API-based evidence acquisition in the cloud - a survey [version 1; peer review: awaiting peer review]

Dr. Dirk Pawlaszczyk¹, Martin Bochmann¹, Philipp Engler¹, Coert Klaver², Christian Hummert³

¹Department of Computer Sciences, Hochschule Mittweida - University of Applied Sciences, Mittweida, Saxonia, 09648, Germany
²Netherlands Forensic Institute (NFI), The Hague, 2497 GB, The Netherlands
³Agentur fuer Innovation in der Cybersicherheit, Halle(Saale), 06110, Germany

Abstract
Cloud services and cloud storage solutions are special challenges in digital forensic investigations. Cloud services allow their users, with relatively little technical knowledge, store, manage and share content with others. At the same time investigators are faced with a wide range of technical, legal and organizational issues. Unfortunately, evidence acquisition for such services still follows the traditional way of collecting artefacts on a client device. In this article, first, an overview of the state of research is given. Next, technical and legal challenges related to the forensically sound acquisition of cloud data are presented. Since accessing these data is highly challenging, basic techniques for acquiring data from the cloud are discussed and compared, using the example of 30 cloud storage services. We introduce the concept of an API-based evidence acquisition for cloud services that utilize the officially supported API of the service. We show how well this approach applies to most current cloud drive services in the survey context. We present the first glance of a proof-of-concept acquisition framework called CLOUDxTRACT, which can acquire evidence from selected cloud service providers.

Keywords
Digitals Forensics, Mobile Phones, Data Acquisition, Cloud, Cloud Security

This article is included in the Societal Challenges gateway.
Corresponding author: Dr. Dirk Pawlaszczyk (pawlaszc@hs-mittweida.de)

Author roles: Pawlaszczyk DD: Conceptualization, Formal Analysis, Funding Acquisition, Investigation, Methodology, Project Administration, Resources, Software, Supervision, Validation, Writing – Original Draft Preparation, Writing – Review & Editing; Bochmann M: Conceptualization, Methodology, Resources; Engler P: Conceptualization, Formal Analysis, Methodology, Software; Klaver C: Conceptualization, Formal Analysis, Project Administration, Supervision; Hummert C: Conceptualization, Supervision, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This research was financially supported by the European Union's Horizon 2020 research and innovation programme under the grant agreement No [832800](https://example.com)[From mobile phones to court – A complete FORnsic investigation chain targeting MOBILE devices [FORMOBILE]].

Copyright: © 2022 Pawlaszczyk DD et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Pawlaszczyk DD, Bochmann M, Engler P et al. API-based evidence acquisition in the cloud - a survey [version 1; peer review: awaiting peer review] Open Research Europe 2022, 2:69 https://doi.org/10.12688/openreseurope.14784.1

First published: 31 May 2022, 2:69 https://doi.org/10.12688/openreseurope.14784.1
Introduction

The usage of cloud storage solutions is permanently increasing. Cloud services like Dropbox and Google Drive (Google, RRID: SCR_005886) allow their users, with relatively little technical knowledge, to store, manage, and share content with others.

Cloud forensics today is an integral part of nearly every criminal investigation. 85% of crime investigations include electronic evidence. An increasing number of this data is managed in the cloud. So, examining cloud storage systems takes on particular importance for law enforcement agencies (LEAs).

Cloud forensics describes cloud services' systematic, scientific, and technical investigation. Data acquisition from the cloud is highly challenging since there are so many providers. Many of the technological, organizational, and legal problems related to retrieving this data are still unsolved. Before we describe different acquisition strategies, we must first introduce a definition and terminology for cloud computing. For this purpose, the technical background of the problem will be examined since this is essential for understanding further explanations.

Technical background

Cloud computing is a central concept of this work. From a technical perspective, cloud computing describes the approach of making IT infrastructures available via a computer network without having to install them on the local computer. It is a model that provides ubiquitous and convenient on demand network access to a shared pool of configurable computing resources (such as networks, servers, storage, applications, and services), which can be deployed and shared quickly with minimal administrative overhead. The term resource pooling in the context of cloud computing environments describes a situation where providers serve multiple clients with scalable services and virtualized hardware. This makes it easy to add additional virtual central processing units (CPUs) and memory on the fly. This process is completely transparent for the user.

The idea behind resource pooling is that through modern, scalable systems involved in cloud computing, providers can create a sense of infinite resource pool. More precisely, the cloud provider's customers share the hardware resources provided. From the customer's point of view, the leased system appears to be practically infinitely expandable.

The definition of the National Institute of Standards and Technology (NIST) distinguishes three fundamental service models for cloud computing, namely infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) (see Figure 1). The main difference between the three models is how much influence the cloud user has on the management of the cloud service.

Infrastructure as a service - IaaS

The cloud supplies resources. The cloud itself needs resources to exist at all. The term “cloud infrastructure” describes both aspects. On the IaaS model, an infrastructure of “server, storage, network, archiving, backup, and data centre infrastructure as an abstract, virtualized service” is made available to the cloud user. The cloud user has complete freedom of choice of operating system (e.g. Linux or Windows Server) and all other applications that are used. The cloud user has no access to the underlying layers of cloud computing. They handle the security of the applications and operating system(s) located on the IaaS.

Platform as a service - PaaS

In the PaaS model, the cloud user gains access to the cloud service provider’s (CSP) cloud infrastructure. Furthermore, the

---

**Figure 1. Basic cloud service models.**
CSP must provide the user with access via interfaces since the user should act with the cloud. Therefore, operating systems and development environments are provided by the CSP. The user does not have to take care of the maintenance of the virtual machines in this model. A virtual machine provides the same functionality as physical computers. Just like these, virtual machines run applications and an operating system. However, virtual machines are computer files that run on a physical computer. Microsoft, with its Azure platform (Microsoft) (RRID: SCR_011880), provides an excellent example of this cloud model.

**Software as a service - SaaS**

In the third scenario, a complete package is provided to the cloud user at the SaaS model, including applications. The user manages data via interfaces with browsers and apps in SaaS. In addition to mail and office programs, other applications such as enterprise resource planning (ERP), enterprise content management (ECM), e-commerce, and more can be offered. In this model, users usually share the infrastructure. The infrastructure is separate, ensuring that only privileged users with rights access to this partition. To avoid vulnerabilities, updates to all the cloud system components are required continuously. This update is the responsibility of the CSP. The third model offers the user the least amount of intervention and design options.

While the IaaS model still offers control over a large part of the infrastructure from the cloud customer’s perspective, this control is reduced in the PaaS model and even more so in the SaaS model. At the technical level, there are hardly any interfaces in practice that cloud providers can use to provide access for forensic investigations. In this article we will take a closer look at cloud storage services like Google Drive. Conceptually, these services can be viewed as a sub branch of SaaS paradigm. This means, that we have little influence on the configuration of these services.

**Issues and challenges**

When considering how to acquire data from a cloud service, we first have to talk about the concept and access strategies for cloud data. Forensics, in general, gives answers to questions about what, where, when and how something has happened. In the digital world, this includes uncovering and interpreting electronic evidence. The goal of digital forensics is to preserve any evidence in its most original form while performing a structured investigation by collecting, identifying and validating the digital information to reconstruct past events.

For a very long time, the topic of IT forensics has received little attention in the environment of cloud computing. In recent years, several research projects have started on problems in this area. However, IT forensics in cloud environments still has open questions that have only been partially answered from technical, organizational and legal perspective.

- **Legal issues.** From a legal point of view, there are a large number of open questions which can be answered in individual cases and not in general terms. It is evident that in practice, many problems arise because CSPs act globally and offer their services across country borders. Therefore, the storage of data and the different, country-specific legal conditions are subject.

- **Organizational issues.** Since access to the spatially separated storage media for an IT forensic analysis is often impossible because the IT systems are now located abroad, investigators often have to request access to data from the operator or are dependent on operators. Even if the servers to be examined are situated in a CSP data centre in the investigator’s home country, which is subject to the access of the investigators, access for backup is often ruled out since it would require disproportionate effort and intervention to read the data. If private cloud solutions operated in one’s own company are not considered, the first problem is that almost always, that at least two parties are involved, the cloud customer and the cloud operator.

- **Technical issues.** At the same time, both providers of cloud services and investigative authorities are confronted with increased technical requirements. Especially in public cloud environments, relevant forensic data is usually mixed with data from uninvolved third parties. The distributed storage of data and the use of distributed storage and computing capacity in cloud computing are other problem areas. Due to the increasingly high availability of broadband Internet, development has begun in which IT resources are no longer stored locally but can be accessed via the Internet from large data centres. Today, this includes storage space on the Internet as a matter of course. The trend toward relocating entire IT systems to remote and software-abstract virtual machines seems irreversible. This development means that less and less data is stored locally on the user’s premises and physical data carriers contain fewer data relevant to evidence.

- **Forensic issues.** Established forensic investigation procedures are designed to collect relevant information locally and are unsuitable for collecting data from a remote, often globally operating cloud provider. Even if physical access to a cloud provider’s infrastructure should be possible, an investigation is hardly possible without the active assistance of the cloud provider.

In addition to the organizational challenges in individual service and organization models of cloud computing, there are also numerous technical difficulties. The technical characteristics of cloud environments present new challenges. Cloud infrastructures can consist of tens of thousands of servers. This search and securing of the right physical system is a problem that should not be underestimated. Established forensic investigation procedures are designed to collect information from local devises and are unsuitable for collecting data from remote cloud servers. Even if physical access to a cloud provider’s is feasible, a full investigation is incredibly difficult without active assistance of the cloud provider. A good overview of these technical challenges can be found in 3, 24, 25 and 11. Well-known data extraction and analysis methods are accompanied by further questions related to the localization of data.
The paradigm shift from classic local IT forensics to the cloud is addressed in the work as mentioned above. In particular, the following challenges are mentioned:

- Loss of control over the data.
- No access to the physical infrastructure.
- Legal difficulties due to different responsibilities, manpower and multiple use.
- Lack of suitable tools for the analysis of virtual environments.
- No standardized interfaces.

As pointed out, there are some unique problems in IT forensics that only come to light in this area of a cloud environment. From a legal point of view, the issues arise from the cross-border structure that is naturally associated with cloud services. Furthermore, relevant forensic data are often mixed with data from uninvolved third parties, especially in public cloud environments. The organizational challenge lies in having two parties, the cloud operator and the cloud user, involved. The current procedural models in IT forensics are not designed for and consequently are unsuitable for collecting data from a global player. It is practically impossible to examine the data without the help of a cloud provider. Even physical access to the cloud provider’s infrastructure is not enough. The main criticism here is the lack of access to individual parts of the system because there are hardly any interfaces at the technical level with which cloud providers can enable access for forensic examinations.

In a cloud scenario, we have four ways to acquire data (Figure 2). Typically, an investigation starts on the client’s device. In addition to a copy of the data from the cloud drive, you can often find the access keys or credentials for the cloud account. However, the computer or smartphone analysis does not have to be complete, since this approach is based on the assumption that all files in the cloud are also stored 1:1 on the device and are fully replicated. In practice, the data is rarely 100 per cent synchronous. Many traces can only be found in the cloud and not local. Thus, we can often even restore old file revisions via the cloud provider’s recovery function. Quite often, these can be retrieved via the trash function. A critical forensic question is also to find out which devices files were shared to and from.

We can use the information on the client to show which data was loaded from it into the cloud. Whether others have access to the cloud drive and which devices have retrieved the data remains hidden. A second starting point would be for the cloud service provider to access the cloud data. Unfortunately, this is where the problems already discussed are found. We are dependent on cooperation with the CSP. In addition, we have to expect considerable waiting times in this case, which means that evidence may already be lost. Virtually all cloud providers also allow access to the content stored in the cloud via a web interface. In addition to a classic website, we almost always find a programming interface. Thus, we can access the data stored in the cloud without delay when we have the correct user credentials. However, accessing and acquiring data via the website can sometimes prove cumbersome. From an investigator’s perspective, automatic preservation of evidence is
naturally preferable. Therefore, we will present the idea of an API-based acquisition in more detail in the next section.

**API-based data acquisition**

There are a large number of publications that discuss the examination of evidence and metadata stored on a local device. The concept of API-based acquisition is a relatively new approach. It was first discussed by Roussev et al. in 2017. We access the data stored in the cloud drive through an API usually provided by the CSP. Sometimes interfaces to special cloud services are also offered by third parties. From a technical perspective, this is almost always a representational state transfer (REST) interface. REST is a paradigm for software architecture of distributed systems, especially for web services. Resources are a core of REST. The resources are uniquely identified by a uniform resource identifier (URI). RESTful APIs provide resources using URIs. A typical URI normally looks like: endpoint-uri/deviceId/disk/id. At this point, there are parallels to a classic hierarchically organized file system. In both cases, a path specification is required to gain access to the data. The call is, therefore, very intuitive. A URI contains the location and name of the resource but not the functionality. Thus, this concept is particularly suitable for making files available via the Internet. Clients obtain storage device resources and storage services using the URIs accordingly (see Figure 3).

Any web provider that only offers entire page content according to the Internet standard HTTP is already REST-compliant. Such pages are requested according to the REST paradigm. REST normally employs the following standard operations to access resources:

- **POST**: creates resources.
- **GET**: queries resources.
- **PUT**: modify resources.
- **DELETE**: deletes resources.

A RESTful API is defined as an application programming interface (API) that uses HTTP requests to access data via GET, PUT, POST, and DELETE. This type of access is often referred to as the CRUD approach. CRUD is an abbreviation for these operations to create, retrieve, update and delete data in a software application or service (Table 1).

To query the data stored with the cloud service provider, we use the official web APIs. In simple terms, a web API describes a data access point that has the following properties:

- Exposed over http(s),
- available over the Internet,
- uses the HTTP response and request schema,
- based on human readable data format.

The respective services essentially differ only in the specification of the endpoint address, which naturally varies from service to service. We can access and store artefacts within the cloud in a controlled manner via corresponding client software. Most access to data is given via a well-defined interface, which makes it much easier to acquire the data in a forensically sound manner and ensures a high degree of automation.

The following example illustrates how easy it is to call a resource provided via REST. The source code shows the login and the query of the directory content for a pCloud account using Python (RRID:SCR_008394):

```python
from pcloud import PyCloud
pc = PyCloud(‘email@example.com’, ‘SecretPassword’, endpoint=”eapi”) pc.listfolder(folderid=0)
```

Data acquisition is straightforward if the investigator has access to the credentials. What’s more, the process can be fully...
automated. In addition to being the only means for guaranteeing a forensically-complete copy of the target data, the API approach supports the reproducibility of results, rigorous tool testing (based on well-defined API semantics) and triage of the data (via hashes or search APIs). The development effort is significantly lower because client-centric approaches eliminate the entire black box reverse engineering component.

**Methodology**

In this study we aim to determine whether this approach is practicable based on the preliminary considerations. For this purpose, we examined 30 cloud providers in a case study. The services were selected primarily on the basis of user numbers. Another criterion for selection was forensic relevance. A cloud service like MEGA (MEGA Ltd, New Zealand), for example, is increasingly the subject of police investigations. Of course, the selection is not exhaustive. The focus of the study was primarily on cloud storage providers and file-sharing vendors. An access account was created for each of the cloud services studied.

The aim was to decide whether or not we could manage a forensic sound data acquisition with our API-based approach. It was first necessary to clarify if a corresponding REST interface is available for each cloud service. For this purpose, the interface documentation provided by each cloud provider and source code repositories made available were analyzed. For better traceability, the names of the respective API have been listed again in Table Table 2. With the help of the API name, all information used for the study can be easily found and verified. All of the information listed below is taken from the descriptions found there. In addition, for 8 of the total of 30 services examined, a sample implementation for access was taken.

In the next step, we determined the technical requirements for each CSP. For this purpose, the API descriptions for each cloud service were analyzed. This involves the architectural style and the parameters that had to be transferred. Beyond this, we had to clarify which information could be queried via the APIs. Of course, the response format also had to be checked. From an investigator’s point of view, it is essential to know whether we can recover deleted data. The interface documentation was reviewed for appropriate access methods to recover deleted data. The user accounts created for testing purposes were also checked for recovery options. Another critical feature of a forensic investigation is retrieving a file’s version history. Adding meta-information about a file, such as the creation and modification dates, should be accessible. Another point we have checked was the possibility of retrieving events logs. All of the cloud services examined in the study can be seen in Table 2. The main focus was on storage cloud providers that take a SaaS approach. However, some messaging and chat services such as Steam Gaming Engine (Valve, 2003) or Telegram (Telegram) are also included, not least because they are increasingly the subject of police investigations. For the type column, the provider notes the primary purpose of the service. The architectural style is indicated in a separate column.

**Results**

From Table 2 we can see that there is a corresponding REST interface in place for any cloud service. Beyond this, some vendors offer alternative RPC-based access options or allow retrieval via WebDAV (web-based distributed authoring and versioning). However, the last alternative is limited to only a few CSP. API-based access is supported by virtually every vendor in our test, a very encouraging result. If we are looking for an access method that works for nearly all cloud storage solutions, then the API approach via REST seems to be a good choice. Except for the ICloud (Apple, 2011) service, all vendors offer official APIs to access the data stored in the cloud via REST. For the ICloud, there is again a plethora of third-party providers providing an interface. For our study, we examined the pyicloud project closely. As pyicloud is an open-source and freely available interface able to provide all necessary functionalities to perform a forensic copy.

As part of the survey, we also wanted to find out which authentication options the respective providers support (see Table 3). This information is of particular importance. A large number of the cloud services examined support login with password and user name. Other providers only support open authorization (OAuth) authentication. In this case, access to the cloud data must always be authorized via a browser session or an app. For a forensic investigation, we need an unlocked phone before we can access the data via the API. After approval via OAuth, the app receives an access key. This time-limited token can then be used to access the data. This way, a password never has to be used directly to access the data. Among the cloud services examined, there was only one service - Telegram - that forces a two-factor authentication (2FA). For others, this login method is optional.

Once we have successfully overcome the authorization hurdle, we can begin to acquire data. Table 4 shows further technical details for accessing the respective cloud services. In addition to the respective API endpoint, the supported request protocol and the supported response formats are displayed. Without exception, all providers follow the CRUD paradigm. Standard HTTP methods such as GET, PUT and DELETE are thus used to access the stored resources. When it comes to the supported response
Table 2. Web application programming interfaces (APIs) for popular cloud storage providers (CSP).

| CSP              | Type             | API name                  | Architectural Style | Official |
|------------------|------------------|---------------------------|---------------------|----------|
| Accellion        | file sharing     | Accellion’s API Guide     | RESTful             | Yes      |
| Amazon Cloud Drive| file sharing     | RESTful Amazon Drive API  | RESTful             | Yes      |
| Box              | file sharing     | Box API                   | RESTful             | Yes      |
| Datto            | backup           | Datto RMM-API             | RESTful             | Yes      |
| Degoo            | file sharing     | Degoo GraphQL API         | RESTful             | No       |
| DropBox          | file sharing     | Dropbox API V2            | RESTful             | Yes      |
| Google Drive     | file sharing     | Drive API/ Drive Activity | RESTful             | Yes      |
| HiDrive          | file sharing     | HiDrive API               | RESTful             | Yes      |
| HIGHTAIL         | file sharing     | Hightail (YouSendIt) API V 1.0 | RESTful | Yes |
| hubiC            | file sharing     | hubiC API (OpenStack API) | RESTful             | Yes      |
| Huddle           | file sharing     | Hudlle API                | RESTful             | Yes      |
| ICloud (pyicloud)| storage, sync    | pyicloud                  | RESTful             | No       |
| IDrive           | backup/file sharing | EVS iDriveSync RESE API  | RESTful             | Yes      |
| MediaFire        | storage, file sharing | mediafire.com/api        | RESTful             | Yes      |
| MEGA             | storage, file sharing | MEGA API                 | RESTful             | Yes      |
| NextCloud        | storage, file sharing | NextCloud OCS API        | RESTful             | Yes      |
| OneDrive         | storage, file sharing | Graph API / OneDrive-API | RESTful             | Yes      |
| Oodrive          | file sharing     | REST API @                | RESTful             | Yes      |
| Orange Cloud     | file sharing     | api.orange.com            | RESTful             | Yes      |
| Open Telekom Cloud| IaaS, Virtualization | Telekom Cloud API    | RESTful             | Yes      |
| ownCloud         | file sharing     | OCS Rest API              | RESTful             | Yes      |
| pCloud           | storage, file sharing | pCloud REST API         | RESTful             | Yes      |
| PowerFolder      | storage, file sharing | Power Folder REST API    | RESTful             | Yes      |
| SendSpace        | file             | SendSpace API 1.2         | RESTful             | Yes      |
| ShareFile        | file sharing     | ShareFile API             | RESTful             | Yes      |
| SpiderOak        | storage, backup  | Accounts API 1.0.0        | RESTful             | Yes      |
| SteamChat        | gaming, chat, share content | SteamWorks Web API | RESTful             | Yes      |
| SugarSync        | storage, backup  | SugarSync API             | RESTful             | Yes      |
| Syncplicity      | content collaboration | Syncplicity-Rest-APIs | RESTful             | Yes      |
| Telegram         | messaging, file sharing | Telegram REST API     | RESTful             | Yes      |
| TeamDrive        | file sharing     | TeamDrive Http Api       | RESTful             | Yes      |

1 REST - Representational state transfer

formats, the picture is twofold. In the vast majority of cases, it is a regular JavaScript Object Notation (JSON) string. Some providers rely on XML. Some CSPs additionally support CSV as a return format. However, parsers are usually available for all offered forms. Decoding the message content should therefore not be a significant problem.

All providers tested allowed the download of individual files as well as cloud directories. The resources can usually be accessed directly as a binary stream. In some cases, the binary information is also embedded directly in the JSON messages. In this case, the providers resort to BASE64 encoding. Of course, the resources stored in the cloud are of the highest interest in a
| Cloud service provider | Authentication/authorization method |
|------------------------|-------------------------------------|
| Accellion              | OAuth 2.0 with access token          |
| Amazon Cloud Drive     | Access token, Authorization Code Grant, Login to amazon |
| Box                    | Access token, OAuth 2.0, JSON Web Tokens (JWT), client credentials |
| Datto                  | OAuth 2.0, API Key and API Secret Key (for account activation) |
| Degoo                  | Credentials + invariant API-key      |
| Dropbox                | OAuth 2.0 with offline access token  |
| Google Drive           | OAuth 2.0 with application access token |
| HiDrive                | OAuth 2.0 with application access token |
| HIGHTAIL               | OpenText Hightail                   |
| hubicC                 | OAuth 2.0 with access token          |
| Huddle                 | OAuth 2.0 with access token          |
| ICloud                 | Credentials (username + password), support for 2SA / 2FA |
| IDrive                 | OAuth or credentials (username + password) |
| MediaFire              | Credentials, OAuth login style       |
| MEGA                   | Credentials (email, password)        |
| NextCloud              | Password based HTTP Basic Auth or valid session cookie |
| OneDrive               | OAuth 2.0 with application access token |
| Orange Cloud           | OAuth 2.0 with access token, Others  |
| Open Telekom Cloud     | Credentials (username + password)    |
| ownCloud               | Credentials (username + password)    |
| pCloud                 | OAuth 2.0 with access token, credentials |
| PowerFolder            | Username + password                  |
| SendSpace              | Username + password                  |
| ShareFile              | OAuth 2.0 with access token, SAML    |
| SpiderOak              | Password based HTTP Basic Auth       |
| SteamChat              | Steam Account info + Web-API key, OAuth |
| SugarSync              | Credentials (username + password)    |
| Syncplicity            | Credentials (username + password)    |
| Telegram               | Credentials + 2FA                    |
| TeamDrive              | Username + password, HTTP Basic Auth |

1 OAuth - Open authorization protocol 2 2SA - Two step authentication 3 2FA - Two factor authentification

forensic investigation. In addition, however, there are other questions to which the investigator may seek answers. So, for the most part, it is just as important to find out when a particular file was uploaded to the cloud. If several versions of the file exist, then it would be desirable to be able to query this information. When and which users have accessed the files is another crucial question in many investigations. Finally, the recovery of previously deleted files from the recycle bin should also be given if possible. In addition, many providers offer the query of user information. In the case of the Box cloud service (Box inc.), for example, this includes the user’s name and user ID, their email address, the time the account was created,
Table 4. Application programming interface (API) - Technical details.

| CSP            | API Endpoint                        | Request Protocol         | Response Format |
|----------------|-------------------------------------|--------------------------|-----------------|
| Accellion      | https://<kiteworks&#95;server>/oauth/ | URI Query String/CRUD    | JSON\(^2\)     |
| Amazon Cloud Drive | https://drive.amazonaws.com        | URI Query String/CRUD    | JSON            |
| Box            | https://app.box.com/2.0/            | URI Query String/CRUD    | JSON            |
| Datto          | https://pinotage-api.centrastage.net/api/ | URI Query String/CRUD    | JSON            |
| Degoo          | https://rest-api.degoo.com/         | URI Query String/CRUD    | JSON            |
| DropBox        | https://content.dropboxapi.com      | URI Query String/CRUD    | JSON            |
| Google Drive   | https://www.googleapis.com/         | URI Query String/CRUD    | JSON            |
| HiDrive        | https://api.hidrive.strato.com/2.1/ | URI Query String/CRUD    | JSON            |
| HIGHTAIL       | https://api.stage.htspaces.com/devapi/v1 | URI Query String/CRUD    | JSON            |
| hubiC          | https://api.hubic.com/oauth/        | URI Query String/CRUD    | JSON            |
| Huddle         | https://api.huddle.net/entry        | URI Query String/CRUD/JSON/XML | JSON/XML       |
| ICloud         | https://ws/com.apple.CloudDocs/     | URI Query String/CRUD    | JSON            |
| IDrive         | https://ews.idrivesync.com/evs/      | URI Query String/CRUD    | XML\(^3\)      |
| MediaFire      | https://www.mediafire.com/api/      | URI Query String/CRUD    | JSON, XML       |
| MEGA           | https://g.api.mega.co.nz/           | URI Query String/CRUD    | JSON            |
| NextCloud      | https://cloud.example.com/ocs/      | URI Query String/CRUD    | XML             |
| OneDrive       | https://graph.microsoft.com/v1.0    | URI Query String/CRUD    | JSON            |
| Orange Cloud   | https://api.orange.com/cloud/v1/    | URI Query String/CRUD    | CSV, JSON, XML  |
| Open Telekom Cloud | https://ecs.eu-de.otc.t-systems.com | URI Query String/CRUD    | (Geo)JSON, XML  |
| ownCloud       | http://api.owncloud.org/            | URI Query String/CRUD    | JSON, XML       |
| pCloud         | https://www.pcloud.com              | URI Query String/CRUD    | JSON            |
| PowerFolder    | https://access.powerfolder.com/api/ | URI Query String/CRUD    | (Geo)JSON, XML  |
| SendSpace      | http://api.sendspace.com/rest/      | URI Query String/CRUD    | XML             |
| ShareFile      | https://subdomain.sharefile.com/rest | URI Query String/CRUD    | JSON(P), Text, XML |
| SpiderOak      | https://alice.diy.spideroak.com/data/ | URI Query String/CRUD    | CSV, (Geo)JSON, XML |
| SteamChat      | https://api.steampowered.com/v/     | URI Query String/CRUD    | JSON            |
| SugarSync      | https://api.sugarsync.com/          | URI Query String/CRUD    | XML             |
| Syncplicity    | https://api.syncplicity.com/        | URI Query String/CRUD    | JSON            |
| Telegram       | https://api.telegram.org/api/       | URI Query String/CRUD    | JSON            |
| TeamDrive      | https://<teamdrive-server>/api      | URI Query String/CRUD    | JSON            |

\(^1\) CRUD - Create Read Update Delete, \(^2\) JSON - JavaScript Object Notation \(^3\) XML - Extensible Markup Language

the time zone setting, the user’s address and the phone number stored. In Table 5 it is shown to what extent these, forensically, interesting questions are supported by the provided interfaces. For categories that are supported by the API we use a ✓ symbol. Properties that could not be found in the API, a minus (-) is used. Of course, the observations represent only a snapshot, as the APIs are constantly evolving.

First of all, it can be stated that practically all providers offer additional information about their resources. Mostly this is
information about the account as well as timestamps. Some providers like Google\textsuperscript{30} or Microsoft\textsuperscript{17} also support the retrieval of user information. For others, the provision of meta-information is limited to resources like files and directories only. There are also significant differences regarding the possibility of accessing deleted data from the recycle bin. Not every CSP provides access here. Services such as SendSpace or Huddle are examples of this. However, some providers do not support a trash function at all - neither via the API nor other methods. During our survey, we could identify artefacts of forensic interest, such as information generated during login, uploading, downloading, deletion, and sharing files. Here again, the providers diverge significantly in terms of the information provided.

Differences also occurred in the retrieval of timestamps. In some cases, these are adjusted to the user’s time zone. In the case of

| Cloud service provider      | Trash Recovery | Meta Data | User Activities | Timestamps |
|-----------------------------|----------------|-----------|-----------------|------------|
| Accellion                   | ✓              | ✓         | ✓               | ✓          |
| Amazon Cloud Drive          | ✓              | ✓         | ✓               | ✓          |
| Box                         | ✓              | ✓         | ✓               | ✓          |
| Datto                       | -              | ✓         | ✓               | ✓          |
| Degoo                       | ✓              | ✓         | -               | ✓          |
| DropBox                     | ✓              | ✓         | ✓               | ✓          |
| Google Drive                | ✓              | ✓         | ✓               | ✓          |
| HiDrive                     | -              | ✓         | ✓               | ✓          |
| HIGHTAIL                    |                |           |                 |            |
| hubicC                      | ✓              | ✓         | ✓               | ✓          |
| Huddle                      | -              | ✓         | ✓               | ✓          |
| iCloud                      | ✓              | ✓         | ✓               | ✓          |
| IDrive                      | ✓              | ✓         | ✓               | ✓          |
| MediaFire                   | ✓              | ✓         | ✓               | ✓          |
| MEGA                        | ✓              | ✓         | ✓               | ✓          |
| NextCloud                   | ✓              | ✓         | ✓               | ✓          |
| OneDrive                    | ✓              | ✓         | ✓               | ✓          |
| Orange Cloud                | ✓              | ✓         | ✓               | ✓          |
| Open Telekom Cloud          | ✓              | ✓         | ✓               | ✓          |
| ownCloud                    | ✓              | ✓         | ✓               | ✓          |
| pCloud                      | ✓              | ✓         | ✓               | ✓          |
| PowerFolder                 | ✓              | ✓         | ✓               | ✓          |
| SendSpace                   | -              | ✓         | -               | ✓          |
| ShareFile                   | ✓              | ✓         | ✓               | ✓          |
| SpiderOak                   | ✓              | ✓         | ✓               | ✓          |
| SugarSync                   | ✓              | ✓         | ✓               | ✓          |
| Syncplicity                 | ✓              | ✓         | ✓               | ✓          |
| Telegram                    |               | -         | -               | ✓          |
| TeamDrive                   | ✓              | ✓         | ✓               | ✓          |
the Box cloud service, for example, the timestamps are stored in Request for Comment (RFC) 3339 format and are always given in Coordinated Universal Time (UTC) - 7 zone time, the timestamp of the folder created in the RFC 3339 format was given in the zone time UTC ± 0 and not in the user’s zone time. In other cases, the UTC timestamps are not returned as strings but as UNIX timestamps in milliseconds or seconds. For example, you can say that on 2021-12-26 05:32 UTC, 1640496741000 milliseconds (or 1640496741 seconds Epoch timestamp) have passed since 1970-01-01 00:00:00 UTC.

However, forensic analyses must always consider that the output time must first be converted to the zone time in which the user carried out the activities to make a correct statement. It is crucial to ensure the integrity and completeness of the data. Therefore, as part of the meta-information, hash values are also offered via the interface in many cases. Mostly, these are classic MD5 checksums. In some cases, SHA-1 is supported as a checksum. Here, again, there is no uniform picture. Regarding the completeness of the forensic copy, it can be stated that in our test, a few pieces of information could not be reconstructed. For example, there were sometimes no entries for login activities, changes to settings, or folder accesses in the user events. The cloud service Box is an example of this.

As our study makes clear, virtually all CSPs offer a REST interface. Thus, data access can be easily automated. However, there are some significant differences in terms of the data that can be retrieved and their structure. An additional access layer is needed to ensure easy and sound data acquisition. Access to the data can be significantly simplified with an appropriate framework in place. All steps - even for different vendors - can be unified. Data preparation issues could thus also be quickly resolved. As a proof-of-concept, based on the results of our study, we developed a framework that supports automated acquisition. At the same time, we can use it to demonstrate practical feasibility of the discussed API-based acquisition approach. The framework is briefly introduced in the next section.

**Proof of concept - The CLOUDxTRACT framework**

Based on the results of our study, we developed an access service that provides prototypical access to numerous cloud services. The novel software solution presented here has the name CLOUDxTRACT. It represents an attempt to provide a simple, automated and forensically sound seizure of cloud resources. Currently, access to eight different storage services is offered: MediaFire, PCloud, Sugarsync, Strato HiDrive, Nextcloud, Skype, Steam Chat and Degoo.

CLOUDxTRACT is designed as a command-line tool without a graphical user interface. This solution offers the highest performance and can be easily integrated in a batch job. In the following, an overview on the principle system design and the interaction of objects is given. Figure 4 illustrates how the CLOUDxTRACT is constructed. The framework consists of multiple classes which are organized into tiers. We start with the Extractor-Class. In the run()-method of the Extractor-Class the workflow is started. It implements the main control logic. The run()-method again calls the concrete procedures for data acquisition from a specialized adapter class. For each supported cloud service provider, a particular adapter class exists which taps the API from the cloud service provider. Specialized adapter classes like MediaFireAdapter use a xxCient classes to handle the communication to the APIs from the CSP. In this way, we obtain a modular framework that can be easily extended with additional cloud services. At the same time, it ensures that all data can be stored in a forensically clean and repeatable manner with timestamps and hash values.

**System requirements and installation**

The installation process is very simple. For the execution of the scripts a Python runtime environment of version 3 or higher is required (more information here). A special operating system is not required. A minimum of 4 GB RAM is recommended. There are no other special hardware requirements.

The installation consists of 4 steps in total:

1. Download the source code of CLOUDxTRACT from the official project page
   
   https://github.com/FORMOBILE/CLOUDxTRACT

2. Copy the library files into a folder of your choice.

3. Open a command line terminal and navigate to the folder where the file setup.py resides.

4. Use pip to register the library in the python environment with the command pip install -e.

This completes the installation process. The software can now be used immediately to acquire data from the cloud.

**MediaFire use case**

Using the tool is very simple. We will show this in the following using the MediaFire cloud service as an example. MediaFire is a file hosting, file synchronization, and cloud storage service based in Shenandoah, Texas, United States. MediaFire has 43 million registered users and attracted 1.3 billion unique visitors to its domains in 2012 (MediaFire about). The input parameter and output are nearly the same for all supported CSPs. For demonstration purposes a test account was filled with simple test files shown in Figure 5. The CLOUDxTRACT can be executed by command line and has a build in help functionality for each service.

To view the help, the optional parameter -h or --help must be provided. To acquire the test account, the following command is used:

```
python -m extractor {service} --logfile file
log -u "username" -p "password" destination
```
This command triggers the extraction process. The name of the respective cloud service must be entered in place of the `service` parameter. In addition to the user name and password, the name of the log file and the destination directory must also be entered.

In Figure 6 the messages of the log file are shown as an example. As you can see in the screenshot, important meta information and timestamps are retrieved in addition to the actual data content. For each file, an additional MD5 hash is automatically calculated during the download. Later modifications to the acquired files can thus be easily detected by comparing the hash values from the download with the most recent once. The source code of this framework is made available as an open-source repository via the FORMOBILE Project Github-Account. In this way, practitioners can directly participate from the results of this project.

**Conclusions and future work**

This paper examined the acquisition of data from cloud drives in more detail. The cloud services of 30 providers analyzed in this context show similarities to a large extent. The
The heterogeneity of cloud services repeatedly emphasized in the literature appears very uniform on closer inspection. Our findings contribute to an up-to-date understanding of cloud storage forensics. As it turns out, all of the cloud services explored can be queried in the same way. All of the interfaces examined are RESTful services. Significant differences arise, in particular, in the meta-information provided by the respective API and the format and structure of the returned data.
The presented software solution CLOUDxTRACT\textsuperscript{35} proves that an automated seizure of data via an API-based approach is possible. A total of eight functional interfaces are implemented at the time of publication. Since the source code is published as open source, it can be easily reviewed and proved by other scientists\textsuperscript{36}.

Hence, many processing steps can be handled by the framework. The straightforward design of the framework enables users to adopt existing and integrate new cloud service providers easily. The number of services provided via CLOUDx-TRACT is to be expanded in the future. Concrete plans for further cloud services to be integrated are already in the pipeline.

**Data availability**

Underlying data

Zenodo: FORMOBILE/CLOUDSURVEY: Survey Datasheet. https://doi.org/10.5281/zenodo.6556669\textsuperscript{36}

This project contains the following underlying data:

- Survey\_DataSheet.xlsx (Complete datasheet for all 30 cloud services surveyed within this article)

Data are available under the terms of the the Creative Commons Zero “No rights reserved” data waiver (CC0 1.0 Public domain dedication).

**Software availability**

Source code of CLOUDxTRACT is available here: https://github.com/FORMOBILE/CLOUDxTRACT

Archived source code at the time of publication: https://doi.org/10.5281/zenodo.6555508\textsuperscript{36}

Licence: GPLv3

**Ethics and consent statement**

Ethical approval and consent were not required.

References

1. Eurostat Press Office: Almost 8 out of 10 internet users in the eu surfed via a mobile or smart phone in 2016. online newsrelease, 12 2017; 260/2016.
2. Pawliszczyz D: Mobile forensics – the end of a golden age? J Forensic Sci & Criminal Inves. 2022; 15: 1–4.
3. Alqahtany S, Clarke N, Furnell S, et al.: Cloud forensics: A review of challenges, solutions and open problems. In 2015 International Conference on Cloud Computing (ICCC). 2015; 1–9.
4. Herman M, Iorga M, Salim AM, et al.: Nist cloud computing forensic science challenges. 2020.
5. Mell P, Grance T: The nist definition of cloud computing. Technical Report 800-145, National Institute of Standards and Technology (NIST), Gaithersburg, MD, 2011.
6. Velte T, Velte A, Elsenpeter R: Cloud Computing, A Practical Approach. McGraw-Hill, Inc., USA, 1 edition. 2009.
7. Kavis M: Cloud Service Models. chapter 2. John Wiley and Sons, Ltd, 2014; 13–22.
8. Copeland M, Soh J, Puca A, et al.: Microsoft Azure: Planning, Deploying, and Managing Your Data Center in the Cloud. Apress, USA, 1st edition. 2015.
9. Pawliszczyz D: Digitalter Tatort, Sicherung und Verfolgung digitaler Spuren. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017; 113-166.
10. Gehhardt T, Reiser HP: Network forensics for cloud computing. In jim Dowling and Francois Taiani, editors, Distributed Applications and Interoperable Systems, Berlin, Heidelberg, Springer Berlin Heidelberg, 2013; 29-42.
11. Pichan A, Lazarescu M, Soh ST: Cloud forensics: Technical challenges, solutions and comparative analysis. Digit Investig. 2015; 13: 38-57.
12. Atefaz A, Zulkipi NH, Atlam HF, et al.: The impact of cloud forensic readiness on security. In Proceedings of the 7th International Conference on Cloud Computing and Services Science - CLOSER. INSTICC, SciTePress, 2017; 539-545.
13. Ab Rahman NH, Choo KKR: A survey of information security incident handling in the cloud. Comput Secur. 2015; 49: 45-69.
14. Zach J, Reiser HP: Livecloudinspector: Towards integrated iaas forensics in the cloud. In Alysson Bessani and Sara Bouchenah, editors, Distributed Applications and Interoperable Systems - 15th IFIP WG 6.1 International Conference, DAS 2015, Held as Part of the 10th International Federated Conference on Distributed Computing Techniques, DisCoTec 2015, Grenoble, France, June 2-4, 2015, Proceedings, of Lecture Notes in Computer Science. 2015; 9038: 207–220.
15. Simou S, Kalloniatis C, Mouaratidis H, et al.: Towards the development of a cloud forensics methodology: A conceptual model. In Anne Persson and Janis Stiirna, editors, Advanced Information Systems Engineering Workshops, Cham, Springer International Publishing. 2015; 470-481.
16. Bansal P, Patidar M: Cloud forensics: An overall research perspective. International Journal of Scientific Research in Network Security and Communication. 2018; 6(2): 5–10.
17. Rousset V, Barreto A, Ahmed T: Api-based forensic acquisition of cloud drives. In Gilbert Peterson and Sujet Shenoi, editors, Advances in Digital Forensics XII, Cham, Springer International Publishing. 2016; 213–235.
18. Ruan K, Carthy J, Kechadi MT, et al.: Cloud forensics definitions and critical criteria for cloud forensic capability: An overview of survey results. Digit Investig. 2013; 10(1): 34-43.
19. Shumba R: Client forensics: An assessment of existing research and future directions. International Journal of Computer Science and Security (ICCSS). 2018; 12(3): 45-54.
20. Simou S, Kalloniatis C, Kavaki E, et al.: Cloud forensics: Identifying the major issues and challenges. In Matthias Jarke, John Mylopoulos, Christoph Quix, Colette Rolland, Yannis Manolopoulos, Haralambs Mouratidis, and Jennifer Hor Kok, editors, Advanced Information Systems Engineering. Cham, Springer International Publishing, 2014; 8484: 271–284.
21. Khan S, Gani A, Wahab AWA, et al.: Cloud log forensics: Foundations, state of the art, and future directions. ACM Comput Surv. 2016; 49(1): 1–42.
22. Almulla S, Ibrahim Y, Jones A: A state-of-the-art review of cloud forensics. Journal of Digital Forensics, Security and Law. 2014; 9: 7–28.
23. Samy GN, Shambagum B, Maanop N, et al.: Digital forensic challenges in the cloud computing environment. In Faisal Saeed, Nadhmi Gazem, Srikanta Patnaik, Ab Saleh Saed Balaid, and Fathey Mohammmd, editors, Recent Trends in Information and Communication Technology. Cham, Springer International Publishing, 2018; 5: 669–676.
24. Sree TR, Bhanu SMS: Data collection techniques for forensic investigation in cloud. In B Suresh Kumar Shetty and Pavanchand Shetty H, editors, Digital
25. Ali SA, Memon S, Sahito F: Challenges and solutions in cloud forensics. In Proceedings of the 2018 2nd International Conference on Cloud and Big Data Computing. ICCBDC’18, New York, NY, USA, Association for Computing Machinery. 2018; 6–10. Publisher Full Text

26. Grispos G, Storer T, Gilsson WB: Calm before the storm: The challenges of cloud computing in digital forensics. CoRR. abs/1410.2123, 2014. Reference Source

27. Daryabar F, Dehghantanha A, Eterovic-Soric B, et al.: Forensic investigation of onedrive, box, googledrive and dropbox applications on android and ios devices. Aust J Forensic Sci. 2016; 48(6): 615–642. Publisher Full Text

28. Daryabar F, Dehghantanha A, Choo KKR: Cloud storage forensics: Mega as a case study. Aust J Forensic Sci. 2017; 49(3): 344–357. Publisher Full Text

29. Pawluczysz D, Friese J, Hummert C: “alexa, tell me . . . ” - a forensic examination of the amazon echo dot 3 rd generation. Int J Comput Sci Eng. 2019; 7(11): 20–29. Publisher Full Text

30. Quick D, Choo KKR: Google drive: Forensic analysis of data remnants. J Netw Comput Appl. 2014; 40: 179–193. Publisher Full Text

31. Fielding RT: REST: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine, 2000. Reference Source

32. Ahmad NH, Hamid ASSA, Shahidan NSS, et al.: Cloud forensic analysis on pcloud: From volatile memory perspectives. In Mahdi H. Miraz, Peter S. Excell, Andrew Ware, Safeeullah Soomro, and Maaruf Ali, editors, Emerging Technologies in Computing. Cham, Springer International Publishing, 2020; 332: 3–15. Publisher Full Text

33. Tabuyo-Benito R, Bahsi H, Peris-Lopez P: Forensics Analysis of an On-line Game over Steam Platform. 10th International EAI Conference, ICDF2C, 2018, New Orleans, LA, USA, September 10-12, 2018, Proceedings, 2019; 259: 106–127. Publisher Full Text

34. Satrya GB, Daely PT, Nugroho MA: Digital forensic analysis of telegram messenger on android devices. 2016 International Conference on Information & Communication Technology and Systems (ICTS). 2016: 1–7. Publisher Full Text

35. FORMOBILE/CLOUDXTRACT. Formobile.eu: cloudxtract 0.1. (published). zenodo. [software] 2022; Accessed: 2022-05-18. http://www.doi.org/10.5281/zenodo.6555508

36. FORMOBILE/CLOUDSURVEY: Survey datasheet (published). zenodo. [dataset]. 2022; Accessed: 2022-05-18. http://www.doi.org/10.5281/zenodo.6556669