# Designing Multi-Layer Security Using Chaotic Map in Cloud Environment

Azita Rezaei [1], Ali Broumandnia [2], Seyed Javad Mirabedini [3]

st_az_rezaei@azad.ac.ir

***Abstract*—Security level is a major subject that is considered rapidly. Major solution is focus on fault prevention against attacks. Many applications are monitored by fault tolerance. This implies clients have intend to tailor their application in special environment. This paper introduces an innovative on creating and managing security level that leads to boost service provider's confidence and user satisfaction. This method allows user to specify and apply their security layer without requiring any knowledge about its implementation with SLA. This study proposes a multi-layer security that includes four main steps: Data Segmentation, Making Fake Services, Heuristic chaotic mapping and private key (PK) segment and Cryptography Code. PK is performed by chaotic maps and broker uses MQTT framework to facilitate security parameters. Experimental results show that the solution can balance the performance and security ranking and the propose scheme demonstrate security penalty cost is descended by 77%; the total penalty cost is decreased by 61.41% and the user satisfaction is grown by 60.67%. As a result, it ascends demanding and performance in cloud computing, compared with exiting approaches in encryption theories.**

***Index Terms*—Cloud computing, Security layers, Cryptography, SLA, Multi-Layer security Algorithm, MQTT**

## I. Introduction

Cloud computing technology is developed rapidly and security level is a main matter. Many companies, organizations and governments tend to immigrate their sensitive and confidential data to cloud storages.

As per four method of cloud architecture [1]: Public cloud, Community cloud, Private cloud and Hybrid cloud. [2] are faced on spoofing [3], Zombies or Service injection attacks; virtualization attack and Metadata spoofing attacks [4], etc. [5] Cloud Service Providers (CSP) are responsible to save and rescue data and applications for end users [6] when data is lost or sniffed on transaction.

In last decade, encryption of digital images is considered. Using standard and non-standard and hybrid methods can converted into data bits stream, and it is divided into equal size blocks and encrypted with the shared key encryption methods. [7]

Using Galois field and utility of matrix can handle data to transfer from source to destination. [8] all thing show that matrix utility is a safety base that can use in cryptographies algorithm.

Clients prefer to surf the web in secure environments when they are transition information. There are many algorithms for cryptography. Each cryptography algorithm uses a symmetric-key, an asymmetric-key and a hashing algorithm [9] [10] [11]. However, the constrained and heterogeneous nature of IoT devices, make it very difficult to apply well-known standard security solutions [12]. And, security policy can cause increase confidence and decrease performance and accessibility [13].

The Message Queuing Telemetry Transport (MQTT) is an application layer protocol that offers publish and subscribe communication model with broker [3]. Managing and cryptography data decline average service time and increscent service transition by broker managing. MQTT pave a novel road for broker to managing service and data due to transition on CSP and clients. Data might be stolen or corrupted when they are communication between two clouds.

In the light of the aforementioned limitations, the proposed multi-security layer scheme includes the following advantages:

1) This study focused on multi-layer security by using a novel security algorithm to prevent attacks when a system sends a request between two-cloud providers and shows impacting security priority on other non-functional requirements such as performance. There are a main question: Are there using security priority can increase customer satisfaction?
2) The main idea propose a novel cryptography process. A novelty Fake service covers true data. System mixed and configuration data to build complex packages that CSP will realize them. The ability of dynamic capacity in packages make them heuristic and making private key for heading and finally, cryptography algorithm covers data that the policy of cryptography refers to SLA.
3) The proposed schema offers a method to growth the confidence for clients, by decreasing the risk and maintenance cost and increase security policy on data protection. Our proposed algorithm is usage of MQTT benefits in SaaS [14]. Our designing can be beneficial on all architectural manufacturers.
4) The main schedule of cloud computing provider is preventing of attacking and catastrophic events. All task of security management is controlled by brokers. The proposed theory will be able to improve the performance of the system in SLA contracts.
5) Our motivation reflected to designing a data packaging and new algorithm to making private key. Datacenters

[1] Faculty of Computer Engineering, Islamic Azad University-South Tehran Branch,Tehran,Iran,
[2] Faculty Member of Computer Engineering, Islamic Azad University-South Tehran Branch,Tehran,Iran, corresponding author
[3] Faculty Member of Computer Engineering, Islamic Azad University-Central Tehran Branch,Tehran,Iran

use matrix utility to build a strong region.

6) Our method constantly keeps service and data integration when they transmit from a resource to a destination on cloud. It is important that all packages are identified by CSP.

In conclusion, this approach designs fake services, heuristic complex packages, and private key (PK) and cryptography algorithm such as RSA. This advantage makes a secure layer to cover data against hacking. Manipulation of fake service and PK are cover true data safety and it helps the package to be secure. This paper tries to balance non-functional requirements vs. security priority. We challenged to prevent mixing data, security and performance ratio, and customer satisfaction.

Most of the researches and methods used to symmetric or chaotic maps to perform security insurance. We intend to implement a new method with less complex and more confidence.

The rest of this paper is presented on VII sections. In section II we present review of related work. In section III we show abstract of previous work and definition that will take for our method. Section IV present a structure and motivation of our work and section V introduces our proposed method and in next section shows our experience and result. Last section draws some conclusions and shows possible future works.

## II. Related Work

In last decade, Cryptography is a paradigm that is considered for many scholar men. Many researchers try to present a survey to demonstrate how asymmetric algorithms such as RSA can defend as CSP against sniffed and attackers. In this section we briefly review some recent usages of security and performance that leads to cryptography on cloud computing.

As per G. Manogaran et al [15]: they used a map reduce framework [16] for big data to observe security on cloud computing. It is suitable for big data to be distributed. System has been solving big data problems. But they might be stolen when they are communication between two clouds. The authors did not specify how their solution performs when some data was stolen or lost.

As per J. Singh et al [17]: They focus on general cloud security issues relevant to the IoT-Cloud. The paper offers a new solution for defending against attacks that works for all levels of the clouds where it transmits service and data. They don't consider low level subsystem-specific security aspects and attacks.

As per Yang et al [18]: they considered a way to grow performance by improved strip partitioning in cloud. Despite, our work focus on security layer and improving performance together.

As per P. Gautam and et al [19] they used Obfuscate Original Source Code and RSA code to consider security on the electronic heath record by cloud computing.

As per M. Rathan [20] improved security feature has been added to the mobile stations in a registered group to eliminate the unnecessary utilization of resource by unauthorized station which maliciously consumes bandwidth and other facility provided by the cloud provider.

As per M. Tyagi et al [21] the framework, cloud service provider (CSP) selects the eminent server using cuckoo algorithm with Markov chain process and Levy's flight. After server selection, user encrypts their data using elliptic curve integrated encryption scheme (ECIES) at the user side and sends it to CSP for storage, where CSP stores the data after applying second encryption on it using advanced encryption standard (AES) at the cloud side.

As per Y. Sharma et al [22] the research paper the use of multiple encryption technique outlines the importance of data security and privacy protection. Also, what nature of attacks and issues might arise that may corrupt the data; therefore, it is essential to apply effective encryption methods to increase data security.

This paper focuses on the multi-layer security algorithm when it wants to send and receive data. In next section we consider on some utility that is impact on our work then we use MQTT properties to introduce our method. Finally, we will show our experience and draw the diagrams. They are helpful for evaluation customer satisfaction.

## III. Architecture and Security

There are some methods that expanding our work simply and it is helpful to make a standard framework. Before displaying the multi-layer security, we need to meet the following requirements:

### A. Service-oriented Architecture

SOA is a method to prepare services in distributed networks easily. Each component can be used in old services or in new modular services. In this situation, each system has a collection of compatible service that is used in different networks and domains. SOA includes processing layers or complex applications that have been developed. Thus, SOA has always repaired and improved properties in a system [23].

Each SOA has a service that is done by a CSP. This service may be a small process such as receiving or saving data, or the service may be complicated such as finding and printing a picture. In this paper, CSP request and respond service on cloud.

Each service has some properties like flexibility, reusability, agility, platform-independent and choreographic.

### B. Service Level Agreement

SLA is a formal contract between a service provider and customers to ensure service has standard quality. It is depend on multi factors such as accessibility, security, performance, service delays, internal power, etc. [24]

SLA agreement (ICTs) have committed to make documents the following standards [25] [26]:

- Expectations and requests from services
- Priorities
- Responsibilities of parties

SLA is essential to use as follows:

- The service supplier has the opportunity to increase their service's performance

- Customers have the opportunity to review their priority
- Uniformity in choosing assessment factors, between customers and CSP
- Creating secure income terms for the service provider

This opportunities will help to tolerant range of computing time by admit of customers.

### C. Security

As previous knowledge, failure, error, and fault words have different functionality [27]. In this paper, we focus on preventing fault damaging. There are plenty of components that offer fault tolerance policy [28] [29], and this paper considers self-protect to prevent damaged messages.

Each cloud is threatened by two positions: internal and external [30]. Internal security depends on internal execution systems such as damaged backup software, damaged repositories, leaked information, hazard confident, and destroyed data in DB. External security depends on data transfer, interface, data interference, stored data and users control accessibility [31]. This paper focuses on external security when data and services are transferred to get service.

However the optimal algorithm for each scenario depends of purpose and budget [32], Customers have a challenge to choose their policy according to their needs and infrastructure of hardware and software. Making decision can decrease cost and grow service flexibility, that it prevents waste of money.

Cryptography is a technique for sending and receiving data in an encrypted form where attackers can't trace the data clearly by encryption and decryption. However, the sending and receiving sources can recognize the data. Cryptography is used to transfer data securely, authenticate information, and provide confidentiality.

When attackers target clouds, triangle non-functional requirements (performance, accessibility and security) become significant increasingly. CSP always try to provide high level features to attract maximum customer satisfaction. Therefore, the main problem can be stated as "How one's security confidence can increase, while performance rate is decrease?"

RSA is an asymmetric algorithm that is the result of multiplying two random big integers. The result is used for making public and private keys. Our approach used RSA to complicate finding and discovering packages.

There are several security standards such as ISO-27000 and IEC which create a specialized standard for everyone. They have a common committee that is named ISO/IEC JTC1. Information Security Management System (ISMS) uses hazard management to save confidence, reality, and accessibility. ISMS guarantee that hazards have been controlled. Acceptance of ISMS is a strategic decision for an organization. An organization's decision depends on requirement, purpose, security requirement, processes, and scale of the organization. It is expected that these variables change over time [33]. In this paper we consider some recommended where is on ISO-27000 that helps us to consider the services availability, integrity and confidentiality. [4]

### D. SOA and Clouds

Flexibility, maintenance and changeability are important components that leads system has a little coherence in distributed systems. Component dependency can be used to increase distribution and scalability in cloud computing. To achieve this goal, cloud computing uses brokers. They are able to control service communication which allows them to coordinate clients and CSP.

There are several advantages to use SOA in cloud such as dynamically, agility, decreased platform cost and repository.

By using this benefits, large scale systems should not depend on services, components and software terminals. They can carry out services, without knowing about the service location. This property allows systems to copy, transfer, and migration data. Loose components in brokers invoke services from anywhere. Then clients respond to find suitable CSP. This paper considers usage of brokers to manage our work.

Service Oriented Cloud Computing Architecture (SOCCA) is one of the concept of combining SOA and cloud computing [34]. This model consists as a hierarchy. On the top of the model, there are SOA and Brokers, then the Cloud Ontology Mapping Layer, followed by the Individual Cloud Provider Layer. MQTT assists this model to have better influence. Combination of two schema has shown in the following **Figure 1**. This concept can give an image for comminute between clouds.

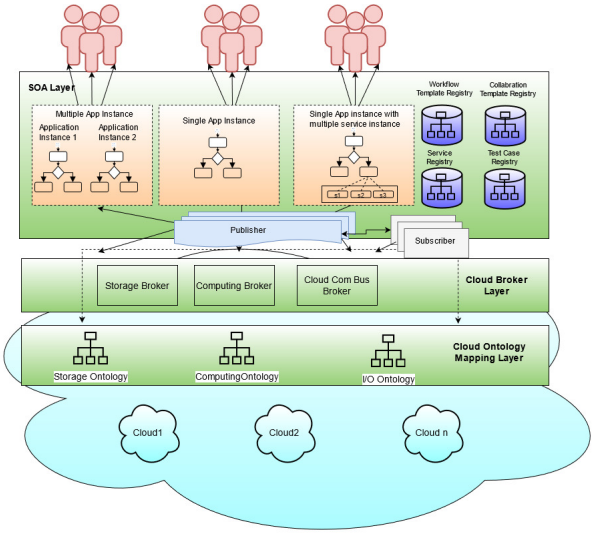



Fig. 1: *Compilation of cloud computing architecture business base and MQTT model*

## IV. Background

MQTT protocol is one of the most extended protocol on the IoT that leads to less capacity for easy implementation on light weight, cheap, low-power and low memory devices [35] . The client can be a publisher or subscriber. The common secure MQTT protocols is used username and password in the "CONNACK" message for authenticating [3].

The cipher attack can miss data whom transition between broker and publisher/subscriber and CSP. Using block cipher

by performing encryption and decryption make key block cipher.

As per A. Cerrada and et al [35] each MQTT has three types of participants: Broker that is charge of the exchange of messages between the other participants. Publisher/ subscriber (client) is send data to the broker and provider service receives data from broker.

Using Hardware Security Manager (HSM) system can covering security in Broker. They selected asymmetric cryptography algorithm (RSA) for block cipher algorithm and symmetric cryptographic for payload encryption. For each transaction, system encrypts data by private key and public key with a random number. Although, Singh Bail and et al [3] proposed three part for cover security, it is time consuming and make redundancy in key producing. As per Amoretti et al [12] proposed a new layer for each line and site by brokers and used RSA for cryptography access token. They presented them idea in industrial IT scheme. It works by making tracing memory that can create redundancy on overhead and it faced on maintenance challenging and cloud manufacturing.

To overcome security challenging on transaction data on network and internet, chaos is performed to encryption and decryption data. The chaotic map base encryption, provides a comprehensive performance as compare to the standard encryption techniques. [7]. In this study, reversible and discrete chaotic maps are used to permutation operations in cryptography. This reversible and discrete chaotic map is defined by Eq. 1 and it's called the 2D modular chaotic map (2DMCM) [7].

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \otimes \begin{bmatrix} x_n \\ y_n \end{bmatrix} \tag{1}$$

$$x_{n+1} = (a \otimes x_n) \oplus (b \otimes y_n)$$

$$y_{n+1} = (c \otimes x_n) \oplus (d \otimes y_n)$$

If (greatest common divisor) $gcd(|A|, n) = 1$ multiplicative inverse is defined by that n is length of matrix:

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} \otimes \begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} \tag{2}$$

## V. Proposed Method

This section introduces 4 step which called multi-layer security data transaction as Figure 2.

Brokers are able to receive a request from clients and demonstrate the appropriate provider service. We consider that broker knows about how it can manage service allocating to clouds. Also, each service can contain of many related data that must be transmitted. So we use service symbol instead data in continue.

Therefore, there are various ways to managing service such as Wiedemann Algorithm [18]. In this section, services will sent to cloud provider in packages. This is a channel for sending and receiving services. The broker receives a request from clients, and completes a cryptography process before

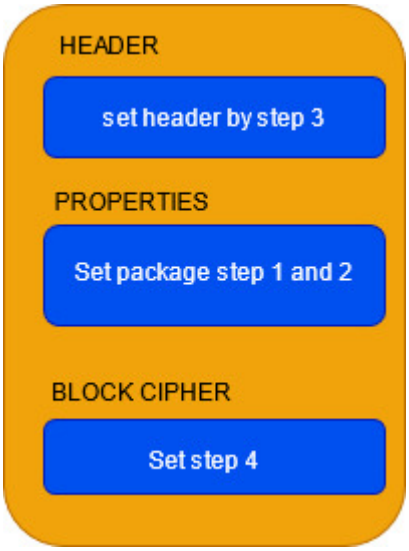



Fig. 2: *connect package*

sending to CSP. This task has four steps and all process has done in broker:

**1. Data Segmentation:** Each service is partitioned to (n) parts which must be greater than 3 (in next section we assume 4 parts to describe our experience) each part must be set in a certain cell in an (n*n) matrix. Each part of the first service is set in the first column a matrix sequentially. The second service is partitioned to n parts like as previous service and is set in the second column of matrix sequentially. By this way, all services is located in matrix until matrix will full. It is important that the minimum (n) amount is 3 and parts is arranged in $(n * n)$ matrixes.

Next formula shows partitioning service that is set in a matrix:

$$n = \begin{cases} 3 & \text{if } \min(x) \leq 3, \\ \lfloor \sqrt{\sum(x)/k} \rfloor & \text{otherwise.} \end{cases} \tag{3}$$

According to this formula there are some assumption as follows:

**n is size of matrix**
**x is length of each service**
**k is amount of all service**

**If n is less than 3 , system assumed n is 3 and matrix is** $3 * 3$

.

**2. Making Fake Services**: System can make a fake service that is set as a real service in a matrix. The header section saves tag of fake service data, due to service receiver will be able to recognize and remove it.

A fake service is an unusable service or intercalary data that is fault and use for covering other real services. It can involve random number or random alphabets. This data is made in a matrix and increase redundancy data. Although, it make a trap to grow complexity of packages, it is effective on covering. So, it is made randomly to prevent redundancy.

**3. Heuristic chaotic mapping and key segment:** each cell of rows has a certain location. So, they can change their location randomly without changing their tags, system converts

decimal format to binary format and locates on a new table. Each bit code is multiplied by primary number as follows:

| B1 (part) | C1 (part) | A1(part) | D1(part) |
|---|---|---|---|

A1=first service = 1, B1=second service= 2 , C1= third service = 3, ...
A2=first service = 1, B2=second service = 2, C3=third service = 3,...

In next step, all cells are displaced randomly by mapping algorithm. Each tag relocated by this mapping. This approach helps to grow complexity and growing complexity is preventing approach against hackers.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1y} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2y} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3y} \\ \vdots & & & & \\ a_{x1} & a_{x2} & a_{x3} & \dots & a_{xy} \end{bmatrix} \tag{4}$$

To finding new located of tags first x and y index is multiplied to $\frac{1}{k}$ that k is odd number. So new position is changed to:

$$\begin{bmatrix} x_1 + ky_1 & x_1 + ky_2 & \dots & x_1 + ky_j \\ x_2 + ky_1 & x_2 + ky_2 & \dots & x_2 + ky_j \\ \vdots & & & \\ x_i + ky_1 & x_i + ky_2 & \dots & x_i + ky_j \end{bmatrix} \tag{5}$$

Mapping matrix can change by (2*1) matrix because broker will send each row separately and column tags is not changed. The matrix is (n*n); so our x and y is calculated as follows:

$$a_{xy} \rightarrow x_n = (x_i + y_j) \quad mod \quad n$$

**n is size of matrix**

next step system makes ptivate key by new matrix as follow:

| Tag | Tag B1 | | | Tag C1 | | | Tag A1 | | | Tag D1 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Tag address | 4 | 2 | 1 | 4 | 2 | 1 | 4 | 2 | 1 | 4 | 2 | 1 |
| Pi | 2 | 3 | 5 | 7 | 11 | 13 | 17 | 19 | 23 | 29 | 31 | 37 |

$$Privatekey = \sum (Bit(Tag \quad ai) * Pi) \tag{6}$$

So, n*n matrix tags is calculated:

$$\begin{bmatrix} PK_1 \\ PK_2 \\ \vdots \\ PK_n \end{bmatrix} = \begin{bmatrix} tag_{11} & tag_{12} & \dots & tag_{1n} \\ tag_{21} & tag_{22} & \dots & tag_{2n} \\ \vdots & & & \\ tag_{n1} & tag_{n2} & \dots & tag_{nn} \end{bmatrix} \otimes \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_n \end{bmatrix} \tag{7}$$

So, each part of data is located on random location with unique tag whom named private key. The private key will be sent to receiver and it will relocated each part of data by reversing.

As instance of Diffie–Hellman algorithm key, our work determine P as prime number that is established on certain cell.

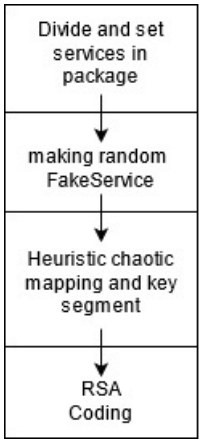



Fig. 3: *Cloud computing architecture business base*

**4. Cryptography Code:** Each matrix is dedicated in independent arrays from rows that contains real service and fake service that is packaged, relocated , made private key and covered by cryptography algorithm and is sent to the provider cloud. Each provider has a broker to decrypt and sets back all services to the original shapes. Also, it can recognize fake services and remove them.

We can define multi-layer security architecture to show how this method encrypts a service. The following figure shows multi-layer security architecture (Figure 3).

The next figure shows the relationship between clouds. (Figure 4)

The element (n) is a private number whom is known between client/CSP and broker.

The following steps shows an example for a 4*4 matrix:

We assume all services have equal length, X=8 char. So, the random number is 4 and we divided each service in to four parts. This means each service and fake service are set in a 4*4 matrix. Each part sets in cells of a column. Thus, first service part is located in the first column. By the same method, the next service is divided in to four parts and is located in the second cells of second column. We are continuing this method until the matrix is be full. The table I shows this process.

| segmentation | part 1 | part 2 | part 3 | part 4 |
|---|---|---|---|---|
| Service one: | A1 | A2 | A3 | A4 |
| Service two: | B1 | B2 | B3 | B4 |
| Service three: | C1 | C2 | C3 | C4 |
| Service four: | D1 | D2 | D3 | D4 |

TABLE I: *First array for crypto*

| A1 | B1 | C1 | D1 |
|---|---|---|---|
| A2 | B2 | C2 | D2 |
| A3 | B3 | C3 | D3 |
| A4 | B4 | C4 | D4 |

Fake services follow the same process as a real service and

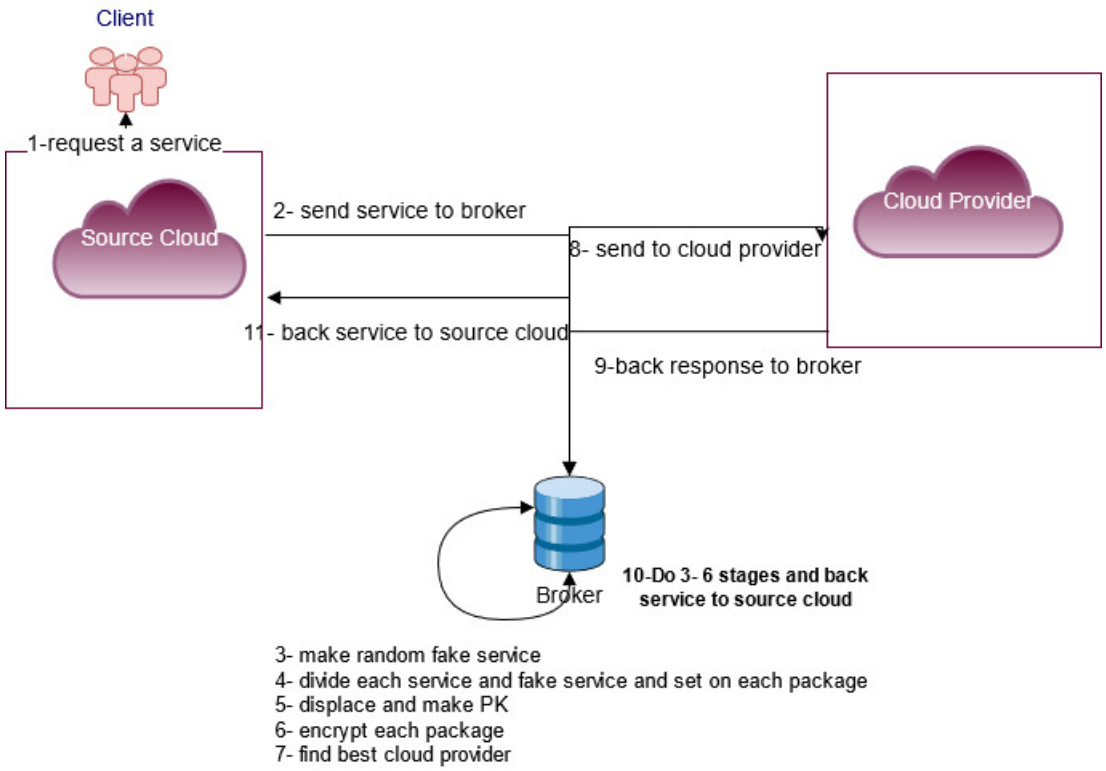



Fig. 4: *The relationship diagram between clouds on multi-layer security base*

are divided to four parts and is set in an array. We can make it for each service, but it prepares redundancy. So, our solution generates it randomly.

For simply explanation, we assume $\frac{1}{1}$ mapping.

$$\begin{bmatrix} x_1+y_1 & x_1+y_2 & \dots & x_1+y_j \\ x_2+y_1 & x_2+y_2 & \dots & x_2+y_j \\ \vdots & & & \\ x_i+y_1 & x_i+y_2 & \dots & x_i+y_j \end{bmatrix}$$

In this matrix n = 4 ; so, x and y set as follows:

$$X_n = (x_i + y_i) \quad mod \quad 4$$

If $(x_n = 0) \rightarrow x_n = n$

In this schema each tag relocated as follows:

| | | | |
|---|---|---|---|
| B1 | C1 | D1 | A1 |
| C2 | D2 | A2 | B2 |
| D3 | A3 | B3 | C3 |
| A4 | B4 | C4 | D4 |

A1=1, B1=2, C1=3, D1=4 , . . .
A2=1, B2=2, C2=3, D2=4,. . .

| **Tag $B_1$** | | | **Tag $C_1$** | | | **Tag $D_1$** | | | **Tag $A_1$** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| **Tag $C_2$** | | | **Tag $D_2$** | | | **Tag $A_2$** | | | **Tag $B_2$** | | |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| **Tag $D_3$** | | | **Tag $A_3$** | | | **Tag $B_3$** | | | **Tag $C_3$** | | |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| **Tag $A_4$** | | | **Tag $B_4$** | | | **Tag $C_4$** | | | **Tag $D_4$** | | |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 2 | 3 | 5 | 7 | 11 | 13 | 17 | 19 | 23 | 29 | 31 | 37 |

$$\begin{bmatrix} 269841 \\ 74865 \\ 566618 \\ 697015 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots \\ 0 & 1 & 1 & 1 & \dots \\ 1 & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & 0 & \dots \end{bmatrix} \otimes \begin{bmatrix} 2 \\ 3 \\ 5 \\ 7 \\ \vdots \end{bmatrix}$$

Each tag has 23 binary code and matrix assumed 4*4 then PK is performed by 12 multiple. If

$$\begin{cases} 1 & \text{if } tag_i * P_i = 0, \\ tag_i * P_i & \text{otherwise} \end{cases} \tag{8}$$

In this example, maximum PK is equal to $1.072.445 \cong 2^{20}$ and minimum Pk is equal to $73.815 \geq 2^{16}$. If tag length extra to 8 digits so $P_i \geq 2^{23}$ and PK for first part length $\geq 2^8 * 2^{20} = 2^{28}$. Consequently, PK for 4 parts is equal to $2^{28} * 2^{44} * 2^{52} * 2^{54} = 2^{178} \cong 3.8 * 10^{53}$ sequentially. The

PK space for each part is produced $2^{178}$ . As a result, if the standard key generation has $2^{128} \cong 3.4 * 10^{38}$ our private key is acceptable. And, if each computer works in a year, all times is $3.1 * 10^7$ s and the fastest computer computes has $2^{80}c/s$ process per second [7], therefore, our PK needs $(3.4 * 10^{38})/(3.8 * 10^{31}) = 10^{23}s$ to decrypt. As a result, this number shows that hackers need $33 * 10^3$ years to decrypt.In conclusion, the more parts make longer PK.

So, each part of data can be located on random location with unique tag that formulas by primary numbers whom was named private key. The private key will be sent to receiver and it is relocated part by reversing.

Finally, we assumed RSA algorithm to cover arrays and each row of the array is sent to the broker by different routers. The broker will receive packages, decrypt them, and delete all fake services. Finally, each service will be set and organized by header number Figure 5.

The following description display this approach:

TABLE II: *Description of the notation used in the considered layered security*

| Symbol | Description |
|---|---|
| EP | Each service is divided to (n) parts |
| T | Tag is organized header for each package to identify |
| IP | This array can save each part of service in specified cell |
| FS | Fake Service |
| PK | Private key that is referred by formula 2 |
| P | An array for saving each part before sending to the 2-dimensional array |

**Algorithm 1** shows how a pub/sub can make multi-layer security by FS and crypto method that we proposed on previous section. For making PK and covering parts we need an **algorithm 2.**

**Algorithm 2** is a sub function of algorithm 1 that show how broker can displace parts and make PK.

Next algorithm (**Algorithm 3**)shows how provider side be able to decrypt packages and recognize FS and reconfigure a service.

We use some variables to show our algorithm. "EP" presents a partitioning service in (n) parts where is calculated by Eq. 3. However, it relates to how much requests there are, and priority of data. If n number is determined long, it leads to grow redundant on arrays and it is time consuming.

"P" is an array for saving each part before sending to 2-dimensional array, "IP" is a 2-dimensional array that saves each part in a form. We must ensure that each package is set in an "IP". While the schedule is running, the system makes random "FS" that are set inside the "IP" array alongside regular services. "T" is symbol of stage that organized header for each package. "PK" is a private key that made by displacing parts and crypto tags by Eq. 5. Finally, RSA coding covers all parts separately and the array is sent to broker.

**Data:** n from Eq. 3
**Result:** Send packages and private key to the broker
initialization;
Set header for each package;
**while** *element of list greater than 0* **do**
  EP is divided to n parts;
  Select each part and set on cells in the array sequential(p);
  **if** $y_i = null$ **then**
    set all remain $column_i$ by null;
  **end**
  **if** *service is traced* **then**
    make random number (0 or 1);
  **end**
  **if** *A = 1 Make FS and set on the array* **then**
    Divide FS to n parts that has calculated;
    Select each part and set on cells in the array sequential(p);
  **end**
  go to next service;
  package list;
  Set T and relocate cells and make PK(Algorithm2)and set on IP;
**end**
**while** *package greater than 0* **do**
  encrypt package by RSA;
  Send packages and PK to the broker;
**end**

**Algorithm 1:** *Encrypt service in customer broker side*

**Data:** matrix n*n
**Result:** making PK row by row
get T of each matrix;
select k;
$Pi \leftarrow 0$
**for** $i = 1$ *to* $n$ **do**
  **for** $j = 1$ *to* $n$ **do**
    $x_{ij} = i + k * j$
    $x_{ij} \leftarrow x_{ij} mod n$;
    **if** $x_{ij} = 0$ **then**
      $x_{ij} \leftarrow n$;
    **end**
    transition each part to new T;
  **end**
  **for** $z = 1$ *to* $n * amountbitword$ **do**
    $PK = (Bitwordtag) * Pi_z * PK$
    **if** $Bitwordtag == 0$ **then**
      go next;
    **end**
  **end**
  save PK and send by each row;
  $PK \leftarrow 0$
**end**

**Algorithm 2:** displacing parts and making PK

Packages are delivered to broker. It decrypts them, find each part of service by private key and sets them together

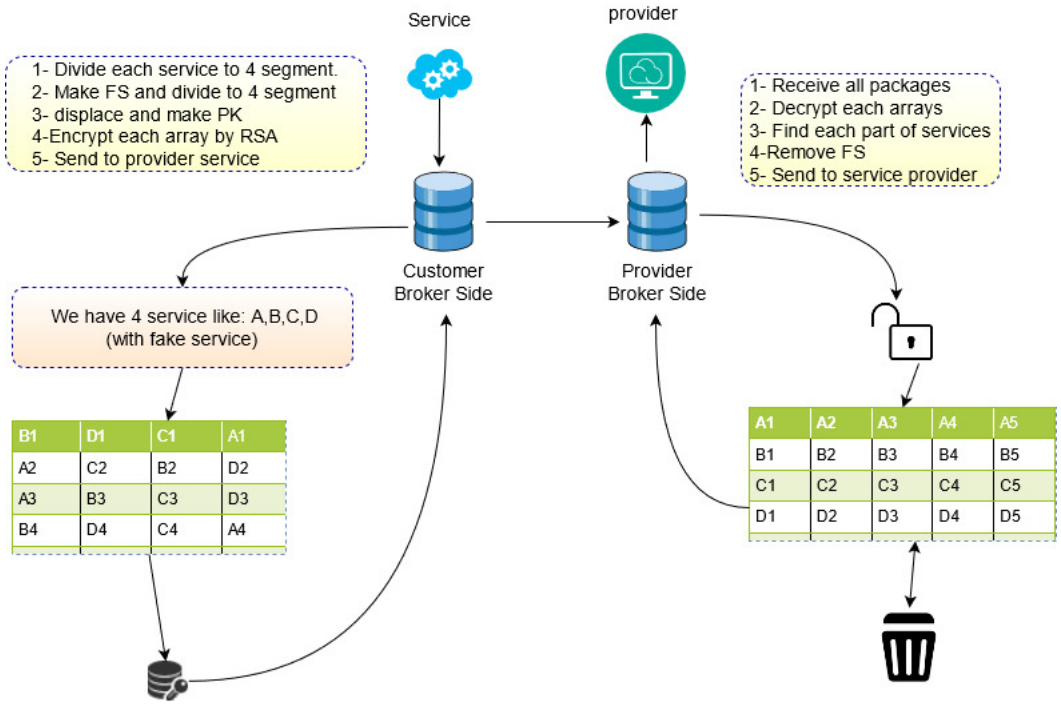



Fig. 5: *multi-layer security schema for services*

**Data:** package
**Result:** Return service to cloud provider
**while** *package greater than 0* **do**
  Get PK and package and decrypt;
  Copy this package in to a list;
  Relocate each cell by PK;
**end**
**while** *list is greater than 0* **do**
  Find FS and remove from list;
  Get each part of element and arrange together;
  Set each element in service part;
  **if** *list is empty* **then**
    select next list
  **else**
    send fault massage to sender
  **end**
**end**

**Algorithm 3:** *Decrypt and unpack service in provider broker side*

by considering headers. If a package is true, it would be a part of a service, however fake service or redundant data must recognized by headers and they are removed. Finally all true packages are set together in a true place and when all parts of the puzzle completed service is received correctly. Otherwise, the broker recall the package by header identification from sender. Finally, broker finds and invokes the appropriate CSP to accomplish its job.

## VI. Experimental Result

We evaluated the efficiency of multi-layer security after performing and measured penalty cost and execution time.

There is a direct relationship among decrease of penalty cost and increase customer satisfaction. For each unit of penalty cost, customer satisfaction increase with three non-functional factors (integrity, availability and confidentiality).

One of the main significance of our implementation was competition between security and performance. Growing time execution can decrease performance and availability and it is a threat against security. This attempt simulated on three condition. First, we trend penalty cost and time execution on normal environment then we performed multi-layer security and in finally round we consider security layer beside of dead time line.

There are some variables that influence on security ranking, such as firewall, password, routers, net traffic, IP confidence, damaged data, and lost data. Failing on each factor effects on SLA contract and it raises penalty cost. Therefore, they impact on customer satisfaction. Cryptography shape, covers service and data until they transmit safety. At our experience each factor effect on coefficient, therefore coefficient makes penalty cost. If each factor faults, it will less index of security and it increases penalty cost.

To provide a processing function to metric security cost, system assumes 0 to 9 index for each factor of security, performance and accessibility. This benchmark could certain our penalty cost at last. Each factor depends on different parameter. Security consists of firewall, crash data, and data loss parameters. If each parameter descend, fault event has happened and the system automatically calculates penalty cost. Accessibility depends on time ranking that system serve service on certain time and performance is rate of execute time on execute CPU time. We choose tolerant impact factor for each requirement. Security has more priority than other

factors (performance and accessibility) [36]. Therefore, there is a reverse relationship between penalty cost and satisfaction that cause fewer penalties due to ascent satisfaction.

This work used the discrete event simulator CloudSim for calculating management resources and performance strategy. This simulator is useful for simulating cloud resource modeling and program scheduling. Also, it has the appropriate datacenter broker class to manage resources, scheduling, sending and receiving services. [37] [38] [39] We used this class for crypto service and sending to broker. This experience used an X86 system architecture, windows OS, and virtual machine Xen were simulated for the environmental conditions. All data has produced randomly by simulation application.

The total penalty consist of all requirements that is mentioned on SLA contrast such as performance, availability and security. In this paper, we focus on 3 nonfunctional requirements and set index for them.

As we mentioned in the previous paragraph, this experience consist of three attempt: **(Table III)**

**Attempt I;** service was done in normal environment.

- This attempt has no crypto layer and the crypto average time is zero.
- The average service time is 254 ms per 100 clients.
- The penalty cost is 1205 of 1577 that is accepted by system. So, the system average percentage penalty cost is 64.41%.

This experience was implemented on basic policy. According to mentioned index, System could not pass fault tolerance completely and accepted a serious penalty.

**Attempt II;** service was done with the multi-layer security (our proposal) without any alternation on dead time therefore if security layer takes time, it is not consider on total dead time.

- This attempt was implemented a security layer. The crypto average time takes 26 ms per 100 clients.
- The average service time is 253 ms that it is nearest time to the first attempt.
- The penalty cost is 1191 of 1413 that is accepted by system. So, the system average percentage penalty is 84%.

This experience used our proposed method with normal dead time without privileged for processing the security layer. So, it leads to increase average percentage penalty.

**Attempt III;** service was done by multi-layer security with adding crypto time on total dead time.

- This attempt is used to multi-layer security and the crypto average time is 30 ms per 100 clients.
- The average service time is 251 ms that it is nearest time to other attempts. This is a benchmark for preparing cryptography time and penalty cost fairly.
- The penalty cost is 195 of 1290 that is accepted by system. So, the system average percentage penalty is down to 15%.

The final experience showed that the multi-layer security with increasing dead time, can transfer data and services perfectly and reduces penalty cost.

We proposed data is made by simulation randomly. And the average service times and average crypto times in each attempt are not equal. So, the predicted penalty and payment penalty are no similar. In next table (**III**), shows result of experience in each situation.

As can be seen in **table IV**, the values of crypto time in second and third attempt is less than other cipher methods. Therefor the proposed method achieves optimal parameters in third round. The second round and third round are the same in crypto and service time, and they are different on penalty cost by different dead time line. As **table IV** presents that our method on attempt III has better ration than other approaches.This table presents the lowest amount of ratio has more performance.

In the following, there are a comparison on 3 nonfunctional requirements on three trials:

*The Performance penalty cost on different condition is shown in **Figure 6(a)**.*

*The Accessibility penalty cost on different condition is shown in **Figure 6(b)**.*

*And the Security penalty cost on different condition is shown in **Figure 6(c)**.*

In **figure 7** there are compressive comparing on three trials. In conclusion, our algorithm can reduce the result by 61.41%.

It has been mentioned that multi-layer security is needed extra time to run, so, dead time was risen in third trial.

Next diagram demonstrates penalty cost comparison in two trials (first and last experience)**(Figure 8(a))**. **Figure 8(b)** shows a percentage comparison on penalty cost in our method versus normal environment. The results are shown in **Figure 8**.

The result in figure **8(b)** shows that the blue column in first trial is not conformities with SLA, and the red column in multi-layer security in third trial is reduction penalty cost. According to previous table and chart, our method shows that it can effective on preventing attacks and cover true service and data. This design is suitable for services which is not limited on dead time line.

We repeated each trials for multi times and we recorded security penalty cost and total penalty cost. The result shows that security penalty cost is decreased by 77% and total penalty cost is decreased by 61.41%. As a result, verifying customers is grown by 60.67%.

In next Figure, we demonstrated our method that can effect on total penalty comparison against other trials. Although multi-layer security can increases average execution time, it can reduce penalty cost obviously **(figure 9a)**. **Figure (9b)** shows average of finish time and **figure (9c)** presents how much time taking for system to execute crypto code.

The next figure demonstrates a comparison by scatter plot on total penalty cost in three attempts **(Figure 10)**. This figure shows our approach can satisfy security policy compared with exiting approaches.

As shown in the plot, third attempt has the lowest penalty cost compared to the others.

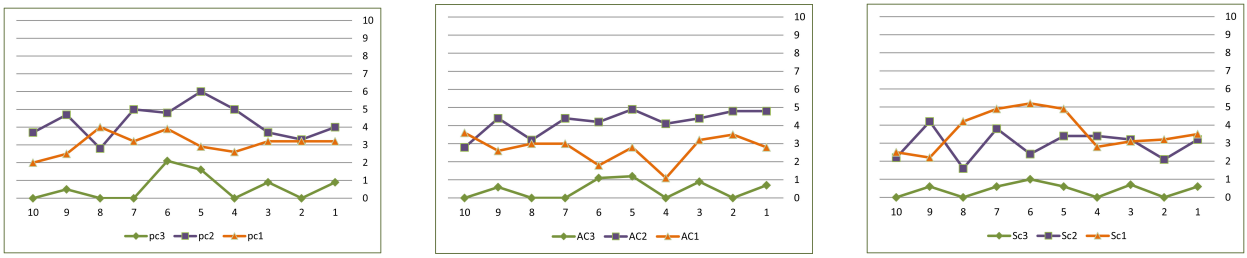
TABLE III: comparison of different multi-layer security implementations

| Experience | Average percentage penalty | Payment penalty | Total penalty | Crypto average time (ms)* | Average service time(ms)* |
|---|---|---|---|---|---|
| Experience I | 76.41% | 1,205 | 1,577 | 0 | 254 |
| Experience II | 84% | 1,191 | 1,413 | 26 | 253 |
| Experience III | 15% | 195 | 1,290 | 30 | 251 |

*average crypto time and average service time is for 100 clients. Crypto time isn't calculated on Service time.

TABLE IV: The crypto average time and service time of various cipher method

| Experience | Crypto average time (ms) | Average service time(ms) | ratio of Crypto/service time |
|---|---|---|---|
| Experience I | 0 | 2.54 | 0 |
| Experience II | 0.26 | 2.53 | 0.1 |
| Experience III | 0.3 | 2.51 | 0.12 |
| Ref. [35] | 1.6 | 2 | 0.8 |
| BS-MQTT [3] | 1.5 | 2.4 | 0.63 |
| X-broker [12] | 3.7 | 7.2 | 0.51 |




(a) (b) (c)

Fig. 6: comparison of penalty cost on three nonfunctional equipment. Performance penalty cost (a), Accessibility penalty cost (b) and Security penalty cost (c)

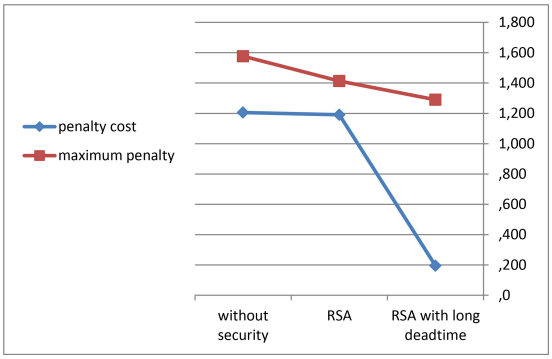



Fig. 7: penalty cost vs. defined maximum penalty assumption in three trial

## VII. Conclusion

We used self-protect with considering ISO-27000 , SOA's benefit , cloud utility and a pattern to image our plan easily (SOCCA) and MQTT together. We made a new multi-layer security Vs performance and accessibility when security cost index increased. Finally, we measured how customer satisfaction changed for each experience.

We assumed that there are many factors effect on destroying a service. We assumed each service has a same service time to show impaction of data covering and crypto time on service time.

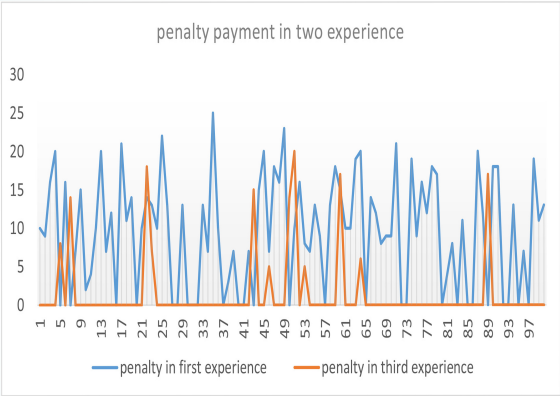
Our algorithm has four layers to encrypting services. The first layer has divided service in to (n) parts and located in a column of a matrix. Then, the system makes fake services randomly and sets in matrix like a real service. The broker makes private key and implement chaotic map on all parts heuristically. We proved the key generation space can increase greater than $2^{128}$.At the end, the array is encrypted by RSA coding. This work makes complex schedule to cover true data. There are no dependency on RSA algorithm, thus it could be replaced with other crypto algorithms such as ECC.

This algorithm has built by a distributed operation, fake services, private key and cryptography algorithm. All layers is encrypted by brokers. Broker divides each service with certain equation and changes shape without changing their properties of them, and the lowest ratio can improve performance and boost it. Therefore, the integrity is protected and transparency is obtained for clients.

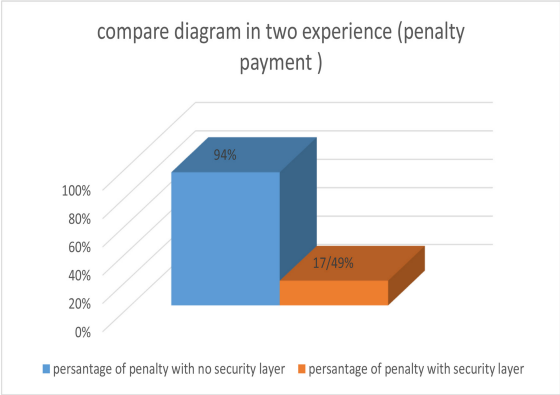
Each layer helps system to build a strong chain. CSP can recognize fake service and private key. So, they can rebuild an original service.This method is boost on cipher attacks due to each package is sent on disparate routers that recognizing and collecting all packages is incredible.

As a result, we demonstrated the percentage of package discover is hard because each package is sent by random routers. But if it is intercepted, multi-layer security prevents to all data was discovered. In this case, the broker can request the lost package again until assuring that all parts has received. Additionally, we intend to ask a few question such as how will memory system could be built to save each package efficient and how is system certain that, the package has received truly?

This method focus on each data parts with certain places in array. So, if we don't care about sequence of numbering what is a solution?




(a)




(b)

Fig. 8: (a) penalty cost comparison on first and last experience; (b) percentage comparison on multi-layer security

Our solution considers on non-functional requirements like performance and accessibility. In addition, customer satisfaction is an important challenge and finding a solution to protect data in process schedule is a problem. This method is multi-layer security, and flexibly allows clients to make better decisions to compare with exiting approaches and considering on their budget and demands.

This work requires an n*n matrix where PK is a private key between the cloud pub/sub, CSP and broker on MQTT. This method depends on the amount of services, service size, and dead time for each service. It has perfect efficient on services with approximately same capacity and it can increase arrays performance.

This method needs extra time and it is suitable for unlimited dead time services. However, dead time can impact on services which is important on real time schedule.

This work assumed multi variable to determine and calculate penalty cost. However, our method covers services when commutes in cloud by making multi-layer solution and it is useful in all architectures.

## References


[1] D. J. Hwang K., Fox G., *Cloud Architecture and Datacenter Design in Distributed Computing: Clusters, Grids and Clouds*. Georgia State University, Chapter7-Cloud-Architecture, 2010.

[2] S. H. Amin Z., Sethi N., "Review on fault tolerance techniques in cloud computing," *International Journal of Computer Applications*, vol. 116, no. 18, 2015.

[3] P. Z. R. Bali, F. Jaafar, "Lightweight authentication for mqtt to improve the security of iot communication," *ICCSP '19: Proceedings of the 3rd International Conference on Cryptography, Security and Privacy*, pp. 6–12, 2019.

[4] M. V. Pawar and J. Anuradha, "Network security and types of attacks in network," *Procedia Computer Science*, vol. 48, pp. 503–506, 2015.

[5] P. D. B. B. e. a. Modi, C., "A survey on security issues and solutions at different layers of cloud," *J Supercomput 63*, 2013561–592 (2013).

[6] V. S. B. J. B. I. Buyya R., Yeoa Ch., "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, pp. 599–616, 2008.

[7] H. Ghazanfaripour and A. Broumandnia, "Designing a digital image encryption scheme using chaotic maps with prime modular," *Optics and Laser Technology*, vol. 131, p. 106339, 2020.

[8] A. Broumandnia, "Image encryption algorithm based on the finite fields in chaotic maps," *Journal of Information Security and Applications*, vol. 54, p. 102553, 2020.

[9] V. R. Pancholi and B. P. Patel, "Enhancement of cloud computing security with secure data storage using aes," *International Journal for Innovative Research in Science and Technology*, vol. 2, no. 9, pp. 18–21, 2016.

[10] E. A. Pansotra and E. P. Singh, "Cloud security algorithms," *International journal of security and its applications*, vol. 9, no. 10, pp. 353–360, 2015.

[11] A. Bhardwaj, G. Subrahmanyam, V. Avasthi, and H. Sastry, "Security algorithms for cloud computing," *Procedia Computer Science*, vol. 85, pp. 535–542, 2016. International Conference on Computational Modelling and Security (CMS 2016).

[12] T. Halabi and M. Bellaiche, "A broker-based framework for standardization and management of cloud security-slas," *Computers and Security*, vol. 75, pp. 59–71, 2018.

[13] Z. Shen and Q. Tong, "The security of cloud computing system enabled by trusted computing technology," in *2nd International Conference on Signal Processing Systems*, vol. 2, pp. V2–11–V2–15, 2010.

[14] L. Tang, J. Dong, Y. Zhao, and L.-J. Zhang, "Enterprise cloud service architecture," in *IEEE 3rd International Conference on Cloud Computing*, pp. 27–34, 2010.

[15] M. V. K. Gunasekaran Manogaran, Chandu Thota, "Metaclouddatastorage architecture for big data security in cloud computing," *Procedia Computer Science*, vol. 87, pp. 128–133, 2016.

[16] Q. Zheng, "Improving mapreduce fault tolerance in the cloud," in *IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW)*, pp. 1–6, 2010.

[17] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "Twenty security considerations for cloud-supported internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 269–284, 2016.

[18] F. J. X. L. T. Yang L., Huang G., "Parallel gnfs algorithm integrated with parallel block wiedemann algorithm for rsa security in cloud computing," *Information Sciences*, vol. 387, pp. 254–265, 2017.

[19] S. S. Gautam P., Dilshad Ansari M., "Enhanced security for electronic health care information using obfuscation and rsa algorithm in cloud computing," *International Journal of Information Security and Privacy (IJISP)*, vol. 13, p. 11, 2019.

[20] R. M., "Resource provision and qos support with added security for client side applications in cloud computing," *International Journal of Information Technology*, vol. 11, 2019.

[21] M. B. Tyagi M., Manoria M., "A framework for data storage security with efficient computing in cloud," in *International Conference on Advanced Computing Networking and Informatics*, vol. 870, pp. 109–116, Springer Singapore, 2019.

[22] K. S. Sharma Y., Gupta H., "A security model for the enhancement of data privacy in cloud computing," in *Amity International Conference on Artificial Intelligence (AICAI)*, pp. 898–902, 2019.

[23] R. G., "Cloud computing and soa," tech. rep., The MITRE Corporation, 2009.

[24] P. Xiong, Y. Chi, S. Zhu, H. J. Moon, C. Pu, and H. Hacigümüş, "Intelligent management of virtualized resources for database systems

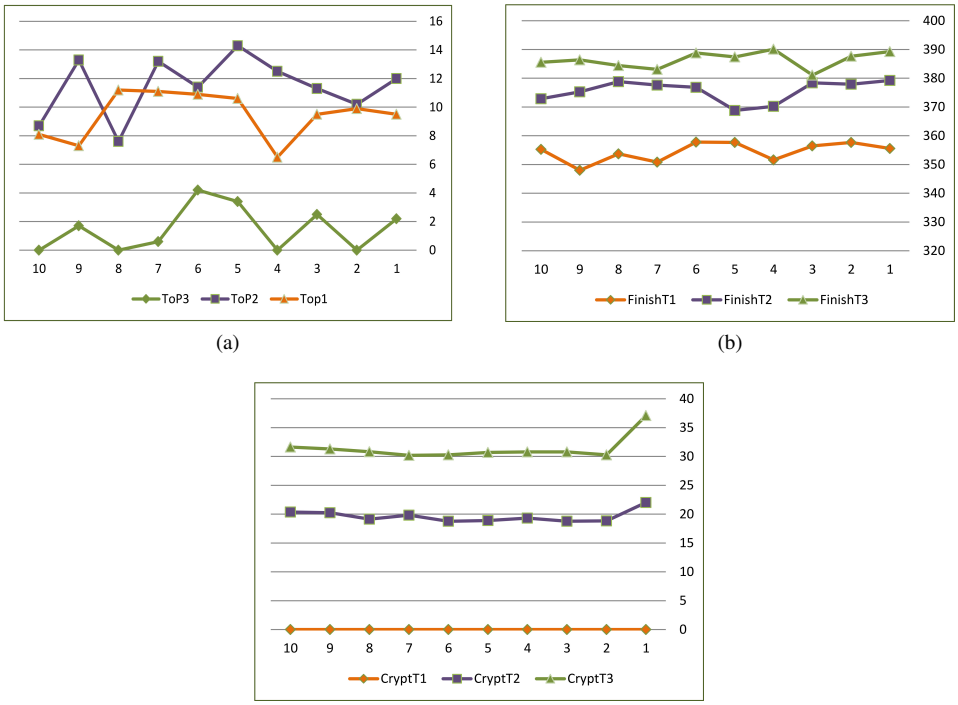



Fig. 9: (a) Comparison penalty cost in three trials; (b) Comparison finish time in three trials; (c) Comparison crypto time in three trials

in cloud environment," in *IEEE 27th International Conference on Data Engineering*, pp. 87–98, 2011.

[25] B. R. Wu L., "Service level agreement (sla) in utility computing systems," *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*, p. 25, 2012.

[26] R. Buyya, S. K. Garg, and R. N. Calheiros, "Sla-oriented resource provisioning for cloud computing: Challenges, architecture, and solutions," in *2011 International Conference on Cloud and Service Computing*, pp. 1–10, 2011.

[27] H. M. Nazari Cheraghlou M., Khadem-Zadeh A., "A survey of fault tolerance architecture in cloud computing," *Journal of Network and Computer Applications*, vol. 61, pp. 81–92, 2016.

[28] S. M. Jhawar R., Piuri V., "A comprehensive conceptual system-level approach to fault tolerance in cloud computing," in *IEEE International Systems Conference SysCon 2012*, pp. 1–5, 2012.

[29] M. R. K. I. Zheng Z., Zhou T., "Ftcloud: A component ranking framework for fault-tolerant cloud applications," in *IEEE 21st International Symposium on Software Reliability Engineering*, pp. 398–407, 2010.

[30] S. K., "A combined approach to ensure data security in cloud computing," *Journal of Network and Computer Applications*, vol. 35, pp. 1831–1838, 2012.

[31] W. P. Y. E. A. M. S. B. K. C. Kuan Lu., Yahyapoura R., "Fault-tolerant service level agreement lifecycle management in clouds using actor system," *Future Generation Computer Systems, Elsevier B.V*, vol. 54, pp. 247–259, 2016.

[32] H. Liang, D. Huang, L. X. Cai, X. Shen, and D. Peng, "Resource allocation for security services in mobile cloud computing," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 191–195, 2011.

[33] Y. Y. YESILYURT M., "New approach for ensuring cloud computing security: using data hiding methods," *Sādhanā(Indian Academy of Sciences)*, vol. 41, p. 1289–1298, 2016.

[34] W.-T. Tsai, X. Sun, and J. Balasooriya, "Service-oriented cloud computing architecture," in *Seventh International Conference on Information Technology: New Generations*, pp. 684–689, 2010.

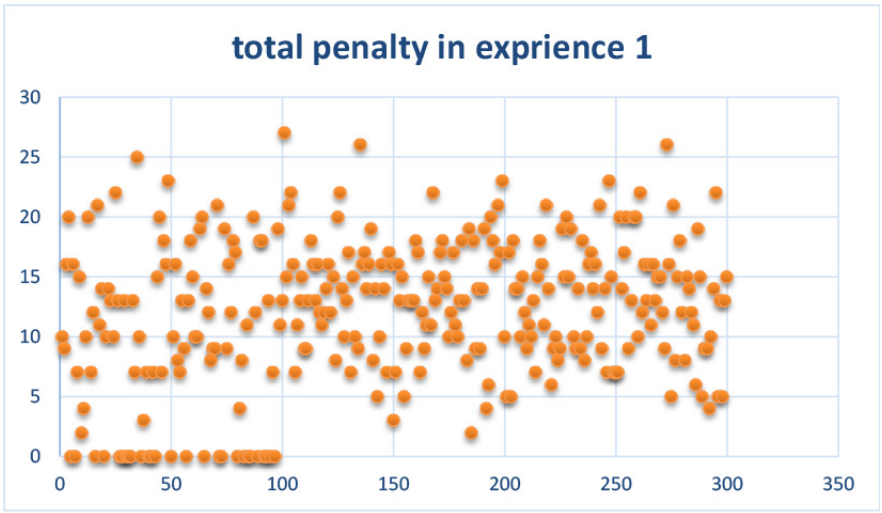
[35] E. B. Sanjuan, I. A. Cardiel, J. A. Cerrada, and C. Cerrada, "Message queuing telemetry transport (mqtt) security: A cryptographic smart card approach," *IEEE Access*, vol. 8, pp. 115051–115062, 2020.

[36] K. A. Rezaei A., Broumandnia A., "new solution for keeping sla in cloud service oriented," master's thesis, azad university Qazvin branch, 2014.

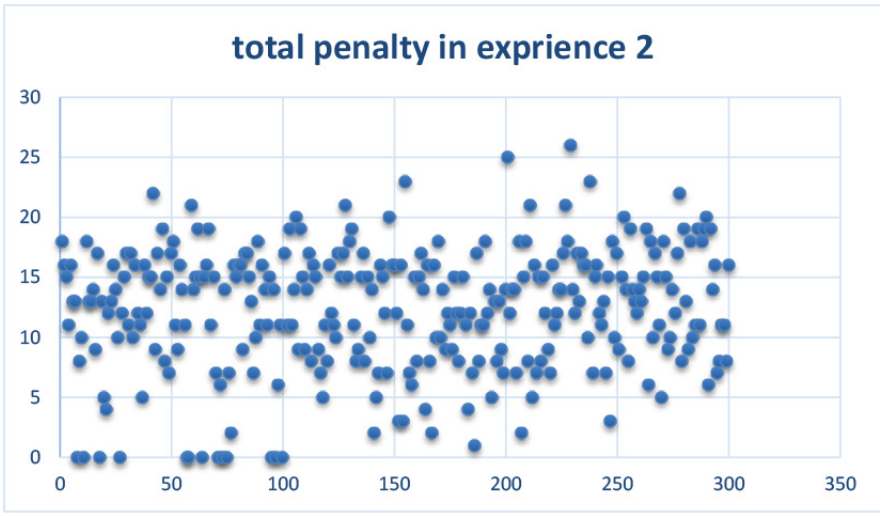
[37] D. R. C. B. R. Calheiros R., Ranjan R., "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," Tech. Rep. echnical Report, GRIDS-TR-2009-1,, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, 2009.

[38] B. A. D. R. C. B. R. Calheiros R., Ranjan R., "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *SOFTWARE – PRACTICE AND EXPERIENCE*, vol. 41, pp. 23–50, 2010.

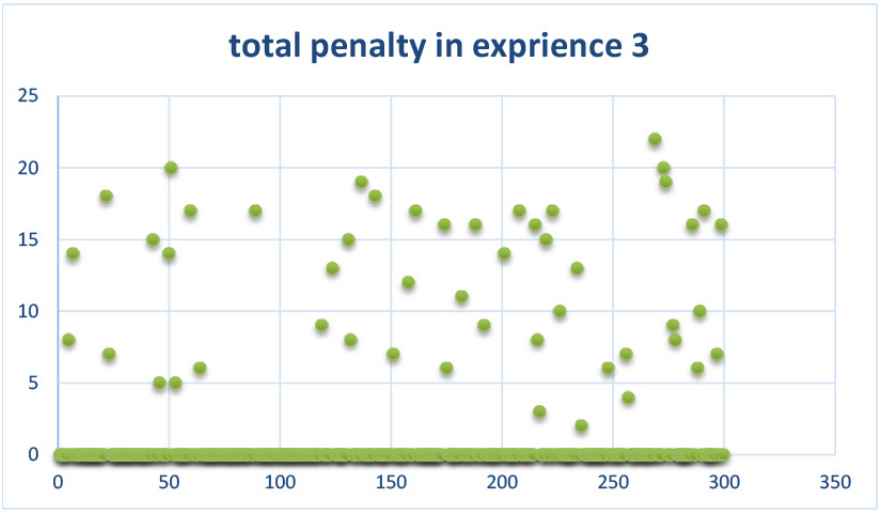
[39] R. C. B. R. Calheiros R., Netto M., "Emusim: An integrated emulation and simulation environment for modeling, evaluation, and validation of performance of cloud computing applications," *Software:Practice and Exprience*, vol. 43, pp. 595–612, 2013.




(a)




(b)




(c)

Fig. 10: scatter plot comparison for penalty cost: (a) experience 1, (b) experience 2 and (c) experience 3