**SECURING INTERCLOUD COMMUNICATIONS: A PRIVACY-PRESERVING PROTOCOL FOR DISTRIBUTED TRUST EVALUATION**

**ABSTRACT**

For managing Intercloud, a distributed trust evaluation technique with privacy protection is presented. The parties involved in this include the cloud service provider (CSP), the user/rater, the inquirer, the distributed feedback storage (FBS), and the secret sharing network. The CSPs collaborate to deliver cloud services to consumers and to one another under the Intercloud system concept. Users of cloud services may be divided into two categories: consumers and businesses. Users of cloud services are consumers. They only have a one-way trust/service relationship with a CSP, meaning that the service is provided by CSPs to consumers. Service exchange between business users is possible (i.e., there is a two-way trust/service connection). Users may score a cloud service or a trust attribute (such availability, response speed, pricing, or technical support) after using it or under specific conditions. As a result, users who provide feedback or rate items are also doing so. You should be aware that we just evaluate one service or trust attribute in order to make the explanation clear. To assess several services or trust traits, it is simple to expand. The Intercloud exchanges or the cloud service providers themselves may implement the FBS via a blockchain-based mechanism. You might also use other distributed storage systems that provide integrity guarantees to store comments. Furthermore, we suppose that encrypted feedback is protected by a covert sharing network. Later, more thorough operations will be explained. To safeguard the anonymity of their ratings, users and raters might create a secret sharing network. An alternative is for a third party to provide it as a service. There is no direct communication between raters throughout the dispersed operation of the secret sharing system. Each rater just needs to respond to the enquirer by decrypting a specific secret published/provided by the FBS over a potentially anonymous secret sharing address (SSA) (such as a server or Intercloud gateway).

1. **INTRODUCTION**
   1. **OVERVIEW OF THE PROJECT**

The project implemented for trust prediction in cloud environment. One of the most important factors for the adoption and success of cloud computing is trust. Trust evaluation is a fundamental component for secure computing in the cloud environment. In this paper a new trust evaluation method has been proposed. In this scheme, each new customer of a cloud service can determine its trust level using the past experiences of previous customers of this cloud service. The proposed method introduces feedback analysis method that helps distinguish between malicious and reliable customer feedbacks. The proposed scheme assigns a reliability weight to each customer feedback. Users who have shared any cloud service before and did rate it similarly are more likely to have the same opinion of a new service. Hence, their evaluation of a new service should be given a higher weight. In cases where no customer has shared a common service with us before, then existing customers who have had a feedback closer to the majority (in any service) may seem to be more reliable and hence their opinion is given a higher weight. The balance between the feedback of customers with shared service and those of majority consensus in the final decision is determined dynamically and based on the rate of malicious attacks. This way the method can respond better to changes in the rate and sources of malicious attacks. The proposed method has been compared with an existing trust evaluation method and proved its superiority in minimizing the effect of malicious feedbacks, and having a faster response time.

* 1. **PROBLEM STATEMENT**

Cloud services are cloud computing resources and applications deployed on the Internet or cloud computing platform, and users can access the required cloud services at any time. However, users face the diversity and complexity of quality of service (QoS) when evaluating and selecting cloud services. Therefore, it is important to study and establish an effective and objective trust model to improve user satisfaction and interaction success rate. In order to ensure the accuracy of direct trust, here proposes a trust evaluation approach based user’s feedback. Feedbacks are collected and analysed to recommend the best cloud services to end users. A recommender system would provide a list of recommended items to the customer based on their individual needs and Preferences. Most of the recommendation system utilizes collaborative filtering. It is a technique used for calculating a prediction for the customers and it is based on the concept that the user gets the perfect recommendation from another user with similar preference. The proposed work has two users namely admin and user. Admin could create the framework which is designed to provide recommendation based cloud service selection process. Users should enroll and access cloud service, then provide the feedback about the service accessing from cloud environment. Registered users are only allowed to post feedbacks. Then recommendation will be performed to shows the best service provider to the user.

1. **SYSTEM ANALYSIS**
   1. **EXISTING SYSTEM**

Cloud service users’ feedback is a good source to assess the overall trustworthiness of cloud services. However, malicious users may collaborate together to disadvantage a cloud service by giving multiple misleading trust feedbacks (i.e., collusion attacks) or trick users into trusting cloud services that are not trustworthy by creating several accounts and giving misleading trust feedbacks (i.e., Sybil attacks). In proposed work have presented a novel technique that help in detecting reputation based attacks and allowing users to effectively identify trustworthy cloud services. Introduce the concept of feedback density to support the determination of credible trust feedbacks. Specifically, we consider the total number of users who give trust feedbacks to a particular cloud service as the feedback mass, the total number of trust feedbacks given to the cloud service as the feedback volume. The feedback volume is influenced by the feedback volume collusion factor which is controlled by a specified volume collusion threshold. This factor regulates the multiple trust feedbacks extent that could collude the overall trusted feedback volume

**2.1.1. DISADVANTAGES OF EXISTING SYSTEM**

* Feedback information cannot be made private.
* Cloud service provider can easily remove negative comments about its services.
* Malicious users can generate a large volume of misleading feedback
* It supports only text feedback process.
  1. **PROPOSED SYSTEM**

Proposed method introduces a trust evaluation method that is used by the trust broker to evaluate the reliability of each customer feedback. In the proposed method, we focus on the reliability of the feedback of the middle customer ‘B’. In the first step, we evaluate the reliability of a middle customer ‘B’ by comparing its feedbacks for commonly rated services. Each customer feedback is weighted based on a newly introduced similarity measure. In the second step, we consider all the feedbacks that customer ‘B’ has rated and compare them with the majority of feedbacks.

In this scheme, each new customer of a cloud service can determine its trust level using the past experiences of previous customers of this cloud service. The proposed method introduces an adaptive method that helps distinguish between malicious and reliable customer feedbacks. The proposed scheme assigns a reliability weight to each customer feedback. Users who have shared any cloud service before and did rate it similarly are more likely to have the same opinion of a new service. Hence, their evaluation of a new service should be given a higher weight. In cases where no customer has shared a common service with us before, then existing customers who have had a feedback closer to the majority (in any service) may seem to be more reliable and hence their opinion is given a higher weight. The balance between the feedback of customers with shared service and those of majority consensus in the final decision is determined dynamically and based on the rate of malicious attacks.

**2.2.1. ADVANTAGES OF PROPOSED SYSTEM**

* It allows the filtering of extreme ratings without leaking privacy information.
* Feedbacks are encrypted and stored securely.
* Fake reviews are avoided efficiently

1. **SYSTEM SPECIFICATION**
   1. **HARDWARE REQUIREMENTS**

* Processor : Dual core processor 2.6.0 GHZ
* RAM : 4GB
* Hard disk : 320 GB
* Compact Disk : 650 Mb
* Keyboard : Standard keyboard
  1. **SOFTWARE REQUIREMENTS**
* Operating system : Windows OS
* Front End : PHP
* Back end : MYSQL Server
* Tool : Macromedia Dreamweaver 8

1. **SOFTWARE DESCRIPTION**
   1. **FRONT END**

PHP is an open-source server-side scripting language that many devs use for web development. It is also a general-purpose language that you can use to make lots of projects, including Graphical User Interfaces (GUIs). In this article, I will help you explore the world of PHP so you can learn how it works and its basic features. By the end, you will be able to write your first Hello World program in PHP. The term PHP is an acronym for PHP: Hypertext Preprocessor. PHP is a server-side scripting language designed specifically for web development. It is open-source which means it is free to download and use. It is very simple to learn and use. The files have the extension “.php”.

Rasmus Lerdorf inspired the first version of PHP and participated in the later versions. It is an interpreted language and it does not require a compiler.

* PHP code is executed in the server.
* It can be integrated with many databases such as Oracle, Microsoft SQL Server, MySQL, PostgreSQL, Sybase, and Informix.
* It is powerful to hold a content management system like WordPress and can be used to control user access.
* It supports main protocols like HTTP Basic, HTTP Digest, IMAP, FTP, and others.
* Websites like www.facebook.com and www.yahoo.com are also built on PHP.
* One of the main reasons behind this is that PHP can be easily embedded in HTML files and HTML codes can also be written in a PHP file.
* The thing that differentiates PHP from the client-side language like HTML is, that PHP codes are executed on the server whereas HTML codes are directly rendered on the browser. PHP codes are first executed on the server and then the result is returned to the browser.
* The only information that the client or browser knows is the result returned after executing the PHP script on the server and not the actual PHP codes present in the PHP file. Also, PHP files can support other client-side scripting languages like CSS and JavaScript.

**Characteristics of PHP**

* Simple and fast
* Efficient
* Secured
* Flexible
* Cross-platform, it works with major operating systems like Windows, Linux, and macOS.
* Open Source
* Powerful Library Support
* Database Connectivity

**Advantages of PHP**

PHP has some advantages that have made it so popular, and it's been the go-to language for web servers for more than 15 years now. Here are some of PHP's benefits:

* Cross-Platform: PHP is platform-independent. You don't have to have a particular OS to use it because it runs on every platform, whether it's Mac, Windows, or Linux.
* Open Source: PHP is open source. The original code is made available to everyone who wants to build upon it. This is one of the reasons why one of its frameworks, Laravel, is so popular.
* Easy to learn: PHP is not hard to learn for absolute beginners. You can pick it up pretty if you already have programming knowledge.
* PHP syncs with all Databases: You can easily connect PHP to all Databases, relational and non-relational. So it can connect in no time to MySQL, Postgress, MongoDB, or any other database.
* Supportive Community: PHP has a very supportive online community. The official documentation provides guides on how to use the features and you can easily get your problem fixed while stuck.

**Syntax of PHP**

The structure which defines PHP computer language is called PHP syntax. The PHP script is executed on the server and the HTML result is sent to the browser. It can normally have HTML and PHP tags. PHP or Hypertext Preprocessor is a widely used open-source general-purpose scripting language and can be embedded with HTML. PHP files are saved with the “.php” extension. PHP scripts can be written anywhere in the document within PHP tags along with normal HTML.

* You can embed PHP code anywhere in a document. It starts with an opening tag of <?php (the PHP code goes in here) and ends with a closing tag ?>.
* All PHP statements end with a semicolon ;. A PHP file is always named with the file extension of .php – for example, index.php or home.php.

**Data Types in PHP**

Variables in PHP store values of different data types. Now let's discuss some data types that work with PHP:

* String
* Integer
* Float
* Boolean

**String data type**

A string is a data type which is represented with some text inside double quotes " ". A string

**Integer data type**

Integers are whole numbers that have no decimal point. Integers can either be negative numbers (-34567) or positive numbers (34567).

**Float data type**

Floats are not whole numbers, but rather they are numbers with decimal points. Floats can also be negative decimal numbers (-34.567) or positive decimal numbers (34.567).

**Boolean data type**

Boolean is a data type that represents two possible outcome, true or false. Booleans are used mostly when we are working with conditional statements like if, else, elseif, and swtich.

**Function of PHP**

PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value. You already have seen many functions like fopen() and fread() etc. They are built-in functions but PHP gives you option to create your own functions as well.

There are two parts which should be clear to you −

* Creating a PHP Function
* Calling a PHP Function

In fact you hardly need to create your own PHP function because there are already more than 1000 of built-in library functions created for different area and you just need to call them according to your requirement.

Please refer to PHP Function Reference for a complete set of useful functions.

**Creating a Function**

While creating a user defined function we need to keep few things in mind:

* Any name ending with an open and closed parenthesis is a function.
* A function name always begins with the keyword function.
* To call a function we just need to write its name followed by the parenthesis
* A function name cannot start with a number. It can start with an alphabet or underscore.
* A function name is not case-sensitive.

**Reusability**: If we have a common code that we would like to use at various parts of a program, we can simply contain it within a function and call it whenever required. This reduces the time and effort of repetition of a single code. This can be done both within a program and also by importing the PHP file, containing the function, in some other program

**Easier error detection**: Since, our code is divided into functions, we can easily detect in which function, the error could lie and fix them fast and easily.

**Easily maintained**: As we have used functions in our program, so if anything or any line of code needs to be changed, we can easily change it inside the function and the change will be reflected everywhere, where the function is called. Hence, easy to maintain.

**Creating PHP Function**

It’s very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called write Message () and then calls it just after creating it.

**PHP Functions with Parameters**

PHP gives you option to pass your parameters inside a function. You can pass as many as parameters your like. These parameters work like variables inside your function. Follow

**Passing Arguments by Reference**

It is possible to pass arguments to functions by reference. This means that a reference to the variable is manipulated by the function rather than a copy of the variable's value. Any changes made to an argument in these cases will change the value of the original variable. You can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition.\

**PHP Functions returning value**

A function can return a value using the return statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code. You can return more than one value from a function using return array(1,2,3,4).

* 1. **BACK END**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

**MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

**MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), http://www.fsf.org/licenses/, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information (http://www.mysql.com/company/legal/licensing/).

**The MySQL Database Server is very fast, reliable, scalable, and easy to use**.

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

**MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multithreaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multithreaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

**A large amount of contributed MySQL software is available.**

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server. The official way to pronounce “MySQL” is “My Ess Que Ell” (not “my sequel”), but we do not mind if you pronounce it as “my sequel” or in some other localized way.

**Connectivity**

* Clients can connect to MySQL Server using several protocols:
* Clients can connect using TCP/IP sockets on any platform.
* On Windows systems, clients can connect using named pipes if the server is started with the named\_pipe system variable enabled. Windows servers also support shared-memory connections if started with the shared\_memory system variable enabled. Clients can connect through shared memory by using the --protocol=memory option.
* On Unix systems, clients can connect using Unix domain socket files.
* MySQL client programs can be written in many languages. A client library written in C is available for clients written in C or C++, or for any language that provides C bindings.
* APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl are available, enabling MySQL clients to be written in many languages. See Chapter 29, Connectors and APIs.
* The Connector/ODBC (MyODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections. For example, you can use MS Access to connect to your MySQL server. Clients can be run on Windows or Unix. Connector/ODBC source is available. All ODBC 2.5 functions are supported, as are many others. See MySQL Connector/ODBC Developer Guide.
* The Connector/J interface provides MySQL support for Java client programs that use JDBC connections. Clients can be run on Windows or Unix. Connector/J source is available. See MySQL Connector/J 5.1 Developer Guide.
* MySQL Connector/NET enables developers to easily create .NET applications that require secure, high-performance data connectivity with MySQL. It implements the required ADO.NET interfaces and integrates into ADO.NET aware tools. Developers can build applications using their choice of .NET languages. MySQL Connector/NET is a fully managed ADO.NET driver written in 100% pure C#. See MySQL Connector/NET Developer Guide.

1. **SYSTEM DESIGN**
   1. **DATAFLOW DIAGRAM**

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

**Data flow Symbols:**

|  |  |  |
| --- | --- | --- |
| **NOTATION** | **SYMBOL** | **FUNCTIONS** |
| External entity |  | The sources of system inputs or sink of system outputs |
| Process |  | Perform some transformation of input data to yield output data |
| Data stores |  | A repository of data the arrow heads indicts net input and net output to store. |
| Data flow |  | An arrow line shows the flow of data into or out of a process or data store. |

**LEVEL 0**

DFD level 0 diagram, also known as context diagram, shows the interaction between the system and other external entities). This diagram is a high level view of a system and is often used early in the project to agree on scope. Context diagrams can be included in the requirements document. Such a diagram pictures the system at the center, with no details of its interior structure, surrounded by all its External entities, interacting systems, and environments.

**Admin**

**Database**

**LEVEL 1**

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

**User**

**Database**

**Level 2**

The next stage is to create the Level 2 Data Flow Diagram. This highlights the main functions carried out by the administrator of the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

**System**

**Database**

* 1. **DATABASE DESIGN**

A table is a collection of rows, where each row holds a data record. Each table row consists of key and data fields, which are defined when a table is created. In addition, a table has a specified storage, can support a defined maximum read and write throughput, and has a maximum size. A table is a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format. For example, databases store data in tables so that information can be quickly accessed from specific rows. Websites often use tables to display multiple rows of data on page. Spreadsheets combine both purposes of a table by storing and displaying data in a structured format.

A table is a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format. For example, databases store data in tables so that information can be quickly accessed from specific rows. Websites often use tables to display multiple rows of data on page. Spreadsheets combine both purposes of a table by storing and displaying data in a structured format.

Databases often contain multiple tables, with each one designed for a specific purpose. For example, a company database may contain separate tables for employees, clients, and suppliers. Each table may include its own set of fields, based on what data the table needs to store. In database tables, each field is considered a column, while each entry (or record), is considered a row. A specific value can be accessed from the table by requesting data from an individual column and row.

**Admin Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| uname | varchar(50) | Yes | NULL |
| password | varchar(50) | Yes | NULL |

**File Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| id | int(50) | Yes | NULL |
| fname | varchar(100) | Yes | NULL |
| filesize | varchar(250) | Yes | NULL |
| details | varchar(250) | Yes | NULL |
| date | varchar(50) | Yes | NULL |
| fkey | varchar(100) | Yes | NULL |
| oid | varchar(50) | Yes | NULL |

**Rating Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| id | int(50) | Yes | NULL |
| sname | varchar(50) | Yes | NULL |
| uid | varchar(50) | Yes | NULL |
| rating | int(50) | Yes | NULL |
| srate | int(250) | Yes | NULL |
| commt | varchar(250) | Yes | NULL |
| st | varchar(50) | Yes | NULL |
| status | varchar(50) | Yes | NULL |

**Register Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| id | int(50) | Yes | NULL |
| name | varchar(50) | Yes | NULL |
| gender | varchar(50) | Yes | NULL |
| address | varchar(50) | Yes | NULL |
| pnumber | varchar(50) | Yes | NULL |
| email | varchar(50) | Yes | NULL |
| uname | varchar(50) | Yes | NULL |
| password | varchar(50) | Yes | NULL |
| sname | varchar(50) | Yes | NULL |

**Server Rating Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Default** |
| id | int(50) | Yes | NULL |
| sname | varchar(50) | Yes | NULL |
| status | varchar(50) | Yes | NULL |