Final deployment, Cloud&Heat SafeCloud-based cloud storage platform

Deliverable 5.5

Project reference no. 653884

August 2018

Horizon 2020
European Union funding
for Research & Innovation

Document information

Scheduled delivery	31.08.2018
Actual delivery	31.08.2018
Version	1.0
Responsible partner	Cloud&Heat

Dissemination level

Public

Revision history

Date	Editor	Status	Version	Changes
27.08.2018	S. Schmerler	Draft	0.1	Initial version
28.08.2018	S. Schmerler	Draft	0.2	Add Legal section
28.08.2018	M. Correia	Draft	0.2	IN-ID review
28.08.2018	S. Totakura	Draft	0.2	TUM review
30.08.2018	S. Schmerler	Draft	0.3	Address reviews
31.08.2018	S. Schmerler	Final	1.0	Final Version

Contributors

S. Schmerler (C&H)
G. Miegel (C&H)
D. Matos (IN-ID)

Internal reviewers

S. Totakura (TUM)
M. Correia (IN-ID)

Acknowledgments

This project is partially funded by the European Commission Horizon 2020 work programme under grant agreement no. 653884

More information

Additional information and public deliverables of SafeCloud can be found at:
http://www.safecloud-project.eu/

Glossary of acronyms

Acronym	Definition
API	Application Programming Interface
AWS	Amazon Web Services
DC	Datacenter
RBD	RADOS Block Device
S3	Amazon Simple Storage Service
VM	Virtual Machine
WAN	Wide Area Network

Contents

Executive Summary

The pilots for the SafeCloudBox and CloudBlockStorage solutions based on SafeCloud technology and developed by C&H are described in detail. Both pilots are storage solutions backed by the Ceph distributed storage software. In this deliverable, we describe working setups and show that SafeCloud technology can be fully integrated into C&H products. Specifically, the SafeCloudBox product is an entirely new development, which relies on the SafeCloud File System (SafeCloudFS, SS3) to provide customers with a secure and fault-tolerant data storage solution. The CloudBlockStorage product extends C&H's block storage cloud offer with inter-datacenter security by using the SafeCloud Private Communication Middleware (SC2).

1 Introduction

This deliverable describes the CloudBlockStorage and SafeCloudBox pilots developed by C&H. Both are storage solutions based on Ceph[1], a distributed storage software which was described in detail in D5.3.

For the full deployment of SafeCloudBox one needs at least four distinct datacenters (DCs), each of which runs a separate Ceph cluster (a storage spanning multiple hard disks within the DC). Each Ceph must offer an AWS S3-compatible API endpoint and will host exactly one S3 storage bucket created by SafeCloudBox. For CloudBlockStorage, two DCs running separate Ceph clusters are needed. At the time of writing, there is only one C&H DC available with a Ceph cluster that meets all requirements. Therefore we tested both pilots as far as is possible given that constraint. In the SafeCloudBox case, we use one instead of four DCs to host the S3 storage buckets, which is technically in no way different to four DCs from SafeCloudBox's point of view. Additionally, we run the coordination service on the same DC as SafeCloudBox itself. In the CloudBlockStorage case, we simulate two DCs using two Virtual Machines (VMs), each of which runs a Ceph cluster inside. Both pilots are fully functional and can be deployed using the required number of DCs once available.

The SafeCloudBox pilot received by far the largest update compared to the state described in D5.3. It is fully functional in terms of the integration of Nextcloud[2] with SafeCloudFS[3]. The current version of SafeCloudFS is a new implementation by IN-ID. This was tested extensively by C&H under real-world conditions, which has been a major part of the work on this pilot. Regarding the CloudBlockStorage pilot, we implemented and tested the OpenStack integration of replicated volumes, which provides the pilot's user interface. Both pilots passed all tests in D4.5.

For comments on further product development and exploitation of the pilots and the work done by C&H within the SafeCloud project period, we refer the reader to Sec. 4.

[1] https://ceph.com
[2] https://nextcloud.com
[3] https://github.com/inesc-id/SafeCloudFS

2 SafeCloudBox

A summary of the design goals described in D5.3 is given here for convenience. The SafeCloudBox product aims to provide users with enhanced data storage security and disaster recovery capabilities compared to services such as DropBox. To accomplish that goal, C&H uses SafeCloudFS (SS3). The product is a SaaS private cloud application which can run, for instance, on a company's infrastructure or a local machine and will provide cloud storage and secure data backups by using SafeCloudFS connected to C&H datacenters. The user interface is provided by Nextcloud, an open source private cloud application, which we adapt transparently to write data to C&H DCs via SafeCloudFS.

2.1 Pilot architecture

The SafeCloudFS client application provides a FUSE-based mount point to which data can be written. The data is then transferred to cloud storage. One advantage of SafeCloudFS is that it writes data encrypted to the cloud storage backends automatically. The second and even more important feature is that SafeCloudFS writes data to multiple cloud backends, thus even allowing data recovery in case a cloud provider's datacenter goes down. These points are very important to customers and as such, are equally important to C&H. In the most common deployment scenario, SafeCloudFS writes data to four backend S3 buckets, which can be distributed freely. One can, for instance, choose to have each bucket on a separate DC or host all four on one DC. Fig. 2.1 shows the overall deployment architecture of SafeCloudBox, with Fig. 2.1a describing the full deployment case using four backend DCs and four different DCs running the DepSpace coordination service for maximal fault tolerance.

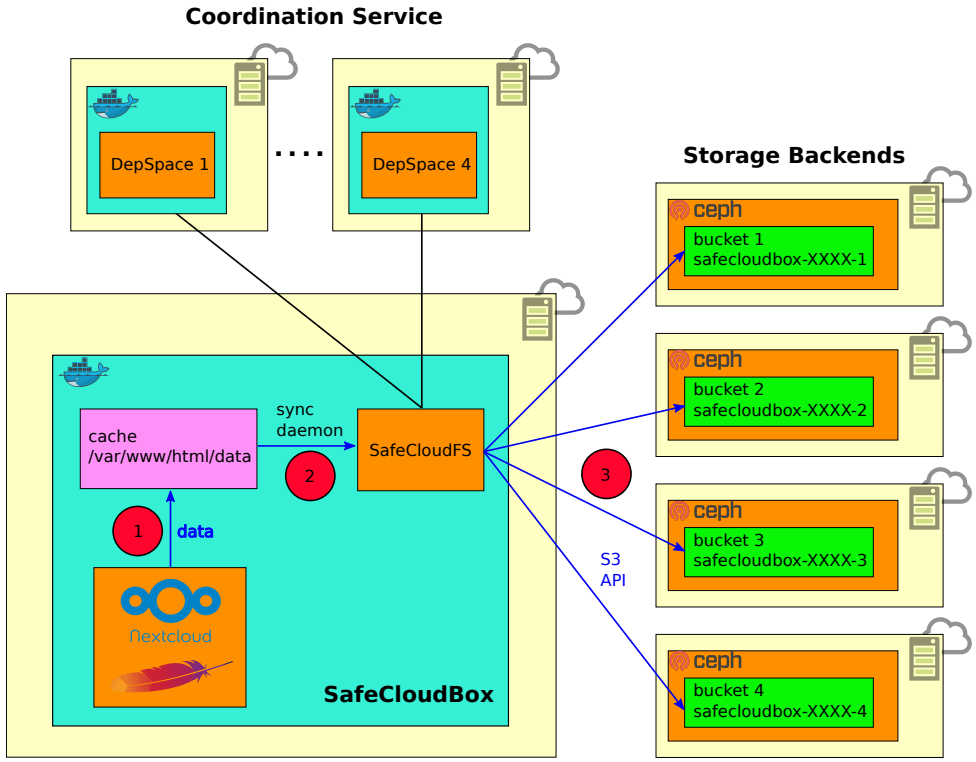
Due to the hardware constraints described earlier, we implemented the pilot in the way shown in Fig. 2.1b. Here, we run DepSpace and SafeCloudBox in the same DC. The S3 backend is run either on the same DC or a different one. We tested (i) production Ceph on same C&H DC, (ii) `ceph/demo` Docker container on same DC and (iii) AWS S3. In each case, all four S3 buckets were provided by one S3 endpoint. We stress that from SafeCloudBox's point of view, there is no technical difference between having one or four S3 backends since SafeCloudBox will write to four configured buckets in any case. In fact, one needs to change exactly one line per bucket in a configuration file.

2.2 Implementation

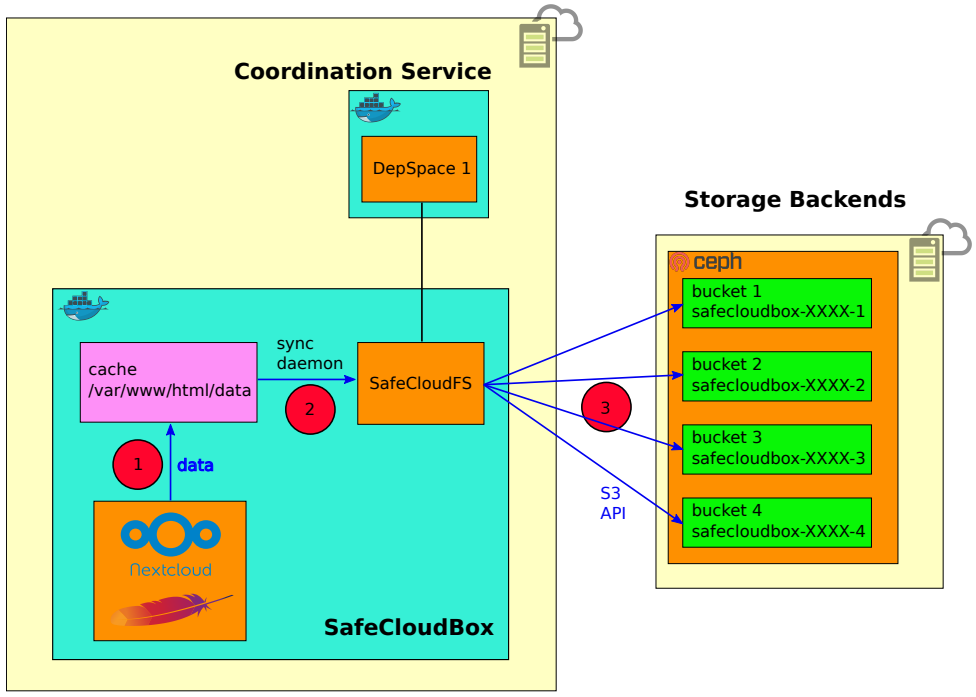
We now describe the implementation of the SafeCloudBox system and in particular the integration of Nextcloud and SafeCloudFS inside SafeCloudBox. The coordination service we use is an improved DepSpace version[1] from IN-ID, where each instance runs in a Docker container.

The core of SafeCloudBox is a combination of Nextcloud (user interface) and SafeCloudFS

[1] https://github.com/inesc-id/DepSpacito



(a) Full deployment using four DepSpace coordination service instances, each running on another DC, and four distinct Ceph backend DCs.



(b) Pilot deployment, running SafeCloudBox and DepSpace on the same DC, as well as one backend DC running Ceph and containing the four data buckets.

Figure 2.1: SafeCloudBox: Full and pilot deployment architecture.

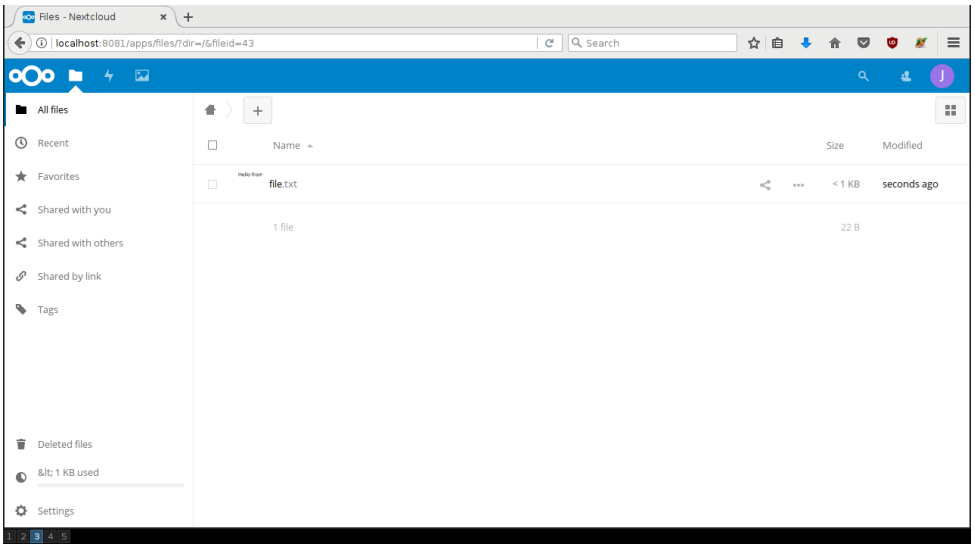


Figure 2.2: Browser window showing the Nextcloud user interface. A user is logged in and sees one file, which has been uploaded before.

(data storage backend). We combine both tools in a single Docker container. For that, C&H has forked[2] IN-ID's SafeCloudFS code[3] and replaced the base image by a Nextcloud base image[4]. By doing so, Nextcloud and SafeCloudFS are now contained in the same Docker image. The user interface of SafeCloudBox is the normal Nextcloud UI, which is a web page served by an Apache web server running in SafeCloudBox (the default Nextcloud configuration). This is shown in Fig. 2.2.

Data is stored in a dir `/var/www/html` on the host running Nextcloud. In our case, the Nextcloud Docker image exports this dir as Docker volume. That way, data gets written to the Docker host's disk instead of the container's OverlayFS file system abstraction layer which is slow. When a user called *user1* now uploads a file `file.txt`, this gets written to `/var/www/html/data/user1/files/file.txt`. The goal of SafeCloudBox is to back up this data using SafeCloudFS (i.e. write user data to SafeCloudFS, which then uploads it to the S3 backends).

SafeCloudFS has two modes of operation for how to synchronize data to the S3 backends: synchronous (*sync*) and asynchronous (*async*). The *async* method caches data locally and writes them to the backends asynchronously, which greatly improves local write speeds, whereas the *sync* method waits until all data uploads to the backends are finished. One would now simply sync `/var/www/html/data` to SafeCloudFS. However, this dir contains a lot of other data besides user data, such as the Nextcloud database which gets written often. One would therefore need to reconfigure Nextcloud to write user data to another dir and then define the mount point created by SafeCloudFS to be that user data dir. For that to work, SafeCloudFS must have an effective *async* method. However, we found the *sync* method to be more stable and thus implemented a separate local cache, while running SafeCloudFS in *sync* mode. With that approach, we bypass any Nextcloud reconfiguration. Also, we can selectively sync data such as `/var/www/html/data/user1`,

[2] https://github.com/CloudAndHeat/SafeCloudFS
[3] https://github.com/inesc-id/SafeCloudFS
[4] Nextcloud Docker image, https://hub.docker.com/_/nextcloud, version 13-apache

`/var/www/html/data/user2`. This is shown in Fig. 2.1. We keep `/var/www/html` unchanged, e.g. we use the Docker volume's fast local write speed as cache. We now mount SafeCloudFS to another dir `/mnt/safecloudfs` and use a directory sync daemon (to this end we use *lsyncd*[5]) to keep both dirs in sync. Files written or deleted on the Nextcloud side get replicated on the SafeCloudFS end.

2.3 Usage

To show how SafeCloudBox might be used, we created a series of screenshots showing the usage of the UI of SafeCloudBox (Nextcloud web interface). The user interaction in this example is a simple file upload. Additionally, we show the C&H backend side, where the user of SafeCloudBox is also logged in to the dashboard of the C&H DC running Ceph which hosts the backend S3 buckets. We show that the backends contain only encrypted data. The images can be found at the end of this document in Sec. A.

In order to use SafeCloudBox, a user needs to register an account with C&H which gives access to C&H's storage products (Ceph S3 in that case). Then, the user needs to fetch the SafeCloudBox code from the SafeCloudBox GitHub repository[6] and follow the instructions contained therein to set up SafeCloudBox. The code will start the pilot described above using *docker-compose*. Apart from SafeCloudBox, the repository contains some tools such as a minimal S3 client (using the *boto3* library[7]) and IO benchmark tools which have been used to measure the performance metrics reported in D4.5.

[5] Live Syncing (Mirror) Daemon: https://axkibe.github.io/lsyncd
[6] https://github.com/CloudAndHeat/safecloudbox
[7] https://aws.amazon.com/sdk-for-python

3 CloudBlockStorage

CloudBlockStorage is a SafeCloud-based security enhancement of inter-DC block data replication, which users can choose when creating block storage volumes in OpenStack. Data will be mirrored to a secondary DC to increase fault tolerance. The WAN connection between the DCs is secured by SafeCloud's SC2.

3.1 Pilot architecture

As described in D5.3, C&H's DCs offers IaaS level products, in particular VMs. C&H also offers block storage using 3-fold replication within a DC, based on Ceph. The block storage interface in Ceph is provided by the RBD layer (RADOS Block Device).

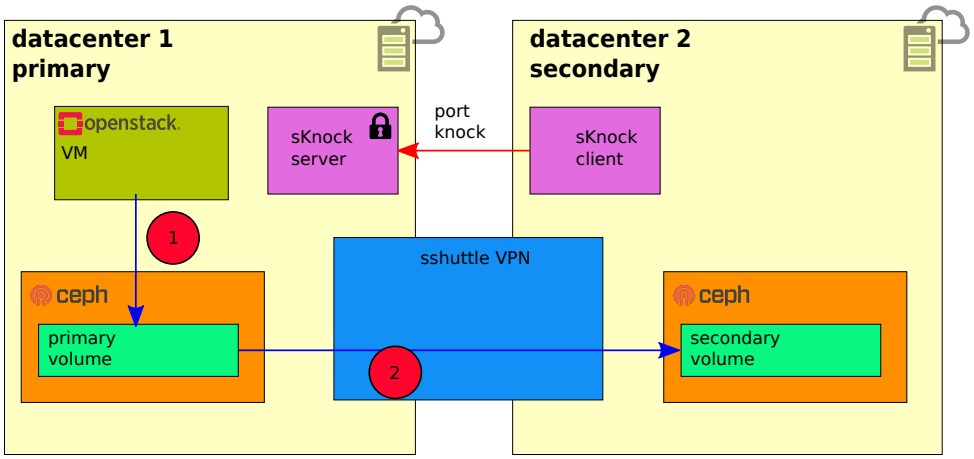
There are two related use cases of block storage. First, users are able to attach block storage volumes to VMs (like adding a hard disk to a computer), which allows persistent data storage, even if VMs are shut down or the compute node running the VM dies. This does not protect the complete VM data, but only the data a user chooses to store on the block storage volume, whereas VM data (e.g. the operating system, user home directory, ...) are stored on the compute node's hard disk by default. To address this issue, users have the option of placing the whole VM on a block storage volume. This has performance implications but adds data redundancy and recovery properties independently from compute nodes. In the SafeCloud context, C&H evaluated the possibilities to go beyond intra-DC data redundancy using inter-DC data replication, where DCs are connected via a WAN. This has, of course, security implications which are met by employing SafeCloud technology. Inter-DC replication gives users additional fault-tolerant data storage options and enhances C&H offers of safe and secure data storage.

The full deployment and pilot setups of CloudBlockStorage are shown in Fig. 3.1. In the pilot, we deploy two Ceph clusters, each in a separate VM which represent a DC in the real-world production setting. One of the clusters (primary) is the block storage backend used by VMs running in that DC. The second cluster (secondary) is the fallback cluster, to which data is synchronized. In terms of data replication technology, we employ a Ceph feature called "RBD mirroring", which allows to synchronize RBD block layer data between Ceph clusters.

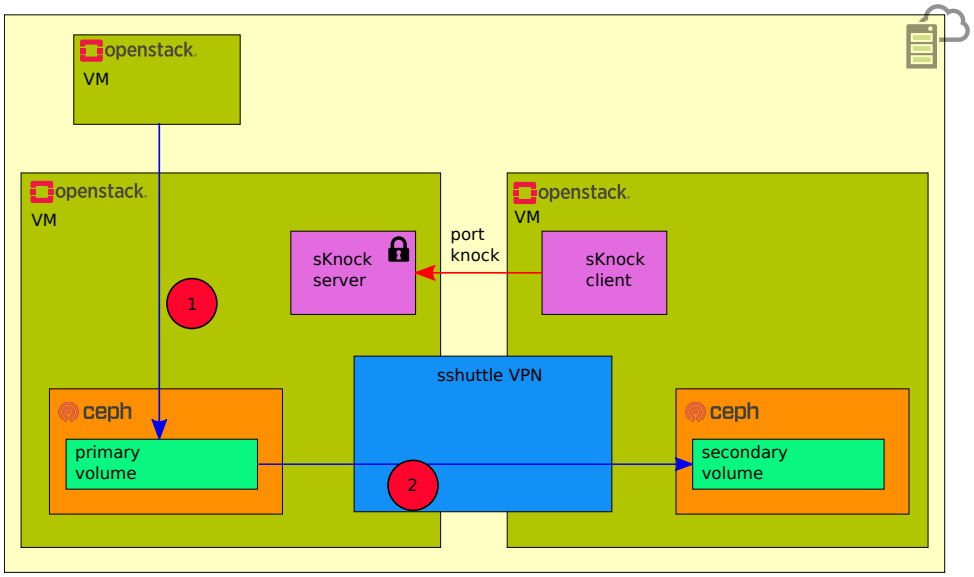
3.2 Implementation

The primary and secondary cluster are, in general, connected by a WAN and the RBD mirror protocol has no security layer. Therefore, one needs to secure that channel by other means. This is where C&H employs SafeCloud technology. As in D5.3, WAN communication between primary and secondary cluster is secured be a combination of *sshuttle*[1] and *sKnock*[2]. *sshuttle* is a simple VPN-like technology based on SSH, while *sKnock* is the port-

[1] https://github.com/sshuttle/sshuttle
[2] https://github.com/tumi8/sKnock



(a) Full deployment using two distinct DCs running Ceph. The DCs are connected by a VPN.



(b) Pilot deployment where DCs are simulated by VMs.

Figure 3.1: CloudBlockStorage: Full and pilot deployment architecture.

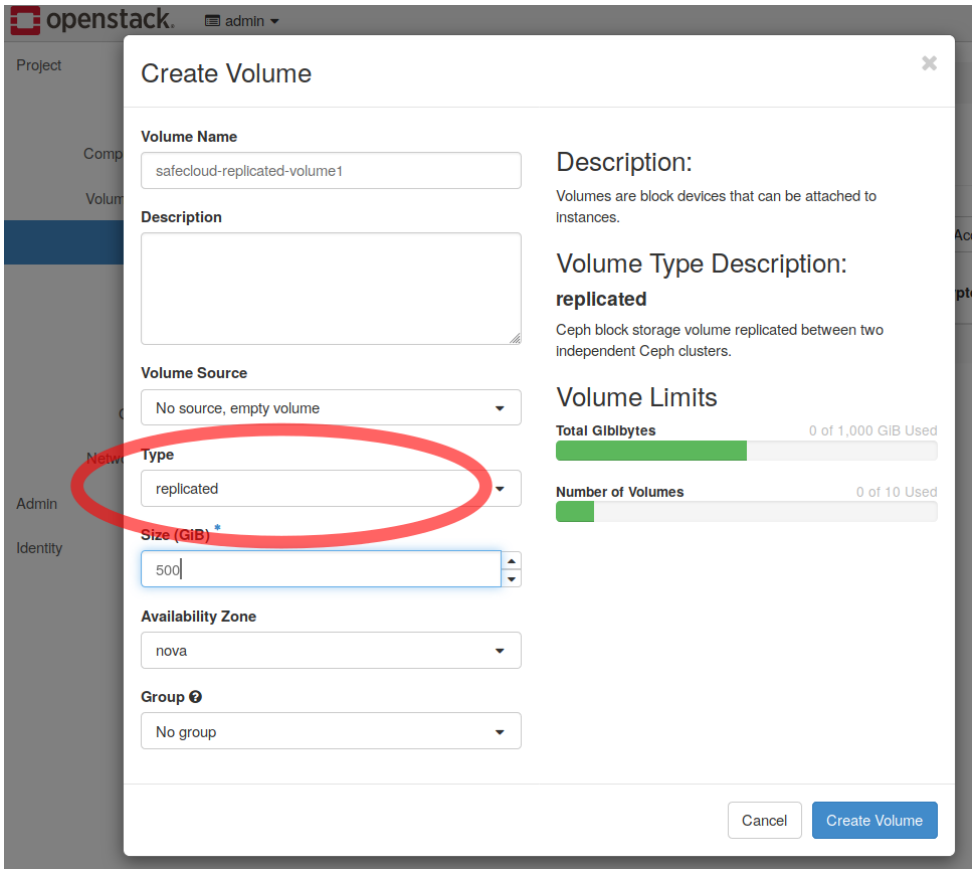


Figure 3.2: CloudBlockStorage UI via the OpenStack dashboard

knocking technology developed by TUM as part of SafeCloud (SC2). The *sKnock* server is running on a host on the primary cluster and gets port-knocked from the *sKnock* client on the secondary cluster to open a specific port. After the *sKnock* server has opened the requested port, *sshuttle* on a host on the secondary cluster is used to establish a transparent and encrypted routing of TCP flows into the layer-3 network of the primary cluster. The *rbd-mirror* daemon running in the secondary cluster then contacts the primary cluster over this tunnel to receive the cluster map and the journal of the primary cluster. The changes in the dataset of the primary cluster are then replicated ("replayed") on the secondary cluster.

In Ceph, volumes are the block devices which are presented to the user by RBD and which can be mounted. Volumes are organized in pools. Different pools can have varying properties, which also affect the contained volumes. In particular, a pool can be configured to replicate all its volumes to a secondary cluster. On the OpenStack side in a C&H DC, volumes are managed by the Cinder OpenStack service. In Cinder, we define a new volume type called *replicated*, which is defined to use a Ceph pool with RBD mirror replication enabled.

3.3 Usage

In terms of user interface, the user logs in to the C&H dashboard of the DC running the primary Ceph. In the dashboard UI, the user creates a volume and sets the *replicated* type. This is show in Fig. 3.2.

4 Conclusion

This deliverable described the two product pilots SafeCloudBox and CloudBlockStorage developed by C&H within the SafeCloud project.

The pilots have passed all tests in D4.5, which is an important step in the product development. Now, C&H is able to further refine the pilots into products and to deploy them on C&H's infrastructure, once more data centers meet the technical requirements. In case of changes in the company's product portfolio, C&H may also decide to postpone the deployment of these specific pilots. Even in that case, C&H will leverage a large part of the work performed within SafeCloud in terms of knowledge about SafeCloud and other technologies used to build the pilots. C&H may decide to use these technologies in other products or internal systems.

As one example, the SafeCloudBox development has pushed the testing and usage of our Ceph S3 API endpoints, which is a standard that every cloud storage provider needs to offer. Also, C&H uses Nextcloud internally and has gained improved operational knowledge while working on SafeCloudBox. Regarding CloudBlockStorage, C&H has internally evaluated various cross-DC data replication techniques. Here, RBD mirroring, which we used in this work, is one of the candidate technologies to approach this problem. Additionally, we have expanded our knowledge about operating and customizing Ceph clusters, which is now the central storage technology used by C&H.

A SafeCloudBox usage screenshots

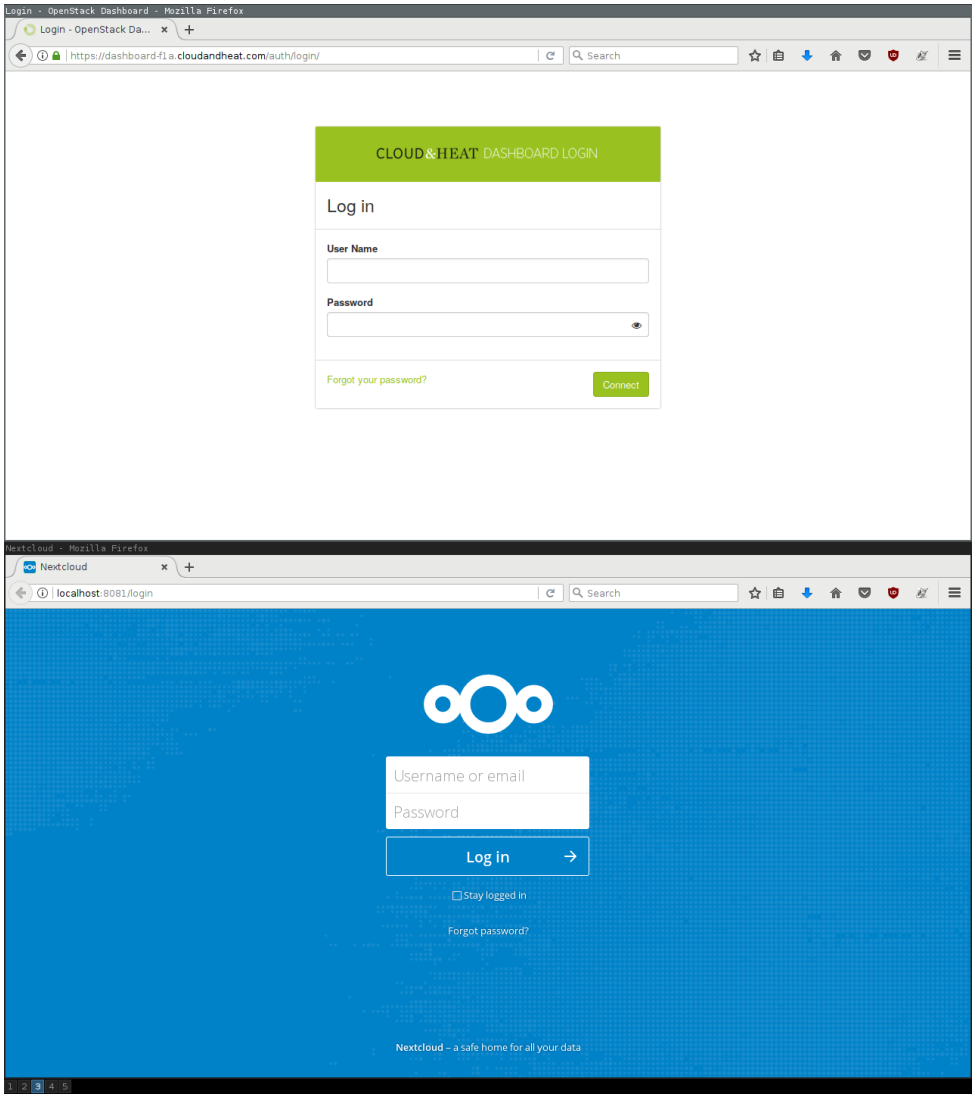


Figure A.1: Login pages: top C&H OpenStack dashboard; bottom: Nextcloud

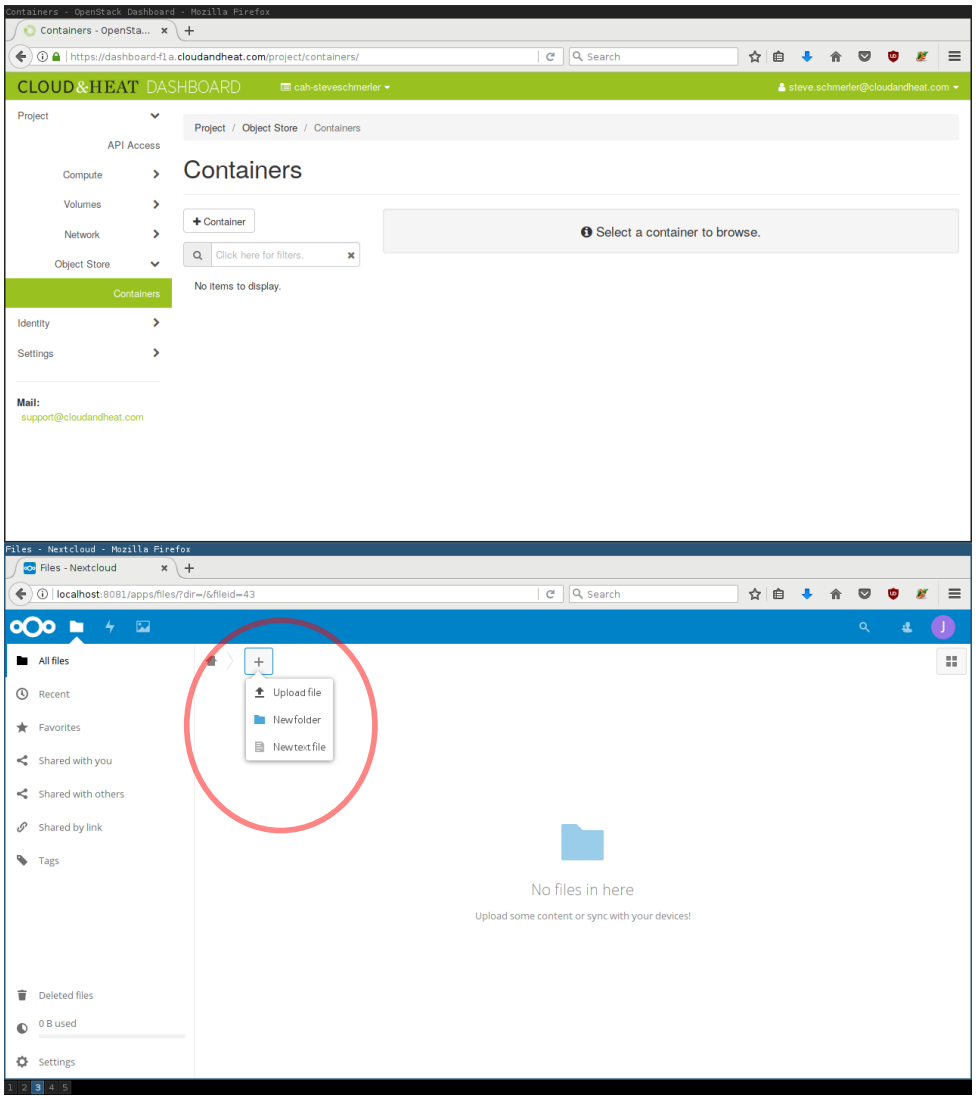


Figure A.2: Upload a file. Note: what we call ”buckets” in the text (S3 terminology) is called ”container” in OpenStack’s object storage service.

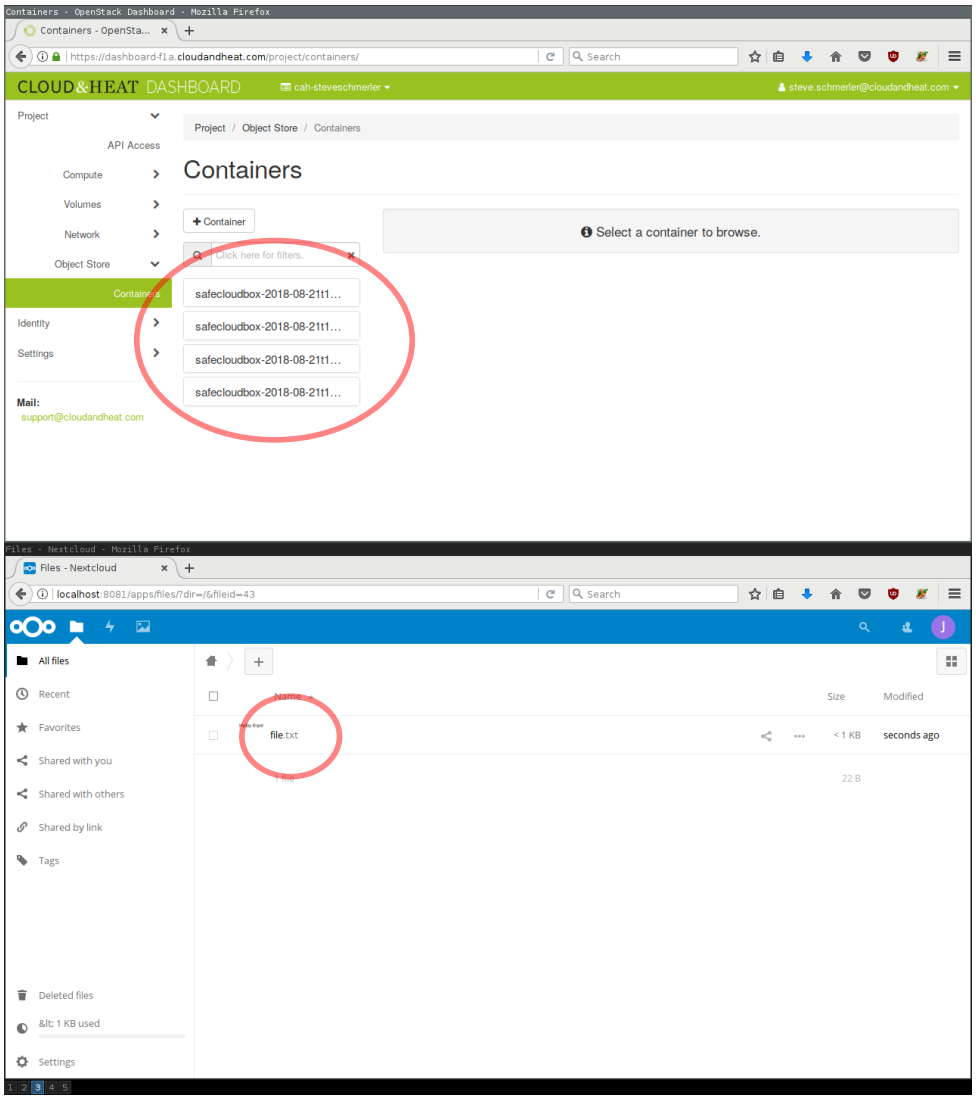


Figure A.3: File uploaded in Nextcloud. On the Ceph end, we see four buckets have been created.

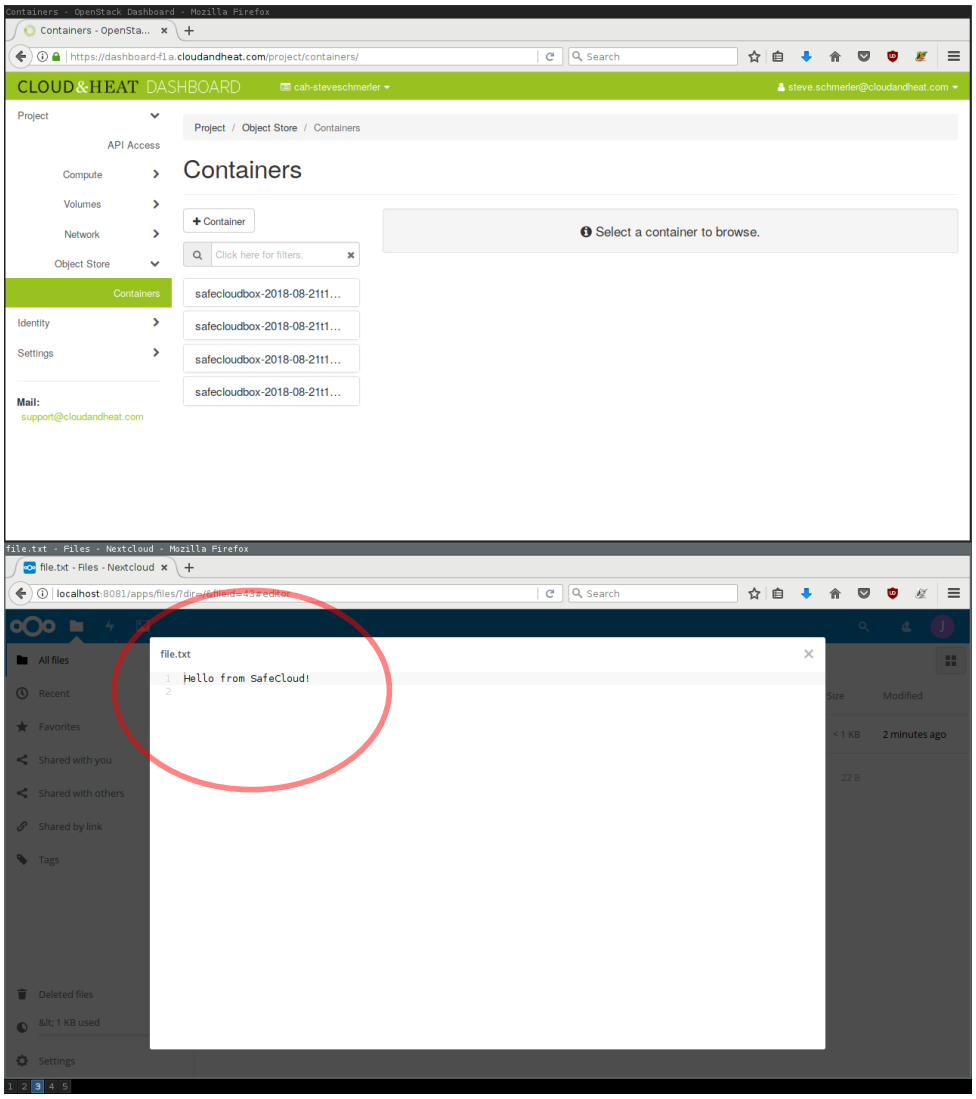


Figure A.4: File content: a simple file containing some text.

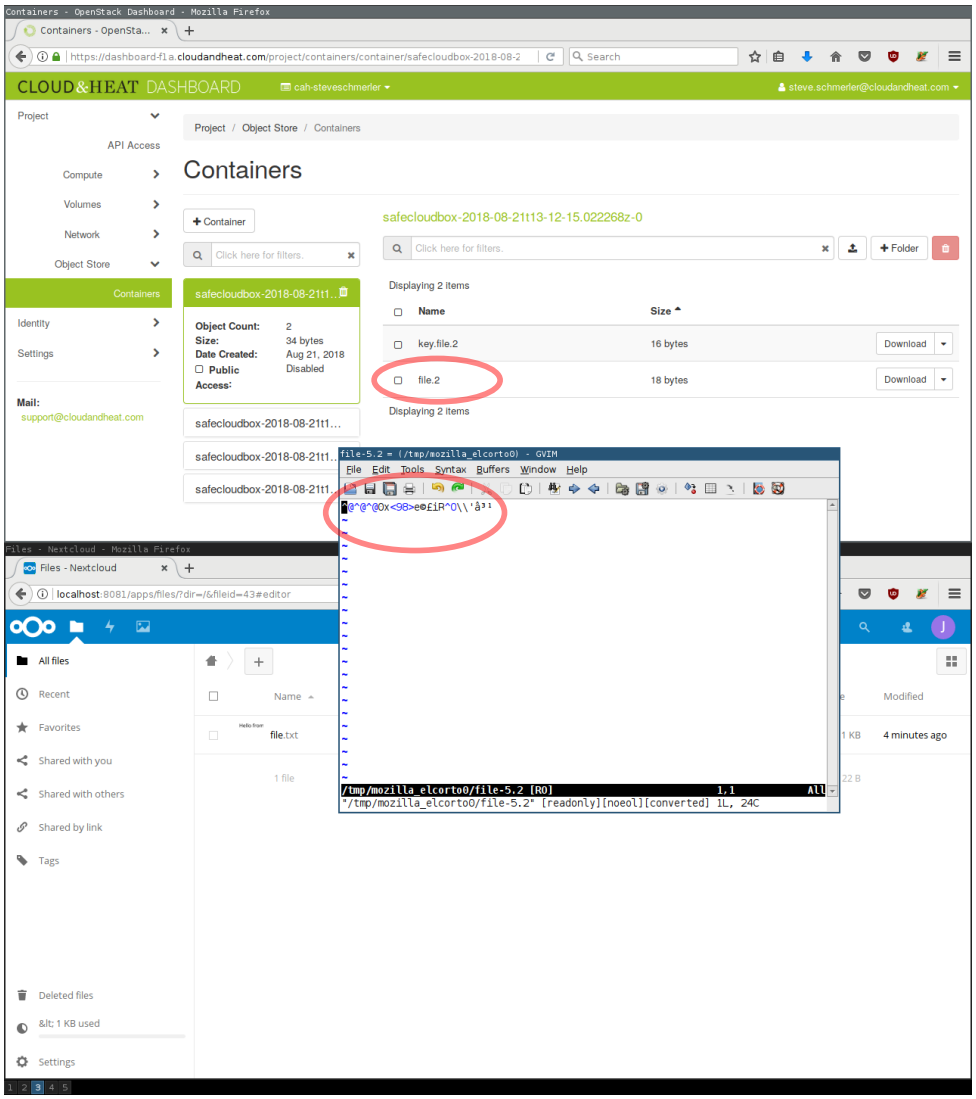


Figure A.5: We now open up one bucket and see data files created by SafeCloudFS. When reading them, we find only encrypted data.

B Legal aspects & GDPR compliance

In each use case, C&H shall be seen, from a data protection perspective, as a data processor, as each use case implies the processing of personal data on behalf of a potential data controller, but never for its own purpose. As such, a processor is required to provide sufficient guarantees to implement appropriate technical and organizational measures to the controller, in order for him to reach compliance with the GDPR. The use of SafeCloud technologies in each of the use cases provides a coherent framework that enables any potential data controller to meet the strong security requirements set by the GDPR against unauthorized or unlawful processing, but also against unwanted destruction or tampering of the data stored through its whole life cycle. Moreover, the use of SafeCloud technologies does not prevent the controller to control access to the data and define security measures that fit their security template. Furthermore, the use of SafeCloud's technologies allows the data controller who takes efficient organizational measures to provide quick, accurate and repeatable answers to potential data subjects' claims. In addition to SafeCloud technologies, C&H already applies strong standard datacenter measures such as TLS, VPNs, hard disk encryption or redundant data storage within a datacenter.