UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Edgar Selihov

CloudTraceBucket: Cloud Trace Visualization and Management Platform

Bachelor's Thesis (9 ECTS)

Supervisor: Chinmaya Kumar Dehury, Ph.D.

Tartu 2022

CloudTraceBucket: Cloud Trace Visualization and Management Platform

Abstract
Lately, the amount of data generated by cloud technologies has been growing exponentially. To cope with this amount of data, various web tools and applications have been developed by software engineers, scientists and researchers for statistics and analysis of cloud traces. The problem is that cloud researchers should use a variety of web resources to find the traces of the cloud data they need. In addition, the data must be manually processed before it can be visualized. The goal of the thesis is to collect publicly available cloud traces (e.g virtual machine's behavior, serverless platforms, server workloads) and create a web application that allows cloud researchers to easily upload, download and visualize data based on user-selected query filters. The creation of this web application would enable researchers to analyze cloud tracking data and consolidate all the data into one place. The thesis mainly describes the cloud providers, their cloud traces along with the development and architecture of the web application.

Keywords:
Web development, back-end development, front-end development, cloud traces, Kotlin, Postgres, Spring Boot, REST, JavaScript, DB, Hibernate

CERCS: P170 Computer science, numerical analysis, systems, control

CloudTraceBucket: Pilve Jälgede Visualiseerimis- ja Haldusplatvorm

Lühikokkuvõte
Viimasel ajal on eksponentsiaalselt kasvanud pilvetehnoloogiate poolt genereeritud andmete hulk. Selle andmemahuga toimetulekuks on tarkvaraarendajad ja teadlased loonud erinevaid veebitööriistu ja rakendusi, et pilvejälgi analüüsida ning teha nende põhjal statistikat. Probleem on selles, et pilveuurijad peavad kasutama erinevaid veebiressursse, et leida pilveandmetest vajalikke jälgi. Lisaks peab andmeid enne visualiseerimist käsitsi töötlema. Lõputöö eesmärk on koguda avalikult kättesaadavad pilvejäljed (nt virtuaalmasinate käitumine, serverita platvormid, serverite töökoormused) ning luua veebirakendus, mis võimaldab pilveuurijatel pilvejälgi lihtsalt üles laadida, alla laadida ja visualiseerida andmed vastavalt kasutaja valitud filtritele. Selle veebirakenduse loomine võimaldab teadlastel analüüsida pilvejälgi ja koondada kõik andmed ühte kohta. Lõputöö kirjeldab peamiselt pilvepakkujaid, nende pakutavaid pilvejälgi, veebirakenduse arendamist ja arhitektuuri.

Võtmesõnad:
Veebiarendus, back-end arendus, front-end arendus, pilvejäljed, Kotlin, Postgres, Spring

Boot, REST, JavaScript, DB, Hibernate

CERCS:P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaat - juhtimisteooria)

Contents

1 Introduction

Cloud technologies are developing rapidly every day. Their impact on the end-user and industry cannot be underestimated as they are an integral part of modern life. Thanks to cloud technologies, companies can optimize their costs and expand their applications without having to purchase physical hardware. Researchers can process data in quantities that were previously only available in highly funded data centres. At the same time, ordinary Internet users can access their programs, file storage and all knowledge of the world with a few clicks.

1.1 Cloud Computing

In recent years, commercial organizations, banks and software companies have become highly interested in cloud computing. A huge amount of data measured in petabytes is generated from a variety of sources, from smart home appliances to weather stations, social networks, streaming services. As a result, companies are slowly migrating their products to cloud servers because cloud servers can handle such a large amount of data. Cloud computing is the provision of resources to an end user via an Internet. A resource can be anything related to calculations and computers such as software, hardware, server network infrastructure, or large network servers [1]. Cloud computing offers several benefits, such as dynamic scalability (the ability to increase workload on demand), flexibility (when a server must allocate more resources to computing), and a "pay as you use" feature. The largest providers of cloud computing and trace data are Google, Yahoo, Alibaba and Amazon. While cloud computing seems promising and attractive, it also comes with some limitations. Not all trace data can be published because it may contain confidential customer information or internal statistics for cloud servers (such as IP addresses or organizational policies). Therefore, cloud data must be filtered out of confidential data before it can be published [2].

1.2 Cloud trace Data

All cloud servers produce so-called cloud trace data. Cloud trace data is data that contain information about the resource usage of virtual machines, HTTP requests from servers, job executions and time performed by the server, and tasks performed within jobs. Trace data can be presented in a variety of formats, such as CSV[1], JSON[2], EXCEL[3] or SQLite[4]. Before cloud data is published, it is anonymized using various obfuscation techniques.

[1]https://csvloader.com/csv-guide/what-is-csv
[2]https://developers.squarespace.com/what-is-json
[3]https://excelx.com/what-is-excel
[4]https://www.sqlite.org/about.html

For example, Google used random hashing for free-text fields, resource sizes have been linearly transformed (scaled), and some other specific values have been mapped to predefined values [3]. After the data is prepared, providers will publish the cloud trace data for research to their public resources.

1.3 Motivation and thesis contributions

The main motivating factor of the thesis is the lack of such a solution among the existing tools, which could allow the researcher to effortlessly process and analyze cloud trace data. Similar solutions are either private and used in small communities, or too complicated to work with.

The main contributions of the thesis are to gather and analyze public cloud trace data, to group analysed cloud trace data by common fields and create generalised tables, to plan and create the CloudTraceBucket architecture with microservices, and to develop a stable, working web application. CloudTraceBucket (CTB) is a web platform, which allows researchers to process and analyze cloud trace files by uploading, downloading and visualizing cloud trace data in a web browser. It is a universal platform, aimed primarily to simplify data processing.

1.4 Thesis outline

This thesis is structured as follows. Section 2 provides an introduction to technologies and frameworks, briefly describes cloud trace providers and their trace files that helped to develop CloudTraceBucket. Section 3 describes the related works and compares them with CloudTraceBucket. Section 4 describes the high-level architecture of the CloudTraceBucket, describing a complete dataset flow in the CloudTraceBucket system. Section 5 describes responsibilities of each service, and how they work. Section 6 summarizes the thesis and discusses the possible improvements.

2 Background

This section gives information about the web application built in the thesis and introduces microservices and technologies used in the implementation. The section will cover each microservice responsibility and tools used for its development.

2.1 Technologies

Before starting any development, it is vital to indicate what technologies will be used to create the application because correctly chosen technology will contribute to a quick and efficient implementation of the functionality required for the application, as well as avoid difficulties in further development. It is important to consider technologies that are not outdated and maintained by the developers' community to avoid unexpected bugs and security holes.

2.1.1 Kotlin

Kotlin[5] is a general purpose, free, open source, statically typed, object-oriented (OOP) programming language initially designed for Java Virtual Machine (JVM) and Android[6]. It was chosen because of its concise, less verbose syntax comparing to Java[7]. Additionally, it is interoperable with Java, which means that Kotlin can work with it and use its libraries. Kotlin was developed by JetBrains in 2010 (Figure 1).

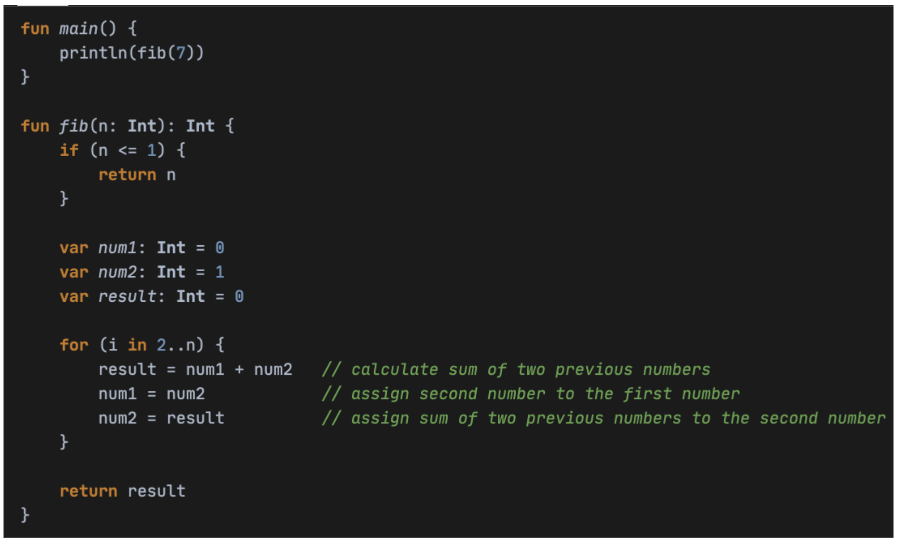
Comparison of Kotlin with other programming languages for performance is difficult because the definition of performance can have several meanings. For example, performance means not only the task execution speed but also stability, scalability, security and user experience [4]. In addition, performance comparisons become more sophisticated as each programming language has been developed for different purposes. But Kotlin can be compared to Java as both run on the JVM.

Java compilation time is 15-20% faster than Kotlin. However, if the Kotlin project is built incrementally, Kotlin shows compilation performance similar to Java [5].

2.1.2 JavaScript

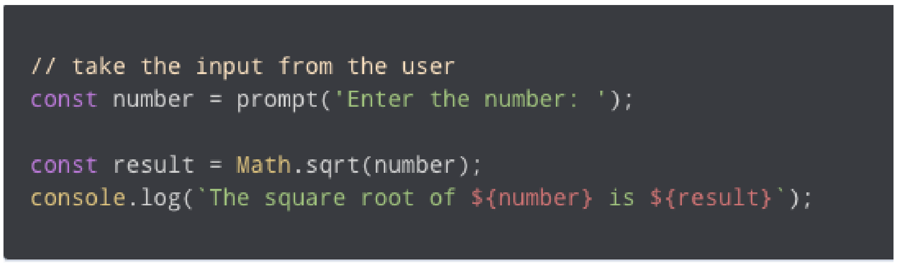
JavaScript[8] is a dynamically typed programming language used both on client- and server-sides. It is lightweight, concise and it allows to make dynamic, interactive webpages and web servers. JavaScript was first introduced in 1995 (Figure 2).

[5]https://kotlinlang.org/
[6]https://www.android.com/
[7]https://www.java.com
[8]https://developer.mozilla.org/en-US/docs/Web/JavaScript



```kotlin
fun main() {
    println(fib(7))
}

fun fib(n: Int): Int {
    if (n <= 1) {
        return n
    }

    var num1: Int = 0
    var num2: Int = 1
    var result: Int = 0

    for (i in 2..n) {
        result = num1 + num2   // calculate sum of two previous numbers
        num1 = num2            // assign second number to the first number
        num2 = result          // assign sum of two previous numbers to the second number
    }

    return result
}
```

Figure 1. Kotlin Fibonacci series example.

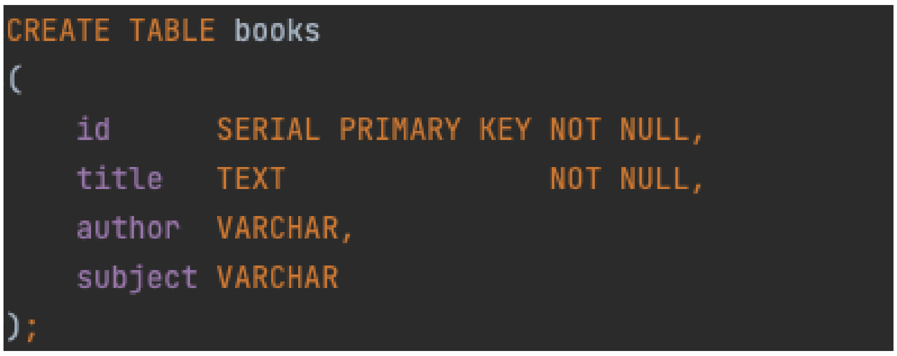


```javascript
// take the input from the user
const number = prompt('Enter the number: ');

const result = Math.sqrt(number);
console.log(`The square root of ${number} is ${result}`);
```

Figure 2. JavaScript Square Root of a Number example.

2.1.3 PostgreSQL

PostgreSQL[9] is an open-source, object-relational database system. It is usually used as a primary database for web applications, mobile and analytics applications. PostgreSQL development started in 1986 and was first released in 1996 by the Berkeley Computer Science Department, University of California (Figure 3).



```
CREATE TABLE books
(
    id      SERIAL PRIMARY KEY NOT NULL,
    title   TEXT               NOT NULL,
    author  VARCHAR,
    subject VARCHAR
);
```

Figure 3. PostgresSQL table creation example.

2.1.4 MiniO Storage

MiniO[10] is an open source, distributed storage server, design for Private Cloud infrastructure providing S3 storage functionality. It allows with minimum effort to set up and store files such as photos, videos, log files, and backups. it is used primarily to save original cloud trace files uploaded from CloudTraceBucket system.

2.2 Frameworks

A framework is a ready-made structure based on which a developer can add his code. The framework defines the structure and rules and provides the necessary set of tools for creating projects [6]. Most often frameworks are used in web development.

2.2.1 Vue.js

Vue.js[11] is a JavaScript open source, framework used for building user interfaces and single-page applications. Vue.js simple to learn, has a decent documentation, customizable, supported by a big community.

[9]https://www.postgresql.org/
[10]https://min.io/
[11]https://vuejs.org/

2.2.2 Spring Boot

Spring Boot[12] is one of the most popular, an open source, enterprise-level Java-based framework used for creating standalone, microservice applications that run without relying on an external web server and run on the Java Virtual Machine.

2.2.3 Express.js

Express.js[13] is a lightweight, JavaScript framework is used for designing and build web applications with minimum effort. It allows to create single-page, multi-page and server-side solutions.

[12]https://spring.io/projects/spring-boot
[13]https://expressjs.com/

2.3 Cloud Trace Providers

Thanks to cloud trace providers, it has become possible to look at the internal structure of providers' systems. It provided valuable information needed for the development of CloudTraceBucket.

All cloud data traces used during the development are publicly available. Most common of the are from Alibaba, Google, TU Delft University, Microsoft Azure.

2.3.1 Microsoft Azure

Microsoft Azure provides cloud traces of its virtual machines [7] and Azure Functions [8]. Virtual machine traces include virtual machine requests with set priority level, the lifetime for each requested virtual machine and normalized resources allocated for each virtual machine. Azure Functions traces contain how many times per minute each function is invoked, how functions are grouped into applications, how applications are grouped by the owner, the distribution of execution times per function and memory usage per application. The traces can be downloaded from Microsoft Azure GitHub[14] repository.

2.3.2 Alibaba

Alibaba provides cloud traces of its FaaSNet [9] system[15].The traces introduce 24-hour-production-level functions invocation logs from two Alibaba datacenters. The traces describe cold start latency such as containers initialization. Traces are available from the Alibaba GitHub[16] repository.

2.3.3 TU Delft Bitbrains

TU Delft University provides cloud traces from Bitbrains distributed datacenter [10]. The dataset contains the performance metrics of 1750 virtual machines and is divided into two traces: fastStorage and Rnd. FastStorage traces include 1250 virtual machines, and Rnd traces include 500 virtual machines. Both traces describe CPU, RAM, hard drive and network performance metrics. Traces are available for downloading from TU Delft University[17] homepage.

[14]https://github.com/Azure/AzurePublicDataset

[15]https://www.usenix.org/conference/atc21/presentation/wang-ao

[16]https://github.com/mason-leap-lab/FaaSNet

[17]http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains

2.3.4 Google Cluster Data

Google provides cloud traces from Borg cluster manager [11] and there is collected cloud traces of about 12.5 thousand machines per cluster. Traces include information about machine jobs, tasks, and system events. Traces are available from Google GitHub repository [18].

[18]https://github.com/google/cluster-data

3 Related works

This section will give a brief overview of some dataset resources created for researchers and how it is related to CloudTraceBucket.

3.1 Kaggle

One of the biggest dataset providers is Kaggle[19]. Kaggle is an online community platform for data science and machine learning researchers. It allows to find, publish datasets and use different data research tools. The Kaggle aims to train and challenge data scientists to solve data science, machine learning and other analytics problems.

3.2 Workflow Trace Archive

The Workflow Trace Archive[20] is a dataset repository, which provides anonymized workload traces from cluster and cloud environments. The aim of the Workflow Trace Archive is to create an open-source workload dataset repository, where datasets can be easily downloaded and standardize open-source datasets to a common format.

3.3 DataDOI

DataDOI[21] is an institutional research data repository hosted by the University of Tartu Library. It gathers research data from different fields such as medicine, chemistry, history, computer science and many more. The aim of the DataDOI is the long-term preservation of research data from various areas, to ensure that research data is findable, accessible, interoperable and reusable [12].

3.4 Yahoo Webscope Program

The Yahoo Webscope Program[22] is a library for scientific datasets available for non-commercial use by academics and scientists. It includes datasets from computing systems, languages, advertising and market and many more. To access the Yahoo Webscope data students or research employees must verify their communion with the university or employer accordingly.

[19]https://www.kaggle.com
[20]https://wta.atlarge-research.com
[21]https://datadoi.ee
[22]https://webscope.sandbox.yahoo.com

3.5 Comparison with CloudTraceBucket

When it comes to comparison with CloudTraceBucket (CTB), the main limitation of existing resources is file uploading. Most resources have an overly complicated process of uploading trace files. For example, in order to upload trace files to the Workflow Trace Archive, the researcher must clone multiple repositories with script files and run these scripts according to the Workflow Trace Archive guidelines. The file uploading into DataDOI has seven intermediate steps before file uploading can be completed. For CTB the researcher should only fill out the form and upload a file.
Even though the existing resources look alike, the CTB's intention is primarily to visualize the cloud trace data to the researcher based on pre-defined query filters. Also, CTB offers to researchers a single-window repository for cloud-only related traces from real production environments. Not to mention, CTB allows researchers to download cloud trace files as soon as files are uploaded to the system.

4 CloudTraceBucket architecture overview

This section gives an overview of the CloudTraceBucket system as a whole, describing the complete processing flow of datasets.

The main idea behind CloudTraceBucket was to develop a generic CSV parser able of handling most of the cloud trace files available without worrying about a manual configuration of file headers for the parser because usually, CSV parsers must know the headers of the file beforehand.

CloudTraceBucket consists of multiple services and sticks to the microservice architecture paradigm. Unlike a monolith, where all functionality is developed in a single application, in a microservice architecture, the application divided into small modules, where each module is responsible only for certain tasks (Figure 4). It makes the application highly maintainable, testable, and independently deployable.

4.1 The dataset flow in CloudTracebucket

Before the file is uploaded, the researcher must fill in a form on the upload page specifying some cloud trace file details (Figure 11), such as cloud trace file provider, file's delimiter and trace type (the trace type is required for a data-collector service to know what generalized table is a target table). As soon as the form is filled, validated, and the file is uploaded, a request with the form and the file is sent to Storage API (Section 5.2). Storage API validates the request to contain CSV file and the form values are not missing. Once validation is done, the file is saved into the MiniO storage server (Section 2.1.4) and the file's metadata is persisted in a database. If the file is new and was not previously uploaded into the CloudTraceBucket system, Storage API creates a new dynamic table from the uploaded CSV file with all its content inserted. Otherwise, Storage API finds the existing table and inserts the uploaded file's content into it.

Once the table is created, Storage API sends a new request to a data-collector service. The request consists of a unique identifier (UUID), the information about a dynamic table and the insert time when the last record was added to the dynamic table. The data-collector service receives the request from Storage API and it starts to process data from the dynamic table. First, it gets information about the dynamic table and finds the last inserted rows in it based on the insert time from the request. Second, it searches for similar columns from the dynamic table and target table (the dynamic table defined from the trace type of the file). When similar columns are found, the data from the dynamic table is converted to data types of the target table columns and converted data is inserted into the target table. After the data-collector service is finished, it sends a response of success to Storage API, and Storage API sends the same response to the researcher.

When the researcher wants to download a cloud trace file (Figure 12), Storage API returns a list of all available files. Once the file is selected, it is downloaded directly from the MiniO Storage server.

When the researcher decides to visualise the data from uploaded data with filters, a request is sent to the Bucket API service (Section 5.4). The Bucket API generates a database query based on the filters set in the request by the user and returns found data accordingly.

4.2 Handling of misconfigured datasets

Quite often happens that data in files come corrupted or misconfigured. Storage API tries to prevent such cases. This subsection will describe Storage API validation rules and handling of misconfigured files.

4.2.1 File validation rules

Storage API has pre-defined validation rules for every file being uploaded:

- `is empty` - validates if the file is empty
- `limit size` - validates that the file's size did not exceed 150 megabytes
- `format` - validates that the incoming file is in a valid CSV format
- `headers` - validates that headers are present on the first row of the file

4.2.2 Handling of misconfigured files

Sometimes CSV file headers contain numeric values, empty column names or special characters. These misconfigurations do not allow to create dynamic tables from CSV because of PostgresSQL limitations.

In Storage API, the file's headers are represented as a list. For empty column names, Storage API replaces missing values with `col_idx`, where `idx` is an index of the element in the headers list. As a result, the column name will look like `col_0`.

For special characters in headers, Storage API applies a regex rule `[^a-zA-Z0-9]`, removes extra spaces at the beginning and the end of the header name. And if the header name consists of multiple words, space characters between words are replaced with an underscore character (_).

If the header name consists only of numeric values (e.g 0.1) then Storage API returns an exception to the user with a message: `File headers have one or more column names consisting of numbers only.`

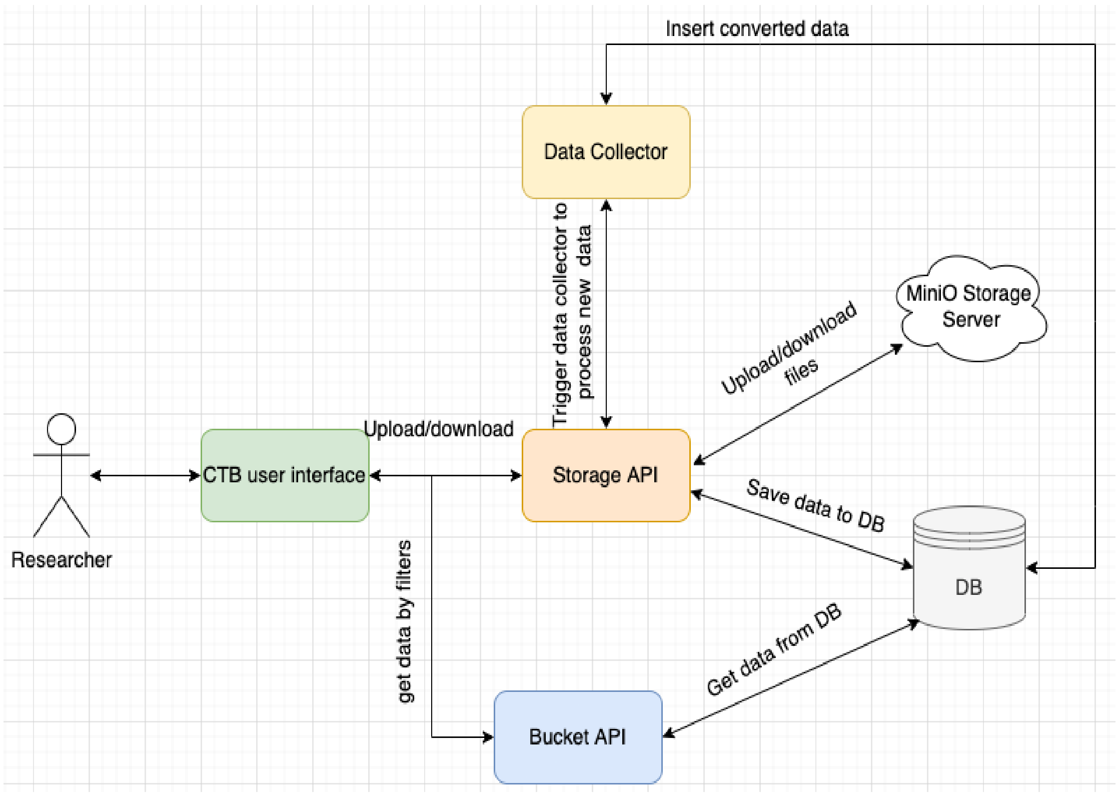


Figure 4. CloudTraceBucket Architecture.

5 CloudTraceBucket in depth

This section describes technical aspects of each microservice and its responsibility in the CloudTraceBucket system.

5.1 Database tables

CloudTraceBucket dynamically creates tables from CSV files, but there will be highlighted the most important tables (Figure 5). The main tables are separated into two groups: intermediate and generalized tables.

5.1.1 Generalized tables

Generalized tables are a result of manual analysis of publicly available cloud traces. The cloud traces provided by Alibaba, TU Delft and Microsoft Azure were grouped by cloud trace type. After the cloud trace groups were formed, cloud trace files were inspected for columns, which have a common column name. All uploaded trace files' data eventually will be inserted into generalized tables:

- `serverless_platform` – a cloud-native development pattern that allows to develop and run applications without setting up development servers locally [13]. It holds information about allocated memory, function invocation count, and the time when the function was invoked
- `cloud_storage` – a cloud computing model that stores and provides access to data or applications online through a cloud provider (e.g. Amazon, Google, Alibaba) and providers manage storage as a service [14]. It holds information about cloud storage read/write operations, what blob type the data is and how many bytes were requested
- `cloud_cluster` – a group of nodes hosted on virtual machines that are connected to a virtual private cloud network [15]. It holds information about nodes of virtual machines, CPU number, disk space, memory requested, what event type is (create, remove)

5.1.2 Intermediate tables

Intermediate tables are required to properly process uploaded files. They give context to the data collector service about what data and what generalized table it should be inserted into.

- `existing_headers` – stores information about dynamically created tables, which is needed for the prevention of creating duplicate tables

- `file_meta` – holds the meta-information of uploaded files
- `column_pointers` – holds the data collector's algorithm results of finding similar columns between dynamic and generalized tables
- `data_collector_logs` – contains inserted rows count of data collector after file processing

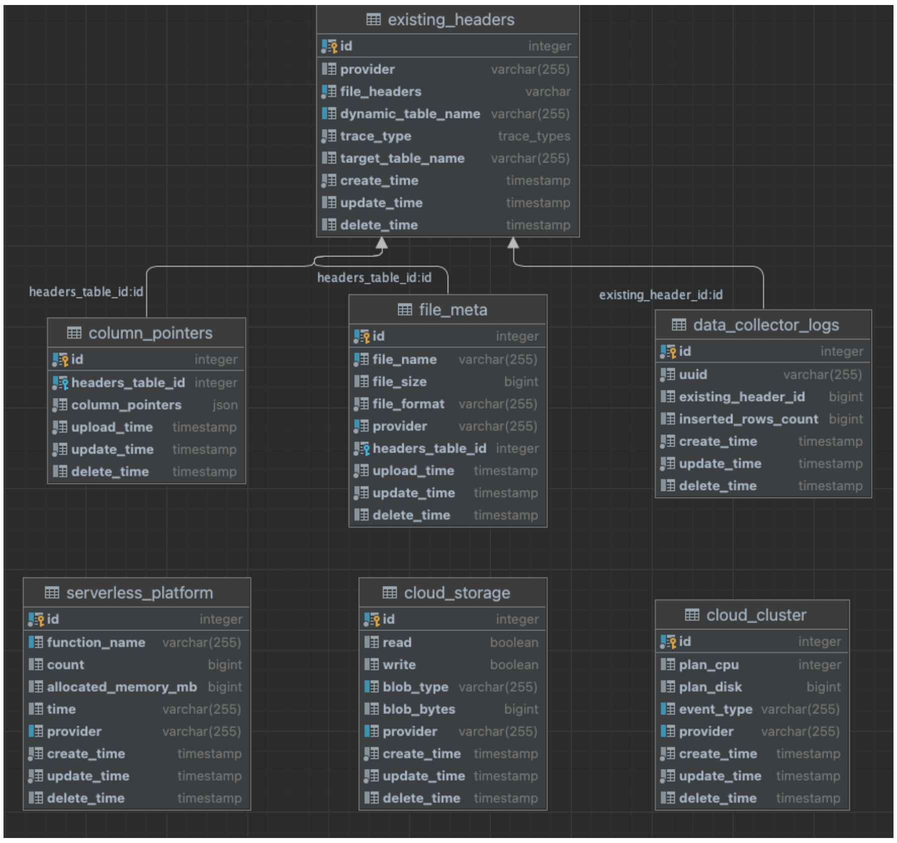


Figure 5. CloudTraceBucket database main tables.

5.2 Storage API

Storage API is a RESTful web service. It is responsible for uploading trace files to file storage, downloading trace files from file storage, creating dynamic tables from uploaded trace files and triggering a data collector service to process newly inserted records from uploaded trace files.

Storage API allows uploading of CSV files only. When the trace file is uploaded by the user, first the file is saved to MiniO storage server. After the file has been uploaded, Storage API takes headers from the file, and checks from the database if a table with such headers exists. If there is no table with such headers, Storage API creates a new table, where the file's headers are presented as table columns. Then the data from the file is inserted into a newly created table. If a table with columns as headers in the database exists, Storage API gets the table name and inserts data from the file into the existing table. When the table was created and data was inserted, Storage API sends a request to the data collector service to process newly inserted data from uploaded trace files. When the user wants to download a file, Storage API validates that the file exists in file storage and returns the file to the user (Figure 6).

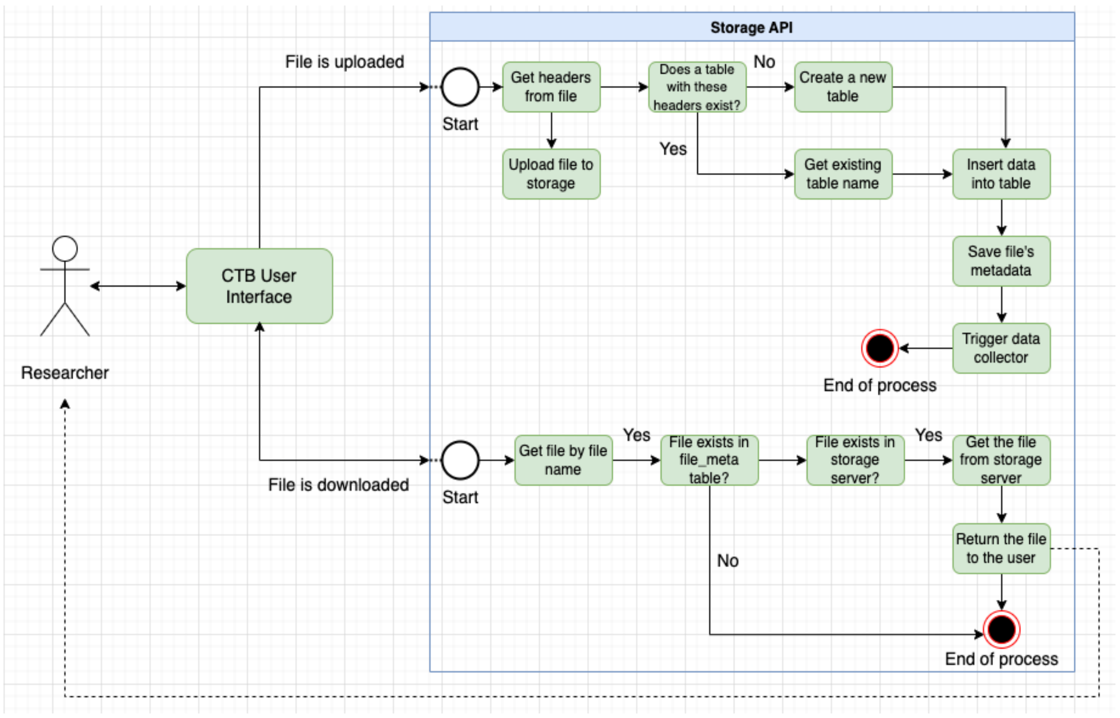


Figure 6. Storage API workflow.

5.3 Data Collector

Data Collector is a RESTful service, responsible for collecting cloud trace data from dynamically created tables by Storage API and inserting the data into generalized tables (Figure 7). Before the data collector inserts the data into the generalized table, it compares column names of the dynamic table to column names of the generalized table using the Jaro-Wrinker string similarity algorithm [16]. Jaro-Wrinker algorithm returns a distance from 0 to 1. If the distance is 1, two strings are identical. If the distance is 0, two strings are different. For example, the distance between "`hello world`" and "`hlelo world`" will be ∼0.98, while for "`cup`" and "`mug`" the distance will be ∼0.55. In case of the data collector, two strings are considered similar if the distance >= 0.9. For optimization purposes, the data collector saves column similarities into the database to reuse them if needed.

When the column similarity check is complete, the data collector inserts data into generalized tables informing the Storage API of a successful insertion.

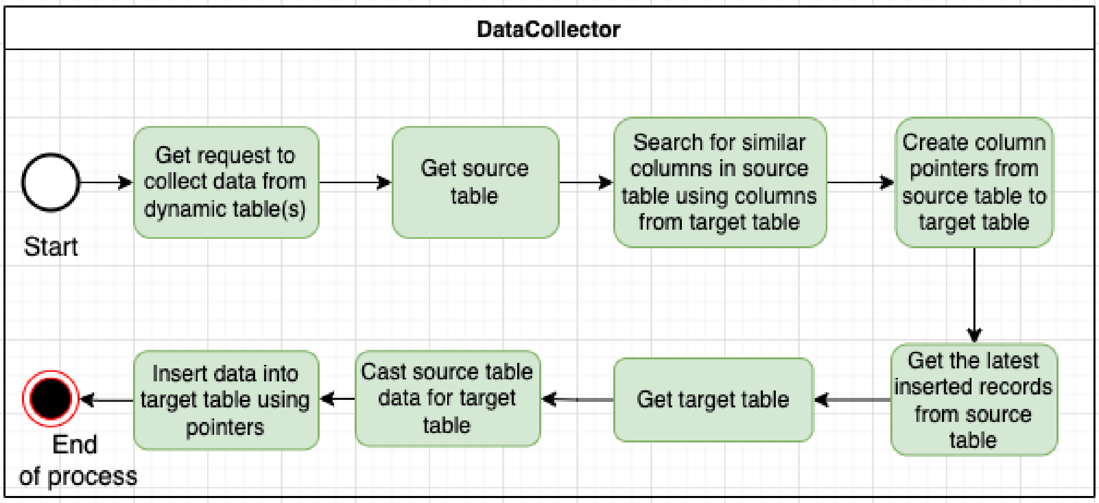

DataCollector
Start
Get request to collect data from dynamic table(s)
Get source table
Search for similar columns in source table using columns from target table
Create column pointers from source table to target table
Get the latest inserted records from source table
Get target table
Cast source table data for target table
Insert data into target table using pointers
End of process

Figure 7. Data Collector workflow.

5.4 Bucket API

Bucket API is a RESTful service, dedicated to getting the cloud trace data from generalized tables based on user-defined filters (Figure 8). Filters will be described for each cloud trace type separately.

serverless_platform can be filtered by:

- `provider`
- `allocated memory`

cloud_storage can be filtered by:

- `provider`
- `blob type`
- `blob bytes`
- `read` operation
- `write` operation

cloud_cluster can be filtered by:

- `provider`
- `plan CPU`
- `plan disk`
- `event type`

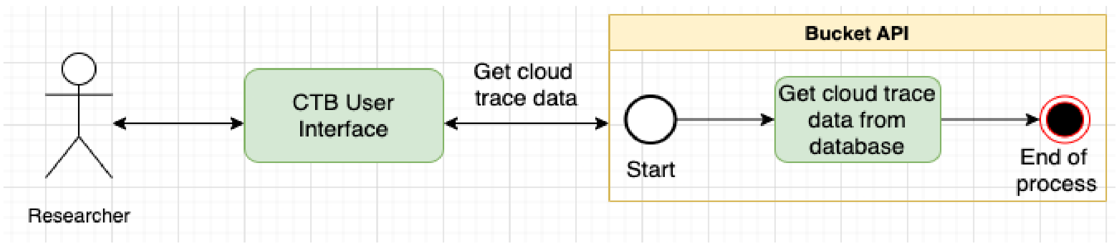


Figure 8. Bucket API workflow.

5.5 CloudTraceBucket User Interface

To interact with CloudTraceBucket there was developed a user interface. The user interface is written using the Vue.js framework.

5.5.1 User interface pages

CloudTraceBucket user interface has four pages:

- Home – an introductory, main page of the website where the visitor can start from (Figure 9)
- About – an informative page describing the definition of the CloudTraceBucket with some additional information (Figure 10)
- Upload Trace Files – a page for uploading cloud trace files into CloudTraceBucket system (Figure 11)
- Download Trace Files – a page for downloading uploaded cloud trace files from CloudTraceBucket system (Figure 12)

5.6 Upload Trace Files overview

The "Upload Trace Files" page is designed for uploading trace data files. Before a file is uploaded, a form must be filled in. The form consists of four fields: provider, trace type, file delimiter and trace file upload button. Below will be a description of each field.

- provider - a mandatory field that specifies who is the provider of the cloud trace file. The field accepts text as an input. There are no limitations what text is written, but the field cannot be empty
- trace type - a mandatory field that specifies the trace type of the cloud trace file. The field is a drop-down list with three values: `SERVERLESS_PLATFORM`, `CLOUD_STORAGE` and `CLOUD_CLUSTER` (Subsection 5.1). At this moment, only three trace types are introduced, because they were described in initial requirements of the thesis. In order to add new trace types to the system, a developer should make code changes in Storage API, Data Collector, Bucket API, user interface and the database
- file delimiter - a mandatory field that specifies the cloud trace file delimiter. By file delimiter, it is possible to separate columns in CSV files. This form field is also a drop-down list and has five values: `COMMA_SEPARATED`, `SEMICOLON_SEPARATED`, `TAB_SEPARATED`, `PIPE_SEPARATED` and `SPACE_SEPARATED`. The form does not set

a default delimiter in case it is not selected, otherwise, the user will be not able to proceed with file uploading. If an incorrect file delimiter is specified it may result in abnormal file processing, for instance, a dynamic table would be created with the wrong number of columns in it

- trace file - a mandatory upload file button. It accepts only files with CSV file extension and maximum allowed size for the file is 150 megabytes

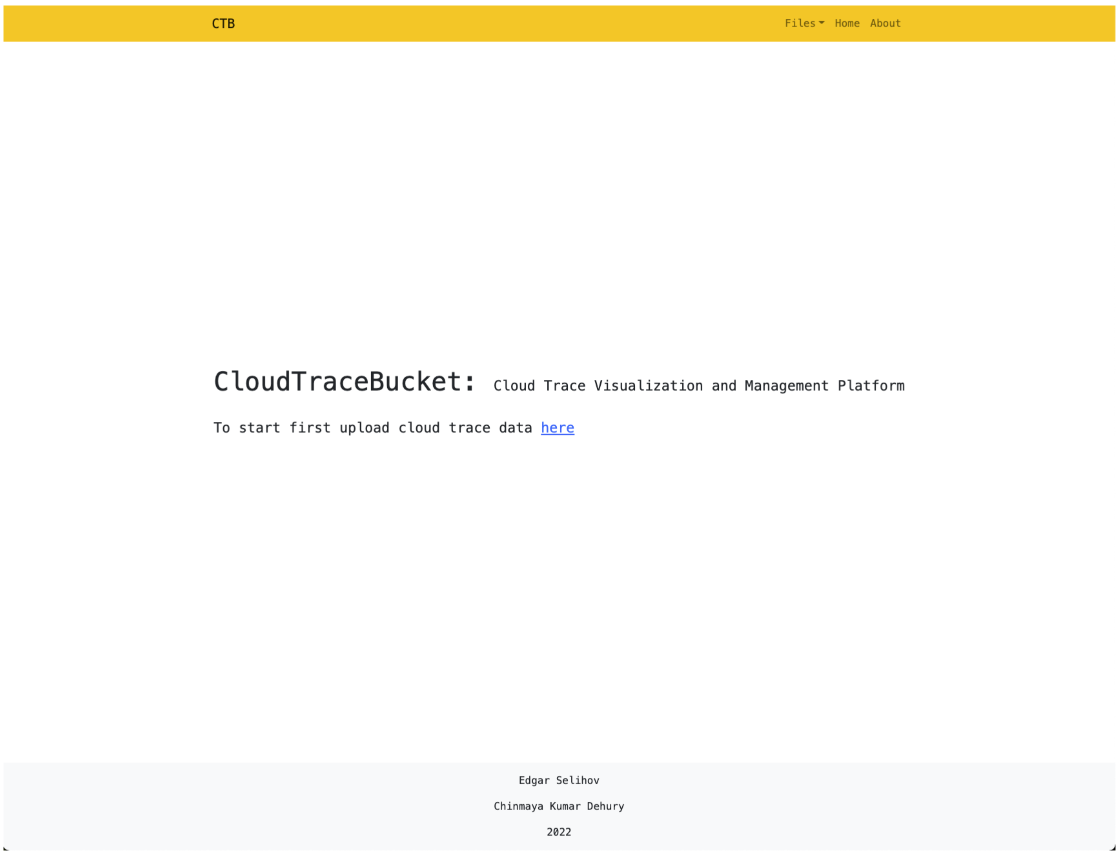
Once all fields in the form are filled in, the "Upload" button will become clickable.

5.7 Download Trace Files page overview

The "Download Trace Files" page allows users to download processed cloud trace files from file storage. All available files are listed in a table with next columns: file name, file format, file size, trace type and upload time:

- `File Name` - displays names of uploaded files. Can be sorted in ascending and descending orders
- `File format` - displays the format of uploaded files
- `Provider` - displays provider name of uploaded files. Can be sorted in ascending and descending orders
- `File size` - displays the size of files in bytes. Can be sorted in ascending and descending orders
- `Trace type` - displays the trace type of uploaded files. Can be sorted in ascending and descending orders
- `Upload Time` - displays when the file was uploaded to file storage. Can be sorted in ascending and descending orders

In the table file names are active links. When user clicks on a file name, it is directly downloaded from file storage.



CloudTraceBucket: Cloud Trace Visualization and Management Platform

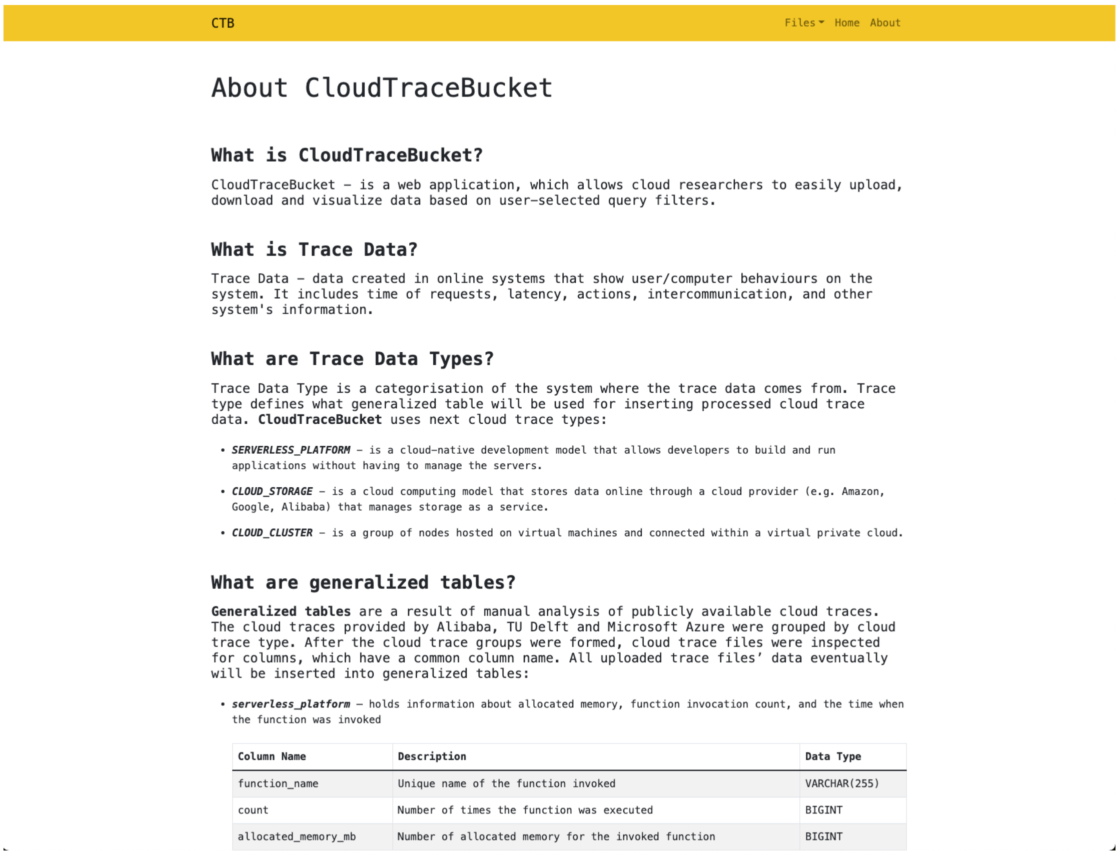
To start first upload cloud trace data here

Edgar Selihov

Chinmaya Kumar Dehury

2022

Figure 9. CloudTraceBucket homepage.



About CloudTraceBucket

What is CloudTraceBucket?

CloudTraceBucket – is a web application, which allows cloud researchers to easily upload, download and visualize data based on user-selected query filters.

What is Trace Data?

Trace Data – data created in online systems that show user/computer behaviours on the system. It includes time of requests, latency, actions, intercommunication, and other system's information.

What are Trace Data Types?

Trace Data Type is a categorisation of the system where the trace data comes from. Trace type defines what generalized table will be used for inserting processed cloud trace data. **CloudTraceBucket** uses next cloud trace types:

- ***SERVERLESS_PLATFORM*** – is a cloud-native development model that allows developers to build and run applications without having to manage the servers.
- ***CLOUD_STORAGE*** – is a cloud computing model that stores data online through a cloud provider (e.g. Amazon, Google, Alibaba) that manages storage as a service.
- ***CLOUD_CLUSTER*** – is a group of nodes hosted on virtual machines and connected within a virtual private cloud.

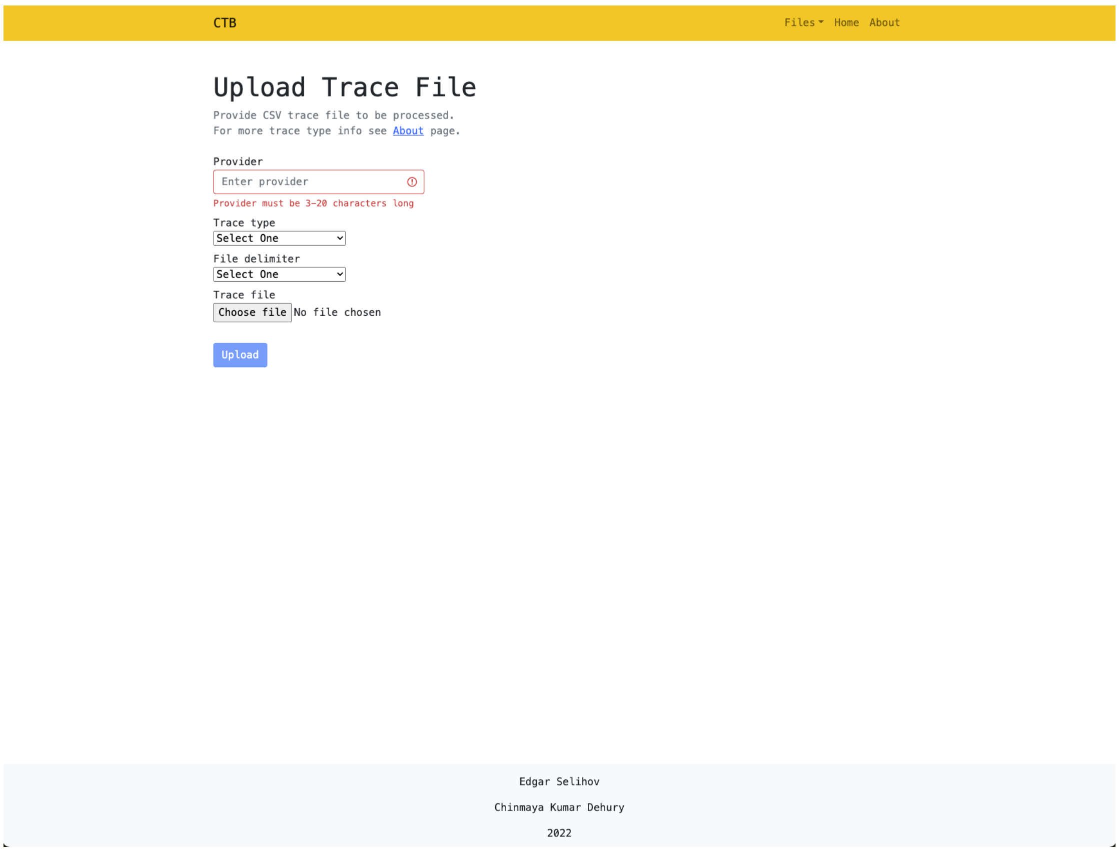
What are generalized tables?

Generalized tables are a result of manual analysis of publicly available cloud traces. The cloud traces provided by Alibaba, TU Delft and Microsoft Azure were grouped by cloud trace type. After the cloud trace groups were formed, cloud trace files were inspected for columns, which have a common column name. All uploaded trace files' data eventually will be inserted into generalized tables:

- ***serverless_platform*** – holds information about allocated memory, function invocation count, and the time when the function was invoked

Column Name	Description	Data Type
function_name	Unique name of the function invoked	VARCHAR(255)
count	Number of times the function was executed	BIGINT
allocated_memory_mb	Number of allocated memory for the invoked function	BIGINT

Figure 10. CloudTraceBucket "About" page.


CTB

Upload Trace File

Provide CSV trace file to be processed.
For more trace type info see About page.

Provider
Enter provider
Provider must be 3-20 characters long

Trace type
Select One

File delimiter
Select One

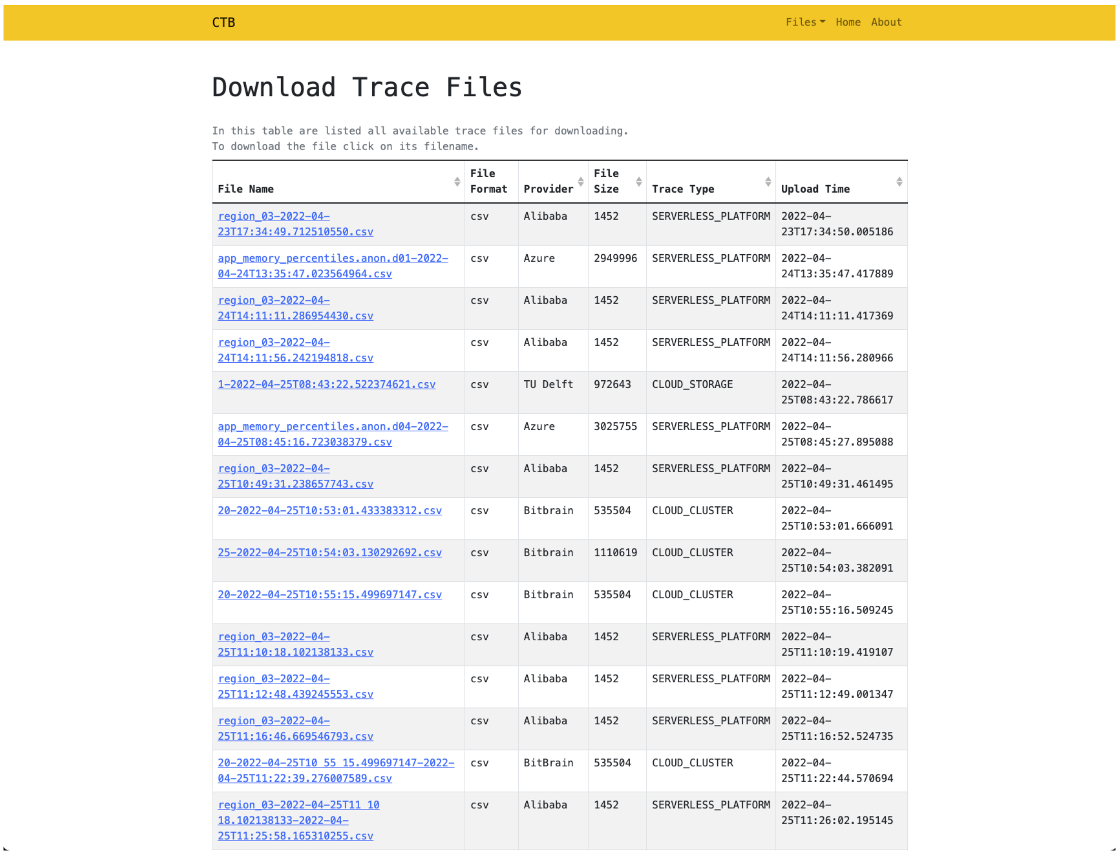
Trace file
Choose file No file chosen

Edgar Selihov

Chinmaya Kumar Dehury

2022

Figure 11. CloudTraceBucket "Upload Trace Files" page.



Download Trace Files

In this table are listed all available trace files for downloading.
To download the file click on its filename.

File Name	File Format	Provider	File Size	Trace Type	Upload Time
region_03-2022-04-23T17:34:49.712510550.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-23T17:34:50.005186
app_memory_percentiles.anon.d01-2022-04-24T13:35:47.023564964.csv	csv	Azure	2949996	SERVERLESS_PLATFORM	2022-04-24T13:35:47.417889
region_03-2022-04-24T14:11:11.286954430.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-24T14:11:11.417369
region_03-2022-04-24T14:11:56.242194818.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-24T14:11:56.280966
1-2022-04-25T08:43:22.522374621.csv	csv	TU Delft	972643	CLOUD_STORAGE	2022-04-25T08:43:22.786617
app_memory_percentiles.anon.d04-2022-04-25T08:45:16.723038379.csv	csv	Azure	3025755	SERVERLESS_PLATFORM	2022-04-25T08:45:27.895088
region_03-2022-04-25T10:49:31.238657743.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-25T10:49:31.461495
20-2022-04-25T10:53:01.433383312.csv	csv	Bitbrain	535504	CLOUD_CLUSTER	2022-04-25T10:53:01.666091
25-2022-04-25T10:54:03.130292692.csv	csv	Bitbrain	1110619	CLOUD_CLUSTER	2022-04-25T10:54:03.382091
20-2022-04-25T10:55:15.499697147.csv	csv	Bitbrain	535504	CLOUD_CLUSTER	2022-04-25T10:55:16.509245
region_03-2022-04-25T11:10:18.102138133.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-25T11:10:19.419107
region_03-2022-04-25T11:12:48.439245553.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-25T11:12:49.001347
region_03-2022-04-25T11:16:46.669546793.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-25T11:16:52.524735
20-2022-04-25T10 55 15.499697147-2022-04-25T11:22:39.276007589.csv	csv	BitBrain	535504	CLOUD_CLUSTER	2022-04-25T11:22:44.570694
region_03-2022-04-25T11 10 18.102138133-2022-04-25T11:25:58.165310255.csv	csv	Alibaba	1452	SERVERLESS_PLATFORM	2022-04-25T11:26:02.195145

Figure 12. CloudTraceBucket "Download Trace Files" page.

6 Future work

As for future work, there is still much room for improvement. First of all, the cloud trace data visualisation is not implemented yet. As all cloud trace providers have different cloud trace files, so there is needed a data visualization tool capable of processing raw data regardless of the file's content. One of such plugins is Grafana[23], but it needs to be investigated more deeply.

Second, API integration can also be added for third-party applications so that other developers could integrate CloudTraceBucket into their solutions.

Third, for better performance and resource management the headers similarity check should be done before the file is saved to MiniO storage server. Otherwise, if the file is structurally valid, but it does not have any similar columns with generalised tables the file is still saved and takes extra space in the storage server, even though the file has no practical use.

Another point is that CloudTraceBucket is only deployed to Tartu University network. In the future, the application is expected to be deployed to a public network so that it could be accessible for everyone.

Finally, new generalised tables can be added. At this moment, CloudTraceBucket supports only three types of cloud traces, but there exist many more of them.

7 Conclusion

This section sums up the thesis and describes possible improvements to the CloudTrace-Bucket system.

This thesis develops a CloudTraceBucket platform for cloud researchers to manage cloud trace data. The thesis introduces cloud computing with cloud trace data, describes the technologies used for development, discusses related works, reviews CloudTrace-Bucket architecture, and describes each microservice in the CloudTraceBucket system in detail.

The results of the thesis are developed multiple independent microservices, working together as a whole system, capable of processing cloud trace data files from different cloud trace providers. Not to mention a lot of new valuable knowledge is acquired about the cloud data processing resources, cloud technologies, and cloud trace data.

[23]https://grafana.com

References

[1] Jake Frankenfield. *Cloud Computing*. July 2020. URL: `https://www.investopedia.com/terms/c/cloud-computing.asp`. Accessed 07.12.2021.

[2] Mansaf Alam, Kashish Ara Shakil, and Shuchi Sethi. “Analysis and clustering of workload in google cluster trace based on resource usage”. In: *2016 IEEE Intl conference on computational science and engineering (CSE) and IEEE Intl conference on embedded and ubiquitous computing (EUC) and 15th Intl symposium on distributed computing and applications for business engineering (DCABES)*. IEEE. 2016, pp. 740–747. Accessed 07.12.2021.

[3] Charles Reiss, John Wilkes, and Joseph L Hellerstein. “Google cluster-usage traces: format+ schema”. In: *Google Inc., White Paper* 1 (2011). Accessed 08.12.2021.

[4] Magda Miu. *High performance with idiomatic Kotlin*. Feb. 2022. URL: `https://magdamiu.medium.com/high-performance-with-idiomatic-kotlin-d52e099d0df0`. Accessed 03.05.2022.

[5] Anshul Bansal. *Java vs. Kotlin*. Nov. 2021. URL: `https://www.baeldung.com/kotlin/java-vs-kotlin`. Accessed 03.05.2022.

[6] *What Is a Framework?* Sept. 2021. URL: `https://www.codecademy.com/resources/blog/what-is-a-framework`. Accessed 04.05.2022.

[7] Ori Hadary et al. “Protean:{VM} allocation service at scale”. In: *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. 2020, pp. 845–861. Accessed 29.04.2022.

[8] Mohammad Shahrad et al. “Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider”. In: *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. 2020, pp. 205–218. Accessed 29.04.2022.

[9] Ao Wang et al. “FaaSNet: Scalable and Fast Provisioning of Custom Serverless Container Runtimes at Alibaba Cloud Function Compute”. In: *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, July 2021, pp. 443–457. ISBN: 978-1-939133-23-6. URL: `https://www.usenix.org/conference/atc21/presentation/wang-ao`. Accessed 29.04.2022.

[10] Siqi Shen, Vincent Van Beek, and Alexandru Iosup. “Statistical characterization of business-critical workloads hosted in cloud datacenters”. In: *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE. 2015, pp. 465–474. Accessed 29.04.2022.

[11] Abhishek Verma et al. “Large-scale cluster management at Google with Borg”. In: *Proceedings of the European Conference on Computer Systems (EuroSys)*. Bordeaux, France, 2015. Accessed 29.04.2022.

[12] Laurens Versluis et al. “The Workflow Trace Archive: Open-Access Data From Public and Private Computing Infrastructures”. In: *IEEE Trans. Parallel Distributed Syst.* 31.9 (2020), pp. 2170–2184. DOI: `10.1109/TPDS.2020.2984821`. URL: `https://doi.org/10.1109/TPDS.2020.2984821`. Accessed 01.05.2022.

[13] *What is serverless?* Oct. 2017. URL: `https://www.redhat.com/en/topics/cloud-native-apps/what-is-serverless`. Accessed 06.05.2022.

[14] *What is Cloud Storage?* URL: `https://aws.amazon.com/what-is-cloud-storage`. Accessed 06.05.2022.

[15] Aaron Nordhoff. *What is a Cluster? An Overview of Clustering in the Cloud.* July 2020. URL: `https://www.capitalone.com/tech/cloud/what-is-a-cluster/`. Accessed 06.05.2022.

[16] *Jaro and Jaro-Winkler similarity*. Feb. 2022. URL: `https://www.geeksforgeeks.org/jaro-and-jaro-winkler-similarity/`. Accessed 05.03.2022.

Appendix

I. Glossary

1. Cloud technologies - IT-technologies, which allow to store and process information on remote servers[24]

2. HTTP (Hypertext Transfer Protocol) - a core format for structuring web requests for proper communication between a client and a server[25]

3. Hashing - a technique of transforming a given key into another value using mathematical algorithms[26]

4. OOP (Object-oriented programming) - a programming paradigm that uses classes and objects. it helps to structure a software program into reusable, simple and maintainable pieces of code, which are used for object instance creation[27]

5. JVM (Java Virtual Machine) - a runtime engine of Java Platform which compiles Java code into a bytecode allowing to run a compiled program on any machine that has a native JVM support[28]

6. S3 Storage - an object storage service that stores objects in buckets where the object is a file and the bucket is a container for objects[29]

7. Trace type - a categorisation of the system where the trace data comes from

8. Blob - a file-like object that represents an immutable raw data. It can be in a format of a text or binary data[30]

9. REST (REpresentational State Transfer) - an architectural approach for establishing web-based standards that make it easier for computer systems to connect with one another[31]

10. API (Application Programming Interface) - a intermediate software that allows two applications to communicate with each other

[24]https://dynamixsolutions.com/what-is-cloud-technology-and-how-does-it-work
[25]https://www.cloudflare.com/en-gb/learning/ddos/glossary/hypertext-transfer-protocol-http
[26]https://www.educative.io/edpresso/what-is-hashing
[27]https://www.educative.io/blog/object-oriented-programming
[28]https://www.pcmag.com/encyclopedia/term/java-virtual-machine
[29]https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
[30]https://developer.mozilla.org/en-US/docs/Web/API/Blob
[31]https://www.codecademy.com/article/what-is-rest

II. Repository

This Appendix contains a link to GitHub repository of CloudTraceBucket.

Source code of CloudTraceBucket can be accessed from here: `https://github.com/chinmaya-dehury/CloudTraces`

III. Access to CloudTraceBucket

To access CloudTraceBucket the user needs to be connected to UT network either directly via eduroam or OpenVPN[32]. After the user successfully connected, CloudTraceBucket can be accessed via `http://172.17.91.248:8080`.

[32]https://wiki.ut.ee/pages/viewpage.action?pageId=17105590

IV. Licence

Non-exclusive licence to reproduce thesis and make thesis public