I J C T A, 10(9), 2017, pp. 931-948 ISSN: 0974-5572

Domain Specific TPA Trust Score Evaluation in Cloud Based Services

K. Shirisha* and M. Bal Raju**

ABSTRACT

In the field of IT, to maintain applications and data most of the organizations are moving towards cloud computing or migrating from traditional client server nature. Most of the data owners are new to cloud computing or not having own facilities to manage data, so data owners are depending on TPA. TPA generated report lets data owners to evaluate usage of the by clients. Most of the owners are depending on public TPA where private TPA is most expensive. Here major challenging issue is how to find trustiness of TPA among available list. If data owners has facility to find the best TPA with good trust score, then owners has chance to setup risk free TPA for their data. There is a difference between services provided by available TPA list for example supporting domains, nature of authentication, mode of encryption etc… First time this paper introducing a mechanism to find trustiness of TPA before committing to subscribe, where trustiness of TPA can be calculated based on different criteria. In this work three new algorithms are defined named as DRM, TCM and CDM.

Index Terms: Cloud Computing, Third Party Auditor, TPA Trust Score, TPA services

1. INTRODUCTION

Cloud computing clean sweeps old computing server structure. In a simple words cloud computing is an environment to provide services in a rental model. There are different types of cloud computing services are available for example providing hardware infrastructure as rental base, providing platforms and providing usage of applications in a rental base. Different IT companies from startups to giant companies are being as Cloud Service Providers (CSP). Cloud users can get different types of services from CSP like infrastructure as a service (IAAS), platform as a service (PAAS), software as a service (SAAS) etc…Major reason to get attraction from more users is due to the reason where cloud is inexpensive and instant setup of computing environment. In daily activities of the cloud applications, vast amount of data is storing in cloud storage. Major issue is privacy and security for the data stored in remote cloud storage media.

Data owners are allowing other individual users or organizations to consume their data to manipulate. For example consider a situation where one retails industry based organization wants to allow their sales data to out sourcing to get reports on sales. So in this case data must be consumed by our sourcing company. In this mentioned situation data owners need an inspector between their data and out sourcing company. This data inspector is none other than TPA. TPA lets data owners to be burden free from management of data outsourcing. Auditing reports helps data owners to take decision on the usage of data, so that internal and external attacks can be predicted and protected data.

TPA is helping data owners of different domains to inspect utilization of sensitive data stored in remote cloud storage. Following figure list some popular domains where TPA usage is required.

Based on literature survey done in cloud auditing, so many researches has been done on data possession, data integrity, dynamic auditing, user identity privacy and some on multi-cloud auditing. Here TPA acts as

* Reddy Research Scholar, Computer science and Engg, JNTUH Hyderabad, India, *Email: Shirishakasireddy20@gmail.com*

** Professor and Principal Computer science and Engg KITE, Hydearabad, India, *Email: principal@kite.edu.in*

a security auditor to assess system utilization from internal sources and external sources. As the nature of inexpensive cloud pricing data owners are interested to use different cloud services to store/ manage their data. In this way to support with security auditing for different clouds data owner has to depend on different TPA Services where excising TPA system is not supporting security auditing in Heterogeneous clouds. At the same time no research is focusing on trustiness of TPA apart from data security. What are the units to trust TPA to handle security auditing of data from data clients. At the same time data owners may not have technical knowledge on security levels and requirements. So it is needed a system to predict and suggest security stands for data owners based on requirement.

This proposed system helps data owners to get a chance to know the trustiness of TPA before committing and facilitate the following three key characteristics

1) TPA Trustiness
2) Predicting Security Requirements
3) Effective auditing in heterogeneous cloud

In the first point of proposed system we are focusing on the technique to get trust score of TPA. Data Owner can get complete trust on their TPA for security auditing to ensure fully assessment on data and avoid security incidents. In the second point we can facilitate intelligent security suggestions to novice cloud users with easy to manage security stands with different domain data and supports security with integration of data. In the third point we can supports security auditing with in heterogeneous multi-cloud to handle different encryption systems and supports homomorphic security. In this paper, our main contribution includes:

1) Health Industry

Operations
- Sharing patient information
- Accessing health records
- Accessing device details
- Other sensitive data

Stakeholders
- Hospitals
- Doctors, Patients
- Offices, Stores
- Medical Representative

Problems
- Loss of data
- Modification of treatment
- Misuse of patients data
- Lack of transparency
- Security and Privacy issues

2) Financial

Operations
- Transactions
- Account Openings
- Lending Operations
- Payments

Stakeholders
- Banks
- Managers
- Tellers
- Account Holders

Problems
- Loss of data
- Modification of Balance
- Misuse of accounts holders data
- Visibility of Transactions
- Security and Privacy issues

3) Government

Operations
- Sharing citizens information
- Accessing schemes data
- Voters data
- Other sensitive data

Stakeholders
- Subsidy teams
- Collectors
- MRO offices
- Local Panchayathi team

Problems
- Loss of data
- Modification of payments
- Voters data privacy
- Lack of transparency
- Security and Privacy issues

Figure 1: List of domains using TPA in cloud

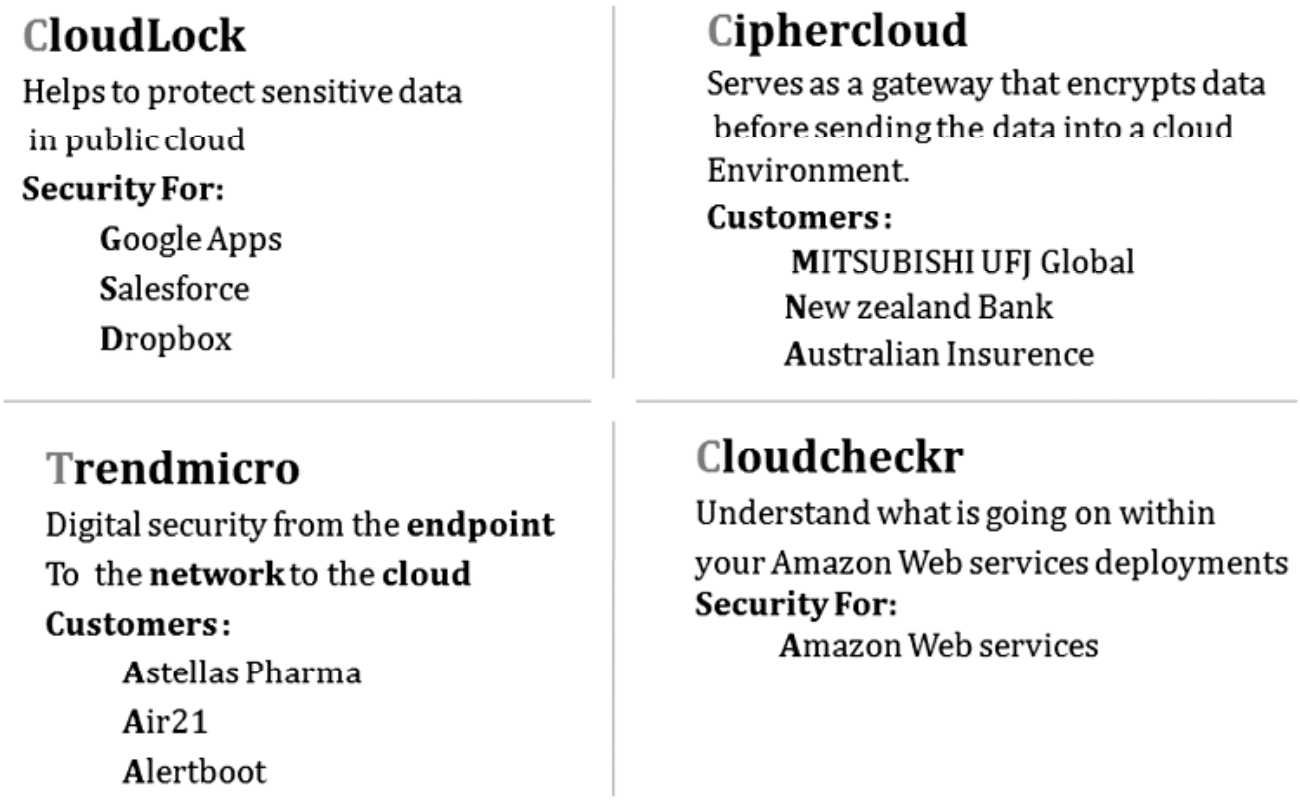


Figure 2: Examples for TPA

1) We develop new vector based matrix called DRM (Domain Relation Matrix), which specifies relation between TPA (Third Party Auditor), CSP (Cloud Service Provider) and Domains. Each element of DRM is a vector.
2) We Develop TCM stands as Trained Configured Matrix, derived by extending DRM, by attaching predefined metrics.
3) We derived CDM tree which specifies pair combination of CSP, Domain and Metric. CRM tree is merging of Level trees.
4) We calculate trust score, with an intersection of TCM and CDM of given testing data.
5) We defined Trust Meter, with the collection of Configuration Metrics and its supporting values in Boolean format.

Rest of the paper organized as given bellow sections: Section II described related work with respect to TPA in cloud. Section III & IV gives detailed explanation of proposed system. Section V described about evaluation of result, followed by Conclusion in Section VI.

2. RELATED WORK

Many of the previous works focused on only implementation of TPA but not including trustiness of TPA performance. One of the previous works “PDP for confirming possession of data files on untrusted storages” [1] it ensure possession of data files on untrusted storages. It is based on the RSA-based homomorphic linear authenticators [1]. It is based on sampling few blocks of data files [1]. “PORs: Proofs of Retrievability for Large Files” [2] it ensure both possession and retrievability of data files on remote archive service systems. It is based on spot-checking, error-correcting codes and back-up service [2]. It encrypts F and randomly embeds a set of randomly-valued check blocks called sentinels in this work main problem is “public auditability is not supported in their main scheme” [2]. “Privacy-Preserving Public Auditing for

Data Storage Security in Cloud Computing" [3] ensure dynamic auditing system to be privacy preserving, is focusing on public auditability, storage correctness, privacy-preserving and batch auditing [3]. Main problem in [3] is due to the large number of data tags, their auditing protocols will incur a heavy storage overhead on the server [3]. "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing" [4] supports public auditability and data dynamics, which is focusing on public auditability for storage correctness assurance, dynamic data operation support and blockless verification [4]. This method may leak the data content to the auditor because it requires the server to send the linear combinations of data blocks to the auditor [4].

"Efficient audit service outsourcing for data integrity in clouds" [5] replaces the traditional Hash-based solution, which is based on probabilistic queries and periodic verification [5]. This method may not worry about trustiness of TPA, due to periodic verification there is heavy computational cost [5]. "Storing Shared Data on the Cloud via Security-Mediator" [6] generates security-mediator (SEM), which is able to generate verification metadata (i.e., signatures) on outsourced data for data owners, which is based on attachment of signature to original data, verification of the signature and based on hash signature [6]. "Identity-Based Distributed Provable Data Possession in Multi-Cloud Storage" [7] is identity-based public key cryptography can eliminate the complicated certificate management, which is based on a Bilinear pairings, Tag system, Signature and Distributed computing [7]. My observation is that none of the previous works are focusing on trusting TPA, and it is not focusing on risk prediction with auditing intelligence.

3. PROPOSED SYSTEM

This section describes key functional components of the proposed system to calculate trust score of TPA. Our proposed method almost the very first one (to the best of our knowledge), to calculate trust score of TPA. Following are main key components in this proposed scheme.

1) Domain Relation Matrix
2) Trained Configure Matrix
3) CDM Tree
4) Trust Meter

3.1. Domain Relation Matrix

In this section, we present the detailed explanation of "Domain Relation Matrix", which can be represented as DRM in next coming sections. DRM is a dynamic matrix which defines the relation among TPA, CSP and Application Domains. Relation in DRM is a service providing by TPA with respect to CSP.

Let $TPA_N = \{TPA_1, TPA_2, - - - TPA_N\}$ is a set of registered TPA's. Where TPA_i is i^{th} TPA, $i \leq N$.

$CSP_J = \{CSP_1, CSP_2, - - - CSP_J\}$ is a cumulated set of CSP's, which are supported by any one of the registered TPA.

$D_K = \{D_1, D_2, - - - D_K\}$ is a set of domains supported by any one of the registered TPA on behalf of CSP.

We can represent set of available services as S, which defined as bellow

$$(TPA \times CSP \times D) \in S \quad (1)$$

Where S defined as list of available services, means TPA_i is providing service to CSP_j for domain D_k can be represented as $(TPA_i \times CSP_j \times D_k) \in S$. DRM contains TPA as columns and supporting CSP's as first row proceeding by rows which defined list of domains supported by TPA for its supported corresponding CSP. Each element of DRM is a vector, where vector consists CSP's or domains as elements. Size of DRM is $(J+1) \times N$, where J is total CSP's and N is total TPA's.

TPA's →	TPA1	TPA2	TPA3
CSP Vector	CSP1 CSP2 CSP3	CSP1 CSP3	CSP2 CSP3
CSP1	D1 D4	D1 D2 D3 D4	Null
CSP2	D2 D3 D4	Null	D1 D2
CSP3	D1 D2 D3 D4	D2 D3	D1 D3 D4

Figure 3: DRM Matrix

For better understanding of DRM element placement, we consider following situation, where 2nd TPA is supporting service for 1st CSP and for 2nd domain then DRM element is represented as ($TPA_2 \times CSP_1 \times D_2$) ∈ S, index of element will be (2, 2), means ($TPA_2 \times CSP_1 \times D_2$) ∈ S is in second row and second cell, so that $DRM_{(2,2)} \neq$ null. Deep element of DRM includes domain position into element index. In considered DRM example, deep element position of D2 for TPA1 and CSP1 is (2, 1, 2). Deep Element Value (DEV) defined as

$$DEV_{(n,j,k)} = \begin{cases} 1, & if\ DRM_{(j+1,n)} \neq null\ and\ D_k \in DRM_{(j+1,n)} \\ 0, & Otherwise \end{cases} \quad (2)$$

$DEV_{(n,j,k)}$ is 1 means TPA_n is supporting CSP_j, and is supporting D_k, $DEV_{(n,j,k)}$ is 0 means D_K is not supporting by TPA_n. DEV is useful to find out whether domain is supporting by TPA and its CSP, or not.

Element Value (EV) is useful to find out whether CSP is supporting by TPA or not. EV can be defined as

$$EV_{(n,j)} = \begin{cases} 1, & if\ DRM_{(j+1,n)} \neq null \\ 0, & Otherwise \end{cases} \quad (3)$$

$EV_{(n,j)}$ is 1 means TPA_n is supporting CSP_j, is 0 means TPA_n is not supporting CSP_j. DEV is useful at the time of Trust Score calculation, when input includes TPA, CSP and Application Domain, EV is useful when input includes TPA and CSP. To calculate Trust Score DEV or EV must be 1 for given input.

DRM is dynamic, when new TPA is registering or new CSP is adding or new Domain is supporting. At the same time if any one of the triplet is deleting then DRM is also be dynamic. Following algorithms describes how to manage DRM.

1) Create DRM
2) Update DRM
3) Visit Element

3.1.1. Algorithm: CreateDRM(TPA_N, CSP_J, D_K)

This algorithm creates DRM by taking S, TPA_N, CSP_J and D_K inputs. DRM can be defined as function f(n,j,k) which notated as

$$DRM \rightarrow f(n, j, k) = \begin{cases} 1, & if \ \left(TPA_n, CSP_j, D_k\right) \in S \\ 0, & Otherwise \end{cases} \quad (4)$$

Where f(n,j,k) is defines DEV of DRM at index (n,j,k).

DRM algorithms uses following functions, these functions are reusable functions while creating DRM, updating DRM and Visiting Element in DRM.

Funtion1: getS (TPA_N, CSP_J, D_K)

This function getS is useful for creating set of available services based in available TPA list, CSP list and Domains list.

Let $T_n(CSP)$ is set of CSP's supported by TPA_n &
$T_n(CSP_j, D)$ *is set of Domains support by* TPA_n *and* CSP_j

1: for each TPA_n perform step 2, $1 \le n \le N$
2: for each CSP_j perform step 3, $1 \le j \le J$
3: if $CSP_j \in T_n(CSP)$ perform step 4
4: for each D_k perform Step 5, $1 \le k \le K$
5: if $D_k \in T_n(CSP_j, D)$ then add (TPA_n, CSP_j, D_k) to S
6: return S

Function 2: getDElement(n,j)

This function is useful to get domain vector element for DRM. This function returns vector of Domain id's. Maximum size of element vector is equal to total domains quantity.

1: Let i = size ($DRM_{(j+1, n)}$), ie size of $DRM_{(j+1, n)}$, initially i is 0
2: Let E is $DRM_{(j+1, n)}$ & E is null initially
3: for each D_k
4: if (TPA_n, CSP_j, D_k), $\in S$,
   add D_k to E at position i
   i++
5: Return E

Function 3: getCSPElement(n)

This function returns vector of CSP's supported by TPA_n

1: Let i = size ($DRM_{(1, n))}$, initially it is 0
2: Let E is $DRM_{(1, n)}$ & E is null initially
3: for each CSP_j
4: if $CSP_j \in T_n(CSP)$, add CSP_j to E at position i, i++
5: return E

Algorithm 1: createDRM(TPA_N, CSP_J, D_K)

1: for each N perform step 2 to Step 6

2: for each J perform step3 to step4

3: $E_{NJ} \leftarrow$ getDElement(N,J)

4: add E_{NJ} to (J+1, N) position of DRM, can be represented as $DRM_{(J+1, N)}$

5: $ECSP_N \leftarrow$ getCSPElement (N)

6: add $ECSP_N$ to (1,N) position of DRM, can be represented as $DRM_{(1, n)}$

7: return DRM

Algorithm 2: updateDRM(DRM, TPA, CSP, D)

Let input DRM is an existing DRM

Let input TPA $\in$ TPA_N or TPA is new TPA

Let input CSP $\in$ CSP_J or CSP is new CSP

Let input D $\in$ D_K or D is new Domain D

1: Let DRM current size is a × b, where DRM has total b TPA's and (a-1) CSP's

2: if CSP is new & TPA $\in$ TPA_N

  2.1: add CSP_{new} to CSP_J & to TPA (CSP), let input CSP is CSP_{new}

  2.2: ECSP $\leftarrow$ getCSLElement(TPA)

  2.3: replace ECSP at $DRM_{(1, n)}$, n is index of current TPA in set of Total TPA's

3: else

  3.1: if CSP $\in$ CSP_J & TPA $\in$ TPA_N

    3.1.1: add CSP to TPA(CSP)

    3.1.2: ECSP $\leftarrow$ getCSPElement (TPA)

    3.1.3: replace ECSP at $DRM_{(1, n)}$

  3.2: if CSP is new & TPA is new

    3.2.1: a++, b++

    3.2.2: add CSP_{new} to CSP_J

    3.2.3: add TPA_{new} to TPA_N

    3.2.4: createDRM(TPA_N,CSP_J,D)

4: if D $\in$ D_K

  4.1: E $\leftarrow$ getDElement(n,j) , where n is current TPA and j is Current CSP

  4.2: replace E at $DRM_{(j+1, n)}$

5: if D is new

  5.1: add D to DK

  5.2: E $\leftarrow$ getDElement(n,j), where n is current TPA and j is Current CSP

  5.3: replace E at $DRM_{(j+1, n)}$

6: return new DRM

Algorithm 3: visitElement(N,J,K)

This algorithm is useful to visit two types of elements (DRM element, Deep element)

1: set K=0 to visit DRM element

2: start from $(j+1)^{th}$ position in DRM

3: E ← getDElement(n,j)

4: if sizeOf(E)=0

return null

5: else

return E

6: if K ≠ 0, visiting Deep element

7: get position of K in E

8: if position (K) in E is not 0

return position(k)

9: else

return null

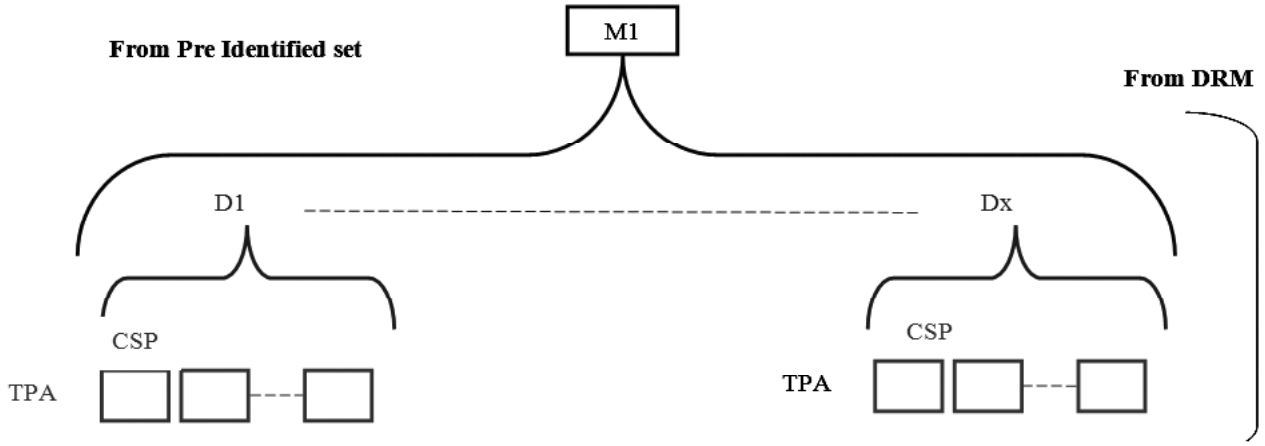


Figure 4: TCM element

3.2. Trained Configure Matrix

TCM stands Trained Configured Matrix. DRM specifies only relation among TPA, CSP and Domains. To calculate Trust Score, metrics needs to include in testing data and training data. With respect to each domain, there must be set of pre identified metrics, where metric is a service property, for example DM[geo]=T means application domain has to support geographical based accessibility of application, DM(geo)=F means not supporting geographical based accessing. Let TDM(geo)=T means TPA T is supporting geographical location service for Domain D, if DM=TDM then one trust score can be increased. Overall trust score described as following

$$TS\left(T_n C_j D_k\right) = \sum_{i=1}^{M} f\left(n, j, k, i\right) = \begin{cases} 1, & if\ \ D_k M = TD_k M \\ 0, & Otherwise \end{cases} \tag{5}$$

$TS(T_i\ C_j\ D_k)$ is overall trust score, if input is triplet.

To calculate f(n, j, k, i), DRM is extending by including metrics. In this extending process first we calculate TCM. Total set of f(n, j, k, i) for TCM is defining as 4D-tensor. One element of 4D-tensor is showing in the figure-4.

From the figure-4 we can say that M_1 is one metric (added or pre identified), which included for Domains D1, D2….Dx.

$$D_{x_m} \ni U_{i=1}^{k} f(m) = \{D_i,\ if\ \ D_i M_m \neq null \tag{6}$$

The above function defines metrics set supported by domain D_x, D_{x_m} means metric m is defined/supported by domain D_x.

Total elements in TCM is equal to

$$\sum_{m=1}^{M} size\ of\ (DM_m) \tag{7}$$

Where M is total size of metrics defined/pre identified for finding optimal trust score.

3.3. Detailed Explanation about TCM element

TCM elements are placing as vertical vector as show in the following figure

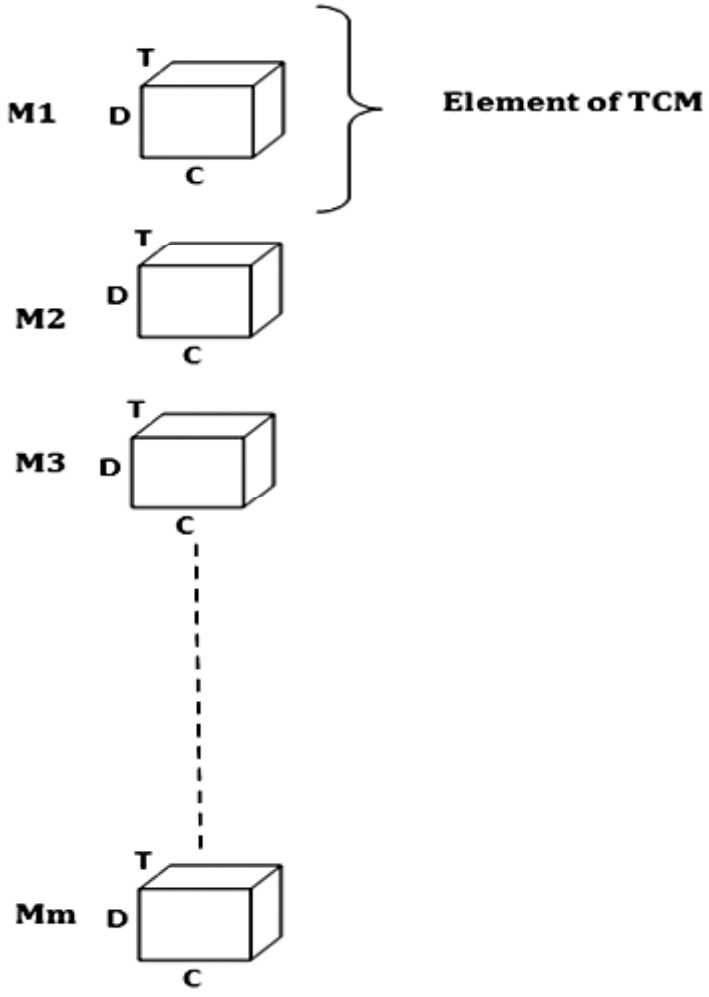


Figure 5: TCM

In TCM element structure, relation between metric and domains are predefined, but relation between TPA and CSP defined by TPA administrator. While creating TCM element, TPA defined metrics can't be considered.

D_{x_m} can be represented as matrix to navigate easily, where elements of D_{x_m} matrix is T or F null. Size of D_{x_m} matrix is |D| × |M|, ie K × M, where "K" total domains and M total metrics.

	M1	**M2**	----------	**M_M**
D_1	T	F	----------	T
D_2	Null	T	----------	T
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
*	*	*	*	*
D_k	F	F	----------	null

Figure 6: D_{x_m} Matrix

If cell value is "null" means, that corresponding metric is not applicable for domain, for example if "null" is at (3, 4) position in D_{x_m} matrix, then we can say metric four is not supporting/applicable to domain three. If cell value is "T", that means domain 3 has to support metric 4 to get positive trust score, if cell neither value if "F", that means domain 3 must not support to get positive trust score. But in the matrix null means, that particular metric is not applicable to that domain.

Creating TCM is integration of DRM & D_{x_m}. f(n, j, k, m) defines an individual element relation among pre identified/defined relations between T, C, D and M.

$TCM \ni f(n, j, k, m)$ which can be defined as

$$\begin{cases} 1, & if \quad D_{k_m} \neq null \\ & \quad and \\ & \quad DRM_{(j+1,n)} \neq null \quad jm \\ 0, & Otherwise \end{cases} \tag{8}$$

Algorithm-4: createTCM(DRM ,)

TCM defines optimal relation defined between registered TPA's and predefined .

1: Let M is set of Metrics

2: D is set of Domains

3: for each metric m=1 to M execute step 4

4: for each Domain k=1 to K execute step5

5: if execute step 6
6: for each TPA n=1 to N execute step 7
7: for each CSP j=1 to J execute step 8 to step 10
8: $E_{jn} \leftarrow get\ D\ Element\ (n, j)$
9: if $E_{jn} \neq null$ execute step10
10: $f(n, j, k, m) = 0$ and
add $f(n, j, k, m)$ to TCM
11: return TCM

3.3. Trained Configure Matrix

To calculate testing TPS trust score, instead of considering all elements of DRM, we are creating CDM tree, where CDM stands combination of CSP, Domain and Metric. Testing TPA may support more than one CSP, so that CSP tree may has more than one root node.

3.3.1. Characteristics of CDM

Following are main key characteristics of CDM to reduce size of DRM to use

1) Root node always will be CSP supported by testing TPA
2) CDM tree always has three levels
3) Child node will be always metric, which is supporting by TPA with respect to CSP and Domain
4) Traversing always starts from Root node to Domain node to Metric node

Fig 7: is an example for CDM tree for TPA which is supporting two CSP's that are C1 and C2.

3.3.2. CDM tree

CDM tree is a part of DRM, by adding metrics at the end of tree, as child nodes. To crate CDM tree we are extracting Level12 tree from DRM for a given testing TPA, and extracting Level 23 tree from TD_{X_M} **for** supported domains of TPA, where TD_{X_M} is a set of metrics supported by Domain D_x in testing TPA.

$$TD_{X_M} \ni f\left(T, D_x, M_m\right) = \begin{cases} T, & if TD_{x_m} \neq null \\ & and \\ & metric\ m\ is\ supporting\ by\ TD_{x_m} \\ F, & if\ TD_{x_m} \neq null\ and \\ & metric\ m\ is\ not\ supporting\ by\ TD_{x_m} \\ & where\ 1 \leq x \leq K\ and\ 1 \leq m \leq M \end{cases}$$

3.3.3. Creating Level12 tree

Level12 tree can be created for given testing TPA. Let us consider Testing TPA is , where I stands for Input. Let ID of TPA_I is "i". Then to create Level12 tree, we need to use DRM. There may be more than one root node for Level12 tree, always Leve12 tree has CSP as root node & Domain as child node as shown in the fig-7.

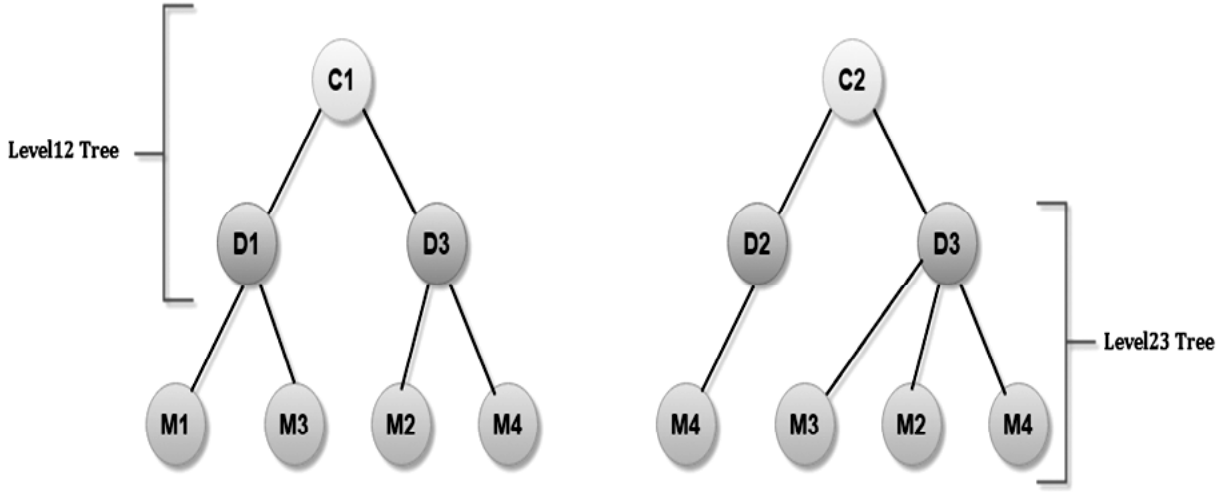


Figure 7: CDM Tree

Let us consider is a set of root nodes for i^{th} TPA, where i^{th} TPA is a testing TPA.

$(RL12)_i$ = Element at DRM(i, j) index position.

In $(RL12)_i$ CSP id may not be equal to index value of CSP, means in position 1 of $(RL12)_i$ there may be CSP 2, and if CSP 4 is in position 2, if CSP 3 is in position 3, then diagrammatical representation of $(RL12)_i$ is shown bellow figure.

Red color values are representing CSP id's and black color value is representing index of CSP element in $(RL12)_i$.

Figure 8: Element of DRM

Each element of $(RL12)_i$ is a root node for Level12 tree. Let size of $(RL12)_i$ is R, so that Level12 tree has R root nodes, based on each root node, there may be each Level12 trees.

Root node of Level12 defined as bellow

$$[RN]_{ir} = \text{Element } [1, r] \in (RL12)_i$$

Where 'r' is the position of element in $(RL12)_i$, which denotes r^{th} root node for Level 12 tree.

$[RN]_{ir}$ is CSP, which is supported by testing TPA. Value of $[RN]_{ir}$ is representing ID of CSP, To find supporting domains by $[RN]_{ir}$ (ie CSP) need to get DRM element at $[RN]_{ir}$ + 1, i].

$[RN]_{ir}$ is root node and DRM ($[RN]_{ir}$ + 1, i) are child nodes.

Algorithm-5: hcreateLevel12Tree(TPA_i)

TPA_i is a testing TPA

1: get [RL12]i ← DRM[1, i]

2: for each element of [RL12]i execute step 3

3: get [RN]ir ← [RL12]i (1, r)

4: Add [RN]ir as root node to $[\text{Level } 12]_r$
(this is r^{th} Level12 tree)
5: get DRM ($[RN]_{ir}$ + 1, i) $\rightarrow$ DEV
6: for each DEV element execute step 7
7: get DEV[i,d], d is index of DEV and
add this as a child node to $[\text{Level } 12]_r$
8: return $[\text{Level } 12]_r$

3.3.4. Level23 tree

Function TD_{X_M} is using to create Level23 tree, in this tree Domains are root nodes, metrics are child nodes. First we have to get CSP's and domains list supporting by TPA. One domain may be supported by more than one CSP.

$[RL12]_i \leftarrow DRM[1, i]$ is set of CSP's supported by TPA and $DRM[([RL1]_i, (1, r)) + 1, i]$ is set of domains supported by *CSP* $[RL12]_i(1, r))$ of TPA i.

Let TD is set domains supported by TPA i. TD can be defined as bellow

$$TD \ni \bigcup_{r=1}^{r=sizeofDRM[1,i]} DRM\left[\left(\left[RL12\right]\right)_i(1,r)+1,i\right]$$

Let M is metrics supported by TPA i then relation among domains TD and Metrics M defines as

$$TDM \leftarrow \overline{TD \oplus M}$$

Algorithm-6: crateLevel23Tree(I, TDM)

1: Let TD is a set of Domains supporting by TPA_i
2: get $[RL12]_i$
3: for each element of $[RL12]_i$
4: let element of $[RL12]_i$ is $[RN]_{ir}$
at r^{th} position in $[RL12]_i$
5: get Domain vector for $[RN]_{ir}$

$$\left[RN\right]_{ir} \leftarrow DRM\left[\left[RN\right]_{ir}+1,i\right]$$

6: for each $[RN]_{ir}$
7: Let $D \leftarrow [RN]_{ir}(i, d)$
8: if $D \notin TD$ then add D to TD
9: Let M is a set of Metrics supporting by TPA_i
10: $TDM \leftarrow \overline{TD \oplus M}$ where $\oplus$ is resulting T or F
11: return TDM

Relation of domains and metrics are not related to CSP, because here TPA trust score is calculating, so that we can consider Relation of Domains and metric with respect to TPA only.

If input includes C1 as CSP then we can consider CSM tree which has C1 as root node. If input is triplet then we can navigate like C-D-M, in this case Boolean matrix contains one row and |M| columns. If input

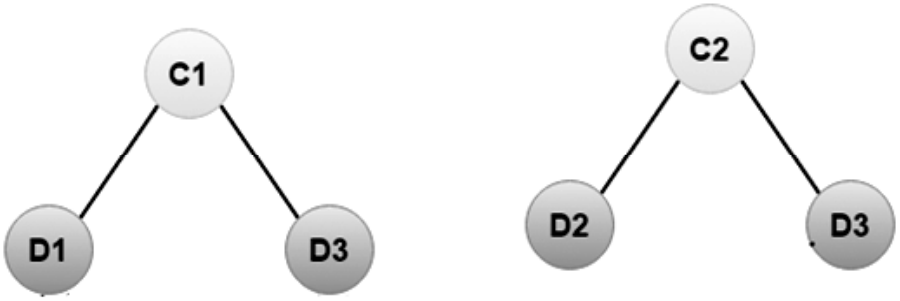


Figure 9: Level12 Tree

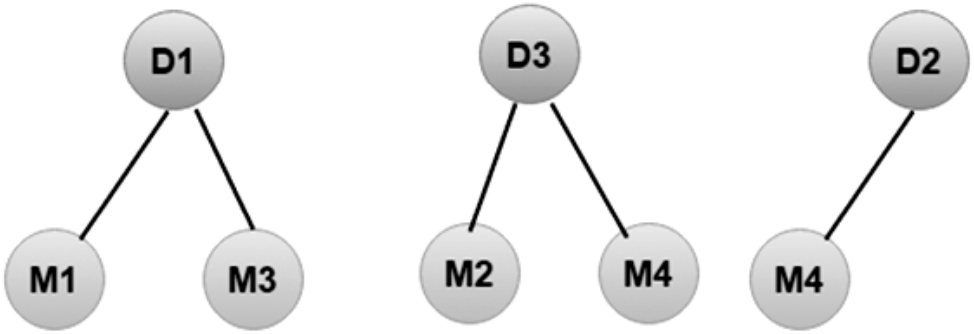


Figure 10: Level23 Tree

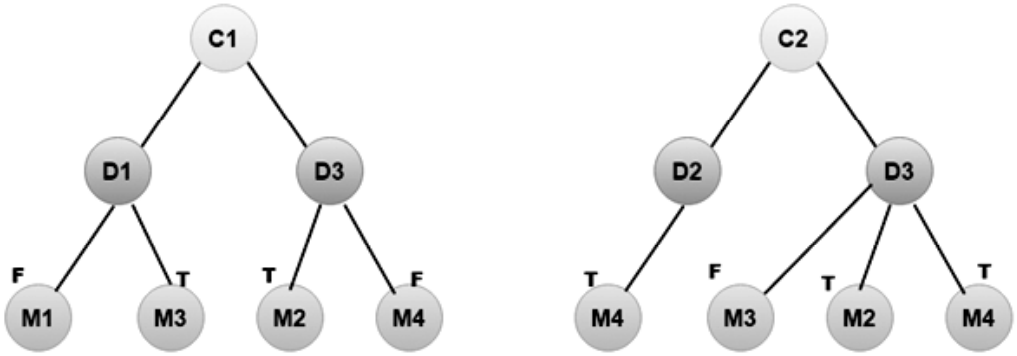


Figure 11: Union of Level Trees

	M1	M2	M3
D1	T	F	T

a: Case 1

	M1	M2	M3
D1	T	F	null
D2	T	T	T
D3	null	F	T

b: Case 2

Figure 12: Cases of CMD tree navigation

includes T & C then navigation in CDM tree will be C-M by omitting D in traversing path. In this case rows count will be |D| supporting by testing CSP and |M| columns, which are cumulated metrics of all domains supporting by testing TPA.

4. CALCULATING TRUST SCORE

Calculating trust score is an intersection of TCM and CDM of given testing data. TCM includes pre identified metrics, where CDM contains testing TPA supported metrics. In this intersection process, we need to get f(i,c,d,m) where m=1 to M, M is size of metric pre identified for optimal performance of TPA for domain d.

TCMElement*(i,c,d) is a part of TCM for given testing TPA I, CSP c and Domain d. TCM is a 4D-tensor, where TCD is triplet for metric view of TCM. Decomposition of tensor is using to get Trained Boolean Matrix for testing TPA. Trained Boolean Matrix has to contains domain & metric relations to current testing TPA, CSP and Domain. TCM has metric as super element, & DRM portion of current TPA as inner element, TBM (Trained Boolean Matrix) can be defined as following

$$TBM(i,c,d) \leftarrow \bigcup_{t=1}^{d} e(r, M_r) = \{TCM(i,c,d,m) \, where m = M_r$$

TBM(i,c,d) contains only current testing Domain & metrics relation.

Algorithm-7: CreateTBM(i,c,d)

Let current testing TPA is TPAi

Let current testing CSP is CSPc

Let current testing domain is Dd

1: get list of metrics supported by Dd,

let it is $D_{d_m} \leftarrow$ is set of metrics

2: Let TBM is null

3: for each element of D_{d_m} perform step 4 to 6

4: if $(D_{d_m} \in TCM)$

$C_{DRM} \leftarrow$ *get DRM of current* D_{d_m}

5: if $C_{DRM} \neq null$

6: Add value of D_{d_m} (*it is T or F*)

at TBM (d,m) position

7: return TBM

B3M is useful for calculating Trust Score by comparing TBM with CDM tree

Algorithm-8: calculationTS(TBM, CDM)

1: Let TS=0

2: for each element of CDM perform step 3 to 5

3: CDM (d, m) ← current CDM element,
which is at d^{th} row and n^{th} column.

4: get values of CDM (d, m)
V(VDM(d,m)) ← T or F

5: if V(VDM(d,m)) = $TBM_{d,m}$ then do TS++

6: return TS

5. EVALUATION OF RESULTS

Proposed scheme evaluated with respect to accuracy while calculating Trust Score of TPA. We have used notation based inputs for testing this mechanism by developing Java based Web Tool. This web tool created based on algorithms defined in this proposed system.

5.1. Trust Score Evaluation

There are different cases based on input given to calculate trust score. Cloud data owner may choose only TPA as an input or may Choose TPA + CSP or may choose TPA + CSP + Domain as input. In every case final output is trust score. To evaluate trust score we created Trust Meter. Trust Meter is a collection of Configuration Metrics and its supporting values in Boolean format. It is a B3M model , where B3M stands for Boolean Metrics Mapping Matrix.

In this model two matrix are mapping with their metric at M(i,j) to their boolean value at B(i,j), where i is index of TPA supporting metric and j is trained supporting metric. We have used Algorithm-7 to create Trained TPA Boolean Matrix. From B3M model we have to find total T values and Total F values to calculate Trust Meter Reading.

Trust Meter Reading for di = (Total True Metrics Count of Domain di/ total trained metrics of domain di).

Total Trust Meter of TPA is = Total of (di) for i=1 to n where n domains are supporting by TPA.

Following graph is comparison between the results generated based on given input, but here considered only five domains.

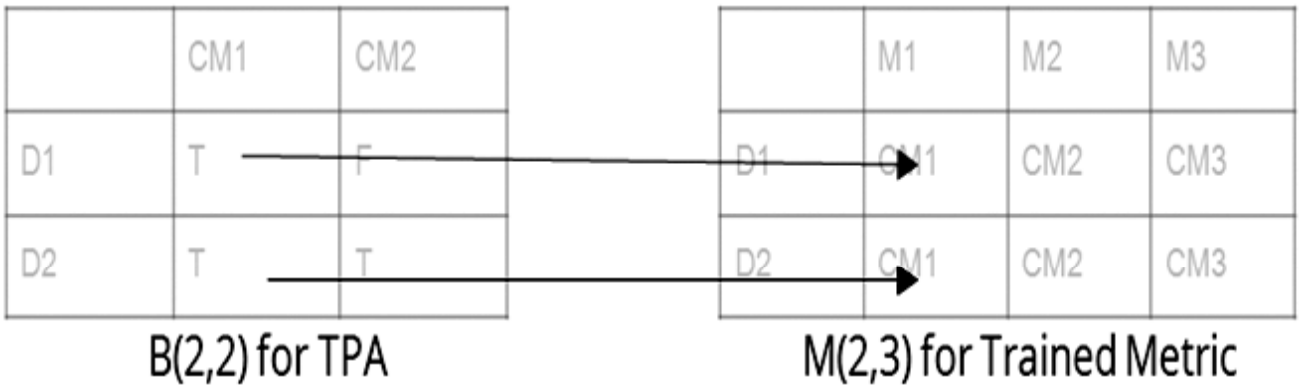
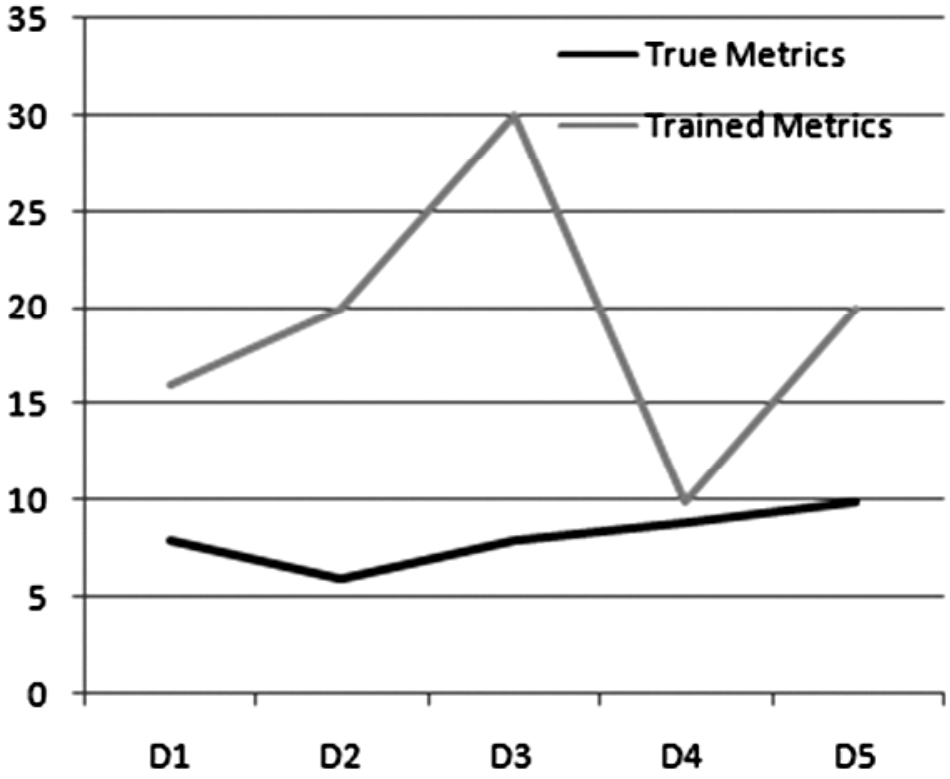


Figure 13: Boolean Mapping



Graph-1: Comparison Obtained metrics and Trained Metrics for Five domains .

6. CONCLUSION

In new era of cloud computing, many users from small to big organization are attacking by cloud environment. Data accessibility if one of the very important activity in cloud, where data is accessing by third party teams, to providing authentication to access data cloud data owners are depending on TPA, many of the works are focusing on functionality of TPA, but not on trustiness of TPA . This proposed work focusing on calculation of TPA trustiness, where cloud data owners can check trustiness of TPA before committing to use. To calculate trustiness of TPA, we have created DRM, TCM, CDM, DM and B3M matrix.

REFERENCES

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc.14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 598-609, 2007.

[2] A. Juels and J. Burton, S. Kaliski, "PORs: Proofs of Retrievability for Large Files," Proc. ACM Conf. Computer and Comm. Security (CCS'07), pp. 584-597, Oct. 2007 First Proposal on Public auditing .

[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," Proc. IEEE INFOCOM, pp. 525-533, 2010.

[4] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans. Parallel Distributed Systems, vol. 22, no. 5, pp. 847-859, May 2011.

[5] Yan Zhua,b, Hongxin Huc, Gail-Joon Ahnc, Stephen S. Yauc. "Efficient audit service outsourcing for data integrity in clouds". In "The Journal of Systems and Software 85 (2012) 1083– 1095".

[6] B. Wang, S. S. M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," in International Conference on Distributed Computing Systems - ICDCS 2013. IEEE, 2013.

[7] Wang, H., Identity-Based Distributed Provable Data Possession in Multi-Cloud Storage. At Services Computing, IEEE Transactions on (Volume:PP , Issue: 99). IEEE, 2014.

[8] H. Shacham and B. Waters, "Compact Proofs of Retrievability", Journal of Cryptology, vol. 26, no. 3, pp. 442-483, 2012.

[9] H. Chen and P. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage: Theory and Implementation", IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 407-416, 2014.

[10] M. Doshi and S. Hiray, “Developing Third Party Auditing Scheme for Secure Cloud Storage Service”, International Journal of Computer Applications, vol. 81, no. 18, pp. 1-3, 2013.

[11] “Amazon Web Services (AWS) - Cloud Computing Services”, Amazon Web Services, Inc., 2016. [Online]. Available: https://aws.amazon.com/. [Accessed: 19- Sep- 2016].

[12] K. Yang and X. Jia, “An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing”, IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 9, pp. 1717-1726, 2013.

[13] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte and J. Quisquater, “Efficient Remote Data Possession Checking in Critical Information Infrastructures”, IEEE Trans. Knowl. Data Eng., vol. 20, no. 8, pp. 1034-1038, 2008.

[14] M. Mowbray, S. Pearson and Y. Shen, “Enhancing privacy in cloud computing via policy-based obfuscation”, J Supercomput, vol. 61, no. 2, pp. 267-291, 2010.

[15] B. Wang and Q. Zhao, “Server-aided batch verification signature schemes in the multiple-signer setting”, Security and Communication Networks, vol. 6, no. 11, pp. 1359-1366, 2013.

[16] N. Khanghahi and R. Ravanmehr, “Cloud Computing Performance Evaluation: Issues and Challenges”, International Journal on Cloud Computing: Services and Architecture, vol. 3, no. 5, pp. 29-41, 2013.

[17] P. Kadam and A. Devare, “A Secure, Scalable, Flexible and Fine-Grained Access Control Using Hierarchical Attribute-Set-Based Encryption (HASBE) in Cloud Computing”, IJCATR, vol. 3, no. 12, pp. 799-808, 2014.

[18] S. Lingwei, Y. Fang, Z. Ru and N. Xinxin, “Method of secure, scalable, and fine-grained data access control with efficient revocation in untrusted cloud”, The Journal of China Universities of Posts and Telecommunications, vol. 22, no. 2, pp. 38-43, 2015.

[19] W. Ng, M. Kirchberg, S. Bressan and K. Tan, “Towards a privacy-aware stream data management system for cloud applications”, IJWGS, vol. 7, no. 3, p. 246, 2011.

[20] “Provable Secure Multi-Authority Attribute Based Signatures”, JCIT, vol. 8, no. 2, pp. 545-553, 2013.

[21] J. Qian and X. Dong, “Fully secure revocable attribute-based encryption”, Journal of Shanghai Jiaotong University (Science), vol. 16, no. 4, pp. 490-496, 2011.

[22] J. Xu, W. Chen, S. Ji, Y. Ren and J. Wang, “A Novel Preserving Client Privacy and Designate Verifier Auditing Scheme for Cloud Storage”, IJSIA, vol. 9, no. 1, pp. 295-304, 2015.

[23] J. Wei, J. Liu, R. Zhang and X. Niu, “Efficient Dynamic Replicated Data Possession Checking in Distributed Cloud Storage Systems”,International Journal of Distributed Sensor Networks, vol. 2016, pp. 1-11, 2016.