CS CAPSTONE TECHNOLOGY REVIEW

NOVEMBER 21, 2017

PRIVACY PRESERVING CLOUD, EMAIL, AND PASSWORD SYSTEMS

PREPARED FOR

OSU

ATTILA YAVUZ

PREPARED BY

GROUP 38

THE SECRET BUNNY TEAM

SCOTT RUSSELL

Abstract

This document reviews technologies and research for three specific items. Email Protocol, Benchmark Comparison, and Hosting Server Cloud. This document reviews ways to implement each item through three specific methods, compares their strengths and weaknesses in the context of this specific project and chooses a choice based on criteria.

CONTENTS

1 Introduction

The purpose of this document is to outline the technical choices for our project, Privacy Preserving Cloud Encryption. The goal of the project is to implement a certain searchable encryption algorithm and investigate how it can be integrated with common internet applications.

2 Email Protocol

2.1 Overview

Email protocols are a standard method used at each end of a communication channel and provide transmission between two distinct sources. This paper will be exploring three of the primary protocols used today, their strengths and weaknesses and finally my choice for a protocol based on what we believe is the most effective specifically for our implementation.

2.2 Criteria

- Security:
  Being a project with a focus on the need for privacy having a secure protocol is paramount to implementation. If we cannot have reliable and secure transfer of data to and from the database than it doesnt matter how secure the system is to begin with. The price of using each cloud hosting service is a very important to usability. We will be working with very large data sets and looking at price per storage and time will drastically affect the usability of each Hosting service.
- Speed:
  The primary concern of DSSE Algorithms is not the security of the system. Security has been proven. It is the speed of search, update and delete functionalities of the database. For this reason, having a protocol with a fast transfer rate will be ideal for our implementation. Functionality: Unlike most Email Protocols that require synching across multiple devices our project works with a single client and server. Thus, functionality of the Email Protocol we select does not have to be complex.

2.3 Potential choices

2.3.1 POP3 (Post Office Protocol v3)

The First protocol is POP3 (Post Office Protocol v3). POP3 is excellent at downloading data to a single client device. POP3 doesnt have to sort through a hierarchy of folders that IMAP must go through. For this reason, POP3 is a much simpler Email Protocol. The security of POP3 allows for the use of TLS to prevent Man in the Middle Attacks. With POP3 receiving messages from the server the user will also need to be able to send data back to the servers to update files. The downside of POP3 is also its biggest strength. That of its simplicity. POP3 does not synchronize across different devices. For example, you read and delete an email on your phone. Now when you get home your personal computer will still have this email. This does not apply to our case as we will only be running POP3 from a single server to a single client. POP3 also allows to access your locally downloaded files without having a connection to the server.

2.3.2 IMAP (Internet message Access Protocol)

IMAP is good at synchronizing two or more applications. lets say youre trying to read the same messages across these multiple devices. In addition, IMAP allows a hierarchy of messages. Lets say you want to separate business, personal, and folders arranged in a hierarchy to pull and sort this information. If you move from one folder to another it will concurrently change all instances of the folder across devices. For this reason, IMAP is very good when it comes to multiple devices accessing the same email server. Automatically synchronize that messages have been read. IMAP also incorporates. IMAP is a more complex protocol but because of this it requires more space and CPU usage than POP3. The security of IMAP and POP3 are both suitable for our implementation.

2.3.3 MAPI (Messaging Applicaion Programming Interface)

MAPI (Messaging Application Programming Interface) is the final Protocol I will be looking at. MAPI is usually implemented to provide communication with Microsoft Exchange servers. MAPI was developed by Microsoft to have the same functionality as IMAP. MAPI allows for messages from the cloud to be stored on a local .PST file. This can be used to create backups for critical information in case the server you are connected too losses connection, allowing for offline viewing of saved messages. Overall MAPI is a product provided to facility Microsoft Office email work and has similar functionality to that of IMAP.

2.4 Discussion

Looking at the criteria for implementation we know that all three choices have a strong security foundation as an Email Protocol. Thus, when choosing a product to use speed will be of paramount importance for our primary goal of this project is to compare algorithm functionality speeds.

2.5 Conclusion

After analyzing the pros and cons of IMAP, POP3 and MAPI I have decided that our Email Protocol will be POP3. Since our data will only be stored on a single device the synchronizing tools that IMAP and MAPI preform are not needed in our implementation. Also since POP3 does note has to synchronously sort headers it will run faster than IMAP in the general case. Primarily because of this speed through simplicity we will be using POP3 for server to client download of information.

3 BENCHMARKING

3.1 Overview

The primary purpose of this project is to text the practicality and efficiency of David Cashs Algorithm. To test results, the runtime speeds of our C++ Implementation of David Cash will be compared to that of other algorithms, specifically the IM-DSSE implementation of a Bit Matrix Algorithm composed by Atilla and Thang using C++ and a Java implementation of David Cashs algorithm. (Clusion) This section will not specifically be comparing and selecting a single way to benchmark our data but explore and explain the many ways we will be comparing our implementation to that of the C++ Bit Matrix and the Java implementation of David Cash.

3.2 Comparison Algorithms

3.2.1 DSSE Bit Matrix

The first implementation explored to compare to David Cashs Algorithm is that of Atilla Yavuzs DSSE (Dynamic Searchable Symmetric Encryption) Bit Matrix Algorithm. As the field of Dynamic Searchable Encryption is a very narrow scope of study it is important for research to be done to not only understand the asymptotical runtime of algorithms but also how these algorithms preform practically and in comparison, to similar ones. We choose Atillas Bit Matrix Algorithm as our first comparison as it will be programmed in the same language as our implementation (C++) but using a different data structure, that of a bit matrix.

3.2.2 David Cash Java Clusion

The second implementation of a DSSE algorithm is that of Clusion Library for Searchable Symmetric Encryption (SSE). This implementation is created using Java using David Cashs Algorithm. In this way we can get a comparison of another implementation of the same algorithm, but using a different language than the one we will be implementing. Through both benchmarks, The C++ Bit Matrix and the Java David Cash algorithm we will be able to develop benchmark testing on runtime speeds and efficiency.

3.3 Primary Benchmark Variables:

3.3.1 Round Trip Delay

Round Trip Delay is the time it takes a packet to transmit across a network from the client to the server and back to the client. It is common to misinterperet with End-to-End delay, which is only acess from the source to the destination (client to server) and not a full round trip. We are using Round Trip Delay in place of End-To-End Delay because of the ability to test both the sending and recieving transmission speed with time lapses. This is important not only to test our implimentation of POP3 but also the connection we will be using between the client and the cloud server API.

3.3.2 Data Size Comparison

When analyzing runtime speed of algorithms we need to test with large databases. Big O() complexity may asymptotically say a specific algorithm is faster than another but in practical implimentation things may be different because of unforseen factors. In the case of comparing large data sets this normalizes the speeds relative to the algorithms themselves rather than being on the basis of outside factors such as transmission speed or computer processing power alone.

3.4 Conclusion

All of these Benchmark types are very important in creating a overview and comparison between these different algorithms. With these three as the primary benchmark variables we can analyze the strengths and weaknesses between the Java implimentation of David Cash, the C++ implimentation using a Bit Matrix and our implimentation of David Cash's Algorithm using C++.

4 Hosting Server Cloud

4.1 Overview

In addition to the Protocol we are using to connect Client to Server we also have done research on what specific Cloud Server Platform we want to be using. The three primary platforms that We will be exploring include Amazon EC2, Google Compute Engine and Microsoft Azures Virtual Machine. Amazon, Google, and Microsoft are all big players in the Cloud Computing business and all provide strong options for storage.

4.2 Criteria

- Price:
  The price of using each cloud hosting service is a very important to usability. We will be working with very large data sets and looking at price per storage and time will drastically affect the usability of each Hosting service.
- Functionality:
  The Cloud Server that we select must be able to handle large data sets, have stability from crashes and downtime as well as be able to backup and store data in the event of a failure on the cloud service.

4.3 Potential choices

4.3.1 Amazon Elastic Compute Cloud (EC2)

The first server we considered to incorporate is Amazon Elastic Compute Cloud (EC2). In the cloud computing market Amazon Web Services (AWS) is the biggest player in Cloud Server databases with 30 percent of the current market. All three of these servers provide CDN, Direct Connection, and DNS network features. Memory for EC2 limits up to 244 GB of storage, with varying price tables based on amount of storage allotted. EC2 provides up to 48 TB of temporary storage across multiple Disks. EC2 provides snapshot backups of data. Also EC2 servers provide file-backup in the case of last data. EC2 provides 750 Hours per month of free instance usage. Overall EC2 provides a large scalability of size based on needs of the server and will provide more than enough memory and storage for our implementation. EC2 provides reliable, flexible and secure server storage. When talking about Azure VM and Google CE I will discuss the differences between those and that of EC2.

4.3.2 Google Compute Engine (CE)

The next server I will be talking about is that of Googles Compute Engine. (CE) Google CE comes with persistent disk storage, and come in flexible sizes and packages. Users can choose to deploy servers to specific regions and zones based on location. Network access to the cloud provides up to five networks per project by default. CE provides Persistent Disk Snapshots to allow for creating backups in the case of failures or maintenance. In comparison to EC2 both provide similar power, performance, backup, and data abilities well within what we would be using for our implementation. CE provides 12 months of usage with 300 dollars of free credit towards product and storage costs.

4.3.3 Microsoft Azure Virtual Machine (VM)

Finally I will be discussing the key features of Microsoft Azure Virtual Machine (VM). Azure VM is an on-demand, scalable computing resource like Google CE and Amazon EC2. It also can dynamically scale in size, limits, and storage space based on needs. For our implementation all three of these servers have similar and adequate speeds, storage, and security for our use case. Azure provides up to 1 GB of ram and 1 GB of storage with their free instance with up to 10 concurrent API apps.

4.4 Discussion

Because of this I understood that the primary test case for choosing a sefrver would be cost oriented. Using an On-Demand price structure Amazon EC2 provides a free 12-month trial of cloud storage with 1 GB of memory and 20 GB of data storage with backup included. Google CE provides 30 GB of persistent disk storage per month for free, depending on implementation benchmark testing this may be enough to have a good estimate of speed comparisons, however CE does have a limit of 300 dollar credit, this may not last for the entire duration of the project implementation. Azure provides 1 GB of storage with unlimited use up to 165 MB outbound network traffic.

4.5 Conclusion

From discussion on pricing and usability we decided to choose Amazon EC2 for our implementation. With a free unlimited 12-month trial including 20 GB of backup storage we concluded that it would be better to maintain a free standing rather than risking using our of our 300 dollar credit that Google CE would of provided. Functionally all three systems provided cloud storage backup for free.