Journal of Theoretical and Applied Information Technology
30th April 2021. Vol.99. No 8
© 2021 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN ENHANCED ACCESS CONTROL MODEL TO ENCRYPTED DATA BASED ON AN XACML FRAMEWORK IN CLOUD ENVIRONMENT

AMJAD ALRUWAILI[1], A. A. EL-AZIZ[2, 3], HEDI HAMDI[1, 4]

[1]Department of Computer Science, College of Computer and Information Sciences. Jouf University, KSA

[2]Department of Information Systems, College of Computer and Information Sciences, Jouf University, KSA

[3]Department of Information Systems & Technology, FGSSR, Cairo University, Egypt

[4]University of Manouba, Manouba,Tunisia

E-mail: [1]401101949@ju.edu.sa, [2]aaeldamarany@ju.edu.sa, [1]hhamdi@ju.edu.sa

ABSTRACT

Cloud computing is a cutting-edge innovation for improving and developing plans of action in associations. It tends to be utilized for giving programming and framework administrations sent in data focuses. Encryption of data by its owner and saving them on the cloud causes many efficiency and secrecy issues. In Cloud computing, a client who has approved certifications ought to be able to get to classified data, such as data owners or cloud providers. In conventional techniques for making data secure, data are encrypted and are kept in trusted hosts and their access is constrained by an access control policy. If the cloud server is penetrated from unapproved clients, the secrecy of touchy data will be uncovered. This paper proposes an enhanced cloud access control approach over encrypted data utilizing an XACML framework system and proof of ownership (POW) procedures. The proposed model controls the access over encrypted data by identifying that the user, who sends requests for accessing the encrypted data, is authorized or not dependent on his/her attributes stored in the XACML policy. By applying the proposed XACML framework, the cloud administrations will play out its concurred capacities with forestalling data spillage, data misfortune, and maltreatment of cloud administrations.

Keywords: *XACML, Cloud Computing, Proof of Ownership, Fingerprint*

1. INTRODUCTION

This XML is standard language for exchanging data over the web. The XML documents contain sensitive information for business requirements and operations. As a result, XML documents must be protected to ensure the secrecy and availability of cloud services. Applying access control over XML documents must enhance the secrecy, efficiency, applicability, and adaptability for transmitted data. The expressiveness of access control must ensure the high-level rules of polices to verify user authorizations. Modularity of access control must maintain the combination of different policies while interoperability must obtain the ability to exchange information. Efficiency of access control must ensure whether the policy should be granted or denied [1]. Different techniques and roles must be proposed to secure XML database documents. XACML is a policy language for executing access control parameters. The policy is formalized and is executed by an access control mechanism for building expressions [2]. An important concern regarding cloud computing is the dynamic provisioning where a single status change in a user credentials is sent across all affected systems from that point [3]. Cloud computing provides a configurable computing resources, such as storage, computing systems, and networks that rapidly provides data management and service availability [4]. The data centers of the cloud stores data based on the cloud infrastructure by providing a sharing pool of storage resources [5]. Although cloud computing became a new technology for corporations and users, it is also becoming a new risks of data security. The risk issue of storing data on cloud storage is trustworthy. Even cloud providers guarantee data security by implement security technologies. Cloud providers encrypt the stored data on their servers using a secret key. The encryption process executed by the providers does not protect the data against any attacks. When there are aggressive actions inside the server, a user's data can be damaged easily. Therefore, a user must trust the service provider [6]. Moreover, the lack of authoritative in cloud storage causes privacy issues. In a cloud data center, the user must ensure that his/her confidential data must be kept private. Therefore, the main problem of cloud storage is the security of user's data [7]. To avoid the security problem in cloud data center, the data should be encrypted and the access of users over the encrypted data must be controlled; i.e., prevent an unauthorized user or cloud provider from using user's encrypted data [7]. In this paper, an enhanced access control model to encrypted data using XACML policy framework is proposed. In the proposed model, the cipher texts and the corresponding symmetric keys are stored together. When an owner encrypts a message, he/she uses XACML access control policy language to specify who can access the cipher texts and the corresponding symmetric keys to decrypt the cipher texts. On other hand, if a user wants to decrypt the cipher, he/she should send a request and his/her attributes must be identical with the predefined attributes stored in the XACML access control policy language. The proposed model controls the access over encrypted data by using the user's fingerprint for consumer's authorization. The proposed model will secure the encrypted data from accessing by unauthorized users or cloud providers. Hence, if the cloud storage is breached, the encrypted data will be confidential. Based on the proposed model, the integrity and confidentiality of data over the cloud are maintained [8]. The remaining of the paper is organized as follows: section 2 presents the problem statement. Section 3 shows the recent related work. Section 4 introduces an introduction about XACML. The proposed model is described in section 5. The discussion is shown in section 6. Finally, in section 7, the conclusion is summarized.

2. PROBLEM STATEMENT

The Cloud is a model of IT that makes it possible to organize and treat computing as a utility, just like electricity or telephony. It is a resourceful model that offers multiple layers of services based on user needs. This model heralds the start of a new era of computing characterized by the advent of new digital services available anywhere, on demand, and for any organization, regardless of size. The expected benefits of the recent emergence of the Cloud are numerous: flexible and dynamic management of resources, the almost unlimited availability of computing, network, or storage resources thanks to visualization techniques leading to economies of scale and significant reductions in infrastructure management costs. However, the increasing convergence of cloud technologies has started to generate many security issues making their security a laborious task that requires more than user authorization with passwords or digital certificates and confidentiality in the data transmission. Many recent works proposed several techniques to solve the security issues of cloud computing, but they didn't guarantee that the owner's resource is accessed by an authorized user. Since XACML [2] is a standard XML language for creating access control policies, we use XACML framework to control the access of encrypted data and make a proof of ownership. Therefore, we enhance the access control of the encrypted data resided on the cloud by using XACML framework.

3. RELATED WORK

As presented in [9], a system for a ciphertext policy attribute based was proposed. In the proposed system, the private keys of the users are identified by a set of features to apply the decryption process. As shown in [10], a security system for cloud data centers based on access control was presented. In that research, all user processing including decryption, encryption, analysis and development of policies were presented. A certificate attribute was proposed for extracting, verifying, and authorizing user permissions. The authors of [11] presented a technique based on attribute encryption. In the proposed system, ciphertext is not encrypted to a receiver as in conventional public key cryptography. The attributes of a policy will be associated with receivers' private keys and ciphertext. If there is a matching between a receiver's private key and its corresponding ciphertext, the receiver will be able to decrypt a ciphertext. Different research methodologies are proposed for identifying fuzzy based encryption schemes [12], [13], [14]. As shown in [15], an attribute encryption scheme is presented based on key policies. In that system, each ciphertext relates to a set of attributes and the key can be connected with any tree access structure. As shown in [16], several implementations of attribute encryption schemes are presented. The authors try to find solutions to the key revocation problem. The authors of [17] presented an encryption system based on multi-authority levels. Each authority manages a set of domain attributes. The main challenge in this issue is to prevent attacks between different users.

Different approaches are applied to secure data on cloud data centers from internal users. As presented in [5], two approaches are presented to secure cloud data center. These approaches are user behavior monitoring and user verification mechanism for tracking and monitoring data from being penetrated. The authors of [18] proposed an access control mechanism for protecting the confidentiality of data based on a set of access control rules. In our proposed framework, the access control policy is defined easier and more secure than the proposed schema in [18] due to the using of XACML. As shown in [19], [20], an access control policy is presented for each data item. After accessing the data item, a policy is created by cloud providers to that data item. If the users' credentials and the policy of that data item are matched, the user will be granted to access the data item. As presented in [21], [22], the authors presented another methodology for not hiding the policies and credentials. The authors of [23] proposed a technique for partitioning the operations of access control between both cloud providers and data owners. The partitioning process for access control rules are based on two basic sets. The visibility of the first set is dedicated to the data owner while the visibility of the second set is dedicated to the cloud provider. Based on this methodology, the access control policies are not completely visible to the data owner. In our proposed framework, the PAP is responsible for storing the access control policies; hence the cloud provider can't know the access control policies. As presented in [24], a scheme for access control was implemented that hides the credentials and rules in a specific form. The authors of [25] presented XACML methodology with server policies of access control by adding request and response languages. The policy language presented access control policies to determine who can request the policy while the request and response language manipulated different queries to verify whether the request is to be granted or denied. The proposed XACML consists of three major parts. The first part is the policy set that contains a set of policies and each policy contains a set of rules which are consisted the smallest components of XACML architecture. Another XACML research was presented in [26]. In that research, an XACML for Islamic financial information system was implemented to enforce Shariah rules in banking sectors. The XACML is embedded with an authorization system to apply Islamic Shariah policies and rules in the banking systems. As shown in [27], the performance limitation problem was improved for policy decision point (PDP) that is considered one of the main components of XACML framework. The improvement is executed by statistically establishing an attribute bitmap for each subject, object and action. The decision engine of the policy analyzes requests and extracts the bitmap attributes. Finally, the policies are matched, and the authorization decision is taken. Another XACML policy was presented in [28] by applying a management optimization scheme for reducing the time taken in storing the bitmap. One of the recent researches for enhancing XACML policy language was presented in [29]. In that paper, the policy set of the access control is based on four components: subject, object, action and condition. After applying the previous components, the request is checked whether is to be granted or denied. The policy framework combines the XACML structure with a proposed authorization framework to group similar functionalities for better authorization process. However, none of these techniques used XACML framework to access the control of the encrypted data stored in the cloud architecture. Moreover, none of the previous techniques used POW to verify that the user is authorized to make a request. As a main methodology, the XACML is based on separating the access decision from the use point. In addition, the proposed framework guarantees the privacy and confidentiality of access control policies.

4. EXTENSIBLE ACCESS CONTROL MARKUP LANGUAGE (XACML)

XACML [2] is an XML-based language for access control that has been developed by OASIS. XACML defines both an access control policy language, and a request and response language.

The policy language is an ABAC mechanism, used to construct expressions that make up an access control policy, which specifies the security mechanisms. In other words, the policy language defines the required constraints and conditions to a subject for accessing a resource and carries out an action through a specific environment. The policy language is an extensible, flexible, highly expressive, standards-based, and general-purpose language. Moreover, it enables the specification of fine-grained policies, used to access the control to resources.

The request and response language describes the subjects making requests for accessing resources, and renders the authorization decisions granting or denying the access request. XACML is used not only for controlling the access to XML documents, but also for controlling the access to any type of resources.

One of the major advantages for using XACML is that it controls not only the access to XML documents, but also controls the access to several resources. Moreover, XACML provides standard data types, functions and combining algorithms.

The XACML contains four main functions presented as follows:

1. Policy Decision Point (PDP) is considered the primary point that evaluates requested policies and provides the authorization decision.

2. Policy Enforcement Point (PEP) is the point that executes the access control mechanism by passing requests to the policy decision point (PDP) and applying the authorization decision.

3. Policy Administration Point (PAP) is the point that creates, manages, and stores policies in an appropriate repository.

4. Policy Information Point (PIP) is the point that retrieves attributes to PDP.

The main functions offered by XACML can be summarized as FOLLOWS:

1. Policy combination: XACML provides a method for combining policies specified independently. Therefore, different entities can define their policies on the same resource. Hence, when an access request on that resource is submitted, the system considers all the applicable policies.

2. Combining algorithms: XACML provides different combining algorithms; each one provides a way of reconciling multiple decisions into a single decision.

3. Attribute-based restrictions: XACML provides the definition of policies based on the attributes of subjects (e.g., name and address) and resources (e.g., creation date and type). Moreover, it includes built-in operators for comparing the attribute values and provides a method for adding nonstandard functions.

4. Policy distribution: Policies can be defined by different parties and carried out at different enforcement points. Moreover, XACML allows one policy to contain another one or refers to it.

5. Implementation independence: XACML provides an abstraction layer that insulates the policy-writer from its implementation details. This insulation guarantees that different implementations are executed consistently.

6. Obligations: XACML provides a method for specifying actions, called obligations, which must be executed in conjunction with the applicable policies that have decisions.

The XACML framework is depicted in Figure 1 and is described as follows:

1. A user starts the mechanism by sending a request which is received by the PEP.

2. The PEP transforms the received request into an XACML authorization request.

3. The PEP passes the authorization request of the user to the PDP.
4. The PDP receives the request and matches the authorization request with the predefined policies (It looks at the data in the request, the appropriate XACML policy, and user attributes in the PIP).

5. If there is a matching between the authorization request and the predefined policies, the PAP manages and acquires the policies and retrieves the attribute values from the PIP.

6. The PDP finalizes the request by determining the final decision whether to permit, deny, not applicable, indeterminate and finally returns the decision to the PEP.

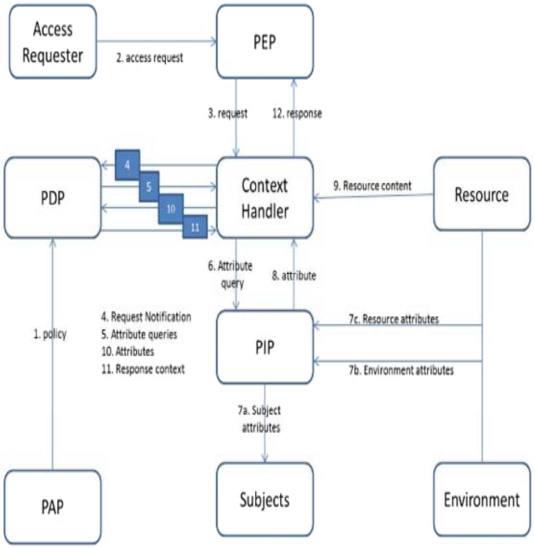
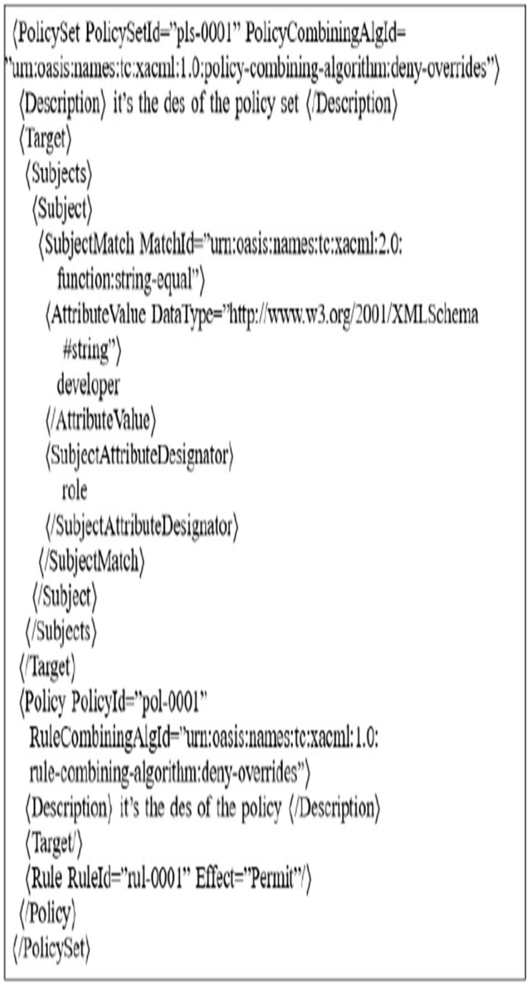


Figure 1: XACML Framework

The main concepts of all XACML policies are a <PolicySet>, <Policy>, and <Rule>, which represent a single access control policy. The root element of each XACML policy document is exactly one <PolicySet>, <Policy>, or <Rule>. The <Policy> consists of one or more <Rule> elements. A <policy> has at least one <Rule>. The <Rule> includes the core logic of an XACML policy. The decision logic of the rules is included in a <Condition>. The <Condition> is a Boolean function that refines the applicability of the rule. If the <Condition> returns true, then the rule's Effect (Permit or Deny) is returned. If the <Condition> returns false, the PDP returns to the PEP the value NotApplicable.

If many <Policy> elements are contained in a <PolicySet>, the PDP needs a way to reconcile the effects returned by all policies. Hence, the concept of the Policy Combining Algorithm is introduced in a <PolicySet>. The final decision value of the policy-combining algorithm is called the authorization decision. Similarly, if many <Rule> elements are contained in a <Policy>, the PDP needs a way to reconcile the effects returned by all rules. Hence, the concept of the Rule Combining Algorithm is introduced in each <Policy>.

Moreover, XACML provides a feature, called a <Target>, which is a set of simplified attribute values for a subject, a resource, an action, and an environment that must be satisfied for a <PolicySet>, <Policy>, or <Rule> to be applicable to a given request. If all the attribute values of a <Target> are satisfied with the associated <PolicySet>, <Policy>, or <Rule>, then the associated <PolicySet>, <Policy>, or <Rule> applies to the request [30]. An example of XACML policy is depicted as in Figure 2:



```
⟨PolicySet PolicySetId=”pls-0001” PolicyCombiningAlgId=
”urn:oasis:names:tc:xacml:1.0:policy-combining-algorithm:deny-overrides”⟩
 ⟨Description⟩ it’s the des of the policy set ⟨/Description⟩
 ⟨Target⟩
  ⟨Subjects⟩
   ⟨Subject⟩
    ⟨SubjectMatch MatchId=”urn:oasis:names:tc:xacml:2.0:
       function:string-equal”⟩
     ⟨AttributeValue DataType=”http://www.w3.org/2001/XMLSchema
        #string”⟩
       developer
     ⟨/AttributeValue⟩
     ⟨SubjectAttributeDesignator⟩
        role
     ⟨/SubjectAttributeDesignator⟩
    ⟨/SubjectMatch⟩
   ⟨/Subject⟩
  ⟨/Subjects⟩
 ⟨/Target⟩
 ⟨Policy PolicyId=”pol-0001”
   RuleCombiningAlgId=”urn:oasis:names:tc:xacml:1.0:
   rule-combining-algorithm:deny-overrides”⟩
  ⟨Description⟩ it’s the des of the policy ⟨/Description⟩
  ⟨Target/⟩
  ⟨Rule RuleId=”rul-0001” Effect=”Permit”/⟩
 ⟨/Policy⟩
⟨/PolicySet⟩
```

Figure 2: An Example of an XACML PolicySet

4.1 XACML Policy Language

XACML policies are XML documents with one <PolicySet>, one <Policy>, or one <Rule> as the root element.

A <PolicySet> contains zero or more <PolicySet> elements (optionally), zero or more <Policy> elements (optionally), one <Target> (required), one identifier for the policy-combining algorithm (required), zero or one <ObligationExpressions> (optionally), and zero or one <AdviceExpressions> (optionally).

A <Policy> contains one or more <Rule> elements (at least one <Rule>), one identifier for the rule- combining algorithm, one <Target>, zero or one <ObligationExpressions>, and zero or one <AdviceExpressions>.

A <Rule> contains zero or one <Target>, zero or one <Condition>, one Effect attribute, zero or one <ObligationExpressions>, and zero or one <AdviceExpressions>.

A <Target> is a set of attribute values to identify uniquely a subject, a resource, an action, and an environment that must be satisfied for a <PolicySet>, <Policy>, or <Rule> to be applicable to a given request. If all the conditions of a <Target> are satisfied with the associated <PolicySet>, <Policy>, or <Rule>, then the associated <PolicySet>, <Policy>, or <Rule> applies to the request. The <Target> must appear as a child of a <PolicySet> and <Policy>. However, it may appear as a child of a <Rule>.

4.2 Rule

A <Rule> contains a RuleId attribute as an identifier, zero or one <Description>, zero or one <Target>, zero or one <Condition>, one Effect attribute, zero or one <ObligationExpressions>, and zero or one <AdviceExpressions>. The required RuleId attribute is a string that assigns a unique name to the <Rule>. The optional <Description> contains a free description to the rule.

The optional <Target> defines the set of attribute values of the requests to which the rule is proposed to apply in the form of a logical expression on the attributes of the request. If the matches defined by the target are satisfied by the attributes of the request, the rule is applicable to the request. The rule is proposed to be applied to all entities of a particular data type, if this entity is omitted from the <Target>. If the <Target> is omitted from a <Rule>, the target of the <Rule> will be the same as the <Target> of its parent <Policy>.

The required Effect attribute of the rule indicates the rule consequence of a True evaluation for the rule. The Effect attribute value is Permit or Deny.

The optional <Condition> is a Boolean expression that must be satisfied (be true) for the rule to be assigned its Effect attribute value. It refines the applicability of the rule. For example, in the sentence "Only allow logins from 10 am to 6 pm", the condition indicates that, the access is allowed only in the interval [10 am - 6 pm].

If the <Condition> is omitted or evaluates to true, the condition value will be True. The condition value will be False if the <Condition> evaluates to false. The condition value will be Indeterminate, if an operational error occurred during the evaluation, such as missing attributes, network errors while retrieving rules, division by zero, or syntax errors in the decision request or in the rule. Therefore, the <Rule> is evaluated as follows:

1. (If the <Rule> has not a <Target>, or the <Target> matches the attributes of the request) and the <Condition> evaluates to true, the rule value will be the value of the Effect attribute (Permit or Deny).

2. (If the <Rule> has not a <Target>, or the <Target> matches the attributes of the request) and the <Condition> evaluates to false, the rule value will be NotApplicable.

3. (If the <Rule> has not a <Target>, or the <Target> matches the attributes of the request) and the <Condition> evaluates to Indeterminate, the rule value will be Indeterminate {P}, if the Effect attribute value is Permit, or Indeterminate {D}, if the Effect attribute value is Deny. Indeterminate {P} means an Indeterminate from a policy or rule, which will be evaluated to Permit but not Deny. Indeterminate {D} means an Indeterminate from a policy or rule, which will be evaluated to Deny but not Permit.

4. If the <Target> of the <Rule> does not match the attributes of the request, the rule value will be NotApplicable. The Rule's condition value will not be considered.

5. If the <Target> of the <Rule> matching evaluates to Indeterminate, the rule's value will be Indeterminate {P}, if the Effect attribute value is Permit, or Indeterminate {D}, if the Effect attribute value is Deny. The rule's condition value will not be considered.

5. THE PROPOSED MODEL

In cloud computing, a user encrypted data can be accessed by a cloud administrator or provider. Hence, to ensure data security form an unauthorized user or cloud provider, we present how to protect a user encrypted data from an unauthorized user or a cloud provider by allowing an encrypter to develop an access control policy for the encrypted data using XACML framework. The proposed model will secure the encrypted data from accessing by unauthorized users or cloud providers. Moreover, it provides more security by using a fingerprint authorization parameter. Hence, the encrypted data will be confidential even if the cloud storage is breached. The enhanced access control model is depicted as in Figure 3.

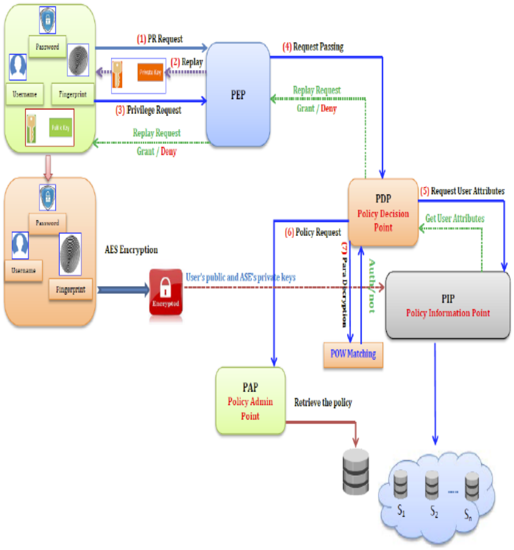


Figure 3: The Enhanced Access Control Model to Encrypted Data based on an XACML Framework in Cloud Environment

As depicted in Figure 3, an enhanced access control mechanism over encrypted data is proposed based on XACML and the Proof of Ownership (POW) methodology. The proposed model is illustrated as follows:

1. **Authorization Layer**: the user provides his own credentials: username, password, public key, and a biometric fingerprint.
2. The user fingerprint, credentials, and are encrypted using Advanced Encryption Standard (AES), which provides longer key length with faster encryption process.
3. After applying AES, the result of AES, the secret key and the user's public key are sent to the PIP.
4. The PIP decrypts the result of AES using the AES secret key and keeps the user's credentials, fingerprint, and user's public key for authorization.
5. **Private Key Request**: before the user sending an access PEP, the user must send a private key request (PRU) to the PEP, which is used as a certificate authority for creating the user's Private Key. The user's request is sent based on formula (1) as follows:

$$\text{Request}= [\text{TU} \mid\mid \text{U}_{\text{username}}]\ (1)$$

Where the $\mathbf{T_U}$ is the user request timestamp and $\mathbf{U_{username}}$ is the username.

6. The PEP replies the user request with the user's private key (PR_U). Now, the user has both a public (PU_U) and a private key (PR_U).
7. Based on step 5, the user sends an access request to the PEP as presented in formula (2):

$$\text{Request=D}\ [\text{PR}_{\text{U}}, [\text{Finger} \mid\mid \text{User Request}\ ]\ (2)$$

Where a user request and a user biometric fingerprint are encrypted using the user's private key (PRu) to proof the identity of the sender.

8. The PEP passes the request to the PDP. The PDP sends a user attribute request to the PIP that stores user's credential attributes from steps 4.
9. The PIP sends the user's credential attributes, user's fingerprint, and user's public key to the PDP.
10. The PDP decrypts the user's request based on formula (3):

$$\text{Request = D}\ [\text{PU}_{\text{U}}, [\text{Finger, User Request}]]\ (3)$$

The decryption process is executed by using the user's public key (PU_U) to extract the user's biometric fingerprint and the user's request.

11. **Proof of Ownership (POW)**: The PDP performs the POW mechanism for matching the user's fingerprints stored in step 4 and extracted from step 10 to verify whether the user is authorized or not. The matching process is based on two separate encryption and decryption paths to ensure the confidentiality and integrity of data.

12. If the user is authorized, the PDP retrieves the access control policy from the PAP and matches between the user request's attributes and the policy target's attributes to check if the user is permitted/denied accessing the stored encrypted data.
13. The permit/deny option is sent throughout the reverse path from the PDP to the PEP.
14. The PEP replies the grant/deny process to the user.

The following Algorithm 1 summarizes the process of the proposed model to control the access model of the encrypted data on the cloud:

Algorithm 1: An Enhanced Access Control Model to Encrypted Data based on an XACML Framework in Cloud Environment

Input: Access Requests Σ.

Output: Decision Request (Permit/Deny).

```
Begin:
    1) Storing the user parameters in PIP.
    2) PR_U Req_i to PEP.
    3) PR_U Reply from PEP.
    4) User Req_i to PEP.
    5) User Req_i Passing to PDP.
    6) User parameters retrieved from PIP.
    7) User Req_i Decryption by PDP.
    8) For each DReqi
    9) If POW(DReqi)= true then
    10) DReri = Grant
    11) Else DReri = Deny
    12) End If
    13) If DReri = Grant then
    14) PDP requests the policy from PAP.
    15) Matching user's Req_i attrs with the policy
    15) End If
    16) Send Response (Reqi) to PEP.
    17) PEP Sends Response (Reqi) to user.
    End.
```

6. DISCUSSION

Based on the proposed model, an enhanced authorization mechanism is applied with proof of ownership (POW) methodology for sending, receiving, and verifying user requests. The proposed model is based on an XACML framework; hence it can reduce the false positive and negative alarms through authorization user identity before receiving XACML policies. The implementation of proposed model can be built on a cloud side either by AT&T XACML 3.0 Implementation in https://github.com/att/XACML or by mapping and storing XACML policies into a relational database [31].

7. CONCLUSION

In this paper, we proposed an enhanced access control model to encrypted data based on an XACML framework in cloud architecture. The proposed model verifies the user authorization using the proof of ownership (POW) methodology. The proposed model can be built on the cloud side to protect the encrypted data from unauthorized users or cloud providers. Our approach allows users who want to decrypt the ciphers to send requests. After verifying them by their fingerprints, their attributes must satisfy the attributes specified in the XACML access control policy language. In our proposed model, the XACML access control policy language enhances the security of the encrypted data over the cloud environment.

REFERENCES:

[1] R. Abassi, F. Jacquemard, M. Rusinowitch, and S. G. El Fatmi, "XML Access Control: from XACML to Annotated Schemas", *Proceedings of the 2nd International Conference on Communications and Networking (ComNet)*, 2010, pp. 1 – 8.

[2] B. Parducci and H. Lockhart, "eXtensible Access Control Markup Language (XACML) Version 3.0. OASIS Standard", http://docs.oasisopen.org/xacml/3.0/xacml-3.0-core-spec-os-en.pdf, 22/01/ 2013.

[3] A. Arora, A. Khanna, A. Rastogi, and A. Agarwal, "Cloud Security Ecosystem for Data Security and Privacy", *Proceedings of the 2017 7th International Conference on Cloud Computing, Data Science Engineering-Confluence*, January, 2017, pp. 288–292.

[4] M. D. Boomija and S.V. Kasmir Raja, "Secure Data Sharing through Additive Similarity Based

Elgamal like Encryption", *Proceedings of the 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB))*, February, 2016, pp. 652–655.

[5] N. Patel and K. B. Kansara, "Ubm uvm approach for preventing insider data theft from cloud storage", *Proceedings of 2018 5th International Symposium on Emerging Trends and Technologies in Libraries and Information Services (ETTLIS)*, February, 2018, pp. 36–40.

[6] P. Suwansrikham and K. She, "Asymmetric Secure Storage Scheme for Big Data on Multiple Cloud Providers", *Proceedings of 2018 IEEE 4th International Conference on Big Data Security on Cloud (BigDataSecurity)*, May, 2018, pp.121–125.

[7] F. Shahzada, "State-of-the-Art Survey on Cloud Computing Security Challenges, Approaches and Solutions", *Proceedings of the 6th International Symposium on Applications of Ad hoc and Sensor Networks (AASNET14)*, *Procedia Computer Science*, Vol. 37, No. 362, 2014.

[8] E. Damiani, S. D. Vimercati, S. Foresti, and et al., "An Experimental Evaluation of Multi-Key Strategies for Data Outsourcing", *Proceedings of FIP International Information Security Conference (SEC 2007), Springer*, May, 2007, pp. 385–396.

[9] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute Based Encryption", *Proceedings of 2007 IEEE Symposium on Security and Privacy (SP '07)*, May 2007, pp. 321–334.

[10] Y. Wang, Q. Sun, Y. Ma, J. Zhang, Z. Liu, and J. Xue, "Security Enhanced Cloud Storage Access Control System Based on Attribute Based Encryption", *Proceedings of 2108 International Conference on 2018 Big Data and Artificial Intelligence,* 2018, pp. 52-57.

[11] A. Sahai and B.Waters, "Fuzzy Identity Based Encryption", *Advances in Cryptology, LNCS, Springer*, Vol. 3494 , 2007, pp. 457–473.

[12] D. Boneh and M. Franklin, "Identity Based Encryption from the Weil Pairing", *Advances in Cryptology CRYPTO, LNCS, Springer*, Vol. 2139, 2007, pp. 213– 229.

[13] C. Cocks, "An Identity Based Encryption Scheme Based on Quadratic Residues", *Proceedings of IMA Int. Conf., 2139 of LNCS, Springer,* Vol. *2139,* 2001, pp.360–363.

[14] A. Shamir, "Identity Based Cryptosystems and Signature Schemes", *Advances in Cryptology CRYPTO, LNCS, Springer*, Vol. 196, 1984, pp. 37–53, 1984.

[15] A. Sahai V. Goyal, O. Pandey and B. Waters, "Attribute Based Encryption for Fine-Grained Access Control of Encrypted Data", *Proceedings of ACM conference on Computer and Communications Security (ACM CCS)*, 2006, pp. 89–98.

[16] M. Pirretti, P. Traynor, P. McDaniel, and B. Waters, "Secure Attribute Based Systems", *Proceedings of ACM conference on Computer and Communications Security (ACM CCS)*, 2006, pp. 99–112.

[17] M. Chase, "Multi-authority Attribute-Based Encryption", *Proceedings of the Fourth Theory of Cryptography Conference (TCC 2007)*, 2007, Vol. 4392, pp. 515-534.

[18] X. Ye, "Access Control for Cloud Applications", *Proceedings of 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*, Aug., 2015, pages 970–977.

[19] D. Song, E. Shi, I. Fischer, and U. Shankar, "Cloud Data Protection for the Masses", *Computer*, Vol. 45, No. 1, January, 2012, pp. 39–45.

[20] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing", *Proceedings of IEEE INFOCOM*, March, 2010, pp. 1–9.

[21] W. She, I. Yen, B. Thuraisingham, and E. Bertino, "The scifc model for information flow control in web service composition", *Proceedings of 2009 IEEE International Conference on Web Services*, July, 2009, pp. 1–8.

[22] A. C. Squicciarini, E. Bertino, E. Ferrari, and I. Ray, "Achieving Privacy in Trust Negotiations with an Ontology-based Approach", *IEEE Transactions on Dependable and Secure Computing*, Vol. 3, No. 1, January, 2006, pp. 13–30.

[23] M. Nabeel and E. Bertino, "Privacy Preserving Delegated Access Control in Public Clouds", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 26, No. 9, September, 2014, pp. 2268–2280.

[24] X. Ye and B. Khoussainov, "Fine-Grained Access Control for Cloud Computing", *Int. J. Grid Util. Comput*, Vol. 4, No. 2/3, September, 2013, pp. 160 - 168.

[25] C. Ramli, H. Nielson, and F. Nielson, “The logic of XACML”, *Science of Computer Programming, ELSEVIER*, Vol. 83, April, 2014, pp. 80–105.

[26] I. Alsmadi and M. Zarour, “Building an Islamic Financial Information System Based on Policy Managements”, *Journal of King Saud University-Computer and Information Sciences, ELSEVIER*, Vol. 27, No. 4, October, 2015, pp. 364–375.

[27] F. Deng, Sh. Wang, L. Zhang, X. Wei, and J. Yu, “Establishment of Attribute Bitmaps for Efficient XACML Policy Evaluation”, *Knowledge-Based Systems, ELSEVIER*, Vol. 143, March, 2018, pp. 93–101.

[28] F. Deng, L. Zhang, C., Zhang, H. Ban, Ch. Wan, M. Shi, Ch.Chen, and E. Zhang, “Establishment of Rule Dictionary for Efficient XACML Policy Management*”, Knowledge-Based Systems*, Vol. 175, July, 2019, pp. 26–35.

[29] N. Skandhakuma, J. Reid, F. Salim, and Ed. Dawson, “A Policy Model for Access Control using Building Information Models”, *International Journal of Critical Infrastructure Protection*, Vol. 23, December, 2018, pp. 1–10.

[30] G. Hsieh, R. Meeks, and L. Marvel, “Supporting Secure Embedded Access Control Policy with XACML+XML Security”, *Proceedings of the 5th International Conference on Future Information Technology (FutureTech)*, May 21-23, 2010, pp. 1 – 6.

[31] A. A. Abd El-Aziz. and A. Kannan, “XML Access Control: Mapping XACML Policies to Relational Database Tables”, *The International Arab Journal of Information Technology (IAJIT)*, Vol. 11, No. 6, November 2014, pp. 532-539.