# A survey on digital forensics investigation of Seafile as a cloud storage

**Kayvan Atefi, Saadiah Yahya, Arash Atefi**

***Abstract*— The internet and digital devices like portable media players, smart phones and digital cameras are now being used by many different people for a variety of reasons. Unfortunately, criminals are also designing their own uses to assist them in committing their crimes. As with most new technologies cloud storage services have the capacity to be used for criminal exploitation and to form the basis of civil litigation. As such those working in both digital forensics and eDiscovery must have the capability to forensically analyze these storage platforms. The storage as a service (StaaS) cloud computing architecture is showing significant growth as users adopt the capability to store data in the cloud environment across a range of devices. Cloud (storage) forensics has recently emerged as a salient area of inquiry. Nowadays with using a widely open source cloud StaaS application, researcher will document a series of digital forensic experiments with the aim of providing forensic researchers and practitioners with an in-depth understanding of the artifacts required to undertake cloud storage forensics. This research will focus upon possible artifacts, which will categories the potential evidential data specified before commencement of the experiments. In this research, researchers will try to find a number of digital forensic artifacts as part of the experiments that will be used to support the selection of artifact categories and provide a technical summary to practitioners of artifact types. Finally general guidelines for future forensic analysis on open source StaaS products and recommendations for future work will be proposed.**

***Index Terms*— Cloud Computing, Digital Forensics in Clouds, Seafile, Seacloud, Private cloud, SaaS, Storage as a service, Cloud forensics, Cloud storage forensics, Open source cloud.**

## I. INTRODUCTION

According to reference [4], Cloud storage services are increasingly used by government, businesses, and consumers to store and access an increasing amount of information. For example, a recent audit by US Government Accountability Office found that many agencies has also identified opportunities for future cloud implementations, such as moving storage and help desk services to a cloud environment' [1].

Law enforcement now has a great demand for investigators with a technology background to join the computer forensics field. This branch of forensic science is responsible for searching through and recovering evidence from digital sources. The evidence is not limited to files or emails found on a computer, but can be from any digital device that stores data. This field has become a vital tool in the fight against crime in cases where little or no physical evidence like DNA or fingerprints exists. Unlike physical evidence, many crooks are aware of the digital trail they left behind and attempt to delete whatever evidence they can [3].

In additional Computer forensics has always been a field which is growing alongside technology. As networks become more and more available and data transfer through networks getting faster, the risks involved gets higher. Malicious software, tools and methodologies are designed and implemented every day to exploit networks and data storage associated with them to extract useful private information that can be used in various crimes. This is where computer forensics and security comes in. The field applies to scientifically collect, preserve, and recover latent evidence from crime scenes with techniques and tools [2]. Computer forensics is the science of identifying, analyzing, preserving, documenting and presenting evidence and information from digital and electronic devices, and it is meant to preserve the privacy of users from being exploited.

Security and privacy issues associated with cloud services are generally better documented and understood than digital forensic issues. By physically displacing the storage from the user cloud storage solutions introduce numerous challenges for digital forensic and eDiscovery practitioners. For example, a report by the European Network and Information Security Agency (ENISA) explained that multi-tenant outsourced services usually cannot give access to raw log data as it contains records of multiple users and thus would compromise the privacy of other customers' (ENISA, 2012, p.45) and, therefore, features synonymous with cloud (storage) services such as multi-tenancy, data security, file encryption and communications encryption also need to be addressed as part of a digital forensics investigation as suggested by various other researchers[7][6][8].

## II. OVERVIEW OF SEAFILE

### 2.1 Introduction

Seafile is an open source cloud storage system with advanced support for file syncing, privacy protection and teamwork. Collections of files are called libraries, and each library can be

Manuscript received Oct 19, 2014

**Kayvan Atefi** , Faculty of Computer and Mathematic science, University Technology Mara (UiTM), Shah Alam, Malaysia

**Prof. Dr Saadiah Yahya**, Faculty of Computer and Mathematic science, University Technology Mara (UiTM), Shah Alam, Malaysia

**Arash Atefi**, Electrical Engineering Department, Science and Research Branch, Islamic Azad University, Kermanshah, Iran

synced separately. A library can be encrypted with a user chosen password. This password is not stored on the server, so even the server admin cannot view a file's contents. Seafile allows users to create groups with file syncing, wiki, and discussion to enable easy collaboration around documents within a team (seafile.com, 2014).

### 2.2 Components Overview

Based on [5], Seafile server and client consist of several components. Understanding how they work together will save you a lot time in deploying and maintaining Seafile.

#### 2.2.1 Server

- Seahub (django) ：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 (fileserver) (It is called HttpServer before version 3.1): 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.
- Seafile server (seaf-server) ：data service daemon
- Ccnet server (ccnet-server) ：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. Fig.1and Fig.2 shows how Seafile desktop client and mobile client, syncs files with Seafile server.

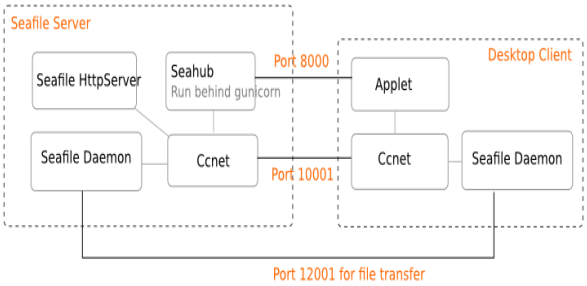


Fig. 1 Seafile desktop client syncs files with Seafile server

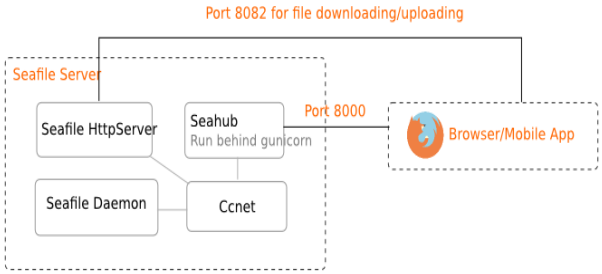


Fig. 2 Seafile mobile client interacts with Seafile server

#### 2.2.2 Client

- Applet (seafile-applet): The GUI front-end
- Seafile daemon (seafile): data service daemon for client
- Ccnet daemon (ccnet): networking service daemon for client

### 2.3 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.

### 2.4 Seafile-server-performance

#### *2.4.1 Single client uploading*

Server parameters:

AMD 2.5GHz CPU 1Core
1GB memory
7200RPM SATA Disk
100Mbps network
Operating system: Ubuntu 12.04
Seafile server uses MySQL as database
Scenario: Single client uploading a file of size 10GB

Table 1.Result of single client uploading a file of size 10GB

| item | value |
|---|---|
| Uploading speed | 10MB/s |
| CPU Usage | 60% |

Result: Table 1. Shows result of single client uploading a file of size 10GB in this case most of the CPU spends in copying file contents in memory.

#### 2.4.2 Multiple clients checking library state

In this Scenario purpose is to create a library in the server, using a python script to simulate 3000 clients checking the library state.

**Server parameters: as above (2.4.1)**

Result: The client will check the library state every 30 seconds. If the state is changed, a syncing operation will be performed. This tests the cost of checking. Table 2 shows result of Multiple clients checking library state.

Table 2. Result of multiple clients checking library state

| item | value |
|---|---|
| CPU Usage | 40%, 20% used by MySQL |

#### 2.4.3 Syncing a lot of small files

Server parameters:

AMD 3.0GHz CPU 1Core
2GB memory
7200RPM SATA Disk
100Mbps network
Operating system: Ubuntu 12.04
Seafile server uses MySQL as database

Result: in this part scenario is attempt to uploading a library with 11000 small text files (145M in total) and downloading a library with 11000 small text files (145M in total). Table 3 Shows result of syncing a lot of small files.

**Table 3 Result of syncing a lot of small files.**

| item | value |
|---|---|
| CPU Usage | 30% ~ 40% |
| Network usage | 500KB~4MB/s |

### 2.4.4 Syncing load testing with 100 clients read

***Server parameters: as above (2.4.3)***

Result: in this phase scenario is for create a library in the server, 100 clients synced this library and add a 10MB file to a client, other clients begins to sync the library after the file is uploaded. Table 4 shows Syncing load testing with 100 clients read.

**Table 4. Syncing load testing with 100 clients read.**

| item | value |
|---|---|
| CPU Usage | 20% ~ 35% |
| Network usage | 6MB/s |

### 2.4.5 Syncing load testing with 100 clients write

***Server parameters: as above (2.4.3)***

Result: in this scenario 100 clients write one 16k file (all data come from /dev/urandom individually) in this case the initial CPU cost is high, because the server has to merge the modification from the clients. Table 5 shows result of Syncing load testing with 100 clients write.

**Table 5 .Result of Syncing load testing with 100 clients write.**

| item | value |
|---|---|
| CPU Usage | 50% ~ 90% in the first 40 seconds, then drop to 20%~30% |
| Network usage | 6MB/s |

### 2.5 Deploying Seafile under Linux

### A. Deploying Seafile with SQLite

In this part researcher consider deploy Seafile in Home/Personal Environment with SQLite under Linux.

- Directory Layout

**Before install seafile :**

```
# 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
```

**After install Seafile :**

```
#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 configuration options**

| Option | Description | Note |
|---|---|---|
| server name | Name of this seafile server | 3-15 characters, only English letters, digits and underscore ('_') are allowed |
| server ip or domain | The IP address or domain name used by this server | Seafile client program will access the server with this address |
| ccnet server port | The TCP port used by ccnet, the underlying networking service of Seafile | Default is 10001. If it's been used by other service, you can set it to another port. |
| seafile data dir | Seafile stores your data in this directory. By default it'll be placed in the current directory. | The size of this directory will increase as you put more and more data into Seafile. Please select a disk partition with enough free space. |
| seafile server port | The TCP port used by Seafile to transfer data | Default is 12001. If it's been used by other service, you can set it to another port. |
| fileserver port | The TCP port used by Seafile fileserver | Default is 8082. If it's been used by other service, you can set it to another port. |

### B. Deploy Seahub/FileServer with Nginx

Seahub is the web interface of Seafile server. File Server is used to handle raw file uploading/downloading through browsers. By default, it listens on port 8082 for HTTP request. Locations for Nginx config file are:

access_log /var/log/nginx/seahub.access.log;
error_log /var/log/nginx/seahub.error.log;
media /root /home/user/haiwen/seafile-server-latest/seahub;

Also location for modify the value of SERVICE_URL is in the /data/haiwen/ccnet/ccnet.conf

Nginx settings "client_max_body_size" is by default 1M. Uploading a file bigger than this limit will give you an error message HTTP error code 413 ("Request Entity Too Large").You should use 0 to disable this feature or write the same value than for the parameter max_upload_size in section [fileserver] of /seafile/seafile-data/seafile.conf

### 2.5.4 Firewall settings:

By default, 4 ports in your firewall are open which if run Seafile behind Nginx/Apache with HTTPS, port of https also should be open. Below shows port of service.
Seahub: 8000, Fileserver: 8082, Ccnet Daemon: 10001, Seafile Daemon: 12001, HTTPS: 443

**2.5.5 Config Files:**
There are three config files:
ccnet/ccnet.conf: contains the network settings
seafile-data/seafile.conf: contains settings for seafile daemon and fileserver.
seahub_settings.py: contains settings for Seahub

**2.5.6 Account Management**
When you setup seahub website, you should have setup an admin account. After you logged in a admin, you may add/delete users and file libraries. Administrator can reset password for a user in "System Admin" page. In a private server, the default settings don't support users to reset their password by email. If you want to enable this, you have first to set up notification email. You may run reset-admin.sh script under seafile-server directory. This script would help you reset the admin account and password.

**2.5.7 Logs**
Log files of seafile server are :
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

**2.5.8 Administrators tools to monitor the system**
access.log: under directory logs/stats-logs
login log: in the web interface
traffic log: in the web interface
public link list: in the web interface

access.log : access.log (under directory logs/stats-logs) records the following information
file download via web
file download via API
The format is:
Date, operation type, user, ip, web agent, library_id, library_name, file path.

**2.5.9 Backup and Recovery**
All your library data is stored under the 'haiwen' directory. Seafile also stores some important metadata data in a few databases. The names and locations of these databases depends on which database software you use. For SQLite, the database files are also under the 'haiwen' directory. The locations are:
ccnet/PeerMgr/usermgr.db: contains user information
ccnet/GroupMgr/groupmgr.db: contains group information
seafile-data/seafile.db: contains library metadata
seahub.db: contains tables used by the web front end (seahub)
For MySQL, the databases are created by the administrator, so the names can be different from one deployment to another. There are 3 databases:
ccnet-db: contains user and group information
seafile-db: contains library metadata
seahub.db: contains tables used by the web front end (seahub)

## CONCLUSION

This research delivered overview of the digital forensics investigation of Seafile as a cloud storage. This research can introduce generic guideline in the area of digital forensics investigation of Seafile. In this paper researcher tried to show server performance of seacloud and also find location of possible artifacts in terms of log files, config file and so on. As acknowledgment, this paper is a survey and researchers summarized overview of digital forensics investigation of Seafile. The most information of this paper is from source of this platform that can be find at http://seafile.com/.

## REFERENCES

[1] (UNODC)., U. N. O. on D. and C. (2012). The use of the Internet for terrorist purposes.
[2] Asou Aminnezhad Mohd Taufik Abdullah, A. D. (2012). A Survey on Privacy Issues in Digital Forensics. International Journal of Cyber-Security and Digital Forensics.
[3] Lei, Z. (2011). Forensic Analysis of Unallocated Space. Faculty of Engineering and Applied Science. University of Ontario Institute of Technology (UOIT), Canada.
[4] Darren Quick, Kim-Kwang Raymond Choo (2013). Google Drive: Forensic analysis of data remnants. Journal of Network and Computer Applications, Elsevier Ltd.
[5] Seafile.com.( 2014) Seacloud , Seafile server manual . Retrieved from https://seacloud.cc/group/3/wiki/
[6] Chung H, Park J, Lee S, Kang C. Digital forensic investigation of cloud storage services. Digital Investigation 2012;9(2):81–95.
[7] Ruan K, Carthy J, Kechadi T, Crosbie M. 'Cloud forensics: an overview. In: 7th IFIP International Conference on Digital Forensics, Orlando, Florida, USA 2011.
[8] Ben Martini, Kim-Kwang Raymond Choo (2013). Cloud storage forensics: ownCloud as a case study. Digital Investigation, Elsevier Ltd.