CDGeB: Cloud Data Geolocation Benchmark

Adi Offer
Cyber@BGU and Department of Software and Information Systems Engineering
Ben-Gurion University of the Negev
Be'er Sheva, Israel

Aviram Zilberman
Department of Computer Science
Jerusalem College of Technology
Jerusalem, Israel Cyber@BGU and Department of Software and Information Systems Engineering
Ben-Gurion University of the Negev
Be'er Sheva, Israel

Asaf Shabtai
Yuval Elovici
Rami Puzis
Cyber@BGU and Department of Software and Information Systems Engineering
Ben-Gurion University of the Negev
Be'er Sheva, Israel

ABSTRACT

Cloud computing has revolutionized data processing and management, offering flexible and scalable infrastructure for the distribution of content, computing power, and services across the globe. Dynamic, flexible, and transparent reallocation of resources increases cloud-based services' use and effectiveness. As rates of cloud adoption soar, privacy regulations, and geopolitical security introduce new challenges, which include the assessment, validation, and enforcement of data geolocation. However, currently, there is no standardized benchmark for this research domain. Therefore, this paper presents a novel dataset of measurements specifically designed to evaluate cloud data geolocation algorithms. In addition to its beneficial role in evaluating data geolocation algorithms, our dataset can be used for other data geolocation subtopics.

CCS CONCEPTS

• **Security and privacy** → Security protocols; **Database and storage security**; **Operating systems security**; • **Software and its engineering** → Contextual software domains; • **Information systems** → **Storage architectures**; **Cloud based storage**.

KEYWORDS

Geolocation, Cloud, Data Geolocation

ACM Reference Format:
Adi Offer, Aviram Zilberman, Asaf Shabtai, Yuval Elovici, and Rami Puzis. 2023. CDGeB: Cloud Data Geolocation Benchmark. In *Proceedings of Proceedings of the 2023 Cloud Computing Security Workshop (CCSW '23).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3605763.3625273

1 INTRODUCTION

The rapid evolution of cloud computing has revolutionized data processing and management, offering organizations unparalleled flexibility and scalability in handling their data [14]. As businesses increasingly rely on cloud infrastructure to store critical information, ensuring data security and compliance with geographical regulations has become a pressing concern [17].

A crucial issue in cloud data management is the geolocation of cloud-stored data. Geolocation involves determining the physical location of data, which has significant implications for data privacy, data residency requirements, and user experience [5]. Governments and regulatory bodies worldwide have established data residency regulations to safeguard sensitive information [14]. For instance, the European Union (EU)'s General Data Protection Regulation (GDPR) mandates that the personal data of EU citizens must be stored and processed within the EU or in countries with equivalent data protection regulations [2]. Failure to comply with the GDPR can result in substantial fines and penalties for businesses operating in the EU or handling EU citizens' data.

Data residency [5] requirements dictate that specific types of data must be stored and processed within designated geographic regions or jurisdictions. As a result, organizations must ensure that the location of their data complies with these regulations to avoid legal repercussions and protect their reputation.

Data sovereignty [17] emphasizes that data is subject to the laws and regulations of the country or region where it is stored. Organizations may prefer storing data within their own country's jurisdiction to maintain sovereignty over sensitive information.

Accurate geolocation of cloud data is challenging due to the distributed nature of cloud infrastructure [11]. Cloud Service Provider (CSP)s often have data centers located in multiple countries to ensure redundancy and availability and improve performance. This distributed setup introduces complexities in tracking and ensuring the precise physical location of data.

In this paper, we propose a novel dataset of measurements specifically designed to enable benchmarking of cloud data-geolocation algorithms. Creating a standardized dataset has become necessary to establish a unified paradigm in the domain of data geolocation within cloud environments. The dataset will promote research and advancements in this research domain, by enabling the evaluation of cloud data geolocation algorithms by measuring their performance and comparing them against each other. The main contributions of this paper can be summarized as follows:

- We introduce and define the challenge of data geolocation in cloud environments.
- We present a dataset of real-world measurements that can be used to evaluate data geolocation algorithms.

* This research was supported by Fujitsu under the Robust Geolocation project.

CCSW '23, November 26, 2023, Copenhagen, Denmark.

ACM ISBN 979-8-4007-0259-4/23/11…$15.00
https://doi.org/10.1145/3605763.3625273

The remainder of this paper is structured as follows: Section 2 provides an overview of various aspects of the cloud architecture and the challenges associated with data geolocation. Section 3 discusses related work in data geolocation. In Section 4, we describe the system model and methodology used to build the dataset, and in Section 5 we provide a description of the dataset. Our conclusions and ideas for future work are presented in Section 6.

2 DATA GEOLOCATION CHALLENGES

Cloud computing has revolutionized the data management landscape, offering organizations a flexible and scalable solution for storing and processing vast amounts of data. Cloud Service Providers (CSPs) maintains a distributed network of data centers and servers in various geographic regions to ensure redundancy, availability, and improved performance. This cloud architecture enables efficient resource allocation and seamless scalability to accommodate fluctuating demands.

2.1 Cloud Architecture

The cloud computing architecture involves the design and configuration of the cloud infrastructure and services. It relies on virtualization technology to create virtualized instances of computing resources, enabling multiple virtual machines (VMs) to run on a single physical server. A hypervisor (Virtual Machine Monitor (VMM)) manages and controls the VMs, ensuring optimized resource allocation and preventing contention, while the networking infrastructure facilitates secure communication and data transfer between cloud resources. Various data storage solutions and cloud databases are used for efficient data handling and management.

Orchestration and automation tools streamline resource provisioning and management, and load balancing mechanisms distribute incoming network traffic across servers to prevent overloading. Security measures, real-time monitoring, logging mechanisms, and API integration contribute to the architecture's robustness, while auto-scaling mechanisms, which adapt resources based on demand, contribute to scalability and elasticity. The latter is also achieved due to cloud applications' decoupling of storage and compute resources; such decoupling enables scaling based on an application's specific needs. Furthermore, the storage and compute nodes' physical (geographical) locations need not be the same.

Data geolocation challenge: On the one hand, the three-tier architecture (illustrated in Figure 1), with its independent distribution of storage and computation resources, increases the flexibility of data deployment, but on the other hand, it obscures the geolocation of the data, hiding it behind the second (application) tier. Geolocating these nodes is a challenging task: The data storage nodes are not visible to the end-user. Consequently, they cannot be pinged and thus cannot be directly geolocated. In addition, the computation nodes' physical properties (e.g., CPU) may affect any delay measurements collected by a data geolocation framework.

2.2 Cloud Caching

Distributed caching is a critical aspect of cloud computing that aims to improve data access times and reduce latency for frequently accessed data. Caching involves storing copies of data closer to the end-users or applications, reducing the need to retrieve the data from the original data source, such as a database or remote server. This caching mechanism significantly enhances cloud-based services and applications' overall performance and responsiveness.

Cache servers are dedicated servers within the cloud infrastructure that host the cached data. Caching algorithms determine which data should be stored in the cache memory and which data should be evicted to make room for new data. Common caching algorithms include Least Recently Used (LRU), First-In-First-Out (FIFO), and random replacement.

In a distributed cloud environment with multiple data centers, cache replication, and distribution techniques ensure that cached data is available across different locations to serve requests from users in various geographic regions. Replicating cache data strategically across data centers enhances data availability and reduces data access latency.

Data geolocation challenge: The dynamic, volatile nature of distributed caching makes data geolocation very difficult when cloud-based services rely on caching to improve the quality of service. First, caching mechanisms and the data movement are opaque, preventing external observers from tracing data between servers. Second, the exact location of the caching servers may not be known to external entities that would like to validate the data location. Third, probing, the most common building block of geolocation, may affect the cache, changing the location of the data items under investigation.

2.3 Cloud DNS Load Balancer

In cloud computing, a Domain Name System (DNS) load balancer is a dynamic and resource-efficient mechanism for distributing incoming traffic across multiple servers or endpoints. Unlike traditional hardware load balancers, a DNS load balancer utilizes the DNS infrastructure to distribute traffic, enabling it to contribute to scalability, fault tolerance, and high availability in cloud environments.

In this approach, multiple IP addresses are associated with a single domain name, pointing to a different server or resource within the cloud infrastructure. When a client initiates a connection by querying the DNS server for the domain name, the DNS load balancer responds with a list of these IP addresses. Clients then establish connections to the respective servers based on the IP addresses listed. This decentralized method ensures the even distribution of incoming requests, preventing any single server from becoming overloaded.

While DNS load balancing is simple and cost-effective, it has limitations. Due to caching mechanisms, changes to IP addresses might not be immediately reflected for all clients, potentially leading to uneven traffic distribution. Furthermore, DNS load balancers lack advanced traffic management features such as session persistence and real-time health checks.

Data geolocation challenge: Each time a probe initiates a session to a service that employs load balancing, the server that handles the requests and its geolocation may change, causing high variability and inconsistency between measurements. As a result, load balancers impair the reproducibility of data geolocation.

3 RELATED WORK

In this section, we review related work in cloud data geolocation.

Gondree and Peterson [8] introduced a general framework for geolocating data that is stored in the cloud. The authors claimed that standard IP geolocation methods do not apply to cloud infrastructure, since the data is stored in data centers that are hidden behind the cloud infrastructure. To overcome this problem, they proposed a solution that combines a standard geolocation method called Constraint-Based Geolocation (CBG) with Proof of Data Possession (PDP) techniques. Their evaluation of the proposed framework demonstrated its ability to achieve state-level accuracy, however, the study is limited given the fact that they did not specify whether their geolocation method is based on the Internet Control Message Protocol (ICMP) or HTTP messages. Moreover, they assumed that all data are held jointly by some set of target data centers whose physical distance from one another is large enough to be distinguishable. This, of course, assumes a very basic cloud architecture.

Zhang *et al.* [15] introduced Splitter, which is based on an improved version of CBG [8]. The proposed solution uses a PDP scheme based on "weak" and "strong" proofs. The "weak" proofs are used to measure the Round-Trip Time (RTT) for probing the server, and the "strong" proofs are used to validate the correctness of data possession in the server. The proposed method employs a random forest model to transform timing information into distances, which is accurate than the linear delay-distance transformation used in [8]. Splitter introduces an improved triangulation method that considers lower distance bounds to the server, however, the triangulation method assumes that probing the data centers with ICMP requests is possible, which is not the case in today's cloud services.

Noman *et al.* [13] proposed HDLAS, which extends their previous work. The proposed solution allows the user to store data in preferred locations and receive a provable assurance from the CSP. The authors assume a direct connection to the data center that stores the data. In addition, the proposed solution proves the authenticity of the data but does not geolocate it.

Albeshri *et al.* [3] proposed GeoProof, validating that the data is stored according to the Service-Level Agreement (SLA). Using the MAC-based Proof of Retrievability (PoR) scheme, a basic verifier-prover distance-bounding protocol and a tamper-proof device that is included in the cloud's LAN networks. Enhanced GeoProof [4] reduces the computational overhead on the server side by using a modified Proof of Storage (PoS) algorithm, improving the geolocation accuracy.

SecLoc [12] is a cryptographic-based framework that securely stores the data encrypted in the cloud so that decryption can only be performed in specified locations. SecLoc does not rely on any trusted equipment in the cloud servers or requires maintaining many keys, however a trusted region server (independent of the cloud) is needed, and it assumes that the CSP and regional servers are not being attacked simultaneously.

Eskandari *et al.* [7] proposed VLOC, which verifies the physical location of a VM on which the customer's applications and data are stored. VLOC uses several arbitrary web servers as external landmarks for localization and employs network latency measurement to estimate distance. Using measurement-based geolocation techniques based on HTTP requests, it probes landmarks and uses machine learning techniques to estimate the location of the target host in which it is executed.

Abid *et al.* [10] proposed a geolocation solution dedicated to Internet of Things (IoT) devices that store data in the cloud. The proposed geolocation scheme is based on PDP, similar to [8]. During the geolocation process, the CSP is challenged to retrieve random data blocks, and the retrieval RTT is measured. Then, the user validates the proofs received from the cloud and evaluates the distance to the cloud.

Reliablebox [9] is a secure framework for ensuring the data storage location. The client first computes integrity tags and then outsources the tags and files to the cloud storage server. In the later attestation, with the precise network delay and distance measurement from location-known verifiers, the client verifies that the outsourced files are intact and backed up on hosts at the specific geolocation. The simulations of Reliablebox show low accuracy of the data geolocation. Table 1 shows a comparison between the related works according to various criteria.

None of the prior studies mentioned above considered the multi-tier cloud architecture and did not reflect the real challenge of data geolocation. We argue that a reliable and accurate data geolocation method should consider an entire architecture. In addition, to the best of our knowledge, no previous work proposed a mechanism or a dataset to evaluate data geolocation methods.

4 DATA COLLECTION FRAMEWORK

4.1 Relevant AWS components

In this research, Amazon Web Services (AWS) was used to create a cloud environment. As officially published by Amazon [6], at the time of this writing, the AWS infrastructure consisted of primary regions in which Amazon data centers were established. Regions are referenced by a state-level name and their region code; for example, N. Virginia (*us-east-1*) or London (*eu-west-2*).

Each region has at least three availability zones, separated by up to 100 km for disaster tolerance. Availability zones are noted by region code followed by an alphabetic letter identifier; for example, *us-east-1a* and *us-east-1b* are both availability zones of N. Virginia.

AWS S3 (Simple Storage Service) is a storage service provided by Amazon. In this research, S3 buckets were used in different regions worldwide to store files. We also used AWS EC2 (Elastic Compute Cloud), a service for computing resources on demand. The EC2 instances were initialized in different regions to run HTTP servers representing a cloud-based service's front end. All EC2 instances were set up using *t2.micro* with the *Amazon Linux 2023 AMI* image.

The Cloud Data Geolocation Benchmark (CDGeB) service supports a single query that retrieves one named file through the REST API. The query should be directed to a specific front end, which locates the file, retrieves it from the relevant S3 bucket, and returns it to the client. The response times measured by the client, depend on the locations of the front-end and the file, as well as on the characteristics of the instances and their local state.

Figure 1 presents the 3-tier data geolocation measurement infrastructure. S3 buckets and EC2 cloud front-end servers are located in two overlapping sets of data centers. First, a client (white tag) requests a file using the REST API. Then, in the second step, the front-end server retrieves the data file. In steps 3 and 4, the file is

Articles	Subject	Year	Method used	Dataset used	Assumes caching	Assumes a front-end server
[3]	cloud data geolocation	2012	PoS and distance bounding protocol	✗	✗	✗
[8]	framework for data geolocation and verification	2013	basic geolocation and PDP	✗	✗	✗
[13]	hardware-based data geolocation for cloud storage.	2014	a trusted platform module and a PDP are implemented on a dedicated chip which a GPS	✗	✗	✗
[4]	enhanced version of GeoProof [3]	2014	modified PoS algorithm	✗	✗	✗
[7]	geolocation of VMs in the cloud	2014	planted software on the VM and delay-based triangulation	✗	✗	✗
[12]	secure data geolocation	2015	PoS with encryption	✗	✗	✗
[16]	secure distributed data geolocation scheme	2019	combination of proof of retrievability and the delay-based geolocation techniques	✗	✗	✗
[10]	cloud data geolocation for IoT	2020	PDP and multiagent-based approach	✗	✗	✗
[15]	determines the geolocation of cloud data stored in a semi-honest CSP publicly	2020	combination of random forest algorithm and an improved triangulation method	✗	✗	✗
[9]	secure and verifiable cloud storage	2021	challenge-based PoS and delay-based triangulation	✗	✗	✗

Table 1: Related work

delivered to the client through the front-end server. The measured RTT includes these four steps.

4.2 Definitions

The following naming conventions were used throughout the rest of this paper:

- *Challenge-i* - describes a featured challenge within the proposed benchmark, with index *i*.
- *cdgeb-probe-i* - describes the probing server located in region *i*, according to the list of probing servers regions described in Section 4.4.
- *cdgeb-server-i* - describes the front-end server located in region *i*, according to the list of regions in Section 4.3.
- *cdgeb-file-i* - describes a file stored in the cloud storage service. *i* does not follow the numbering system described in Section 4.3. *cdgeb-file-1* in particular does not necessarily reside in the same region as *cdgeb-server-1*.

4.3 System Model

Our service comprises S3 buckets, sample files stored in S3 buckets, and EC2 servers that function as the front-end servers of our application. The AWS regions used are:

(1) US East (N. Virginia) *us-east-1*
(2) EU (London) *eu-west-2*
(3) South America (São Paulo) *sa-east-1*
(4) Asia Pacific (Tokyo) *ap-northeast-1*
(5) Asia Pacific (Singapore) *ap-southeast-1*
(6) Canada (Central) *ca-central-1*
(7) US East (Ohio) *us-east-2*
(8) US West (N. California) *us-west-1*
(9) US West (Oregon) *us-west-2*
(10) Asia Pacific (Mumbai) *ap-south-1*
(11) Asia Pacific (Osaka) *ap-northeast-3*
(12) Asia Pacific (Seoul) *ap-northeast-2*
(13) Asia Pacific (Sidney) *ap-southeast-2*
(14) EU (Frankfurt) *eu-central-1*
(15) EU (Ireland) *eu-west-1*
(16) EU (Paris) *eu-west-3*
(17) EU (Stockholm) *eu-north-1*

The regions are denoted using a numerical convention throughout this paper (e.g., region *AWS-01* refers to *us-east-1*). Correspondingly, all EC2 instances are named based on the same numerical listing (e.g., *cdgeb-server-1* is located in region *AWS–01*). However, this numerical convention does not apply to file naming conventions (e.g., *cdgeb-file-1* does not necessarily imply that the file is located in region *AWS-01*).

We developed server-side software that runs on each EC2 instance. The server exposes a REST API, enabling Internet clients to request specific files from our cloud-based service. Upon receiving such a request, the EC2 instance initiates a retrieve request from the corresponding S3 bucket. Then, the requested file is transferred from the region in which the S3 bucket resides to the region in which the EC2 instance is located. Finally, the requested file is transferred back to the Internet client. The probing process is illustrated in Figure 1.

4.4 Probing

To probe the front-end servers, a total of 14 instances of Google Cloud Platform services were employed as Internet clients, distributed across the following locations:

(1) northamerica-northeast1 (Montreal)
(2) northamerica-northeast2 (Toronto)
(3) southamerica-east1 (Sao Paulo)
(4) us-west1 (Oregon)
(5) us-west2 (Los Angeles)
(6) europe-central2 (Warsaw)
(7) europe-north1 (Finland)
(8) europe-southwest1 (Madrid)
(9) me-central1 (Doha)
(10) me-central2 (Tel Aviv)
(11) asia-east1 (Taiwan)
(12) asia-east2 (Hong Kong)
(13) asia-northeast1 (Tokyo)
(14) asia-northeast2 (Osaka)

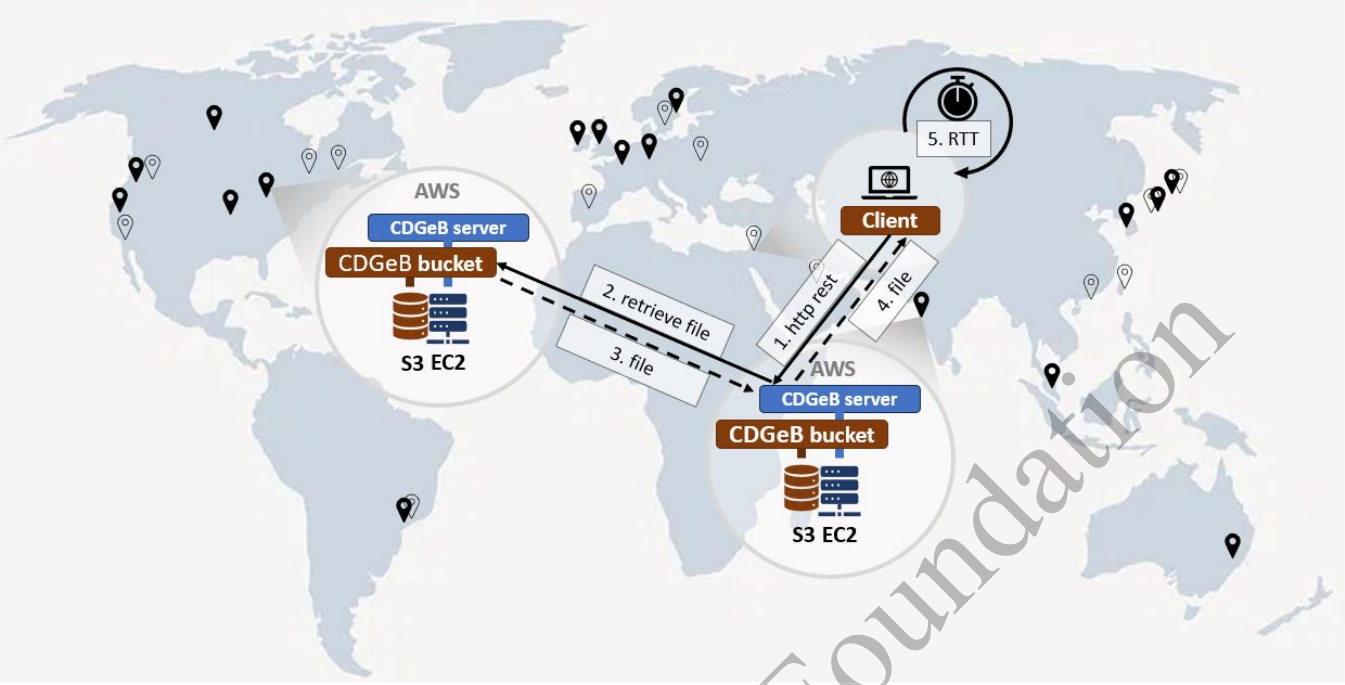

Figure 1: The 3-tier data geolocation measurement infrastructure. Black tags represent the real locations of the AWS data centers used to generate the data. White tags represent the real locations of the client applications hosted at Google data centers.

Each of these clients probed the cloud-based service. Each client is configured to probe a specific file in a specific location by requesting it from a front-end, however, the requested file and the front-end server are not necessarily in the same location.

5 BENCHMARK

This section describes the dataset used for the data geolocation benchmark.

The dataset consists of timing measurements from the probings conducted using the described system. Each one of the probe clients (*cdgeb-probe-i*) retrieves each one of the files (*cdgeb-file-i*) through multiple front-end servers (*cdgeb-server-i*).

The entries within the dataset are categorized into two distinct challenges that we introduce within this benchmark. The two challenges have varying levels of complexity in the task of geolocating cloud data. Using the provided dataset, a researcher is expected to determine the specific physical region in which any file within a challenge is situated.

Each line in the dataset contains the following information:

- *Difficulty - Challenge-i*, (*i=1,2*)
- *Probe - cdgeb-probe-i*, (*i=1..14*)
- *Front Server - cdgeb-server-i*, (*i=1..17*)
- *Data File - cdgeb-file-i*, (*i=1..17*)
- RTTs - values of 20 sequential probings, in units of seconds

Challenge-1 and Challenge-2 correspond to two difficulty levels. *Challenge-1* consists of the measurement of files 1-5 (*cdgeb-file-01*, .., *cdgeb-file-05*), and *Challenge-2* consists of the measurement of files 6-17.

	Challenge-1	Challenge-2
Total number of measurements	11,200	26,880
Number of probes	14	14
Number of files	5	12
Number of front-end servers (per file)	8	8
Repetitions per probe-server-file tuple	20	20
First RTT - mean (STD)	0.899 (0.468)	0.967 (0.399)
RTT without first – mean (STD)	0.445 (0.203)	0.453 (0.238)
Avg. value of min. RTT out of any 20 repetitions (STD)	0.438 (0.200)	0.446 (0.186)

Table 2: Dataset statistics

In both *Challenge-1* and *Challenge-2*, a random set of 8 front-end servers is chosen for the retrieval of each individual file. However, the challenges differ in their level of difficulty. In the easier *Challenge-1*, one server out of the 8 is located in the same geographical region as the retrieved file. In contrast, in the more difficult *Challenge-2*, none of the front-end servers within the set reside in the same geographical region as the corresponding file.

To collect the measurements, 14 different probes were employed, probing 17 front-end servers for a selection of 14 files. However, not

Difficulty	Probe	Front Server	Data File	RTT #1	RTT #2	RTT #3	RTT #4	RTT #5
Challenge-1	cdgeb-probe-14	cdgeb-server-12	cdgeb-file-01	0.271192	0.10165	0.104269	0.114672	0.111449
Challenge-2	cdgeb-probe-08	cdgeb-server-11	cdgeb-file-07	1.120666	0.62014	0.619905	0.614746	0.620332
Challenge-2	cdgeb-probe-14	cdgeb-server-16	cdgeb-file-06	1.104429	0.629728	0.619708	0.639373	0.624838
Challenge-2	cdgeb-probe-07	cdgeb-server-11	cdgeb-file-13	0.905342	0.600389	0.601311	0.597241	0.595656
Challenge-1	cdgeb-probe-13	cdgeb-server-15	cdgeb-file-03	1.434679	0.687423	0.692032	0.690057	0.693404
Challenge-1	cdgeb-probe-05	cdgeb-server-12	cdgeb-file-04	1.430711	0.548741	0.547811	0.547604	0.548021
Challenge-2	cdgeb-probe-01	cdgeb-server-12	cdgeb-file-10	2.401201	0.609209	0.609948	0.61107	0.611052
Challenge-2	cdgeb-probe-14	cdgeb-server-12	cdgeb-file-16	1.287669	0.365719	0.360802	0.361502	0.354872
Challenge-2	cdgeb-probe-04	cdgeb-server-05	cdgeb-file-10	1.033584	0.489651	0.484646	0.488195	0.488279

Table 3: Dataset demonstration

all possible combinations of probe-server-file were used, specifically to present the defined challenges. Within this framework, a total of 1,904 unique probe-server-file tuples were employed. For each tuple, the probing process was iterated 20 times consecutively, culminating in a total of 38,080 measurements. Table 2 provides statistical information about the dataset, as well as Standard Deviation (STD).

Table 3 provides a dataset example, presenting a partial subset of the dataset due to its extensive size. The leftmost column indicates the challenge corresponding to the record. The three to the right of it refer to the probing client used in the measurement, the front-end server used, and the name of the file requested. The remaining columns contain the RTT values from sequential measurements.

The benchmark was published on September 19, 2023, on the Kaggle platform as a community competition: https://www.kaggle.com/c/cloud-data-geolocation-1-multi-tier-services [1]

6 CONCLUSION AND FUTURE PLANS

Cloud data geolocation plays a pivotal role in modern cloud computing, focusing on strategically distributing data across global regions. This paper explored its significance, mechanisms, and implications.

Organizations address data sovereignty, compliance, and performance optimization by situating data in specific locations. Techniques like Content Delivery Networks (CDN) and distributed storage systems enhance data access speed and fault tolerance.

However, challenges arise from the various regulations, demanding a balanced approach. Effective implementation requires collaboration between legal experts, IT professionals, and cloud providers. Related works proposed data geolocation solutions without considering the multi-tier cloud architecture. In addition, a framework for benchmarking a geolocation system is required.

Therefore, in this paper, we proposed a novel dataset of measurements specifically designed to enable benchmarking of cloud data geolocation algorithms. We believe that the publication of this dataset will facilitate the entry of researchers into the field and promote advancements in this research domain.

In future research, our dataset could be expanded to consider data caching or DNS load balancing, and an evaluation scheme for the comparison of data geolocation methods could be explored. In addition, further research should be performed to create new data geolocation algorithms that consider multi-tier cloud architecture.

REFERENCES

[1] Aviram Zilberman Adi Offer, Asaf Shabtai and Rami Puzis. 2023. Cloud Data Geolocation 1 - Multi-Tier Services. https://kaggle.com/competitions/cloud-data-geolocation-1-multi-tier-services

[2] Haris Ahmad and Gagangeet Singh Aujla. 2023. GDPR compliance verification through a user-centric blockchain approach in multi-cloud environment. *Computers and Electrical Engineering* 109 (2023), 108747.

[3] Aiiad Albeshri, Colin Boyd, and Juan Gonzalez Nieto. 2012. Geoproof: Proofs of geographic location for cloud computing environment. In *2012 32nd International Conference on Distributed Computing Systems Workshops*. IEEE, 506–514.

[4] Aiiad Albeshri, Colin Boyd, and Juan González Nieto. 2014. Enhanced geoproof: improved geographic assurance for data in the cloud. *International Journal of Information Security* 13 (2014), 191–198.

[5] Bader Alouffi, Muhammad Hasnain, Abdullah Alharbi, Wael Alosaimi, Hashem Alyami, and Muhammad Ayaz. 2021. A systematic literature review on cloud computing security: threats and mitigation strategies. *IEEE Access* 9 (2021), 57792–57807.

[6] Amazon. 2023. Overview of Amazon Web Services. https://docs.aws.amazon.com/whitepapers/latest/aws-overview/introduction.html

[7] Mojtaba Eskandari, Anderson Santana De Oliveira, and Bruno Crispo. 2014. Vloc: An approach to verify the physical location of a virtual machine in cloud. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*. IEEE, 86–94.

[8] Mark Gondree and Zachary NJ Peterson. 2013. Geolocation of data in the cloud. In *Proceedings of the third ACM conference on Data and application security and privacy*. 25–36.

[9] Tao Jiang, Wenjuan Meng, Xu Yuan, Liangmin Wang, Jianhua Ge, and Jianfeng Ma. 2021. ReliableBox: Secure and Verifiable Cloud Storage With Location-Aware Backup. *IEEE Transactions on Parallel and Distributed Systems* 32, 12 (2021), 2996–3010.

[10] Abid Khan, Sadia Din, Gwanggil Jeon, and Francesco Piccialli. 2020. Lucy with agents in the sky: trustworthiness of cloud storage for industrial internet of things. *IEEE Transactions on Industrial Informatics* 17, 2 (2020), 953–960.

[11] Rakesh Kumar and Rinkaj Goyal. 2019. On cloud security requirements, threats, vulnerabilities and countermeasures: A survey. *Computer Science Review* 33 (2019), 1–48.

[12] Jingwei Li, Anna Squicciarini, Dan Lin, Shuang Liang, and Chunfu Jia. 2015. Secloc: Securing location-sensitive storage in the cloud. In *Proceedings of the 20th ACM Symposium on Access Control Models and Technologies*. 51–61.

[13] Ali Noman and Carlisle Adams. 2014. Hardware-based DLAS: Achieving geolocation guarantees for cloud data using TPM and provable data possession. In *2014 17th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 280–285.

[14] Dilli Babu Salvakkam and Rajendra Pamula. 2023. An improved lattice based certificateless data integrity verification techniques for cloud computing. *Journal of Ambient Intelligence and Humanized Computing* (2023), 1–20.

[15] Yang Zhang, Dongzheng Jia, Shijie Jia, Limin Liu, and Jingqiang Lin. 2020. Splitter: an efficient scheme to determine the geolocation of cloud data publicly. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 1–11.

[16] Yinyuan Zhao, Haoran Yuan, Tao Jiang, and Xiaofeng Chen. 2019. Secure distributed data geolocation scheme against location forgery attack. *Journal of Information Security and Applications* 47 (2019), 50–58.

[17] Jinglin Zou, Debiao He, Sherali Zeadally, Neeraj Kumar, Huaqun Wang, and Kkwang Raymond Choo. 2021. Integrated blockchain and cloud computing systems: A systematic survey, solutions, and challenges. *ACM Computing Surveys (CSUR)* 54, 8 (2021), 1–36.