A SECURE MULTI CLOUD STORAGE SYSTEM FOR SHARING OF STORING BIGDATA USING CHARON

[1]*K.POORNAMBIGAI CSE & STANNE'S COLLEGE OF ENGINEERING AND TECHNOLOGY*

[2]*S.DHIVYA CSE & STANNE'S COLLEGE OF ENGINEERING AND TECHNOLOGY*

[3]R.*NISHANTHI CSE& STANNE'S COLLEGE OF ENGINEERING AND TECHNOLOY*

[4]N.*SUGANYA CSE & STANNE'S COLLEGE OF ENGINEERING AND TECHNOLOGY*

[5]J.*KEERTHIGA CSE& STANNE'S COLLEGE OF ENGINEERING AND TECHNOLOGY*

---***---

ABSTRACT

CHARON, a cloud-backed storage system capable of storing and sharing big data in a secure, reliable, and efficient way using multiple cloud providers and storage repositories to comply with the legal requirements of sensitive personal data.CHARON implements three distinguishing features: (1) it does not require trust on any single entity,(2) it does not require any client-managed server, and (3) it efficiently deals with large files over a set of geo-dispersed storage services. Besides that, we developed a novel Byzantine-resilient data-centric leasing protocol to avoid write-write conflicts between clients accessing shared repositories. CHARON using micro and application-based benchmarks simulating representative workflows from bioinformatics, a prominent big data domain. The results show that our unique design is not only feasible but also presents an end-to-end performance of up to 2:5_ better than other cloud-backed solutions.

INTRODUCTION

The high volume, velocity, and variety of data being produced by diverse scientific and business domains challenge standard solutions of data management, requiring them to scale while ensuring security and dependability. A fundamental problem is where and how to store the vast amount of data that is being continuously generated. Private infrastructures are the first option for many organizations. However, creating and maintaining data centers is expensive, requires specialized workforce, and can create hurdles to sharing. Conversely, attributes like cost-effectiveness, ease of use, and (almost) infinite scalability make public cloud services natural candidates to address data storage problems.

Unfortunately, many organizations are still reticent to adopt public cloud services. First, few tools are already integrated with clouds, introducing difficulties to non-technical users. Second, as with most organizations dealing with critical information, there are concerns about trusting data to externally-controlled services that occasionally suffer from unavailability and security incidents. Finally, depending on the nature of the data being analyzed, there are legal restrictions impeding such institutions to outsource the storage and manipulation of some of the datasets, especially when involving personal information.

We present CHARON, a near-POSIX cloud-backed storage system capable of storing and sharing big data with minimal management and no dedicated infrastructure. The main motivation for building this system was to support the management of genomic data, as required by bioinformatics and life sciences organizations. As an example, consider the case of biobanks.

These institutions were originally designed to keep physical samples that could be later retrieved for research purposes. More recently, they are becoming responsible also for storing and analyzing the data related to such samples. A sequenced human genome can reach up to 300GB, and each individual may have his genome sequenced many times during his life. The problem is that biobanks lack the scalable infrastructure for storing and managing this potentially vast data volume. Public clouds have plenty of resources for that.

Furthermore, the use of widely-accessible cloud services would facilitate the sharing of data among biobanks, hospitals, and laboratories, serving as a managed repository for public and access-controlled datasets. This would enable research initiatives that are not possible today due to the lack of a sufficient number of samples in a single institution. For example, 20–50k samples are required to study the interactions between genes, environment, and lifestyle that enable (or inhibit) a complex disease.

The rarer a disease is, the longer it takes to gather all necessary samples. Given the rarity of some diseases, it is unlikely that a single hospital or research institute will ever be able to collect the required number of samples. The problem is how to exploit the benefits of public clouds for data storage and sharing without endangering the security and dependability of biobanks' data.

CHARON OVERVIEW

we consider an unconstrained set of clients and a group of cloud providers. Each client has a unique id, an account for each cloud, and limited local storage. Every cloud provider offers one or more services,
which implement access control mechanisms to ensure that only authorized accounts can access them.To minimize the trust assumptions in the system, we consider that an unbounded number of clients and up to f cloud services can experience arbitrary (or Byzantine) faults. As in most storage systems, malicious clients can jeopardize the security properties of the files they have access to: they can leak information about the data they can read and modify or destroy the data they can write.Furthermore, the security properties of all information stored in faulty parties (clients or clouds) can be also compromised. As partof our assumptions, we include the extreme scenario where an active adversary takes control of all faulty parties, making themact together maliciously.

CHARON implements a security model where the owner of the file pays for its storage and defines its permissions. This is enforced by mapping the file system permissions (POSIX ACLs) to cloud services access control mechanisms (see details in x4.3).Therefore, a malicious client can only see, modify, and delete its own files and the files shared with him.

DESIGN OVERVIEW

CHARON is a distributed file system that provides a near-POSIXinterface to access an ecosystem of multiple cloud services and allows data transfer between clients. The preference for a POSIX interface rather than using data objects resorts to the fact the envisioned users are likely to be non-experts, and existent life sciencestools use files as their input most of the times. In particular, the system needs to (1) efficiently deal with multiple storage locations,(2) support reasonably big files, and (3) offer controlled filesharing. These challenges are amplified by our goals of excluding user-deployed servers and of requiring no modifications to existing cloud services (for immediate deployability).Addressing these challenges requires a novel data-centricprotocol to coordinate metadata writes, and the adaptation ofseveral multi-cloud storage techniques for working in a practicalsetting. The demand for these novel designs comes from the lackof efficient solutions to fulfill our needs (i.e., data-centric design,dependability, and performance).All techniques used in CHARON were combined considering two important design decisions. First, the system absorbs file writes in the client's local disk and, in the background, uploads
them to their storage location. Similarly, prefetching and parallel downloads are widely employed for accelerating reads. Thisimproves the usability of CHARON since transferring large filesto/from the clouds take significant time (see x5). Second, the system avoids write-write conflicts, ruling out any optimistic mechanism that relies on users/applications for conflict resolution. The expected size of the files and the envisioned users justify this decision. More specifically, (1) solving conflicts manually in big files can be hard and time-consuming; (2) users are likely to be non-experts, normally unaware of how to repair such conflicts; and (3) the cost of maintaining duplicate copies of big files can be significant. For instance, collaborative repositories, such as the Google Genomics , require such control since they allow users to read data about available samples, process them,and aggregate novel knowledge on them by sharing the resulting derived data into the bucket containing the sample of interest.

CHARON separates file data and metadata in different objectsstored in diverse locations and manages them using different strategies, as illustrated in Figure 1. File data locations are ofthree kinds in CHARON: cloud-of-clouds, single (public) storagecloud, and private repository . These alternativesexplore various cost-dependability tradeoffs and address allplacement requirements we have encountered with life sciences and big data applications. For example, the cloud-of-clouds canstore critical data (CHARON'S namespace and file B) that needsthe availability and confidentiality assured by the multi-cloud
scenario (provider-fault tolerance). A single cloud can store noncritical public studies and anonymized datasets (file D) (provider dependent and potentially less expensive). Finally, private repositories must be used to keep clinical data from human samples that cannot leave the boundaries of a particular institution (file A) or country (file C) (subject to local infrastructure restrictions).

CHARON maintains the namespace tree, together with the files'metadata, replicated in the cloud-of-clouds storage.The objective is to have only soft state in clients, which can be reconstructed after a crash by fetching data from the clouds.

EXISTING SYSTEM:

BiobankCloud are building a Hadoop-based platform for the secure storage, sharing, and parallel processing of genomic data. We extended Hadoop to include support for multi-tenant studies, reduced storage requirements with erasure coding, and added support for extensible and consistent metadata.

RELATED WORK

Choosing a cloud storage system and specific operations for reading and writing data requires developers to make decisions that trade off consistency for availability and performance. Applications may be locked into a choice that is not ideal for all clients and changing conditions. Pileus is a replicated key-value store that allows applications to declare their consistency and latency priorities via consistency- based service level agreements (SLAs). It dynamically selects which servers to access in order to deliver the best service given the current configuration and system conditions.

We design, implement, and evaluate MetaSync, a secure and reliable file synchronization service that uses multiple cloud synchronization services as untrusted storage providers. To provide global consistency among storage providers, we devise a novel variant of Paxos that enables efficient updates on top of the unmodified APIs exported by each service. MetaSync provides better availability, and performance, stronger confidentiality and integrity, and larger storage for end users.

In the BiobankCloud project, we are building a Hadoop-based platform for the secure storage, sharing, and parallel processing of genomic data. We extended Hadoop to include support for multi-tenant studies, reduced storage requirements with erasure coding, and added support for extensible and consistent metadata. On top of Hadoop, we built a scalable scientific workflow engine featuring a proper workflow definition language focusing on simple integration and chaining of existing tools, adaptive scheduling on Apache Yarn, and support for iterative dataflows.

DISADVANTAGES

- How to store the vast amount of data that is being continuously generated.
- creating and maintaining data centers is expensive, requires specializedwork force, and can create hurdles to sharing.

BYZANTINE-RESILIENT COMPOSITE LEASE

Previous data-centric fault-tolerant mutual exclusion algorithms were designed to work directly on top of storage services. In this paper, we propose a more modular approach in which we build non-fault-tolerant base lease objects, each on top of a specific cloud-provided service, and 3 f +1 of these services are combined in an f -fault-tolerant composite lease object. This approach allows the design of more efficient base lease objects on top of any cloud-provided service (e.g., queues) instead of relying on fault-tolerant register constructions as in previous works . We still provide the design of base lease objects on top of storage services because they are the only abstraction available in certain cloud providers (e.g., [40]). The lease operation of the composite lease object appears in Algorithm 1. To acquire a composite lease, a client simultaneously calls lease in each of the 3 f +1 base lease objects (Lines 5–6) and waits for either 2 f +1 successes or f +1 failures. The client acquired the lease. Otherwise, the lease is unavailable or under contention, and the client needs to release all potentially obtained leases (the successful and the unanswered ones) and back offs (Lines 11–13). This algorithm is repeated until it succeeds or a timeout is triggered.

Releasing a lease requires invoking the release operation in all base objects (as in Lines 11–12). The renew algorithm is similar to the one for lease, but with one additional cloud access to remove the old version of the lease being renewed. These two operations (release and renew) are never executed in the critical path of CHARON (see x4.1), and thus have little effect onthe latency of the system.

PROPOSED SYSTEM

CHARON, a cloud-backed storage system capable of storing and sharing big data in a secure, reliable, and efficient way using multiple cloud providers and storage repositories to comply with the legal requirements of sensitive personal data. CHARON implements three distinguishing features: (1) it does not require trust on any single entity, (2) it does not require any client-managed server, and (3) it efficiently deals with large files over a set of geo-dispersed storage services. Besides that, we developed a novel Byzantine-resilient data-centric leasing protocol to avoid write-write conflicts between clients accessing shared repositories.

CHARON is a cloud-backed file system for storing and sharing bigdata. Its design relies on two important principles: Files metadata and data are stored in multiple clouds, without requiring trust on any of them individually, and The system is completely datacentric. This design has led us to develop a novel Byzantineresilient leasing protocol to avoid write-write conflicts without any custom server.

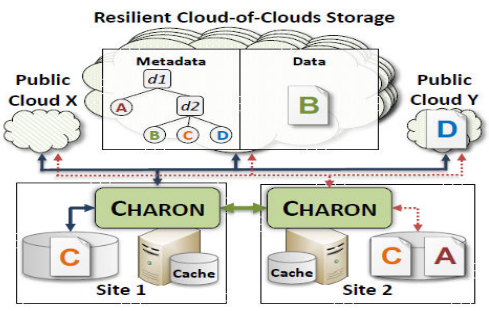


Fig .1System Architecture

CHARON implements a security model where the owner of the file pays for its storage and is able to define its permissions. This means that each client pays for all private data and all the shared data associated with the shared folders he created (independently on who wrote it).

CHARON clients are not required to be trusted since access control is performed by the cloud providers, which enforce the permissions for each object. Moreover, the cloud-of clouds access control is satisfied even if up to f cloud providers misbehave. This happens because if an object is read from up to faulty providers, no useful information will be obtained (recall that data is encrypted and keys are stored using secret sharing in a SWMR register).

CHARON is the first data-centric cloud-backed file system to support the main requirements of life sciences and other big data domains.

- A multiple cloud providers to improve the integrity and availability of stored data.
- Cloudserver is used to store large amount of data and also store owner and client details.

IMPLEMENTATION

CHARON is a user-space file system implemented using FUSE-J, a Java wrapper for the FUSE library . The system is fully implemented at the client side, using cloud services for storage and coordination, and is publicly available as open-source software at https://github.com/cloud-of-clouds/charon-fs.

Metadata Organization

Metadata is the set of attributes assigned to a file/directory (e.g., name, permissions). Independently of the location of the data chunks, CHARON stores all metadata in the cloud-of-clouds using single-writer multi-reader registers to improve their accessibility and availability guarantees (see x2.3). More specifically, we redesigned and optimized the SWMR register implementation of Dep Sky to improve the performance and concurrency as described in the remaining of this section.

Managing namespaces

All metadata is stored within namespace objects, which encapsulate the hierarchical structure of files and directories in a subdirectory tree. CHARON uses two types of namespaces: personal namespace (PNS) and shared namespace (SNS). A PNS stores the metadata for all non-shared objects of a client, i.e., files and directories that can only be accessed by their owner. Each client has only one associated PNS. On the other hand, a client has access to as many SNSs as the shared folders it can access. Each shared folder is associated with exactly one SNS, which is referenced in the PNSs of the clients sharing it. Although similar, personal and shared namespaces differ in the way the hashes of the most recent versions of the files' data chunks are stored. These hashes are primarily used to validate the cached file chunks. In PNSs, these hashes are serialized together with the rest of files' metadata before they are stored in the clouds. In the case of SNSs, the hashes are stored in a separate Chunk Hashes (CH) object, explained in the next section. Another difference between a PNS and a SNS is that the latter is associated with a lease to coordinate concurrent write accesses between different users which must be acquired before any update is executed on afile or directory in the namespace. Since each SNS is associated in Charon.

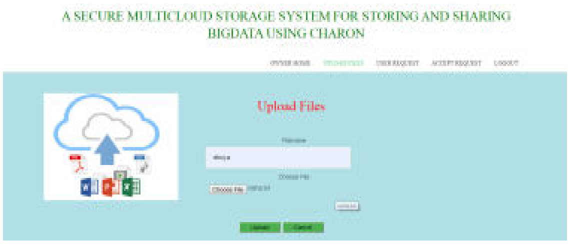
WRITE TO WRITE CONFLICT AVOID THE PROCESS

SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
HOME OWNER LOGIN USER LOGIN CLOUD LOGIN REGISTRATION
CLOUD COMPUTING

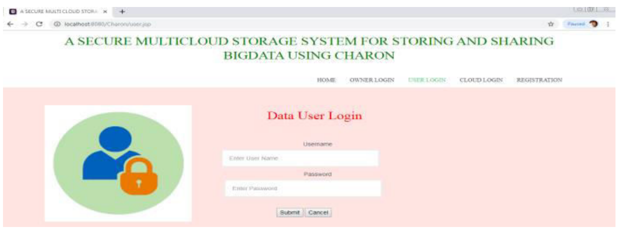
REGISTRATION

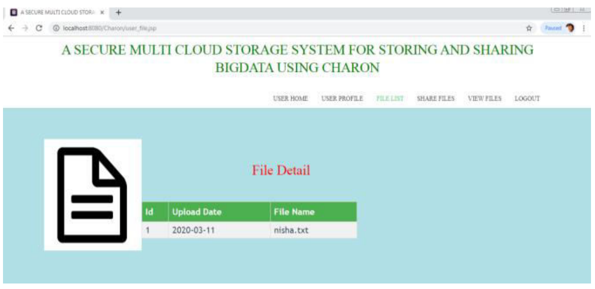
A SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
HOME OWNER LOGIN USER LOGIN CLOUD LOGIN REGISTRATION
Registration
Registration

DATA OWNER


A SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
OWNER HOME UPLOAD FILES USER REQUEST ACCEPT REQUEST LOGOUT
Upload Files
Filename
Choose File
Choose File
Upload
Cancel

INFORMATION CONSUMER

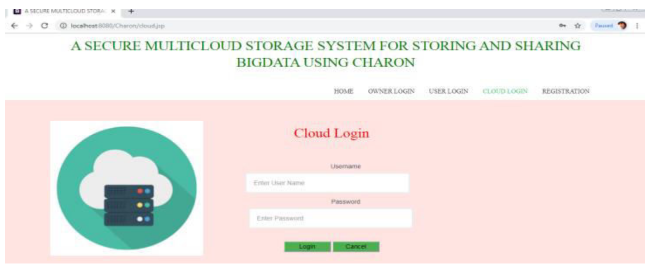

A SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
HOME OWNER LOGIN USER LOGIN CLOUD LOGIN REGISTRATION
Data User Login
Username
Enter User Name
Password
Enter Password
Submit
Cancel

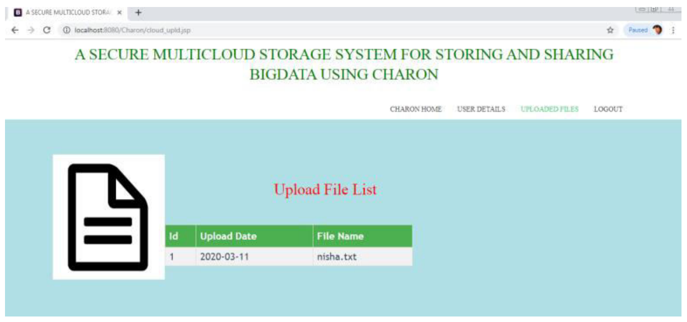

A SECURE MULTI CLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
USER HOME USER PROFILE FILE LIST SHARE FILES VIEW FILES LOGOUT
File Detail

Id	Upload Date	File Name
1	2020-03-11	nisha.txt

Fig file detail

CLOUD SERVER


A SECURE MULTICLOUD STORA
localhost:8080/Charon/cloud.jsp
Paused
A SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
HOME
OWNER LOGIN
USER LOGIN
CLOUD LOGIN
REGISTRATION
Cloud Login
Username
Enter User Name
Password
Enter Password
Login
Cancel


A SECURE MULTICLOUD STORA
localhost:8080/Charon/cloud_upld.jsp
Paused
A SECURE MULTICLOUD STORAGE SYSTEM FOR STORING AND SHARING BIGDATA USING CHARON
CHARON HOME
USER DETAILS
UPLOADED FILES
LOGOUT
Upload File List

Id	Upload Date	File Name
1	2020-03-11	nisha.txt

ADVANTAGES

- It including a novel leasing protocol to avoid write-write conflicts on shared files.
- Low cost and Less expensive
- It stores Large Amount of Data

CONCLUSION

CHARON is a cloud-backed file system for storing and sharing big data. Its design relies on two important principles: files metadata and data are stored in multiple clouds, without requiring truston any of them individually, and the system is completely datacentric.This design has led us to develop a novel Byzantineresilient leasing protocol to avoid write-write conflicts without any custom server. Our results show that this design is feasible and can be employed in real-world institutions that need to store and share large critical datasets in a controlled way.

REFERENCES

[1] Cloud Harmony, "Service Status," https://cloudharmony.com/
status-of-storage-group-by-regions, 2019.

[2] Cloud Security Alliance, "Top Threats," https://cloudsecurityalliance.org/
group/top-threats/, 2016.

[3] M. A. C. Dekker, "Critical Cloud Computing: A CIIP perspective on
cloud computing services (v1.0)," European Network and Information
Security Agency (ENISA), Tech. Rep., 2012.

[4] H. S. Gunawi et al., "Why does the cloud stop computing?: Lessons from
hundreds of service outages," in Proc. of the SoCC, 2016.

[5] European Commission, "Data protection," https://ec.europa.eu/info/law/
law-topic/data-protection en, 2018.

[6] G. Gaskell and M. W. Bauer, Genomics and Society: Legal, Ethical and
Social Dimensions.Routledge, 2013.

[7] A. Bessani et al., "BiobankCloud: a platform for the secure storage,
sharing, and processing of large biomedical data sets," in DMAH, 2015.

[8] H. Gottweis et al., "Biobanks for Europe: A challenge for governance,"
European Commission, Directorate-General for Research and Innovation,
Tech. Rep., 2012.

[9] P. E. Verissimo and A. Bessani, "E-biobanking: What have you done
to my cell samples?" IEEE Security Privacy, vol. 11, no. 6, pp. 62–65,
2013.

[10] P. R. Burton et al., "Size matters: just how big is big? Quantifying
realistic sample size requirements for human genome epidemiology," Int
J Epidemiol, vol. 38, no. 1, pp. 263–273, 2009.

[11] D. Haussler et al., "A million cancer genome warehouse," University of
Berkley, Dept. of Electrical Engineering and Computer Science, Tech.
Rep., 2012.

[12] R. W. G. Watson, E. W. Kay, and D. Smith, "Integrating
biobanks:
addressing the practical and ethical issues to deliver a valuable
tool for
cancer research," Nature Reviews Cancer, vol. 10, no. 9, pp.
646–651,
2010.
[13] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon,
"RACS: A case
for cloud storage diversity." SoCC, pp. 229–240, 2010.
[14] C. Basescu et al., "Robust data sharing with key-value
stores," in Proc.
of the