认为 `True`.
# 如设置为 `False`, 需管理员手动激活.
# NOTE: 1.8版本后可用
ACTIVATE_AFTER_REGISTRATION = False

# 管理员新增用户后是否给用户发送邮件. 默认为 `True`.
# NOTE: 1.4版本后可用.
SEND_EMAIL_ON_ADDING_SYSTEM_MEMBER = True

 # 管理员重置用户密码后是否给用户发送邮件. 默认为 `True`.
# NOTE: 1.4版本后可用.
SEND_EMAIL_ON_RESETTING_USER_PASSWD = True

# 隐藏 `Organization` 标签 .
# 如果你希望你的私人 Seafile 像 https://cloud.seafile.com/ 一样运行,  请设置.
CLOUD_MODE = True

# 在线查看文件大小限制, 默认为 30M.
FILE_PREVIEW_MAX_SIZE = 30 * 1024 * 1024

# cookie的保存时限, (默认为 2 周).
SESSION_COOKIE_AGE = 60 * 60 * 24 * 7 * 2

# 是否存储每次请求的会话数据.
SESSION_SAVE_EVERY_REQUEST = False

# 浏览器关闭后, 是否清空用户会话 cookie
SESSION_EXPIRE_AT_BROWSER_CLOSE = False
```

```
# 是否可以把一个群组设为公开.
ENABLE_MAKE_GROUP_PUBLIC = False

# 登录记住天数. 默认 7 天
LOGIN_REMEMBER_DAYS = 7
```

注意:

- 请重启 Seahub 以使更改生效.
- 如果更改没有生效，请删除`seahub_setting.pyc`缓存文件.

```
./seahub.sh restart
```

发送邮件提醒

邮件提醒会使某些功能有更好的用户体验, 比如发送邮件提醒用户新消息到达. 请在 `seahub_settings.py` 中加入以下语句以安装邮件提醒功能 (同时需要对你的邮箱进行设置).

```
EMAIL_USE_TLS = False
EMAIL_HOST = 'smtp.domain.com'        # smpt 服务器
EMAIL_HOST_USER = 'username@domain.com'    # 用户名和域名
EMAIL_HOST_PASSWORD = 'password'    # 密码
EMAIL_PORT = '25'
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER
SERVER_EMAIL = EMAIL_HOST_USER
```

Gmail 用户请加入以下语句:

```
EMAIL_USE_TLS = True
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_HOST_USER = 'username@gmail.com'
EMAIL_HOST_PASSWORD = 'password'
EMAIL_PORT = 587
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER
SERVER_EMAIL = EMAIL_HOST_USER
```

注意**1**:如果邮件功能不能正常使用，请在 `logs/seahub.log` 日志文件中查看问题原因. 更多信息请见 Email notification list.

注意**2**: 如果你想在非用户验证情况下使用邮件服务，请将 `EMAIL_HOST_USER` 和 `EMAIL_HOST_PASSWORD` 置为**blank** (`''`). (但是注意一点，这种情况下，邮件将不会记录发件人 `From:` 信息.)

注意**3**:

- 请重启 Seahub 以使更改生效.
- 如果更改没有生效，请删除 `seahub_setting.pyc` 缓存文件.

```
./seahub.sh restart
```

个性化邮件提醒

注意: 不同版本之间有所差异，本文档基于 2.0.1 版本编写。请按提示，自行更改相应代码，以实现个性化功能。重启 Seahub 以使更改生效。

用户重置密码

Subject

seahub/seahub/auth/forms.py line:103

Body

seahub/seahub/templates/registration/password_reset_email.html

管理员添加新用户

Subject

seahub/seahub/views/sysadmin.py line:424

Body

seahub/seahub/templates/sysadmin/user_add_email.html

管理员重置用户密码

Subject

seahub/seahub/views/sysadmin.py line:368

Body

seahub/seahub/templates/sysadmin/user_reset_email.html

用户发送文件/文件夹外链

Subject

seahub/seahub/share/views.py line:668

Body

seahub/seahub/templates/shared_link_email.html

用户管理

在 `seahub_settings.py` 配置文件中可以个性化用户管理

```
# 是否开启用户注册功能，默认为 `False`，不开启.
ENABLE_SIGNUP = False

# 用户注册后是否立即激活，默认为 `True`，立即激活。
# 如果设置为 `False`，需管理员在系统管理界面激活用户。
ACTIVATE_AFTER_REGISTRATION = False

# 管理员添加新用户后，是否给此用户发送邮件提醒。默认为 `True`，发送邮件提醒。
# 此功能只支持 1.4 及之后版本。
SEND_EMAIL_ON_ADDING_SYSTEM_MEMBER = True

# 用户登录时，输入几次错误后，显示验证码。
LOGIN_ATTEMPT_LIMIT = 3

# 关闭浏览器后，是否保存 Session Cookie，默认为 `False`，不保存。
SESSION_EXPIRE_AT_BROWSER_CLOSE = False
```

存储容量与文件上传/下载大小限制

存储容量

可以通过在 `seafile-data/seafile.conf` 文件中增加以下语句，来为所有用户设置默认存储容量（比如，2GB）。

```
[quota]
# 用户存储容量，单位默认为 GB，要求为整数。
default = 2
```

此设置对所有用户有效，如果想为某一用户单独设置，请在管理员界面更改。

文件修改历史保存期限 (seafile.conf)

如果你不想保存所有的文件修改历史，可以在 `seafile-data/seafile.conf` 中设置:

```
[history]
# 文件修改历史保存期限（单位为“天”）
keep_days = 10
```

文件上传/下载大小限制

在 `seafile-data/seafile.conf` 中:

```
[fileserver]
# 设置最大上传文件为 200M.
max_upload_size=200

# 设置最大下载文件/目录为 200M.
max_download_dir_size=200
```

个性化 Seahub

个性化 Logo 及 CSS 样式

假设你目前在使用 2.1.0 版本。 在 `seafile-server-2.1.0/seahub/media` 下新建 `custom` 。将所有的个性化文件放到这个文件夹下。 当你升级到 2.1.1 版本的时候，升级脚本会自动的将此文件夹复制到 `seafile-server-2.1.1/seahub/media` 下。

自定义 Logo

1. 将 Logo 文件放在 `seahub/media/custom/` 文件夹下
2. 在 `seahub_settings.py` 中，重新定义 `LOGO_PATH` 的值。

```
LOGO_PATH = 'custom/mylogo.png'
```

3. 在 `seahub_settings.py` 中，重新定义 `LOGO_URL` 的值。

```
LOGO_URL = 'http://your-seafile.com'
```

自定义 Seahub CSS 样式

1. 在 `seahub/media/custom/` 中新建 CSS 文件，比如： `custom.css` 。
2. 在 `seahub_settings.py` 中，重新定义 `BRANDING_CSS` 的值。

```
BRANDING_CSS = 'custom/custom.css'
```

个性化 Seahub 页面

注意: 仅支持 2.1 及之后的版本

在 `<seafile-install-path>/seahub-data/custom` 目录下，新建 `templates` 文件夹。

个性化“页脚”页面

1. 复制 `seahub/seahub/templates/footer.html` 到 `seahub-data/custom/templates` 。
2. 自行编写 `footer.html` 。

个性化“下载”页面

1. 复制 `seahub/seahub/templates/download.html` 到 `seahub-data/custom/templates` 。
2. 自行编写 `download.html` 。

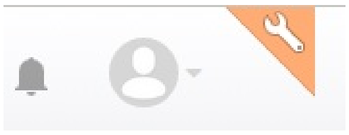
个性化“帮助”页面

1. 复制 `seahub/seahub/help/templates/help.html` 到 `seahub-data/custom/templates` 。
2. 自行编写 `help.html` 。

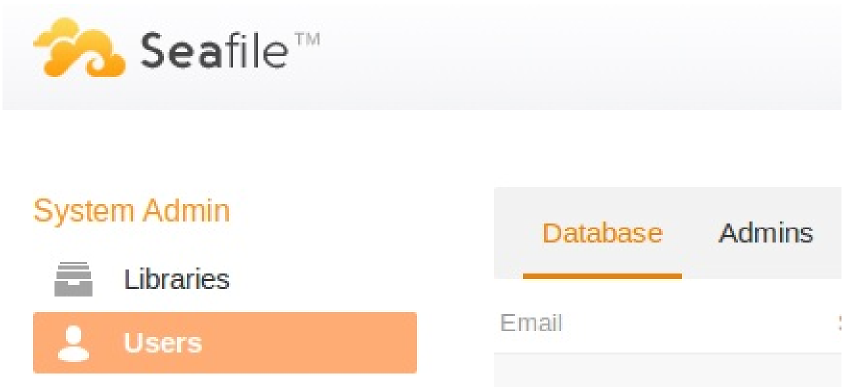
管理员

进入管理界面

作为系统管理员，你可以通过点击网页左上侧的 `tools` 按钮（在个人头像旁)进入管理界面：



点击完 `tools` 按钮，你便进入管理界面：



账号管理

- 账号管理

日志

- 日志文件位置

备份和恢复

备份和恢复:

- 备份和恢复

服务器强制关闭或系统坏掉后，恢复损坏的文件：

- Seafile FSCK

你可以运行Seafile GC来删除无用的文件:

- Seafile GC

账号管理

用户管理

当你部署好 Seahub 网站，你应该已经创建好一个管理员账号。用管理员账号登陆后，便可添加、删除，用户、资料库等。

重置用户密码

在 **System Admin** 页面，管理员可以重置用户密码。

对于私有服务器，默认设置不支持用户通过邮箱来重置密码。如果你想采用这种方式，你必须首先设置邮件通知。

如果忘记管理员账号或密码如何处理？

你可以进入 *seafile-server* 目录，运行 `reset-admin.sh` 脚本。这个脚本可以帮助你重置管理员账号和密码。

日志

Seafile 服务器有如下日志文件:

- Ccnet Log: logs/ccnet.log
- Seafile server： logs/seafile.log
- FileServer: logs/http.log
- Controller: logs/controller.log
- Seahub：logs/seahub_django_request.log, logs/seahub.log

概述

一般来说，Seafile 备份分为两部分内容：

- Seafile 资料库数据
- 数据库

如果你根据我们的手册来安装 Seafile 服务器，你应该有如下目录结构：

```
haiwen       # 根目录，haiwen 为示例文件名，如果你安装到其他目录则为相应的目录名
  --seafile-server-2.x.x # Seafile 安装包解压缩后目录
  --seafile-data   # Seafile 配置文件和数据（如果你选择默认方式）
  --seahub-data    # Seahub 数据
  --ccnet          # Ccnet 配置文件和数据
  --seahub.db      # Seahub 用到的 sqlite3 数据库文件
  --seahub_settings.py # seahub可选属性配置文件
```

你所有的资料库数据都存储在 **haiwen** 目录。

Seafile 也在数据库中存储一些重要的元数据。数据库的命名和存储路径取决于你所使用的数据库。

对于 SQLite, 数据库文件也存储在 **haiwen** 目录。相应的数据文件如下:

- ccnet/PeerMgr/usermgr.db: 包含用户信息
- ccnet/GroupMgr/groupmgr.db: 包含群组信息
- seafile-data/seafile.db: 包含资料库元数据信息
- seahub.db: 包含网站前端（Seahub）所用到的数据库表信息

对于 MySQL, 数据库由管理员来创建，所以不同的人部署，可能会有不同的文件名。大体而言，有如下三个数据库会被创建:

- ccnet-db: 包含用户和群组信息
- seafile-db: 包含资料库元数据信息
- seahub.db: 包含网站前端（seahub）所用到的数据库表信息

备份步骤

备份需要如下三步：

1. 可选步: 如果你选择 SQLite 作为数据库，首先停掉 Seafile 服务器；
2. 备份数据库；
3. 备份存放 Seafile 数据的目录；

我们假设你的 Seafile 数据位于 `/data/haiwen` 目录下，并且你想将其备份到 `/backup` 目录（`/backup` 目录可以是 NFS（网络文件系统），可以是另一台机器的 Windows 共享，或者是外部磁盘）。请在 `/backup` 目录下创建如下目录结构：

```
/backup
---- databases/  包含数据库备份
---- data/  包含 Seafile 数据备份
```

备份数据库

我们建议你每次将数据库备份到另一个单独文件，并且不要覆盖最近一周来备份过的旧数据库文件。

MySQL

假设你的数据库名分别为 `ccnet-db`，`seafile-db` 和 `seahub-db`。 `mysqldump` 会自动锁住表，所以在你备份 MySql 数据库的时候，不需要停掉 Seafile 服务器。通常因为数据库表非常小，所以执行以下命令备份不会花太长时间。

```
mysqldump -h [mysqlhost] -u[username] -p[password] --opt ccnet-db > /backup/databases/ccn

mysqldump -h [mysqlhost] -u[username] -p[password] --opt seafile-db > /backup/databases/s

mysqldump -h [mysqlhost] -u[username] -p[password] --opt seahub-db > /backup/databases/se
```

SQLite

对于 SQLite 数据库，在备份前你需要停掉 Seafile 服务器。

```
sqlite3 /data/haiwen/ccnet/GroupMgr/groupmgr.db .dump > /backup/databases/groupmgr.db.bak

sqlite3 /data/haiwen/ccnet/PeerMgr/usermgr.db .dump > /backup/databases/usermgr.db.bak.`d

sqlite3 /data/haiwen/seafile-data/seafile.db .dump > /backup/databases/seafile.db.bak.`da

sqlite3 /data/haiwen/seahub.db .dump > /backup/databases/seahub.db.bak.`date +"%Y-%m-%d-%
```

备份 Seafile 资料库数据

由于所有的数据文件都存储在 `/data/haiwen` 目录，备份整个目录即可。你可以直接拷贝整个目录到备份目录，或者你也可以用 rsync 做增量备份。

直接拷贝整个数据目录，

```
cp -R /data/haiwen /backup/data/haiwen-`date +"%Y-%m-%d-%H-%M-%S"`
```

这样每次都会产生一个新的备份文件夹，完成后，可以删掉旧的备份。

如果你有很多数据，拷贝整个数据目录会花很多时间，这时你可以用rsync做增量备份。

```
rsync -az /data/haiwen /backup/data
```

这个命令数据备份到 `/backup/data/haiwen` 下。

让拷贝和 **rsync** 过程成功结束是非常重要的，否则你最近的一些数据将会丢失。

恢复备份

如果你当前的 Seafile 服务器已经坏掉，将使用另一台机器来提供服务，需要恢复数据:

1. 假设在新机器中，Seafile 也被部署在了 `/data/haiwen` 目录中，拷贝 `/backup/data/haiwen` 到新机器中即可。
2. 恢复数据库。

恢复数据库

现在你已经拥有了数据库备份文件，你可以按如下步骤来进行恢复。

MySQL

```
mysql -u[username] -p[password] ccnet-db < ccnet-db.sql.2013-10-19-16-00-05
mysql -u[username] -p[password] seafile-db < seafile-db.sql.2013-10-19-16-00-20
mysql -u[username] -p[password] seahub-db.sql.2013-10-19-16-01-05
```

SQLite

```
cd /data/haiwen
mv ccnet/PeerMgr/usermgr.db ccnet/PeerMgr/usermgr.db.old
mv ccnet/GroupMgr/groupmgr.db ccnet/GroupMgr/groupmgr.db.old
mv seafile-data/seafile.db seafile-data/seafile.db.old
mv seahub.db seahub.db.old
sqlite3 ccnet/PeerMgr/usermgr.db < usermgr.db.bak.xxxx
sqlite3 ccnet/GroupMgr/groupmgr.db < groupmgr.db.bak.xxxx
sqlite3 seafile-data/seafile.db < seafile.db.bak.xxxx
sqlite3 seahub.db < seahub.db.bak.xxxx
```

Seafile FSCK

在服务器端，Seafile 通过一种内部格式将文件存储在资料库中。Seafile 对于文件和目录有其独有的保存方式（类似于Git）。

默认安装下，这些内部对象，会被直接存储在服务器的文件系统中（例如 Ext4，NTFS）。由于大多数文件系统，不能在服务器非正常关闭或系统崩溃后，保证文件内容的完整性。所以，如果当系统崩溃时，正在有新的内部对象被写入，那么当系统重启时，这些文件就会被损坏，相应的资料库也无法使用。

注意: 如果你把 seafile-data 目录存储在有后备电源的 NAS（例如 EMC 或 NetApp）系统中，或者使用 S3 作为专业版的服务器，内部对象不会被损坏。

Seafile 服务器包含了 `seafile-fsck` 工具来帮助你恢复这些毁坏的对象（类似于git-fsck工具）。这个工具将会进行如下两项工作：

1. 检查 Seafile 内部对象的完整性，并且删除毁坏的对象。
2. 恢复所有受影响的资料库到最近一致，可用的状态。

执行流程如下所示：

```
cd seafile-server-latest
./seaf-fsck.sh [--repair|-r] [--enable-sync|-e] [repo_id_1 [repo_id_2 ...]]
```

seaf-fsck 有检查资料库完整性和修复损坏资料库两种运行模式。

检查资料库完整性

执行 seaf-fsck.sh 不加任何参数将以只读方式检查所有资料库的完整性。

```
cd seafile-server-latest
./seaf-fsck.sh
```

如果你想检查指定资料库的完整性，只需将要检查的资料库 ID 作为参数即可：

```
cd seafile-server-latest
./seaf-fsck.sh [library-id1] [library-id2] ...
```

运行输出如下:

```
[02/13/15 16:21:07] fsck.c(470): Running fsck for repo ca1a860d-e1c1-4a52-8123-0bf9def869
[02/13/15 16:21:07] fsck.c(413): Checking file system integrity of repo fsck(ca1a860d)...
[02/13/15 16:21:07] fsck.c(35): Dir 9c09d937397b51e1283d68ee7590cd9ce01fe4c9 is missing.
[02/13/15 16:21:07] fsck.c(200): Dir /bf/pk/(9c09d937) is curropted.
[02/13/15 16:21:07] fsck.c(105): Block 36e3dd8757edeb97758b3b4d8530a4a8a045d3cb is corrup
[02/13/15 16:21:07] fsck.c(178): File /bf/02.1.md(ef37e350) is curropted.
```

```
[02/13/15 16:21:07] fsck.c(85): Block 650fb22495b0b199cff0f1e1ebf036e548fcb95a is missing
[02/13/15 16:21:07] fsck.c(178): File /01.2.md(4a73621f) is curropted.
[02/13/15 16:21:07] fsck.c(514): Fsck finished for repo ca1a860d.
```

被损坏的文件和目录将显示在输出的结果中。

有时，你会看到如下的输出结果：

```
[02/13/15 16:36:11] Commit 6259251e2b0dd9a8e99925ae6199cbf4c134ec10 is missing
[02/13/15 16:36:11] fsck.c(476): Repo ca1a860d HEAD commit is corrupted, need to restore
[02/13/15 16:36:11] fsck.c(314): Scanning available commits...
[02/13/15 16:36:11] fsck.c(376): Find available commit 1b26b13c(created at 2015-02-13 16:
```

这意味着记录在数据库中的 "head commit" （当前资料库状态的标识）与数据目录中的记录不一致。这种情况下，fsck 会试着找出最近可用的一致状态，并检查其完整性。

建议: 如果你有很多资料库要检查，保存 **fsck** 的输出到日志文件中将有助于后面的进一步分析。

修复损坏的资料库

fsck 修复损坏的资料库有如下两步流程:

1. 如果记录在数据库中的资料库当前状态无法在数据目录中找出，fsck 将会在数据目录中找到最近可用状态。
2. 检查第一步中可用状态的完整性。如果文件或目录损坏，fsck 将会将其置空并报告损坏的路径，用户便可根据损坏的路径来进行恢复操作。

执行如下命令来修复所有资料库：

```
cd seafile-server-latest
./seaf-fsck.sh --repair
```

大多数情况下我们建议你首先以只读方式检查资料库的完整性，找出损坏的资料库后，执行如下命令来修复指定的资料库：

```
cd seafile-server-latest
./seaf-fsck.sh --repair [library-id1] [library-id2] ...
```

由于被损坏的文件或目录在修复过后会被置空，在客户端同步被修复的损坏资料库将会导致数据丢失，即客户端先前好的完整的文件或目录拷贝将被空文件或空目录替代。为了避免这种情况发生，服务器将会阻止客户端同步被修复的损坏资料库。系统管理员应该通知用户来恢复损坏的文件或目录，然后执行如下命令让此资料库可以再次同步：

```
cd seafile-server-latest
```

```
./seaf-fsck.sh --enable-sync [library-id1] [library-id2] ...
```

Seafile GC

Seafile 利用存储去重技术来减少存储资源的利用。 简单来说，这包含如下两层含义：

- 不同版本的文件或许会共享一些数据块。
- 不同的资料库也或许会共享一些数据块。

运用这项技术之后，在你删除一个资料库时，会导致底层数据块不会被立即删除，因此 Seafile 服务器端没用的数据块将会增多。

通过运行垃圾回收程序，可以清理无用的数据块，释放无用数据块所占用的存储空间。

垃圾回收程序将会清理如下两种无用数据块：

1. 未被资料库所引用的数据块即数据块属于被删除的资料库。
2. 设置了历史长度限制的资料库的过期数据块。

如果使用社区版服务器，运行垃圾回收程序之前，请先在服务器端停掉 **Seafile** 程序。这是因为垃圾回收程序，会错误的删除刚刚写入 **Seafile** 的新的数据块。对于专业版，**3.1.11** 及之后的版本，支持在线垃圾回收即如果使用 **MySQL** 或 **PostgreSQL** 数据库，你不需要暂停 **Seafile** 程序来进行垃圾回收。

4.1.1 及之后的版本

从社区版 4.1.1 和 专业版 4.1.0开始， 我们改善了垃圾回收的命令参数和执行结果输出。

Dry-run 模式

如果仅为了查看有多少垃圾可以回收而不进行删除操作，用 dry-run 选项：

```
seaf-gc.sh --dry-run [repo-id1] [repo-id2] ...
```

运行输出如下所示:

```
[03/19/15 19:41:49] seafserv-gc.c(115): GC version 1 repo My Library(ffa57d93)
[03/19/15 19:41:49] gc-core.c(394): GC started. Total block number is 265.
[03/19/15 19:41:49] gc-core.c(75): GC index size is 1024 Byte.
[03/19/15 19:41:49] gc-core.c(408): Populating index.
[03/19/15 19:41:49] gc-core.c(262): Populating index for repo ffa57d93.
[03/19/15 19:41:49] gc-core.c(308): Traversed 5 commits, 265 blocks.
[03/19/15 19:41:49] gc-core.c(440): Scanning unused blocks.
[03/19/15 19:41:49] gc-core.c(472): GC finished. 265 blocks total, about 265 reachable bl

[03/19/15 19:41:49] seafserv-gc.c(115): GC version 1 repo aa(f3d0a8d0)
[03/19/15 19:41:49] gc-core.c(394): GC started. Total block number is 5.
[03/19/15 19:41:49] gc-core.c(75): GC index size is 1024 Byte.
[03/19/15 19:41:49] gc-core.c(408): Populating index.
[03/19/15 19:41:49] gc-core.c(262): Populating index for repo f3d0a8d0.
[03/19/15 19:41:49] gc-core.c(308): Traversed 8 commits, 5 blocks.
```

```
[03/19/15 19:41:49] gc-core.c(264): Populating index for sub-repo 9217622a.
[03/19/15 19:41:49] gc-core.c(308): Traversed 4 commits, 4 blocks.
[03/19/15 19:41:49] gc-core.c(440): Scanning unused blocks.
[03/19/15 19:41:49] gc-core.c(472): GC finished. 5 blocks total, about 9 reachable blocks

[03/19/15 19:41:49] seafserv-gc.c(115): GC version 1 repo test2(e7d26d93)
[03/19/15 19:41:49] gc-core.c(394): GC started. Total block number is 507.
[03/19/15 19:41:49] gc-core.c(75): GC index size is 1024 Byte.
[03/19/15 19:41:49] gc-core.c(408): Populating index.
[03/19/15 19:41:49] gc-core.c(262): Populating index for repo e7d26d93.
[03/19/15 19:41:49] gc-core.c(308): Traversed 577 commits, 507 blocks.
[03/19/15 19:41:49] gc-core.c(440): Scanning unused blocks.
[03/19/15 19:41:49] gc-core.c(472): GC finished. 507 blocks total, about 507 reachable bl

[03/19/15 19:41:50] seafserv-gc.c(124): === Repos deleted by users ===
[03/19/15 19:41:50] seafserv-gc.c(145): === GC is finished ===

[03/19/15 19:41:50] Following repos have blocks to be removed:
repo-id1
repo-id2
repo-id3
```

如果在参数中指定资料库 ID，则程序只检查指定的资料库，否则所有的资料库将会被检查。

在程序输出的结尾，你会看到 "repos have blocks to be removed" 部分，这部分内容会列出含有可回收垃圾块的资料库的 ID，后续你可以运行程序不加 --dry-run 选项来回收这些资料库的垃圾数据块。

删除垃圾数据块

运行垃圾回收程序，不加 --dry-run 选项来删除垃圾数据块：

```
seaf-gc.sh [repo-id1] [repo-id2] ...
```

如果在参数中指定资料库 ID, 则程序只检查和删除指定的资料库。

正如前面所说，有两种类型的垃圾数据块可被回收，有时仅删除第一类无用数据块（属于删除的资料库）便可达到回收的目的，这种情况下，垃圾回收程序将不会检查未被删除的资料库，加入 “-r” 选项便可实现这个功能：

```
seaf-gc.sh -r
```

Seafile 4.1.1 及之后的版本，被用户删除的资料库不会直接从系统中删除，它们会被转移到系统管理员界面的垃圾箱。垃圾箱中的资料库，只有在从垃圾箱中清除以后，它们的数据块才可被回收。

3.1.2 及之后版本

运行垃圾回收程序

```
./seaf-gc.sh run
```

程序结束之后，运行以下命令，检查是否误删了还在使用的数据块，如果误删，会显示警告信息。

```
./seaf-gc.sh verify
```

可以通过 `dry-run` 选项，设置在运行垃圾回收程序前，进行完整性检查

程序将会显示 所有的数据块数量 和 将要被删除的数据块数量

```
./seaf-gc.sh dry-run
```

如果资料库已损坏，因为无法判断数据块是否还在被其他资料库使用，所以垃圾回收程序将会停止运行。

可以通过 `force` 选项，强制删除已损坏资料库的数据。通过将已损坏资料库的数据块标记为“未使用”，来将其删除。

```
./seaf-gc.sh force
```

3.1.2 及之前版本

运行垃圾回收程序

```
cd seafile-server-{version}/seafile
export LD_LIBRARY_PATH=./lib:${LD_LIBRARY_PATH}
./bin/seafserv-gc -c ../../ccnet -d ../../seafile-data
```

如果你源码编译安装 Seafile 服务器，仅仅运行

```
seafserv-gc -c ../../ccnet -d ../../seafile-data
```

当垃圾回收程序结束后，你也可以检查是否一些有用的数据块被错误的删除：

```
seafserv-gc -c ../../ccnet -d ../../seafile-data --verify
```

如果一些有用的数据块丢失，它将会打印一些警告信息。

如果你想在真正删除一些数据块之前，做一些常规检查，可以使用--dry-run选项

```
seafserv-gc -c ../../ccnet -d ../../seafile-data --dry-run
```

这将会向你展示数据块总数量和将被删除数据块数量。

如果在服务器端一些库的元数据被毁坏，垃圾回收程序将会停止处理，因为它无法识别是否一个数据块被一些毁坏的资料库所使用。如果你不想保留毁坏库的数据块，可以运行垃圾回收程序并使用--ignore-errors或-i选项。

```
seafserv-gc -c ../../ccnet -d ../../seafile-data --ignore-errors
```

这将会屏蔽毁坏资料库的数据块为无用状态并删除掉它们。

WebDAV和FUSE扩展

Seafile WebDAV和FUSE扩展使得Seafile能够很容易的与第三方应用协调工作。例如，你可以在IOS上通过WebDAV接口访问Seafile上的文件。

WebDAV扩展

Seafile WebDAV Server(SeafDAV)在Seafile Server 2.1.0版本中被加入.

在下面的维基中, 我们假设你将Seafile安装到 `/data/haiwen` 目录下。

SeafDAV配置

SeafDAV配置文件是 `/data/haiwen/conf/seafdav.conf` . 如果它还没有被创建，你可以自行创建它。

```
[WEBDAV]

# 默认值是false。改为true来使用SeafDAV server。
enabled = true

port = 8080

# 如果fastcgi将被使用则更改fastcgi的值为true。
fastcgi = false

# 如果你将seafdav部署到nginx/apache，你需要更改“share_name”的值。
share_name = /
```

每次配置文件被修改后，你需要重启Seafile服务器使之生效。

```
./seafile.sh restart
```

示例配置 1: No nginx/apache

你的WebDAV客户端将在地址 `http://example.com:8080` 访问WebDAV服务器。

```
[WEBDAV]
enabled = true
port = 8080
fastcgi = false
share_name = /
```

示例配置 2: With Nginx/Apache

你的WebDAV客户端将在地址 `http://example.com/seafdav` 访问WebDAV服务器。

```
[WEBDAV]
enabled = true
port = 8080
fastcgi = true
share_name = /seafdav
```

在上面的配置中，"'share_name'"的值被改为"'/seafdav'", 它是你指定给seafdav服务器的地址后缀。

Nginx 无 HTTPS

相应的Nginx配置如下 (无 https):

```
    location /seafdav {
       fastcgi_pass    127.0.0.1:8080;
       fastcgi_param   SCRIPT_FILENAME     $document_root$fastcgi_script_name;
       fastcgi_param   PATH_INFO           $fastcgi_script_name;

       fastcgi_param   SERVER_PROTOCOL     $server_protocol;
       fastcgi_param   QUERY_STRING        $query_string;
       fastcgi_param   REQUEST_METHOD      $request_method;
       fastcgi_param   CONTENT_TYPE        $content_type;
       fastcgi_param   CONTENT_LENGTH      $content_length;
       fastcgi_param   SERVER_ADDR         $server_addr;
       fastcgi_param   SERVER_PORT         $server_port;
       fastcgi_param   SERVER_NAME         $server_name;

       client_max_body_size 0;

       access_log      /var/log/nginx/seafdav.access.log;
       error_log       /var/log/nginx/seafdav.error.log;
   }
```

Nginx 有 HTTPS

Nginx配置为https:

```
    location /seafdav {
       fastcgi_pass    127.0.0.1:8080;
       fastcgi_param   SCRIPT_FILENAME     $document_root$fastcgi_script_name;
       fastcgi_param   PATH_INFO           $fastcgi_script_name;

       fastcgi_param   SERVER_PROTOCOL     $server_protocol;
       fastcgi_param   QUERY_STRING        $query_string;
       fastcgi_param   REQUEST_METHOD      $request_method;
       fastcgi_param   CONTENT_TYPE        $content_type;
       fastcgi_param   CONTENT_LENGTH      $content_length;
       fastcgi_param   SERVER_ADDR         $server_addr;
       fastcgi_param   SERVER_PORT         $server_port;
       fastcgi_param   SERVER_NAME         $server_name;

       client_max_body_size 0;

       fastcgi_param   HTTPS               on;

       access_log      /var/log/nginx/seafdav.access.log;
       error_log       /var/log/nginx/seafdav.error.log;
   }
```

Apache

首先编辑 `apache2.conf` 文件, 添加如下这行到文件结尾(或者根据你的Linux发行版将其添加到 `httpd.conf`):

```
FastCGIExternalServer /var/www/seafdav.fcgi -host 127.0.0.1:8080
```

注意, `/var/www/seafdav.fcgi` 仅仅只是一个占位符, 实际在你的系统并不需要有此文件。

第二, 修改Apache配置文件 (site-enabled/000-default):

Apache 无 HTTPS

根据你的Apache配置当你[将要部署 Seafile 和 Apache|已经部署 Seafile 和 Apache], 加入Seafdav的相关配置:

```


ServerName www.myseafile.com
  DocumentRoot /var/www
  Alias /media  /home/user/haiwen/seafile-server/seahub/media

  RewriteEngine On

  #
  # seafile fileserver
  #
  ProxyPass /seafhttp http://127.0.0.1:8082
  ProxyPassReverse /seafhttp http://127.0.0.1:8082
  RewriteRule ^/seafhttp - [QSA,L]

  #
  # seafile webdav
  #
  RewriteCond %{HTTP:Authorization} (.+)
  RewriteRule ^(/seafdav.*)$ /seafdav.fcgi$1 [QSA,L,e=HTTP_AUTHORIZATION:%1]
  RewriteRule ^(/seafdav.*)$ /seafdav.fcgi$1 [QSA,L]

  #
  # seahub
  #
  RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.*)$ /seahub.fcgi$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
```

Apache 有 HTTPS

根据你的apache配置当你配置Seafile网站和Apache并启用Https, 加入seafdav的相关配置:

```
ServerName www.myseafile.com
  DocumentRoot /var/www
  Alias /media  /home/user/haiwen/seafile-server/seahub/media

  SSLEngine On
  SSLCertificateFile /etc/ssl/cacert.pem
  SSLCertificateKeyFile /etc/ssl/privkey.pem

  RewriteEngine On

  #
  # seafile fileserver
  #
  ProxyPass /seafhttp http://127.0.0.1:8082
  ProxyPassReverse /seafhttp http://127.0.0.1:8082
  RewriteRule ^/seafhttp - [QSA,L]

  #
  # seafile webdav
  #
  RewriteCond %{HTTP:Authorization} (.+)
  RewriteRule ^(/seafdav.*)$ /seafdav.fcgi$1 [QSA,L,e=HTTP_AUTHORIZATION:%1]
  RewriteRule ^(/seafdav.*)$ /seafdav.fcgi$1 [QSA,L]

  #
  # seahub
  #
  RewriteRule ^/(media.*)$ /$1 [QSA,L,PT]
  RewriteCond %{REQUEST_FILENAME} !-f
  RewriteRule ^(.*)$ /seahub.fcgi$1 [QSA,L,E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
```

关于客户端的注意事项

Windows

在Windows平台，我们推荐使用webdav客户端软件例如Cyberduck或BitKinex. webdav对于Windows浏览器的支持实现并不是十分可用，因为：

```
Windows 浏览器需要利用HTTP数字认证。但是由于Seafile在服务器端不存储纯文本密码，所以它不支持这个特性。HT
```

结论就是如果你有一个合法的ssl证书，你应该能过通过Windows浏览器来访问seafdav。否则你应该使用客户端软件。Windows XP被声明不支持HTTPS webdav.

Linux

在Linux平台你有更多的选择。你可以利用文件管理器例如Nautilus来连接webdav服务器，或者在命令行使用davfs2。

使用davfs2

```
sudo apt-get install davfs2
sudo mount -t davfs -o uid= https://example.com/seafdav /media/seafdav/
```

-o选项设置挂载目录的拥有者为，使得非root用户拥有可写权限。

我们建议对于davfs2，禁用锁操作。你需要编辑/etc/davfs2/davfs2.conf

```
 use_locks       0
```

Mac OS X

Finder对于WebDAV的支持不稳定而且较慢. 所以我们建议使用webdav客户端软件如Cyberduck.

常见问题

客户端无法连接**seafdav**服务器

默认, seafdav是未被启用的。检查你是否在`seafdav.conf`中设置`enabled = true`。 如果没有，更改配置文件并重启seafle服务器。

客户端得到**"Error: 404 Not Found"**错误

如果你将SeafDAV部署在Nginx/Apache, 请确保像上面的配置文件一样更改`share_name`的值。重启Seafile服务器后重新测试。

FUSE扩展

在Seafile系统上文件被分割成数据块，这意味着在你的Seafile服务器上存储的并不是完整的文件而是数据块。这种设计能够方便有效的运用数据去重技术。

然而，有时系统管理员想要直接访问服务器上的文件，你可以使用seaf-fuse来做到这点。

`Seaf-fuse` 是一种FUSE虚拟文件系统的实现. 一句话来说就是，它挂载所有的Seafile文件到一个目录（它被称为'''挂载点'''），所以你可以像访问服务器上的正常目录一样来访问由Seafile服务器管理的所有文件。

`注意:`

- 加密的目录不可以被seaf-fuse来访问。
- Seaf-fuse的当前实现是只读访问，这意味着你不能通过挂载的目录来修改文件。
- 对于debian/centos系统，你需要在“fuse”组才有权限来挂载一个FUSE目录。

如何启动seaf-fuse

假设你想挂载到 `/data/seafile-fuse` .

创建一个目录作为挂载点

```
mkdir -p /data/seafile-fuse
```

用脚本来启动seaf-fuse

`注意:` 在启动seaf-fuse之前, 你应该已经通过执行 `./seafile.sh start` 启动好Seafile服务器。

```
./seaf-fuse.sh start /data/seafile-fuse
```

停止seaf-fuse

```
./seaf-fuse.sh stop
```

挂载目录的内容

顶层目录

现在你可以列出 `/data/seafile-fuse` 目录的内容

```
$ ls -lhp /data/seafile-fuse

drwxr-xr-x 2 root root 4.0K Jan  1  1970 abc@abc.com/
drwxr-xr-x 2 root root 4.0K Jan  1  1970 foo@foo.com/
```

```
drwxr-xr-x 2 root root 4.0K Jan  1  1970 plus@plus.com/
drwxr-xr-x 2 root root 4.0K Jan  1  1970 sharp@sharp.com/
drwxr-xr-x 2 root root 4.0K Jan  1  1970 test@test.com/
```

- 顶层目录包含许多子目录，每个子目录对应一个用户
- 文件和目录的时间戳不会被保存

每个用户的目录

```
$ ls -lhp /data/seafile-fuse/abc@abc.com

drwxr-xr-x 2 root root  924 Jan  1  1970 5403ac56-5552-4e31-a4f1-1de4eb889a5f_Photos/
drwxr-xr-x 2 root root 1.6K Jan  1  1970 a09ab9fc-7bd0-49f1-929d-6abeb8491397_My Notes/
```

从上面的列表可以看出，在用户目录下有一些子目录，每个子目录代表此用户的一个资料库，并且以"'{库id}-{库名字}'"的格式来命名。

资料库的目录

```
$ ls -lhp /data/seafile-fuse/abc@abc.com/5403ac56-5552-4e31-a4f1-1de4eb889a5f_Photos/

-rw-r--r-- 1 root root 501K Jan  1  1970 image.png
-rw-r--r-- 1 root root 501K Jan  1  1970 sample.jpng
```

如果出现**"Permission denied"**的错误

如果你运行 `./seaf-fuse.sh start` 时，遇到"Permission denied"的错误信息, 很有可能你没有在“fuse用户组”解决方法：

- 把你的用户加到fuse组

  ```
  sudo usermod -a -G fuse
  ```

- 退出shell重新登陆

- 现在试着再一次执行 `./seaf-fuse.sh start <path>`。

安全和审计

安全特性

- 安全特性

访问日志和审计

- 访问日志和审计

安全机制

客户端和服务器间的通信加密

Seafile 在服务器配置了 HTTPS 后，客户端会自动使用 HTTPS 协议和服务器通信。

加密资料库如何工作？

当你创建一个加密资料库，你将为其提供一个密码。所有资料库中的数据在上传到服务器之前都将用密码进行加密。

加密流程：

1. 生成一个32字节长的加密的强随机数。它将被用作文件加密键（“文件键”）。
2. 用用户提供的密码对文件键进行加密。我们首先用PBKDF2算法从密码中获取到一个键/值对，然后用AES 256/CBC来加密文件键，所得结果被称之为“加密的文件键”。加密的文件键将会被发送到服务器并保存下来。当你需要访问那部分数据，你可以从加密的文件键中解密出文件键。
3. 所有的文件数据都将用AES 256/CBC加密的文件键进行加密。我们用PBKDF2算法从文件键中获取键/值对。加密完成后，数据将会被传送到服务器端。

上述加密过程即可在桌面客户端执行也可在网站浏览器中执行。浏览器端加密功能可在服务器端使用。当你从加密的资料库中上传和下载时，如下过程将会发生：

- 服务器端发回加密的数据，浏览器将会在客户端用JavaScript解密它们。
- 浏览器在客户端用JavaScript加密后，将加密后的数据发回服务器。服务器端直接保存加密后的结果。

在上述过程中，你的密码将不会在服务器端传输。

当你同步一个加密资料库到桌面客户端或者在网站浏览器中浏览一个资料库，桌面客户端/浏览器需要确认你的密码。当你创建一个资料库，一个“魔力标志”将会在密码和资料库id中获得。这个标志和资料库一起存储到服务器端。客户端用这个标志检查你的密码是否正确在你同步和浏览资料库之前。魔力标志是通过PBKDF2算法经过1000次迭代产生，所以它将非常安全抵抗蛮力破解。

为了最大安全性，纯文本的密码也不会保存在客户端。客户端只保存从“文件键”获得的键/值对，它用来解密数据。所以如果你忘记密码，你将不能恢复和访问服务器端的数据。

Access log and auditing

In pro edition, Seafile offers four auditing logs in system admin panel:

- Login log
- File access log
- File update log
- Permission change log

The logging feature is turned off by default. See config options for pro edition for how to turn it on.

file access log

Access log (under directory logs/stats-logs) records the following information

- file download via Web
- file download via API (including mobile clients and cloud file browser)
- file sync via desktop clients

The format is:

```
user, operation type, ip/device, date, library, path.
```

file update log

The format is:

```
user, date, library, path, detail
```

permission change log

The format is:

```
user, grant to, operation, permission, library, folder, date
```

Develop Documents

- How to Build Seafile
- How to Setup Develop Envirnoment
- Seafile Code Standard

Seafile Open API

- Seafile Web API
- Seafile Python API

Seafile Implement Detail

- Seafile Data Model
- Seafile Server Components
- Seafile Sync algorithm

How to Build Seafile

You can build Seafile from our source code package or from the Github repo directly.

Client

- Linux
- Max OS X

Server

- Build Seafile server

Linux

Preparation

The following list is what you need to install on your development machine. **You should install all of them before you build seafile**.

Package names are according to Ubuntu 12.04. For other Linux distros, please find their corresponding names yourself.

- autoconf/automake/libtool
- libevent-dev (2.0 or later)
- libcurl4-openssl-dev (1.0.0 or later)
- libgtk2.0-dev (2.24 or later)
- uuid-dev
- intltool (0.40 or later)
- libsqlite3-dev (3.7 or later)
- valac (only needed if you build from git repo)
- libjansson-dev
- libqt4-dev
- valac
- cmake
- libfuse-dev (for seafile >= 2.1)
- python-simplejson (for seaf-cli)

```
sudo apt-get install autoconf automake libtool libevent-dev libcurl4-openssl-dev libgtk2.
```

For a fresh Fedora 20 installation, the following will install all dependencies via YUM:

```
$ sudo yum install wget gcc libevent-devel openssl-devel gtk2-devel libuuid-devel sqlite-
```

Building

First you should get the latest source of libsearpc/ccnet/seafile/seafile-client:

Download the source tarball of the latest tag from

- https://github.com/haiwen/libsearpc/tags (use v3.0-latest)
- https://github.com/haiwen/ccnet/tags
- https://github.com/haiwen/seafile/tags
- https://github.com/haiwen/seafile-client/tags

For example, if the latest released seafile client is 3.0.2, then just use the **v3.0.2** tags of the four projects.

You should get four tarballs:

- libsearpc-v3.0-latest.tar.gz
- ccnet-3.0.2.tar.gz
- seafile-3.0.2.tar.gz
- seafile-client-3.0.2.tar.gz

```
export version=3.0.2
alias wget='wget --content-disposition -nc'
wget https://github.com/haiwen/libsearpc/archive/v3.0-latest.tar.gz
wget https://github.com/haiwen/ccnet/archive/v${version}.tar.gz
wget https://github.com/haiwen/seafile/archive/v${version}.tar.gz
wget https://github.com/haiwen/seafile-client/archive/v${version}.tar.gz
```

Now uncompress them:

```
tar xf libsearpc-v3.0-latest.tar.gz
tar xf ccnet-${version}.tar.gz
tar xf seafile-${version}.tar.gz
tar xf seafile-client-${version}.tar.gz
```

To build Seafile client, you need first build **libsearpc** and **ccnet**, **seafile**.

set paths

```
export PREFIX=/usr
export PKG_CONFIG_PATH="$PREFIX/lib/pkgconfig:$PKG_CONFIG_PATH"
export PATH="$PREFIX/bin:$PATH"
```

libsearpc

```
cd libsearpc-${version}
./autogen.sh
./configure --prefix=$PREFIX
make
sudo make install
```

ccnet

```
cd ccnet-${version}
./autogen.sh
./configure --prefix=$PREFIX
make
sudo make install
```

seafile

```
cd seafile-${version}/
./autogen.sh
./configure --prefix=$PREFIX --disable-gui
make
sudo make install
```

seafile-client

```
cd seafile-client-${version}
cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=$PREFIX .
make
sudo make install
```

custom prefix

when installing to a custom `$PREFIX`, i.e. `/opt`, you may need a script to set the path variables correctly

```
cat >$PREFIX/bin/seafile-applet.sh <<END
#!/bin/bash
export LD_LIBRARY_PATH="$PREFIX/lib:$LD_LIBRARY_PATH"
export PATH="$PREFIX/bin:$PATH"
exec seafile-applet $@
END
cat >$PREFIX/bin/seaf-cli.sh <<END
export LD_LIBRARY_PATH="$PREFIX/lib:$LD_LIBRARY_PATH"
export PATH="$PREFIX/bin:$PATH"
export PYTHONPATH=$PREFIX/lib/python2.7/site-packages
exec seaf-cli $@
END
chmod +x $PREFIX/bin/seafile-applet.sh $PREFIX/bin/seaf-cli.sh
```

you can now start the client with `$PREFIX/bin/seafile-applet.sh`.

Windows

Mac OS X

Setup development environment

1. Install xcode
2. Install macports

   Modify file `/opt/local/share/macports/Tcl/port1.0/portconfigure.tcl` , change the line

   ```
   default configure.ldflags {-L${prefix}/lib}
   ```

   to

   ```
   default configure.ldflags {"-L${prefix}/lib -Xlinker -headerpad_max_install_names"}
   ```

3. Install following libraries and tools using `port`

   ```
   autoconf, intltool, automake,  pkgconfig, libtool, glib2, ossp-uuid, libevent, vala, op
   ```

4. Install python

   ```
   port install python27
   port select python python27
   export PATH=/opt/local/Library/Frameworks/Python.framework/Versions/2.7/bin:$PATH
   ```

   Then download and install pip from http://pypi.python.org/pypi/pip

5. Set pkg config environment

   ```
   export PKG_CONFIG_PATH=/opt/local/lib/pkgconfig:/usr/local/lib/pkgconfig
   ```

Compiling libsearpc

Download libsearpc, then:

```
./autogen.sh
LDFLAGS="-Xlinker -headerpad_max_install_names" ./configure
make
sudo make install
```

Compiling ccnet

Download ccnet, then:

```
./autogen.sh
CFLAGS="-Wall" LDFLAGS="-L/opt/local/lib -Xlinker -headerpad_max_install_names" ./configu
make
sudo make install
```

Compiling seafile

1. Download seafile
2. Install python libs and tools

   ```
   sudo pip-2.7 install py2app web.py mako simplejson
   ```

3. Compile

   ```
   ./autogen.sh
   LDFLAGS="-L/opt/local/lib -Xlinker -headerpad_max_install_names -framework CoreServi
   make
   sudo make install
   ```

Packaging

1. seafileweb. First setup python path:

   ```
   export PYTHONPATH=.:/usr/local/lib/python2.7/site-packages
   ```

   This path is where pyccnet and pysearpc installed.

   ```
   ./setupmac.sh web
   ```

   This will generate `seafileweb.app` , and copy it to `gui/mac/seafile`

2. ccnet, seaf-daemon:

   ```
   ./setupmac.sh dylib
   ```

   This will copy ccnet, seaf-daemon and other libraries to gui/mac/seafile, and use `install_name_tool` to modify the library paths in ccnet, seaf-daemon.

3. Compile seafile.app：

   ```
   ./setupmac.sh 10.6 or ./setupmac.sh 10.7
   ```

   After compiling, it will copy seafile.app to `${top_dir}/../seafile-${VERSION}` . You can also compiling seafile.app in xcode.

4. Go to seafile-${VERSION} and see if it can run correctly.

5. Construct dmg using dropdmg. Use `dmg-backgroud.jpg` as dmg background, add link of `/Application` to seafile-${VERSION}, then packaging seafile-${VERSION} to seafile-${VERSION}.dmg.

Problem you may encounter

1. If `install_name_tool` reports "malformed object" "unknown load command", It may be the version of xcode command line tools incompatible with `install_name_tool` .
2. If xcode can't find glib, Corrects xcode's "build settings/search paths/header search".

Server

Preparation

The following list is all the libraries you need to install on your machine. '''You should install all of them before you build seafile'''.

Package names are according to Ubuntu 12.04. For other Linux distros, please find their corresponding names yourself.

- libevent-dev (2.0 or later)
- libcurl4-openssl-dev (1.0.0 or later)
- libglib2.0-dev (2.28 or later)
- uuid-dev
- intltool (0.40 or later)
- libsqlite3-dev (3.7 or later)
- libmysqlclient-dev (5.5 or later)
- libarchive-dev
- libtool
- libjansson-dev
- valac
- libfuse-dev

The following libraries need to be compiled from source.

- libzdb [http://www.tildeslash.com/libzdb/dist/libzdb-2.12.tar.gz]
- libevhtp [https://github.com/ellzey/libevhtp/archive/1.1.6.zip]

libzdb relies on two packages: `re2c` and `flex`. libevhtp can be build by `cmake .; make; sudo make install`. libevhtp's version should be 1.1.6 or 1.1.7.

'''Seahub''' is the web front end of Seafile. It's written in the [http://djangoproject.com django] framework. Seahub requires Python 2.6(2.7) installed on your server, and it needs the following python libraries:

- [https://www.djangoproject.com/download/1.5.2/tarball/ django 1.5]
- [https://github.com/djblets/djblets/tarball/release-0.6.14 djblets]
- sqlite3
- simplejson (python-simplejson)
- PIL (aka. python imaging library, python-image)
- chardet
- gunicorn

The module '''argparser''' is required by the `seafile-admin` script which you'll see later. If you use Python 2.7, '''argparser''' is distributed with python's standard library, so you don't need to install it. But if you use Python 2.6, you should install it manually.

Before continue, make sure you have all the above libraries available in your system.

Prepare the directory layout

In the following sections, you'll be guided to build and setup the seafile server step by step. Seafile server is consisted of several components. In order for them to function correctly, you must:

- Follow our instructions step by step
- Make sure your directory layout is exactly the same with the guide in each step.

First create the top level directory. In the following sections, we'll use "/data/haiwen" as the top level directory.

```
mkdir /data/haiwen/
cd /data/haiwen/
mkdir seafile-server
cd seafile-server
```

The currently layout is:

```
haiwen/
└── seafile-server
```

Get the source

First you should get the latest source of libsearpc/ccnet/seafile/seahub

Download the source tarball of the latest tag from

- [https://github.com/haiwen/libsearpc/tags]
- [https://github.com/haiwen/ccnet/tags]
- [https://github.com/haiwen/seafile/tags]
- [https://github.com/haiwen/seahub/tags]

For example, if the latest released seafile client is 2.0.3, then just use the '''v2.0.3-server''' tags of the four projects. You should get four tarballs:

- libsearpc-2.0.3-server.tar.gz
- ccnet-2.0.3-server.tar.gz
- seafile-2.0.3-server.tar.gz
- seahub-2.0.3-server.tar.gz

Create a folder `haiwen/src`, and uncompress libsearpc/ccnet/seafile source to it.

```
cd haiwen/seafile-server
mkdir src
cd src
```

```
tar xf /path/to/libsearpc-2.0.3-server.tar.gz
tar xf /path/to/ccnet-2.0.3-server.tar.gz
tar xf /path/to/seafile-2.0.3-server.tar.gz
```

And uncompress seahub tarball to `haiwen/seafile-server` :

```
cd haiwen/seafile-server
tar xf /path/to/seahub-2.0.3-server.tar.gz
mv seahub-2.0.3-server seahub
```

So far, The current directory layout is:

```
haiwen/
└── seafile-server
    └── seahub
    └── src
        ├── libsearpc-2.0.3-server
        ├── ccnet-2.0.3-server
        ├── seafile-2.0.3-server
        ├── ... (other files)
```

Building

To build seafile server, you need first build '''libsearpc''' and '''ccnet'''.

libsearpc

```
cd libsearpc-${version}
./autogen.sh
./configure
make
make install
```

ccnet

```
cd ccnet-${version}
./autogen.sh
./configure --disable-client --enable-server   # `export PKG_CONFIG_PATH=/usr/local/lib/p
make
make install
```

seafile

```
cd seafile-${version}
./autogen.sh
./configure --disable-client --enable-server
make
```

```
make install
```

Deploy Seafile Server

Components of the Seafile Server

The seafile server consists of the following components:

Process Name	Functionality
ccnet-server	underlying networking
seaf-server	data management
Seahub	website front-end of seafile server
fileserver	handles raw file upload/download for Seahub

[[images/server-arch.png]]

- '''ccnet''' stores its configuration and metadata is a directory named `ccnet`.
- '''seaf-server''' store its configuration and data in a directory, normally named `seafile-data`.
- '''seahub''' is written in Django. If you have any experience with Django, you should know the `syncdb` command must be run to create all the database tables.
- An '''admin account''' has to be created, so that you, the admin, can login with this account to manage the server.

These are the essential steps to create the configuration:

- ensure seafile is already installed and all the python libraries seahub needs are installed.
- create the ccnet configuration with the '''ccnet-init''' program
- create the seafile configuration with '''seaf-server-init''' program
- run Django '''syncdb''' command for seahub
- create an admin account for the seafile server

To create the configurations, you can either:

- use the seafile-admin script(see below)
- [[create server configuration by hand]]

Create Configurations with the seafile-admin script

`seafile-admin` should have been installed to system path after you have built and installed Seafile from source.

```
usage: seafile-admin [-h] {setup,start,stop,reset-admin} ...

optional arguments:
  -h, --help            show this help message and exit
```

```
subcommands:

  {setup,start,stop,reset-admin}
    setup               setup the seafile server
    start               start the seafile server
    stop                stop the seafile server
    reset-admin         reset seafile admin account
```

Go to the top level directory(in this guide it's '''/data/haiwen/'''), and run '''seafile-admin setup''' to create all the configuration:

```
cd /data/haiwen
export PYTHONPATH=/data/haiwen/seafile-server/seahub/thirdpart
seafile-admin setup
```

The script would ask you a series of questions, and create all the configuration for you.

Name	Usage	Default	Requirement
server name	The name of the server that would be shown on the client		3 ~ 15 letters or digits
ip or domain	The ip address or domain name of the server		Make sure to use the right ip or domain, or the client would have trouble connecting it
ccnet port	the tcp port used by ccnet	10001	
seafile port	tcp port used by seafile	12001	
seafile fileserver port	tcp port used by seafile fileserver	8082	
admin email	Email address of the admin account		
admin password	password of the admin account		

This is a screenshot of the '''seafile-admin setup''' command: [[images/seafile-admin-1.png]]

And a screenshot after setup is finished successfully: [[images/seafile-admin-2.png]]

At this time, the directory layout would be like this:

```
haiwen/
└── ccnet # ccnet config directory
    └── ccnet.conf # ccnet config file
└── seafile-data # seafile configuration and data
    └── seafile.conf # seafile config file
└── seahub-data/ # seahub data
└── seahub.db # seahub sqlite3 database
└── seahub_settings.py # custom settings for seahub
└── seafile-server
    └── seahub/
    └── seafile-{VERSION} # seafile source code
```

Start the Seafile Server

After configuration successfully created, run '''seafile-admin start''' in the top directory to start the all components of Seafile. ('''You should always run the seafile-admin script in the top directory''').

```
cd /data/haiwen # go to the top level directory
seafile-admin start
```

At this moment, all the components should be running and seahub can be visited at '''http://yourserver-ip-or-domain:8000'''

'''Note''' You may want to deploy seahub with nginx or apache. In this case, follow the instructions on [[Deploy Seafile Web With Nginx/Apache]].

Stop the Seafile Server

To stop seafile server, run '''seafile-admin stop'''.

```
cd /data/haiwen # go to the top level directory
seafile-admin stop
```

Upgrade the Seafile Server

When you want to upgrade to a new vesrion of seafile server, you need to:

- Stop the seafile server if it's running

```
cd /data/haiwen
seafile-admin stop
```

- Get and latest source code and build libsearpc/ccnet/seafile, just as what you do in a fresh setup.
- Run the upgrade script. The upgrade script mainly updates database used by seafile for you. For example, create a new database table that is used in the latest seafile server but not in the previous version.

Get and compile the latest libsearpc/ccnet/seafile

See the '''Building''' section above.

Get the new seahub tarball and uncompress it

```
cd haiwen/seafile-server
mv seahub/ seahub-old # move away the old seahub folder
tar xf /path/to/new/seahub-x.x.x-server.tar.gz
mv seahub-x.x.x-server seahub
```

Do the upgrade

- copy the scripts/upgrade/ subdir outside

The upgrade scripts is distributed in the `scripts/upgrade` subdir of seafile source code, we need to copy it to '''seafile-server''' directory before run the scripts.

```
cd /data/haiwen/seafile-server
cp -rf seafile-{version}/scripts/upgrade .
```

Continuous Upgrade (like from 1.1 to 1.2)

Continuous upgrade means to upgrade from one version of seafile server to the next version. For example, upgrading from 1.1.0 to 1.2.0 is a continuous upgrade.

'''Note:''' Minor upgrade, like upgrade from 1.3.0 to 1.3.1, is documented in a separate section below.

Say you are upgrading from 1.1.0 to 1.2.0, you should run the script '''upgrade_1.1_1.2.sh''' in `seafile-server` directory.

```
cd /data/haiwen/seafile-server
./upgrade/upgrade_1.1_1.2.sh
```

Non-continous version upgrade(like from 1.1 to 1.3)

If you upgrade a few versions at once, e.g. from 1.1.0 to 1.3.0. The procedure is:

- upgrade from 1.1.0 to 1.2.0
- upgrade from 1.2.0 to 1.3.0

Just run the upgrade scripts in sequence.

Minor Upgrade (like from 1.3.0 to 1.3.1)

Minor upgrade Minor upgrade is like an upgrade from 1.3.0 to 1.3.1. For this type of upgrade, you only need to update the avatar link:

```
cd /data/haiwen/seafile-server/seahub/media
cp -rf avatars/* ../../../seahub-data/avatars/
rm -rf avatars
ln -s ../../../seahub-data/avatars
```

Problems Report

If you encounter any problem when building/deploying Seafile, please leave us a message or [https://github.com/haiwen/seafile/issues open an issue].

Setup Develop Environment

Preparation

Package names are according to Ubuntu 12.04. For other Linux distros, please find their corresponding names yourself.

- libevent-dev (2.0 or later)
- libcurl4-openssl-dev (1.0.0 or later)
- libglib2.0-dev (2.28 or later)
- uuid-dev
- intltool (0.40 or later)
- libsqlite3-dev (3.7 or later)
- libmysqlclient-dev (5.5 or later)
- libarchive-dev
- libtool
- libjansson-dev
- valac
- libfuse-dev
- python-dateutil

The following libraries need to be compiled from source.

- libzdb
- libevhtp

libzdb relies on two packages: `re2c` and `flex`. libevhtp can be build by `cmake .; make; sudo make install`. libevhtp's version should be 1.1.6 or 1.1.7.

'''Seahub''' is the web front end of Seafile. It's written in the Django framework. Seahub requires Python 2.6(2.7) installed on your server, and it needs the following python libraries:

- Django1.5
- Djblets
- sqlite3
- simplejson (python-simplejson)
- PIL (aka. python imaging library, python-image) or Pillow
- chardet

Download & Compile

Clone libsearpc, ccnet, seafile, seahub to ~/dev (or wherever you want).

Complie libsearpc with

```
cd ~/dev/libsearpc
./autogen.sh
./configure
make
make install
```

Compile ccnet with

```
cd ~/dev/ccnet
./autogen.sh
./configure --disable-client --enable-server   # `export PKG_CONFIG_PATH=/usr/local/lib/p
make
make install
```

Compile seafile with

```
cd ~/dev/seafile
./autogen.sh
./configure --disable-client --enable-server
make
make install
```

Run seafile

Run seafile with

```
cd ~/dev/seafile/tests/basic
./seafile.sh 2
```

Or you can start ccnet, seafile and fileserver manually by:

```
ccnet-server -c ~/dev/seafile/test/basic/conf2/ -D all -f -
seaf-server -c ~/dev/seafile/test/basic/conf2/ -d ~/dev/seafile/test/basic/conf2/seafile-
fileserver -c ~/dev/seafile/test/basic/conf2/ -d ~/dev/seafile/test/basic/conf2/seafile-d
```

Prepare seahub

Go to seahub

```
cd ~/dev/seahub
```

Download django-1.5 to thirdpart. And create and modify setenv.sh from templates

```
cp setenv.sh.template setenv.sh
```

Create database

```
. setenv.sh
python manage.py syncdb
```

Create admin account (assume seafile is under ~/dev/seafile)

```
python tools/seahub-admin.py ~/dev/seafile/tests/basic/conf2
```

Start seahub

```
./run-seahub.sh.template
```

Code Standard

The source code of seafile is ISO/IEC 9899:1999 (E) (a.k.a. C99) compatible.

Indent

- Use only spaces, and indent 4 spaces at a time.

Inline

- Define functions inline only when they are sufficiently small.

Const

- We are using keyword `const` in the source.Please look into the code for detail.

Web API

Seafile Web API V2

API Basics

All API calls must be authenticated with a valid Seafile API key.

```
curl -H 'Authorization: Token 24fd3c026886e3121b2ca630805ed425c272cb96' https://cloud.sea
```

The api key can be retrieved by the obtain auth api. See the Quick Start for details.

For each API, we provide `curl` examples to illustrate the usage.

Status Code

- 200: OK
- 201: CREATED
- 202: ACCEPTED
- 301: MOVED_PERMANENTLY
- 400: BAD_REQUEST
- 403: FORBIDDEN
- 404: NOT_FOUND
- 409: CONFLICT
- 429: TOO_MANY_REQUESTS
- 440: REPO_PASSWD_REQUIRED

- 441: REPO_PASSWD_MAGIC_REQUIRED
- 500: INTERNAL_SERVER_ERROR
- 520: OPERATION_FAILED

Quick Start

ping

```
curl https://cloud.seafile.com/api2/ping/

"pong"
```

obtain auth token

```
curl -d "username=username@example.com&password=123456" https://cloud.seafile.com/api2/au

{"token": "24fd3c026886e3121b2ca630805ed425c272cb96"}
```

auth ping

```
curl -H 'Authorization: Token 24fd3c026886e3121b2ca630805ed425c272cb96' https://cloud.sea

"pong"
```

Account

List Accounts

GET https://cloud.seafile.com/api2/accounts/

Request parameters

- start (default to 0)
- limit (default to 100)
- scope (default None, accepted values: 'LDAP' or 'DB')

To retrieve all users, just set both `start` and `limit` to `-1`.

If scope parameter is passed then accounts will be searched inside the specific scope, otherwise it will be used the old approach: first LDAP and, if no account is found, DB.

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -H 'Accept: appli
```

Sample response

```
[
{
    "email": "foo@foo.com"
},
{
    "email": "bar@bar.com"
}
]
```

Errors

- 403 Permission error, only administrator can perform this action

Get Account Info

GET https://cloud.seafile.com/api2/accounts/{email}/

Request parameters

Sample request

```
curl -v -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -H 'Accept: ap
```

Sample response

```
{
"is_staff": false,
"is_active": true,
"id": 2,
"create_time": 1356061187741686,
"usage": 651463187,
"total": 107374182400,
"email": "user@mail.com"
}
```

Errors

- 403 Permission error, only administrator can perform this action

Check Account Info

GET https://cloud.seafile.com/api2/account/info/

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -H 'Accept: appli
```

Sample response

```
{
"usage": 26038531,
"total": 104857600,
"email": "user@example.com"
}
```

Errors

- 403 Invalid token

Create Account

PUT https://cloud.seafile.com/api2/accounts/{email}/

Request parameters

- password
- is_staff (defaults to False)
- is_active (defaults to True)

Sample request

```
curl -v -X PUT -d "password=123456" -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8
```

Sample response

```
...
< HTTP/1.0 201 CREATED
< Location: https://cloud.seafile.com/api2/accounts/newaccount@gmail.com/
...

"success"
```

Success

```
Response code 201(Created) is returned and the Location header provides shared link.
```

Errors

- 403 Permission error, only administrator can perform this action

Update Account

PUT https://cloud.seafile.com/api2/accounts/{email}/

Request parameters

At least one of followings:

- password
- is_staff
- is_active
- name
- note
- storage

Sample request

```
curl -v -X PUT -d "password=654321&is_staff=true&storage=1073741824" -H "Authorization: T
```

Sample response

```
...
< HTTP/1.0 200 OK
...

"success"
```

Success

```
Response code 200(OK) is returned.
```

Errors

- 400 Bad Request, keyword password is required
- 403 Permission error, only administrator can perform this action

Delete Account

DELETE https://cloud.seafile.com/api2/accounts/{email}/

Sample request

```
curl -v -X DELETE -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -H '
```

Sample response

```
"success"
```

Errors

- 403 Permission error, only administrator can perform this action

Get Server Information

GET https://cloud.seafile.com/api2/server-info

Note:

- No authentication required.
- Added in seafile community edition server `4.0.5` or pro edition server `4.0.3`

Sample request

```
curl https://cloud.seafile.com/api2/server-info/
```

Sample response

Sample response from a seafile community edition server:

```
{
    "version": "4.0.6",
    "features": [
    "seafile-basic",
    ]
}
```

Sample response from a seafile pro edition server:

```
{
    "version": "4.0.6",
    "features": [
    "seafile-basic",
    "seafile-pro",
    "office-preview",
    "file-search"
    ]
}
```

Starred Files

List starred files

GET https://cloud.seafile.com/api2/starredfiles/

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e6199b477fd" -H 'Accept: applic
```

Sample response

```
[
{
    "repo": "99b758e6-91ab-4265-b705-925367374cf0",
    "mtime": 1355198150,
    "org": -1,
    "path": "/foo/bar.doc",
    "dir": false,
    "size": 0
},
{
    "repo": "99b758e6-91ab-4265-b705-925367374cf0",
    "mtime": 1353751237,
    "org": -1,
    "path": "/add_folder-blue.png",
    "dir": false,
    "size": 3170
}
]
```

Star A File

POST https://cloud.seafile.com/api2/starredfiles/

Request parameters

- repo_id (post)
- p (post)

Sample request

```
curl -v -d "repo_id=dae8cecc-2359-4d33-aa42-01b7846c4b32&p=/foo.md" -H 'Authorization: To
```

Sample response

```
...
< HTTP/1.0 201 CREATED
< Location: https://cloud.seafile.com/api2/starredfiles/
```

```
...
"success"
```

Success

Response code is 201(Created) and Location header provides url of starred file list.

Errors

- 400 `repo_id` or `p` is missing, or `p` is not valid file path(e.g. /foo/bar/).

Unstar A File

DELETE https://cloud.seafile.com/api2/starredfiles/

Request parameters

- repo_id
- p

Sample request

```
curl -X DELETE -v  -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' -H
```

Sample response

```
...
< HTTP/1.0 200 OK
...
"success"
```

Success

Response code is 200(OK), and a string named "success" is returned.

Errors

- 400 `repo_id` or `p` is missing, or `p` is not valid file path(e.g. /foo/bar/).

User Messages

List User Messages

GET https://cloud.seafile.com/api2/user/msgs/{id_or_email}/

Request parameters

- id_or_email

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "to_email": "user@example.com",
    "next_page": -1,
    "msgs": [
        {
            "attachments": [
                {
                    "path": "/123.md",
                    "repo_id": "c7436518-5f46-4296-97db-2fcba4c8c8db"
                }
            ],
            "timestamp": 1398233096,
            "from_email": "user@example.com",
            "msgid": 3,
            "msg": "another test msg",
            "nickname": "user"
        },
        {
            "attachments": [],
            "timestamp": 1398233067,
            "from_email": "user@example.com",
            "msgid": 2,
            "msg": "a test msg",
            "nickname": "user"
        }
    ]
}
```

Errors

- 404 user not found

Reply A User Message

POST https://cloud.seafile.com/api2/user/msgs/{id_or_email}/

Request parameters

- id_or_email
- message

Sample request

```
curl -d "message=this is a user msg reply" -H 'Authorization: Token f2210dacd9c6ccb813360
```

Sample response

```
{
    "msgid": 4
}
```

Errors

- 404 user not found

Count Unseen Messages

GET https://cloud.seafile.com/api2/unseen_messages/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "count": 1
}
```

Group

List Groups

GET https://cloud.seafile.com/api2/groups/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "replynum": 0,
    "groups": [
        {
            "ctime": 1398134171327948,
            "creator": "user@example.com",
            "msgnum": 0,
            "mtime": 1398231100,
            "id": 1,
            "name": "lian"
        },
        {
            "ctime": 1398236081042441,
            "creator": "user@example.com",
            "msgnum": 0,
            "mtime": 0,
            "id": 2,
            "name": "123"
        }
    ]
}
```

Add A Group

PUT https://cloud.seafile.com/api2/groups/

Request parameters

- group_name

Sample request

```
curl -X PUT -d "group_name=newgroup" -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff
```

Sample response

```
{"group_id": 3, "success": true}
```

Errors

- 400 There is already a group with that name.

Delete Group

DELETE https://cloud.seafile.com/api2/groups/{group_id}/

Request parameters

None

Sample request

```
curl -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https:
```

Success

200 if everything is fine.

Errors

- 400 if ad group id format
- 404 if Group not found
- 403 if Forbid to delete group
- 520 if Failed to remove group (generic error)

Rename Group

POST https://cloud.seafile.com/api2/groups/{group_id}/

Request parameters

- operation (value must be 'rename')
- newname (the new name for the group)

Sample request

```
curl -d "operation=rename&newname=pinkfloyd_lovers" -H 'Authorization: Token f2210dacd9c6
```

Success

200 if everything is fine.

Errors

- 404 if Group not found
- 403 if Forbid to rename group
- 400 if Newname is missing or if Group name is not valid of if There is already a group with that name or Operation can only be rename.

Group Member

Add A Group Member

PUT https://cloud.seafile.com/api2/groups/{group_id}/members/

Request parameters

- user_name

Sample request

```
curl -X PUT -d "user_name=user@example.com"-H 'Authorization: Token f2210dacd9c6ccb813360
```

Sample response

```
{"success": true}
```

Errors

- 400 invalid group id
- 403 only administrators can add group members
- 404 unable to find group

Delete A Group Member

DELETE https://cloud.seafile.com/api2/groups/{group_id}/members/

Request parameters

- user_name

Sample request

```
curl -X DELETE -d "user_name=user@example.com" -H 'Authorization: Token f2210dacd9c6ccb81
```

Sample response

```
{"success": true}
```

Errors

- 400 invalid group id
- 403 only administrators can remove group members
- 404 unable to find group

Group Message

Get Group Messages

GET https://cloud.seafile.com/api2/group/msgs/{group_id}/

Request parameters

- group_id

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "next_page": -1,
    "msgs": [
        {
            "reply_cnt": 0,
            "timestamp": 1398230602,
            "replies": [],
            "from_email": "user@example.com",
            "msgid": 1,
            "msg": "test discuss",
            "nickname": "user"
        }
    ]
}
```

Get Group Message Detail

GET https://cloud.seafile.com/api2/group/{group_id}/msg/{msg_id}/

Request parameters

- group_id
- msg_id

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "reply_cnt": 2,
    "timestamp": 1398230602,
    "replies": [
        {
            "msg": "this is another test",
            "timestamp": 1398232319,
            "nickname": "user",
            "msgid": 1,
            "from_email": "user@example.com"
        },
        {
            "msg": "this is another test",
            "timestamp": 1398232508,
            "nickname": "user",
            "msgid": 3,
            "from_email": "user@example.com"
        }
    ],
    "from_email": "user@example.com",
    "msgid": 1,
    "msg": "test discuss",
    "nickname": "user"
}
```

Errors

- 404 message not found

Send A Group Message

POST https://cloud.seafile.com/api2/group/msgs/{group_id}/

Request parameters

- message
- group_id
- repo_id(optional)
- path(optional)

Sample request

```
curl -d "message=this is another test&repo_id=c7436518-5f46-4296-97db-2fcba4c8c8db&path=/
```

Sample response

```
{
    "msgid": 3
}
```

Reply A Group Message

POST https://cloud.seafile.com/api2/group/{group_id}/msg/{msg_id}

Request parameters

- group_id
- msg_id
- message

Sample request

```
curl -d "message=this is a reply" -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e6
```

Sample response

```
{
    "msgid": 3
}
```

Errors

- 404 message not found

Get Group Message Replies

GET https://cloud.seafile.com/api2/new_replies/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
[
    {
        "reply_cnt": 1,
        "timestamp": 1398231100,
        "replies": [
            {
                "msg": "@user test reply",
                "timestamp": 1398234493,
                "nickname": "123",
                "msgid": 5,
                "from_email": "user@example.com"
            }
        ],
        "from_email": "user@example.com",
        "att": {
            "repo": "c7436518-5f46-4296-97db-2fcba4c8c8db",
            "path": "/123.md",
            "type": "file",
            "src": "recommend"
        },
        "msgid": 3,
        "msg": "this is another test",
        "nickname": "user"
    }
]
```

Share

File Share Link

List File Share Links

GET https://cloud.seafile.com/api2/shared-links/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{"fileshares": [{"username": "user@example.com", "repo_id": "a582d3bc-bcf5-421e-9125-741f
```

Create File Share Link

PUT https://cloud.seafile.com/api2/repos/{repo-id}/file/shared-link/

Request parameters

- repo-id
- p (Path to the file)
- share_type (optional, `download` or `upload`, default `download`)
- password (optional)
- expire (optional)

Sample request

Create download link for file

```
curl -v  -X PUT -d "p=/foo.md" -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9
```

Create download link for directory with password and expire date

```
curl -v  -X PUT -d "password=password&expire=6&p=/123/" -H 'Authorization: Token f2210dac
```

Create upload link for directory

```
curl -v -X PUT -d "share_type=upload&p=/123/" -H 'Authorization: Token f2210dacd9c6ccb813
```

Sample response

```
...
< HTTP/1.0 201 CREATED
< Location: https://cloud.seafile.com/f/9b437a7e55/
...
```

Success

```
Response code 201(Created) is returned and the Location header provides shared link.
```

Errors

- 400 Path is missing
- 400 Password(if link is encrypted) is missing
- 500 Internal server error

Delete File Share Link

DELETE https://cloud.seafile.com/api2/shared-links/?t=0ae587a7d1

Request parameters

- t

Sample request

```
curl -v -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "htt
```

Sample response

```
...
< HTTP/1.0 200 OK
...
```

List Direntry in Dir Download Link

GET https://cloud.seafile.com/api2/d/{token}/dir/

Request parameters

- token (upload link token)
- p (sub folder path)
- password (if link is encrypted)

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
[{"mtime": 1436846750, "type": "dir", "name": "sadof", "id": "1806dbdb700b7bcd49e6275107c
```

Shared Libraries

List Shared Libraries

GET https://cloud.seafile.com/api2/shared-repos/

Sample request

```
curl -v -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' -H 'Accept: ap
```

Sample response

```
[{"repo_id": "7d42522b-1f6f-465d-b9c9-879f8eed7c6c", "share_type": "personal", "permissio
```

List Be Shared Libraries

GET https://cloud.seafile.com/api2/beshared-repos/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
"[{"user": "user@example.com", "repo_id": "989e3952-9d6f-4427-ab16-4bf9b53212eb", "share_
```

Share A Library

PUT https://cloud.seafile.com/api2/shared-repos/{repo-id}/

Request parameters

- share_type ('personal', 'group' or 'public')
- user (or users)
- group_id
- permission

If share_type is 'personal' then 'user' or 'users' param are required, if share_type is 'group' then 'group_id' parameter is required. If share_type is 'public' no other params is required.

'user' or 'users' parameters can be a comma separated list of emails, in this case the share will be done for more users at the same time. If a problem is encountered during multiple users sharing then the sharing process is aborted.

Sample request

```
curl -X PUT -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://c
```

Sample response

```
"success"
```

Unshare A Library

DELETE https://cloud.seafile.com/api2/shared-repos/{repo-id}/

Request parameters

- share_type ('personal', 'group' or 'public')
- user
- group_id

If share_type is 'personal' then 'user' param is required, if share_type is 'group' then 'group_id' parameter is required. If share_type is 'public' no other params is required.

Sample request

```
curl -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https:
```

Sample response

```
"success"
```

Shared Files

List Shared Files

GET https://cloud.seafile.com/api2/shared-files/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{"priv_share_in": [{"s_type": "f", "repo_id": "989e3952-9d6f-4427-ab16-4bf9b53212eb", "pe
```

Download Shared File

GET https://cloud.seafile.com/api2/f/{token}/

Request parameters

- token(file share token)

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
"http://192.168.1.101:8082/files/89223601/lib.md"
```

Errors

- 404 repo/token/file not found
- 520 OPERATION FAILED, fail to get file id by path

Get Shared File Detail

GET https://cloud.seafile.com/api2/f/{token}/detail/

Request parameters

- token(file share token)

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{"repo_id": "989e3952-9d6f-4427-ab16-4bf9b53212eb", "name": "lib.md", "mtime": 1398218747
```

Errors

- 404 repo/token/file not found
- 520 OPERATION FAILED, fail to get file id by path

Delete Shared File

DELETE https://cloud.seafile.com/api2/shared-files/?t=0ae587a7d1

Request parameters

- t

Sample request

```
curl -v -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "htt
```

Sample response

```
...
< HTTP/1.0 200 OK
...
```

Download Private Shared File

GET https://cloud.seafile.com/api2/s/f/{token}/

Request parameters

- token(private file share token)

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
"http://192.168.1.101:8082/files/6960d5a4/lib.md"
```

Errors

- 404 repo/token/file not found
- 520 OPERATION FAILED, fail to get file id by path

Get Private Shared File Detail

GET https://cloud.seafile.com/api2/s/f/{token}/detail/

Request parameters

- token(private file share token)

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{"repo_id": "989e3952-9d6f-4427-ab16-4bf9b53212eb", "name": "lib.md", "shared_by": "user@
```

Errors

- 404 repo/token/file not found
- 520 OPERATION FAILED, fail to get file id by path

Library

Library

Get Default Library

GET https://cloud.seafile.com/api2/default-repo/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "repo_id": "691b3e24-d05e-43cd-a9f2-6f32bd6b800e",
    "exists": true
}
```

Create Default Library

POST https://cloud.seafile.com/api2/default-repo/

Sample request

```
curl -X POST -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://
```

Sample response

```
{
    "repo_id": "691b3e24-d05e-43cd-a9f2-6f32bd6b800e",
    "exists": true
}
```

List Libraries

GET https://cloud.seafile.com/api2/repos/

Sample request

```
curl -H 'Authorization: Token 24fd3c026886e3121b2ca630805ed425c272cb96' -H 'Accept: appli
```

Sample response

```
[
{
    "permission": "rw",
    "encrypted": false,
    "mtime": 1400054900,
    "owner": "user@mail.com",
    "id": "f158d1dd-cc19-412c-b143-2ac83f352290",
    "size": 0,
    "name": "foo",
    "type": "repo",
    "virtual": false,
    "desc": "new library",
    "root": "0000000000000000000000000000000000000000"
},
{
    "permission": "rw",
    "encrypted": false,
    "mtime": 1400054802,
    "owner": "user@mail.com",
    "id": "0536b11a-a5fd-4482-9314-728cb3472f54",
    "size": 0,
    "name": "foo",
    "type": "repo",
    "virtual": false,
    "desc": "new library",
    "root": "0000000000000000000000000000000000000000"
}
]
```

Get Library Info

GET https://cloud.seafile.com/api2/repos/{repo-id}/

Request parameters

- repo-id

Sample request

```
curl -G -H 'Authorization: Token 24fd3c026886e3121b2ca630805ed425c272cb96' -H 'Accept: ap
```

Sample response

```
{
"encrypted": false,
"password_need": null,
"mtime": null,
"owner": "self",
"id": "632ab8a8-ecf9-4435-93bf-f495d5bfe975",
"size": 1356155,
"name": "org",
"root": "b5227040de360dd22c5717f9563628fe5510cbce",
"desc": "org file",
"type": "repo"
}
```

Get Library Owner

GET https://cloud.seafile.com/api2/repos/{repo-id}/owner/

Request parameters

- repo-id

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Accept: applic
```

Sample response

```
{
"owner": "user@example.com"
}
```

Errors

- 403 Permission error, only administrator can perform this action

Get Library History

GET https://cloud.seafile.com/api2/repos/{repo-id}/history/

Request parameters

- repo-id

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Accept: applic
```

Sample response

```
{"commits": [{"rev_file_size": 0, "rev_file_id": null, "ctime": 1398045167, "creator_name
```

Create Library

POST https://cloud.seafile.com/api2/repos/

Request parameters

- name
- desc (defaults to "new repo")
- passwd (needed by encrypt library)

Sample request

```
curl -v -d "name=foo&desc=new library" -H 'Authorization: Token f2210dacd9c6ccb8133606d94
```

Sample response

```
{
"encrypted": "",
"enc_version": 0,
"repo_id": "f15811fd-5c19-412c-b143-2ac83f352290",
"magic": "",
"relay_id": "c5e41170db250ea497075e2911104faf0105b7fb",
"repo_version": 1,
"relay_addr": "cloud.seafile.com",
"token": "c1f3defe9ba408cd7964427ec276843e9d10c23b",
"relay_port": "10001",
"random_key": "",
"email": "user@mail.com",
"repo_name": "foo"
}
```

Success

Response code 200 and newly created library information are returned.

Errors

- 400 Library name missing.
- 520 Operation failed.

Check/Create Sub Library

check if a dir has a corresponding sub_repo, if it does not have, create one

GET https://cloud.seafile.com/api2/repos/{repo-id}/dir/sub_repo/?p=/\&name=sub_lib

Request parameters

- repo-id
- p
- name

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Accept: applic
```

Sample response

```
{"sub_repo_id": "c0a3283c-013c-4a7c-8f68-006f06fa6dec"}
```

Errors

- 400 Argument missing
- 500 INTERNAL SERVER ERROR

Delete Library

DELETE https://cloud.seafile.com/api2/repos/{repo-id}/

Sample request

```
curl -v -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' -H '
```

Sample response

"success"

Errors

- 400 Library does not exist.

- 403 Only library owner can perform this operation.

Decrypt Library

POST https://cloud.seafile.com/api2/repos/{repo-id}/

Request parameters

- password

Sample request

```
curl -v -d "password=123" -H 'Authorization: Token e6a33d61954f219a96b60f635cf02717964e43
```

Sample response

"success"

Errors

- 400 Incorrect password
- 409 Repo is not encrypt
- 500 Internal server error

Create Public Library

POST https://cloud.seafile.com/api2/repos/{repo-id}/public/

Request parameters

- repo-id

Sample request

```
curl -X POST -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Accept
```

Sample response

```
...
< HTTP/1.0 200 OK
...
```

Success

```
Response code is 200(OK), and a string "success" is returned.
```

Errors

- 404 Repo not found
- 403 Forbid to access this repo
- 500 INTERNAL SERVER ERROR, Unable to make repo public

Remove Public Library

DELETE https://cloud.seafile.com/api2/repos/{repo-id}/public/

Request parameters

- repo-id

Sample request

```
curl -X DELETE -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Acce
```

Sample response

```
...
< HTTP/1.0 200 OK
...
```

Success

```
Response code is 200(OK), and a string "success" is returned.
```

Errors

- 404 Repo not found
- 403 Forbid to access this repo
- 500 INTERNAL SERVER ERROR, Unable to remove public repo

Fetch library download info

GET https://cloud.seafile.com/api2/repos/{repo-id}/download-info/

Request parameters

- repo-id

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd' -H 'Accept: applic
```

Sample response

```
{
"applet_root": "https://localhost:13420",
"relay_addr": "localhost",
"token": "46acc4d9ca3d6a5c7102ef379f82ecc1edc629e1",
"repo_id": "dae8cecc-2359-4d33-aa42-01b7846c4b32",
"relay_port": "10002",
"encrypted": "",
"repo_name": "test",
"relay_id": "8e4b13b49ca79f35732d9f44a0804940d985627c",
"email": "user@example.com"
}
```

List Virtual Libraries

GET https://cloud.seafile.com/api2/virtual-repos/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{"virtual-repos":
    [
        {"virtual_perm": "rw", "store_id": null, "worktree_invalid": false, "encrypted":
        {"virtual_perm": "rw", "store_id": null, "worktree_invalid": false, "encrypted":
    ]
}
```

Search Libraries

GET https://cloud.seafile.com/api2/search/

Request parameters

- q
- per_page (optional)

Sample request

```
curl -G -H 'Authorization: Token 24fd3c026886e3121b2ca630805ed425c272cb96' -H 'Accept: ap
```

Sample response

```
{
    "has_more": false,
    "total": 3,
    "results": [
        {
            "repo_id": "691b3e24-d05e-43cd-a9f2-6f32bd6b800e",
            "name": "api.md",
            "oid": "8ea78453bb474359cd9d8e2c4c4d8d9cbdcef0a2",
            "last_modified": 1398045167,
            "fullpath": "/api.md",
            "size": 18939
        },
        {
            "repo_id": "c5509062-9bca-4933-a7e0-c6da1d5f82be",
            "name": "home.md",
            "oid": "dda57aaffa5179829e064c7d0c142f47a8a65d3b",
            "last_modified": 1397096831,
            "fullpath": "/home.md",
            "size": 1954
        },
        {
            "repo_id": "c5509062-9bca-4933-a7e0-c6da1d5f82be",
            "name": "\u5e38\u89c1\u5b89\u88c5\u95ee\u9898.md",
            "oid": "8573f982eeb478b932a55ec13218f4f90a7c5a27",
            "last_modified": 1397188959,
            "fullpath": "/\u5e38\u89c1\u5b89\u88c5\u95ee\u9898.md",
            "size": 1050
        }
    ]
}
```

Errors

- 404 Search not supported.
- 400 Missing argument q.

File

Download File

GET https://cloud.seafile.com/api2/repos/{repo-id}/file/?p=/foo

Request parameters

- repo-id
- p

Sample request

```
curl  -v  -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' -H 'Accept:
```

Sample response

```
"https://cloud.seafile.com:8082/files/adee6094/foo.c"
```

Errors

- 400 Path is missing
- 404 File not found
- 520 Operation failed.

Get File Detail

GET https://cloud.seafile.com/api2/repos/{repo-id}/file/detail/?p=/foo.c

- repo-id
- p

Sample request

```
curl -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept: application/js
```

Sample response

```
{
"id": "013d3d38fed38b3e8e26b21bb3463eab6831194f",
"mtime": 1398148877,
"type": "file",
"name": "foo.py",
"size": 22
}
```

Errors

- 400 Path is missing
- 520 Operation failed.

Get File History

GET https://cloud.seafile.com/api2/repos/{repo-id}/file/history/?p=/foo.c

Request parameters

- repo-id
- p

Sample request

```
curl -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept: application/js
```

Sample response

```
{
"commits":
    [
        {
        "rev_file_size": 0,
        "repo_id": "a582d3bc-bcf5-421e-9125-741fa56d18d4",
        "ctime": 1398149763,
        "creator_name": "user@example.com",
        "creator": "0000000000000000000000000000000000000000",
        "root_id": "b64d413d9894c9206beac3faf9c2a0d75b4a8ebf",
        "rev_renamed_old_path": null,
        "parent_id": "8e546762e1657ab22dad83e9cb1e5ea31a767c9a",
        "new_merge": false,
        "version": 1,
        "conflict": false,
        "desc": "Added \"foo.c\"",
        "id": "9464f7499bfa7363d563282361339eaf96a93318",
        "rev_file_id": "0000000000000000000000000000000000000000",
        "second_parent_id": null
        },
        {
        "rev_file_size": 0,
        "repo_id": "a582d3bc-bcf5-421e-9125-741fa56d18d4",
        "ctime": 1398146059,
        "creator_name": "user@example.com",
        "creator": "0000000000000000000000000000000000000000",
        "root_id": "572413414257c76039897e00aeb35f819471206b",
        "rev_renamed_old_path": null,
        "parent_id": "f977bdb0ebb205645c3b42216c2817e511c3f68f",
        "new_merge": false,
        "version": 1,
        "conflict": false,
        "desc": "Added \"foo.c\"",
        "id": "a1ec20709675f4dc8db825cdbca296be245d189b",
        "rev_file_id": "0000000000000000000000000000000000000000",
        "second_parent_id": null
        }
    ]
}
```

Errors

- 400 Path is missing
- 404 File not found

Download File From a Revision

GET https://cloud.seafile.com/api2/repos/{repo-id}/file/revision/?p=/foo.c&commit_id=a1ec20709675f4dc8db825cdbca296be245d189b

Request parameters

- repo-id
- p
- commit_id

Sample request

```
curl -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept: application/js
```

Sample response

```
"https://cloud.seafile.com:8082/files/adee6094/foo.c"
```

Errors

- 400 Path is missing
- 404 Revision not found

Create File

POST https://cloud.seafile.com/api2/repos/{repo-id}/file/?p=/foo.c

Request parameters

- repo-id
- p
- operation

Sample request

```
curl -v -d "operation=create" -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99
```

Sample response

```
...
< HTTP/1.1 201 CREATED
...
"success"
```

Success

Response code is 201, and a string `"success"` is returned.

Errors

- 403 FORBIDDEN, You do not have permission to move file
- 520 OPERATION FAILED, fail to create file

Rename File

POST https://cloud.seafile.com/api2/repos/{repo-id}/file/?p=/foo.c

Request parameters

- repo-id
- p
- operation
- newname

Sample request

```
curl -v -d "operation=rename&newname=newfoo.c" -H 'Authorization: Token f2210dacd9c6ccb81
```

Sample response

```
...
< HTTP/1.1 301 MOVED PERMANENTLY
...
"success"
```

Success

Response code is 301, and a string `"success"` is returned.

Errors

- 400 BAD REQUEST, Path is missing or invalid(e.g. p=/) or newname is missing(newname too long)
- 403 FORBIDDEN, You do not have permission to rename file
- 404 NOT FOUND, repo not found
- 409 CONFLICT, the newname is the same to the old
- 520 OPERATION FAILED, fail to rename file

Lock File

PUT https://cloud.seafile.com/api2/repos/{repo-id}/file/

Request parameters

- repo-id
- p
- operation

Sample request

```
curl -v -X PUT -d "operation=lock&p=/foo.c" -H 'Authorization: Token f2210dacd9c6ccb81336
```

Sample response

```
...
< HTTP/1.0 200 OK
...
"success"
```

Success

Response code is 200, and a string `"success"` is returned.

Errors

- 400 BAD REQUEST, Path is missing or invalid(e.g. p=/)
- 403 FORBIDDEN, You do not have permission to lock file
- 404 NOT FOUND, repo not found
- 520 OPERATION FAILED, fail to lock file

Unlock File

PUT https://cloud.seafile.com/api2/repos/{repo-id}/file/

Request parameters

- repo-id
- p
- operation

Sample request

```
curl -v -X PUT -d "operation=unlock&p=/foo.c" -H 'Authorization: Token f2210dacd9c6ccb813
```

Sample response

```
...
< HTTP/1.0 200 OK
...
"success"
```

Success

Response code is 200, and a string `"success"` is returned.

Errors

- 400 BAD REQUEST, Path is missing or invalid(e.g. p=/)
- 403 FORBIDDEN, You do not have permission to lock file
- 404 NOT FOUND, repo not found
- 520 OPERATION FAILED, fail to unlock file

Move File

POST https://cloud.seafile.com/api2/repos/{repo-id}/file/?p=/foo.c

Request parameters

- repo-id
- p
- operation
- dst_repo
- dst_dir

Sample request

```
curl -v -d "operation=move&dst_repo=affc837f-7fdd-4e91-b88a-32caf99897f2&dst_dir=/" -H 'A
```

Sample response

```
...
< HTTP/1.1 301 MOVED PERMANENTLY
...
"success"
```

Success

Response code is 301, and a string `"success"` is returned.

Errors

- 400 BAD REQUEST, Path is missing or invalid(e.g. p=/)
- 403 FORBIDDEN, You do not have permission to move file
- 404 NOT FOUND, repo not found
- 500 INTERNAL SERVER ERROR

Copy File

POST https://cloud.seafile.com/api2/repos/{repo_id}/fileops/copy/

Request parameters

- p: source folder path, defaults to `"/"`
- file_names: list of file/folder names to copy. Multiple file/folder names can be seperated by `:` .
- dst_repo: the destination repo id
- dst_dir: the destination folder in `dst_repo`

Sample request

```
curl -d "dst_repo=73ddb2b8-dda8-471b-b7a7-ca742b07483c&dst_dir=/&file_names=foo.c" -H 'Au
```

Sample response

```
"success"
```

Errors

- 400 missing argument
- 403 You do not have permission to copy file
- 404 repo not found
- 502 failed to copy file

Revert File

PUT https://cloud.seafile.com/api2/repos/{repo_id}/file/revert/

Request parameters

- repo_id
- p
- commit_id

Sample request

```
curl -v -X PUT -d "commit_id=a1ec20709675f4dc8db825cdbca296be245d189b&p=/foo.c" -H "Autho
```

Sample response

```
...
< HTTP/1.0 200 OK
...

{"ret": 0}
```

Success

```
Response code 200(OK) is returned.
```

Errors

- 400 Path is missing

Delete File

DELETE https://cloud.seafile.com/api2/repos/{repo-id}/file/?p=/foo

Request parameters

- repo-id
- p

Sample request

```
curl -X DELETE -v  -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept:
```

Sample response

```
...
< HTTP/1.0 200 OK
...
"success"
```

Errors

- 400 Path is missing
- 520 Operation failed.

Note

This can also be used to delete directory.

Upload File

Get Upload Link

GET https://cloud.seafile.com/api2/repos/{repo-id}/upload-link/

Request parameters

- repo-id

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" https://cloud.sea
```

Sample response

```
"http://cloud.seafile.com:8082/upload-api/ef881b22"
```

Errors

```
500 Run out of quota
```

Upload File

After getting the upload link, POST to this link for uploading files.

POST http://cloud.seafile.com:8082/upload-api/ef881b22

Errors

```
400 Bad request
440 Invalid filename
441 File already exists
500 Internal server error
```

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -F file=@test.txt
```

Sample response

```
"adc83b19e793491b1c6ea0fd8b46cd9f32e592fc"
```

Note

For python client uploading, see https://github.com/haiwen/webapi-examples/blob/master/python/upload-file.py

Update file

Get Update Link

GET https://cloud.seafile.com/api2/repos/{repo-id}/update-link/

Request parameters

- repo-id

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" https://cloud.sea
```

Sample response

```
"http://cloud.seafile.com:8082/update-api/ef881b22"
```

Errors

```
500 Run out of quota
```

Update File

After getting the upload link, POST to this link for uploading files.

POST http://cloud.seafile.com:8082/update-api/ef881b22

Request parameters

- target_file

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" -F file=@test.txt
```

Returns

The id of the updated file

Sample response

```
"adc83b19e793491b1c6ea0fd8b46cd9f32e592fc"
```

Errors

- 400 Bad request
- 440 Invalid filename
- 500 Internal server error

Get Upload Blocks Link

GET https://cloud.seafile.com/api2/repos/{repo-id}/upload-blks-link/

Request parameters

- repo-id

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" https://cloud.sea
```

Sample response

```
"https://cloud.seafile.com/seafhttp/upload-blks-api/c1e6823d"
```

Errors

- 403 Can not access repo
- 520 above quota

Get Update Blocks Link

GET https://cloud.seafile.com/api2/repos/{repo-id}/update-blks-link/

Request parameters

- repo-id

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd" https://cloud.sea
```

Sample response

```
"https://cloud.seafile.com/seafhttp/update-blks-api/c1e6823d"
```

Errors

- 403 Can not access repo
- 520 above quota

Directory

List Directory Entries

GET https://cloud.seafile.com/api2/repos/{repo-id}/dir/

- repo-id
- p (optional): The path to a directory. If `p` is missing, then defaults to '/' which is the top directory.
- oid (optional)

Sample request

```
curl -H "Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d9b477fd" -H 'Accept: applic
```

Sample response

If oid is the latest oid of the directory, returns `"uptodate"` , else returns

```
[
{
    "id": "0000000000000000000000000000000000000000",
    "type": "file",
    "name": "test1.c",
    "size": 0
},
{
    "id": "e4fe14c8cda2206bb9606907cf4fca6b30221cf9",
    "type": "dir",
    "name": "test_dir"
}
]
```

Errors

- 404 The path is not exist.
- 440 Repo is encrypted, and password is not provided.
- 520 Operation failed..

Create New Directory

POST https://cloud.seafile.com/api2/repos/{repo-id}/dir/

- repo-id
- p
- operation=mkdir (post)

Sample request

```
curl -d  "operation=mkdir" -v  -H 'Authorization: Tokacd9c6ccb8133606d94ff8e61d99b477fd'
```

Sample response

```
...
< HTTP/1.0 201 CREATED
< Location: https://cloud.seafile.com/api2/repos/dae8cecc-2359-4d33-aa42-01b7846c4b32/dir
...

"success"
```

Success

Response code 201(Created) is returned, and Location header provides the url of created directory.

Errors

- 400 Path is missing or invalid(e.g. p=/)
- 520 Operation failed.

Notes

Newly created directory will be renamed if the name is duplicated.

Rename Directory

POST https://cloud.seafile.com/api2/repos/{repo-id}/dir/

Parameters

- p (path)
- operation=rename (post)
- newname (the new name for directory)

Sample request

```
curl -d  "operation=rename&newname=pinkfloyd_newfolder" -v  -H 'Authorization: Tokacd9c6c
```

Success

Response code 200 if everything is ok

Errors

- 403 if You do not have permission to rename a folder
- 400 if newname is not given
- 520 if Failed to rename directory (generic problem)

Notes

If the new name is the same of the old name no operation will be done.

Delete Directory

DELETE https://cloud.seafile.com/api2/repos/{repo-id}/dir/

- repo-id
- p

Sample request

```
curl -X DELETE -v  -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept:
```

Sample response

```
...
< HTTP/1.0 200 OK
...
"success"
```

Success

Response code is 200(OK), and a string `"success"` is returned.

Errors

- 400 Path is missing or invalid(e.g. p=/)
- 520 Operation failed.

Note

This can also be used to delete file.

Download Directory

GET https://cloud.seafile.com/api2/repos/{repo-id}/dir/download/?p=/foo

- repo-id
- p

Sample request

```
curl -H 'Authorization: Token f2210dacd3606d94ff8e61d99b477fd' -H 'Accept: application/js
```

Sample response

```
"https://cloud.seafile.com:8082/files/adee6094/foo"
```

Errors

- 400 Path is missing or invalid(e.g. p=/), or unable to download directory, size is too large
- 404 Repo(path) not found(exist)
- 520 Operation failed.

Share Directory

POST https://cloud.seafile.com/api2/repos/{repo-id}/dir/share/

- repo-id
- emails
- s_type
- path
- perm

Sample request

```
curl -v -X POST -d "emails=user@example.com&s_type=d&path=/dir&perm=r" -H 'Authorization:
```

Sample response

```
...
< HTTP/1.0 200 OK
...
```

Success

Response code is 200(OK).

Batch Delete

Pipelining over HTTP/1.1 can be used to delete multiple files and directories without losing performance.

A sample request looks like `curl -X DELETE https://cloud.seafile.com/api2/repos/{repo-id}/dir/?p=/foo http://cloud.seafile.com/api2/repos/{repo-id}/dir/?p=/bar` . This code snippet shows how to use Python client to batch delete multiple files and directories. See http://cloud.seafile.com/f/f7fd5d5b9d/

Get Avatar

Get User Avatar

GET https://cloud.seafile.com/api2/avatars/user/{user}/resized/{size}/

Request parameters

- user
- size

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "url": "http://127.0.0.1:8000/media/avatars/default.png",
    "is_default": true,
    "mtime": 0
}
```

Get Group Avatar

GET https://cloud.seafile.com/api2/avatars/group/{group_id}/resized/{size}/

Request parameters

- group_id
- size

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "url": "http://127.0.0.1:8000/media/avatars/groups/default.png",
    "is_default": true,
    "mtime": 0
}
```

Get Thumbnail

Get Thumbnail Image

GET https://cloud.seafile.com/api2/repos/{repo_id}/thumbnail/

Request parameters

- repo_id
- p
- size

Sample request

```
curl -H 'Authorization: Token 40f9a510a0629430865dc199a3880898ad2e48fc' https://cloud.sea
```

List Group And Contacts

GET https://cloud.seafile.com/api2/groupandcontacts/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
{
    "contacts": [
        {
            "msgnum": 0,
            "mtime": 0,
            "lastmsg": null,
            "email": "user@example.com",
            "name": "123"
        }
    ],
    "umsgnum": 0,
    "replynum": 0,
    "groups": [
        {
            "ctime": 1398134171327948,
            "creator": "user@example.com",
            "msgnum": 0,
            "mtime": 0,
            "lastmsg": null,
            "id": 1,
            "name": "lian"
        }
    ],
    "gmsgnum": 0,
    "newreplies": []
}
```

Get File Activities

GET https://cloud.seafile.com/api2/events/

Sample request

```
curl -H 'Authorization: Token f2210dacd9c6ccb8133606d94ff8e61d99b477fd' "https://cloud.se
```

Sample response

```
 {"more_offset": 16, "events":[{"repo_id": "6f3d28a4-73ae-4d01-a727-26774379dcb9", "autho
```

Add Organization

POST https://cloud.seafile.com/api2/organization/

Request parameters

- username
- password
- org_name
- prefix
- quota
- member_limit

Sample request

```
curl -v -X POST -d "username=example@example.com&password=example&org_name=example&prefix
```

Sample response

```
"success"
```

Python API

Seafile Python API

This tutorial show you how to use seafile-api, and will accomplish a "library copy" work under **Ubuntu** as example.

Install Seafile Server

First of all, make sure you have Download and Setup Seafile Server successfully. And your directory layout will be like this:

```
# tree . -L 3
.
├── ccnet
│   ├── ccnet.conf
│   ├── ......
│......
├── seafile-server-3.0.3
│   ├── seafile
│   ├── seafile.sh
│   ├── seahub
│   ├── seahub.sh
│   ├── setup-seafile.sh
│   ├── upgrade
│       ├── README
│       ├── seaf_migrate_3.py
│       ├── seaf_migrate_3.sh
│       ├── ......
│   ├── ......
│......
```

Example: Copy Library

In this example, two script files will be used: `seaf_migrate_3.sh` and `seaf_migrate_3.py`. We put them in the **upgrade** directory as you see above.

Set Environment Variable

If you want use Seafile-API, set environment variable first. That's what `seaf_migrate_3.sh` does:

1. get ccnet/seafile config file path and export them;
2. export Python path;
3. call `seaf_migrate_3.py`.

Example code

```
#!/bin/bash

#get path of ccnet.conf
SCRIPT=$(readlink -f "$0") # haiwen/seafile-server-3.0.3/upgrade/seaf_migrate_3.sh
UPGRADE_DIR=$(dirname "$SCRIPT") # haiwen/seafile-server-3.0.3/upgrade/
INSTALLPATH=$(dirname "$UPGRADE_DIR") # haiwen/seafile-server-3.0.3/
TOPDIR=$(dirname "${INSTALLPATH}") # haiwen/
default_ccnet_conf_dir=${TOPDIR}/ccnet

#get path of seafile.conf
function read_seafile_data_dir () {
    seafile_ini=${default_ccnet_conf_dir}/seafile.ini
    if [[ ! -f ${seafile_ini} ]]; then
        echo "${seafile_ini} not found. Now quit"
        exit 1
    fi
    seafile_data_dir=$(cat "${seafile_ini}")
    if [[ ! -d ${seafile_data_dir} ]]; then
        echo "Your seafile server data directory \"${seafile_data_dir}\" is invalid or do
        echo "Please check it first, or create this directory yourself."
        echo ""
        exit 1;
    fi

    export SEAFILE_CONF_DIR=$seafile_data_dir
}

export CCNET_CONF_DIR=${default_ccnet_conf_dir}
read_seafile_data_dir;

export PYTHONPATH=${INSTALLPATH}/seafile/lib/python2.6/site-packages:${INSTALLPATH}/seafi
export PYTHONPATH=${INSTALLPATH}/seafile/lib/python2.7/site-packages:${INSTALLPATH}/seafi

function usage () {
    echo "Usage: `basename $0` <repo-id>"
    echo "exit."
    exit 1
}
if [ $# != 1 ]; then
    usage
fi

python seaf_migrate_3.py $1
```

> **NOTE:** You can get `repo_id` at address bar of Seahub or through Seafile web API

Copy Library

Then `seaf_migrate_3.py` will call Seafile-API to copy library:

1. Get library ID from input.
2. Get origin_repo object.
3. Create a new library, set name, desc and owner.
4. Copy stuffs from old library to new library.

Example code

```
#!/usr/bin/env python

import os
import stat
import sys
from seaserv import seafile_api

def count_files_recursive(repo_id, path='/'):
    num_files = 0
    for e in seafile_api.list_dir_by_path(repo_id, path):
        if stat.S_ISDIR(e.mode):
            num_files += count_files_recursive(repo_id,
                                               os.path.join(path, e.obj_name))
        else:
            num_files += 1
    return num_files

#Get library ID from input
origin_repo_id = sys.argv[1]

#Get origin_repo object
origin_repo = seafile_api.get_repo(origin_repo_id)
username = seafile_api.get_repo_owner(origin_repo_id)

#Create a new library, set name, desc and owner
new_repo_id = seafile_api.create_repo(name=origin_repo.name,
                                      desc=origin_repo.desc,
                                      username=username, passwd=None)

#Copy stuffs from old library to new library
dirents = seafile_api.list_dir_by_path(origin_repo_id, '/')
for e in dirents:
    print "copying: " + e.obj_name
    obj_name = e.obj_name
    seafile_api.copy_file(origin_repo_id, '/', obj_name, new_repo_id, '/',
                          obj_name, username, 0, 1)

print "*" * 60
print "OK, verifying..."
print "Origin library(%s): %d files. New Library(%s): %d files." % (
    origin_repo_id[:8], count_files_recursive(origin_repo_id),
    new_repo_id[:8], count_files_recursive(new_repo_id))
print "*" * 60
```

If you execute script file successfully, you will see these output, and of course a new library at myhome page of Seahub.

```
foo@foo:~/haiwen/seafile-server-3.0.3/upgrade$ ./seaf_migrate_test.sh c8bbb088-cbaf-411d-
Loading ccnet config from /home/foo/haiwen/ccnet
Loading seafile config from /home/foo/haiwen/seafile-data
copying: test.html
copying: test-dir-2
copying: test-dir
copying: solar.html
copying: examples.desktop
*************************************************************
OK, verifying...
Origin library(c8bbb088): 10 files. New Library(4d6f4837): 10 files.
*************************************************************
```

List Of Seafile-API

This list is based on **seafile-server-3.0.3**, and parameter was omitted.

For more infomation about Seafile-API, please see api.py.

> - seafile_api.add_inner_pub_repo()
> - seafile_api.cancel_copy_task()
> - seafile_api.change_repo_passwd()
> - seafile_api.check_passwd()
> - seafile_api.check_permission()
> - seafile_api.check_quota()
> - seafile_api.check_repo_access_permission()
> - seafile_api.copy_file()
> - seafile_api.count_inner_pub_repos()
> - seafile_api.create_enc_repo()
> - seafile_api.create_repo()
> - seafile_api.create_virtual_repo()
> - seafile_api.del_file()
> - seafile_api.delete_repo_token()
> - seafile_api.delete_repo_tokens_by_peer_id()
> - seafile_api.diff_commits()
> - seafile_api.edit_repo()
> - seafile_api.generate_repo_token()
> - seafile_api.get_commit_list()
> - seafile_api.get_copy_task()
> - seafile_api.get_decrypt_key()
> - seafile_api.get_deleted()
> - seafile_api.get_dir_id_by_commit_and_path()
> - seafile_api.get_dir_id_by_path()
> - seafile_api.get_file_id_by_commit_and_path()

> - seafile_api.get_file_id_by_path()
> - seafile_api.get_file_revisions()
> - seafile_api.get_file_size()
> - seafile_api.get_files_last_modified()
> - seafile_api.get_group_repo_list()
> - seafile_api.get_group_repoids()
> - seafile_api.get_group_repos_by_owner()
> - seafile_api.get_fileserver_access_token()
> - seafile_api.get_inner_pub_repo_list()
> - seafile_api.get_orphan_repo_list()
> - seafile_api.get_owned_repo_list()
> - seafile_api.get_repo()
> - seafile_api.get_repo_list()
> - seafile_api.get_repo_owner()
> - seafile_api.get_repo_size()
> - seafile_api.get_share_in_repo_list()
> - seafile_api.get_share_out_repo_list()
> - seafile_api.get_shared_groups_by_repo()
> - seafile_api.get_user_quota()
> - seafile_api.get_user_self_usage()
> - seafile_api.get_user_share_usage()
> - seafile_api.get_virtual_repo()
> - seafile_api.get_virtual_repos_by_owner()
> - seafile_api.group_share_repo()
> - seafile_api.group_unshare_repo()
> - seafile_api.is_inner_pub_repo()
> - seafile_api.is_password_set()
> - seafile_api.is_repo_owner()
> - seafile_api.is_valid_filename()
> - seafile_api.list_dir_by_commit_and_path()
> - seafile_api.list_dir_by_dir_id()
> - seafile_api.list_dir_by_path()
> - seafile_api.list_file_by_file_id()
> - seafile_api.list_repo_tokens()
> - seafile_api.list_repo_tokens_by_email()
> - seafile_api.move_file()
> - seafile_api.post_dir()
> - seafile_api.post_empty_file()
> - seafile_api.post_file()
> - seafile_api.put_file()
> - seafile_api.query_fileserver_access_token()
> - seafile_api.remove_inner_pub_repo()
> - seafile_api.remove_repo()
> - seafile_api.remove_share()
> - seafile_api.rename_file()
> - seafile_api.revert_dir()

> - seafile_api.revert_file()
> - seafile_api.revert_repo()
> - seafile_api.set_group_repo_permission()
> - seafile_api.set_passwd()
> - seafile_api.set_repo_owner()
> - seafile_api.set_share_permission()
> - seafile_api.set_user_quota()
> - seafile_api.share_repo()
> - seafile_api.unset_passwd()

Data Model

Seafile internally uses a data model similar to GIT's. It consists of `Repo` , `Branch` , `Commit` , `FS` , and `Block` .

Repo

A repo is also called a library. Every repo has an unique id (UUID), and attributes like description, creator, password.

Branch

Unlike git, only two predefined branches is used, i.e., `local` and `master` .

In PC client, modifications will first be committed to the `local` branch. Then the `master` branch is downloaded from server, and merged into `local` branch. After that the `local` branch will be uploaded to server. Then the server will fast-forward its `master` branch to the head commit of the just uploaded branch.

When users update a repo on the web, modifications will first be committed to temporary branch on the server, then merged into the `master` branch.

Commit

Like in GIT.

FS

There are two types of FS objects, `SeafDir Object` and `Seafile Object` . `SeafDir Object` represents a directory, and `Seafile Object` represents a file.

Block

A file is further divided into blocks with variable lengths. We use Content Defined Chunking algorithm to divide file into blocks. A clear overview of this algorithm can be found at http://pdos.csail.mit.edu/papers/lbfs:sosp01/lbfs.pdf. On average, a block's size is around 1MB.

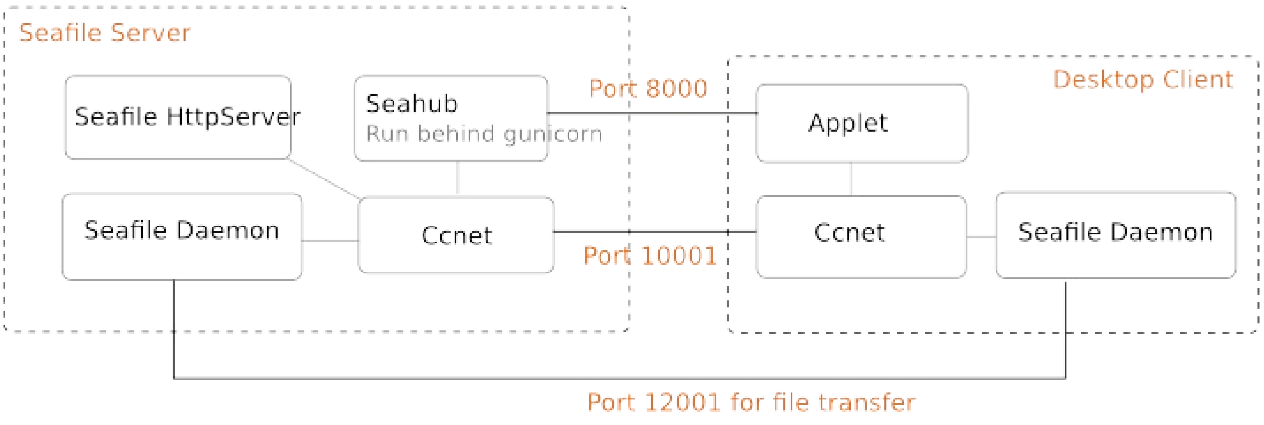
This mechanism makes it possible to deduplicate data between different versions of frequently updated files, improving storage efficiency. It also enables transferring data to/from multiple servers in parallel.

Components of Seafile Server

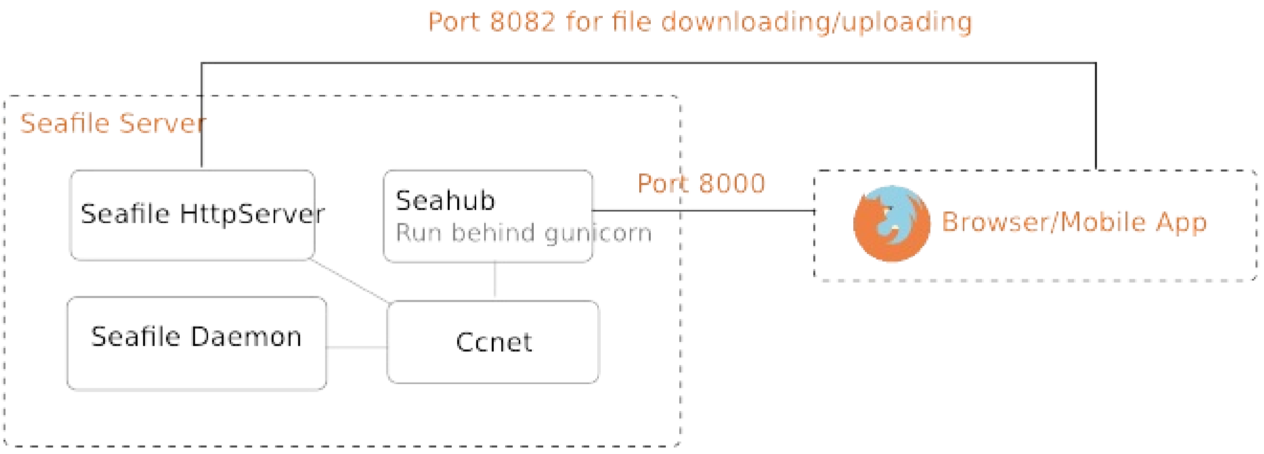
Seafile server comprises of the following services.

- **Ccnet daemon** (ccnet for client side or ccnet-server for server side)：networking service daemon. In our initial design, Ccnet worked like a traffic bus. All the network traffic between client, server and internal traffic between different components would go through Ccnet. After further development we found that file transfer is improved by utilizing the Seafile daemon component directly.
- **Seafile daemon**：data service daemon
- **Seahub**：the website. Seafile server package contains a light-weight Python HTTP server `gunicorn` that serves the website. Seahub runs as an application within gunicorn.
- **FileServer**: handles raw file upload/download functions for Seahub. Due to Gunicorn being poor at handling large files, so we wrote this "FileServer" in the C programming language to serve raw file upload/download.
- **Controller**: monitors ccnet and Seafile daemons, restarts them if necessary.

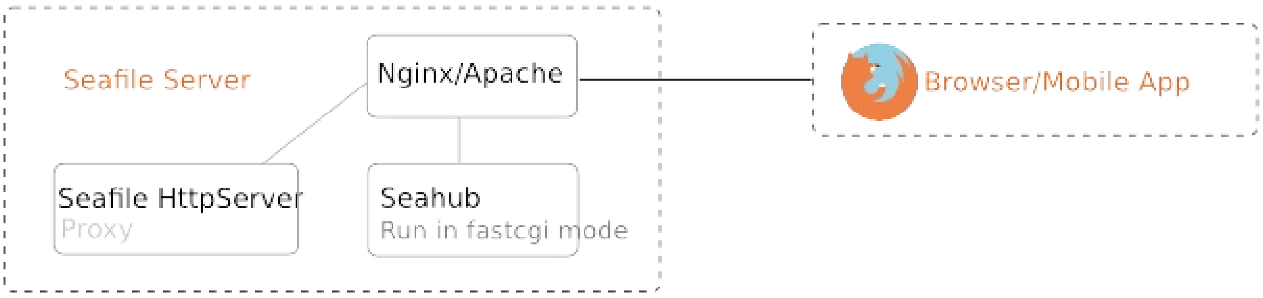
The picture below shows how Seafile desktop client syncs files with Seafile server:



The picture below shows how Seafile mobile client interacts with Seafile server:



The picture below shows how Seafile mobile client interacts with Seafile server if the server is configured behind Nginx/Apache:



Synchronization algorithm

This article tries to give an overview on Seafile's file synchronization algorithm. For clarity, some details are deliberately omitted, but it should help you get the big picture.

To better understand this article, you should first read [[Seafile data model]].

The Basic Work Flow

Each downloaded repo is bound to an ordinary local folder. Using Git's terminology, we call this local folder the "worktree".

A typical synchronization work flow consists of the following steps:

1. Seafile client daemon detects changes in the worktree (via inotify etc).
2. The daemon commits the changes to the `local` branch.
3. Download new changes from the `master` branch on the server (if any).
4. Merge the downloaded branch into `local` branch (also checkout changes to worktree).
5. Fast-forward upload `local` branch to server's `master` branch.

Since the above work flow may be interrupted at any point by shutting down the program or computer, after reboot we lose all notifications from the OS. We need a reliable and efficient way to determine which files in the worktree has been changed (even after reboots).

We use Git's index file to do this. It caches the timestamps of every file in the worktree when the last commit is generated. So we can easily and reliably detect changed files in the worktree since the latest commit by comparing timestamps.

Another notable case is what happens if two clients try to upload to the server simultaneously. The commit procedure on the server ensures atomicity. So only one client will update the `master` branch successfully, while the other will fail.

The failing client will restart the sync work flow later. It will first merge the changes from the succeeded client then upload again.

Merge

The most tricky part of the syncing algorithm is merging.

Git's merge algorithm doesn't work well enough for auto synchronization.

Firstly, if a merge is interrupted, git requires you to reset to the latest commit and merge again. It's not a problem for Git since it's a single command. But seafile runs as a daemon and may be kill at any time. The user may have changed some files in the worktree between the interruption and restart. Resetting the worktree will LOSE user's uncommitted data.

Secondly, Git's merge command will fail if it fails to update a file in the worktree. But on Windows, an opened Office document will be write-protected by the Office process. So the merge may fail in this case.

That's why programs use Git directly for auto-sync is not reliable.

Seafile implement its own merge algorithm based on the ideas from Git's merge algorithm.

It handles the first problem by "redoing" the merge carefully after restart. It handles the second problem by not starting merge until no file is write-protected in the worktree.

Seafile's merge algorithm also handles all the conflict cases handled by Git.