TEST Engineering & Management

May-June 2020
ISSN: 0193-4120 Page No. 4788-4792

A Certificate Technique and Two Factor Authentication for Cloud Security Adaptation

[1]Tejaswini B, [2]K.Amuthabala
[1,2]School of C&IT, REVA University, Bangalore
[1]reach2teju@gmail.com, [2]amuthabala.p@reva.edu.in

Article Info
Volume 83
Page Number: 4788-4792
Publication Issue:
May-June 2020

Abstract

The unforeseen nature of Cloud Computing visibility as well as manage among the data owners, applications, along with cloud providers raises tenants vagueness when utilizing cloud based services. Adaptation methods offer a reliable infrastructure based on cloud with assured behavior that preserves the excellence of service for tenants. Existing adaptation methods mainly concentrate on work qualities as well as they use non verifiable proof that is gathered in non-trust worthy manner. In this work, we recommend a technique for security-oriented adaptation for the cloud, dependent on the evidence gathered through trustworthy resources of a certificate procedure along with two factor authentication mechanisms. With this approach the cloud will keep stable security attribute over the time, through continuously confirming certificate validity with a digital signature with Google cloud KMS along with two factor authentication of the certificate. To protect users against digital identity theft, 2FA (Two-factor authentication) is a famous technique. Two-factor authentication connections access online information as well as accounts to having 2 or more than two tokens which show permission or ownership. It utilizes the output of verification actions and provides a model, in which corresponding configuration is utilized as the origin for adaptation.

Article History
Article Received:*19 November 2019*
Revised: *27 January 2020*
Accepted: *24 February 2020*
Publication: *16 May 2020*

Keywords: *Adaptive Cloud, Certificate, Two Factor Authentication (2FA), HMAC, Google KMS*

1. Introduction

The cloud computing is diverting the traditional IT made up of on-premise as well as static resources along with applications, component of a physical infrastructure run via a particular enterprise. It offers an environment in those platforms, applications, along with infrastructures are introduced as a service based on pay-as-you-go schedule to distant tenants. Cloud includes numerous benefits that are ease-of-use, reduced costs along with flexibility; as its adoption is usually damaged through the way it makes it difficult for clients to recognize the actions of an application/ service in a dynamic environment. These situations are partially because segregation of control as well as visibility among the cloud providers, applications, along with data owners that might establish a point of uncertainty with predictable limit of the complete IT system behavior, in comparison with entirely manage IT systems. Cloud services are subsequently concerned through constant situation modifications along with (unexpected as well as new) cloud functions, possibly distancing the monitor cloud service conduct through the probable one at runtime. Autonomic models and adaptation techniques have likewise been recommended to make a trustworthy cloud among reliable behavior that reacts to context modifications as well as functions to protect a constant excellence of service for tenants. Present methods mostly concentrate on adapting the cloud based on performance as well as accessibility qualities; through expanding resource management methods through reliability, elasticity, along with scalability algorithms. They are usually supplied as a

TEST Engineering & Management

May-June 2020
ISSN: 0193-4120 Page No. 4788-4792

black box, in which actions triggering adaptation actions, in addition to monitoring outcomes, are usually unavailable to end users. The end users will likely to belief in data readily becoming available on deployment or access time, through decreased visibility on the reasons triggering an adaptation. This is usually not a concern for regular users, but symbolizes a problem for users in significant domains related to security as well as regularly stops the movement of theirs to the cloud. In this paper, the technique handles the mentioned issues through specifying an adaptation method determined by veritable and trustworthy proof. Evidence is collected utilizing the certificate procedure that desires to shed light on cloud behavior growing the Cloud's transparency at the backend. Here we maintain constant security qualities for a cloud system, through ensuring that the configurations ensure the validity of the certification procedure as well as corresponding certificates in the long run. System changes that alter the certification procedure as well as gave certified security properties that are specifically pinpointed as well as utilized to question a feature model indicating equivalent cloud configurations. This paper involved in the defining a security adaptation method based on a certification procedure using the Cloud KMS (Cloud Hosted Key management service and an extra layer of authentication using the two factor Authentication technique. With Two-factor authentication facilitated for a cloud service ,any effort to sign in on an unrecognized device needs which you go in a secret code obtained as a text message or produced by an authenticator apps, which follow an open standard for generating time based one-time passwords using HMAC.

2. Literature Survey

In cloud computing, there is numerous research works that are related to data security. Various models, schemes as well as techniques are planned for fearlessly uploading the information to cloud.

Krotsiani, M., Kloukinas, C. &Spanoudakis, G. [1] Proposed framework in which the certification begins while an authority of certification submits a certification model to the cumulus certification platform and the certification platform have3majorparts (CM2Monitor translator, Certification Manager, Anamoly Manager). This recognized model allows us to evaluate the procedure for various qualities of adherence to the probable high-level certificate life cycle, to minimum/maximum probabilities of canceling a given certificate and so on. Translating the CM to code being performed at run-time for certifying the cloud service of issue. The code is made together with Prism model as well as follows its behavior, consequently that the analysis outcomes stay valid. So the total certificate process depends on the Prism model and each time checking for the probabilities would be time consuming. DananThilakanatha et al. [4] proposed system utilizing proxy re-encryption for security of information. In the proposed system data users encrypt the data utilizing his key part after that proxy encrypt the data utilizing his key part. Decryption is also approved in fashion that is similar. Nevertheless, if proxy is fake then information is converted into in secure. In the projected model information is sheltered from every threats that is internal as well as external, thread throughout, transits in addition to while data at rest. Supreeth [5] proposed a model where code is analyzed on encrypted data to be able to have minimum overhead over data owner. Thus third party is engaged. In this model the author follows 2- phases called uploading and downloading. The key generation here happens by a third party so there are chances of the key being by the third party who can be an internal attacker.

In this case the data integrity is maintained only when the HMAC (Uploading) =HMAC (downloading).This involves a tedious process of uploading and downloading the instructions.

In our proposed model we are creating a digital certificate for the verification of the cloud user and then further creating a verification process by using the HMAC algorithm and key generation using the two factor authentication mechanism

3. Proposed System

A. HMAC Algorithm

HMAC is a symmetric process which utilizes a secret key as well as a hash algorithm like SHA to produce a message authentication code.
This authentication code securely offers authenticity and data integrity because the secret key is necessary to recreate the code.

B. Equations

This definition is taken from RFC 2104:
Where
HMAC (K, m) =H ((K' ⊕ OPAD) ||H ((K' ⊕ IPAD) ||M))
K'= {H (K) K is larger than block size
{K otherwise…….equation (1)
Cryptographic hash function is denotes with H
M will be the message to be authenticated
Secret key is denotes with K

A block-sized key derived from the secret key, *K*is symbolize *with K'* ; also through padding to the right by the block sizing, or even through hashing down to less than the block size first as well as then padding to the right with zeros
|| represents the Concatenation
⊕ symbolizes the bitwise exclusive or (XOR)
Opad may be the block-sized outer padding, comprising of repeated bytes valued 0x5c

TEST Engineering & Management

May-June 2020
ISSN: 0193-4120 Page No. 4788-4792

*Ipad*may be the block-sized inner padding, comprising of repeated bytes valued 0x36

C. Cloud KMS

A cloud-hosted key management service is known as Cloud KMS which enables user to regulate cryptographic keys for user's cloud services the exact similar manner as user do on-premises. It includes support for encryption, decryption, signing, and verification using a variety of key types and sources including Cloud HSM for hardware-backed keys.

There are many encryption algorithms to give security to the cloud and we follow two procedures
1. Generating a certificate using the Google KMS
2. Validating the certificate using the key generation HMAC algorithm.

1. Generating a certificate using the Google KMS
A digital signature is generally created utilizing the private key piece of an asymmetric key. The signature is validated utilizing the public key part of the same asymmetric key.

When creating digital signatures, you must use a key which has the key purpose of ASYMMETRIC_SIGN. When you create the key, use ASYMMETRIC_SIGN.

To validate a signature, you need to know the full algorithm that was used when creating the key. For command-line instructions below that use the openssl command, you need to pass this information to those commands.

Grant the cloudkms.cryptoKeyVersions.useToSignpermission on the asymmetric key to the user or service that will perform the signing. You can learn about permissions in Cloud Key Management Service at Permissions and roles.

If you are going to validate a signature, grant cloudkms.cryptoKeyVersions.viewPublicKey permiss ion on the asymmetric key to the user or service that will download the public key to use for validation.

If you are going to use the command line, install OpenSSL if you do not already have it. If you use Cloud Shell, OpenSSL is already installed.

The following scenarios are offered by Cloud KMS for key functions:

Table 1: Cloud KMS Methodsand Scenarios

Scenario	Key_Purpose	Supported methods
Symmetric encryption	ENCRYPT_DECRYPT	cryptoKeys.encrypt, cryptoKeys decrypt
Asymmetric signing	ASYMMETRIC_SIGN	cryptoKeys.asymmetricSign, cryptoKeyVersions.getPublicKey
Asymmetric encryption	ASYMMETRIC _DECRYPT	cryptoKeyVersions.asymmetricDecrypt, cryptoKeyVersions.getPublicKey

The creation of the keys and the versions depend on the algorithm being used for encryption and decryption

A. Symmetric Encryption Algorithms

The ENCRYPT_DECRYPT key purpose allows symmetric encryption. Each key with this key purpose ENCRYPT_DECRYPT use the GOOGLE_SYMMETRIC_ENCRYPTION algorithm. In this algorithm, no parameters are utilized. This algorithm utilizes 256-bit Advanced Encryption Standard (AES-256) keys in Galois Counter Mode (GCM), padded through Cloud KMS-internal metadata.

B. Asymmetric Signing Algorithms

The ASYMMETRIC_SIGN key purpose allows asymmetric signing. Keys with key purpose ASYMMETRIC_SIGN utilize various algorithms, based on the key that supports RSA signing or elliptic curve signing .

For a key which has purpose ASYMMETRIC_SIGN, you could switch between different size keys as well as different signature schemes through the algorithm.

1) Elliptical Curve Signing Algorithm (ECDSA)
In the context of signing data at rest using RSA or ECDSA, where performance is not the critical factor, would it make sense to use HMAC-SHA256 instead of plain SHA256 for the digest function
s=k‖S(k‖H(m,k))s=k‖S(k‖H(m,k))

Here SS is some asymmetric signing operation such as RSA or ECDSA, kk is a randomly generated key, HH is HMAC-SHA256.

2) RSA Signing Algorithm
The HMAC in the end is a hash. You could then sign this hash with your private key (different than the key used in the HMAC). A third party with the symmetric key and your public key could verify you signed the HMAC, and could generate their own HMAC with the message and the shared

May-June 2020
ISSN: 0193-4120 Page No. 4788-4792

key to make sure it matched. This would also give you non-repudiation.

The layout of an RSA signing algorithm is:

RSA_SIGN_***[PADDING_ALGORITHM]_[MODULUS_BIT_LENGTH]_[DIGEST_ALGORITHM]***

The possible algorithms for RSA keys with purpose ASYMMETRIC_SIGN are given in table 2.

Table 2: Algorithms for Asymmetric Sign

ALGORITHM	DESCRPTION
RSA_SIGN_PSS_2048_SHA256	RSASSA-PSS 2048 bit key with a SHA-256 digest
RSA_SIGN_ PSS_3072_SHA256 (recommended)	RSASSA-PSS 3072 bit key with a SHA-256 digest
RSA_SIGN_PSS_4096_SHA256	RSASSA-PSS 4096 bit key with a SHA-256 digest
RSA_SIGN_PSS_4096_SHA512	RSASSA-PSS 4096 bit key with a SHA-512 digest
RSA_SIGN_PKCS1_2048_SHA256	RSASSA-PKCS1-v1_5 with a 2048 bit key and a SHA-256 digest
RSA_SIGN_PKCS1_3072_SHA256	RSASSA-PKCS1-v1_5 with a 3072 bit key and a SHA-256 digest
RSA_SIGN_ PKCS1_4096_SHA256	RSASSA-PKCS1-v1_5 with a 4096 bit key and a SHA-256 digest
RSA_SIGN_PKCS1_4096_SHA512	RSASSA-PKCS1-v1_5 with a 4096 bit key and a SHA-512 digest

C. Asymmetric Encryption Algorithms

The ASYMMETRIC_DECRYPT key purpose allows RSA encryption. The layout of an ASYMMETRIC_DECRYPT algorithm is

RSA_DECRYPT_***[PADDING_ALGORITHM]_[MODULUS_BIT_LENGTH]_[DIGEST_ALGORITHM]***

The possible algorithms for RSA keys with purpose ASYMMETRIC_DECRYPT are given in table 3.

Table 3: Algorithms for Asymmetric Decrypt

ALGORITHM	DESCRIPTION
RSA_DECRYPT_OAEP_ 2048_SHA256	RSAES-OAEP 2048 bit key with a SHA-256 digest
RSA_DECRYPT_OAEP_ 3072_SHA256 (recommended)	RSAES-OAEP 3072 bit key with a SHA-256 digest
RSA_DECRYPT_OAEP_ 4096_SHA256	RSAES-OAEP 4096 bit key with a SHA-256 digest
RSA_DECRYPT_OAEP_ 4096_SHA512	RSAES-OAEP 4096 bit key with a SHA-512 digest

Validating the certificate using the key generation HMAC algorithm
Algorithm: hmac
Step1: declare key, Message as array of bytes
Step2: declare function hash
Declare inputblockSize, outputblockSize for the hash Function
Step3: if (length (key) >blockSize) then
Key becomes*outputSize* bytes long
Else
Keys shorter than *blockSize* are padded to *blockSize*
By padding with zeros on the right

Step4: if (length (key) <blockSize) then
Pad key with zeros to make it *blockSize* bytes long

Step5:o_key_pad ← key xor [0x5c * blockSize]
Step6:i_key_pad ← key xor [0x36 * blockSize
Step 7: return hash (o_key_pad‖hash (i_key_pad‖ message

4. Implementation

Digital signatures in Google KMS are generated using a cryptographic URL signing secret, and they are accessible on the Google Cloud Platform Console. The secret, referred to as a private key, is encoded in a customized Base64 using the HMAC algorithm for URLs. This secret is shared among you and Google, as well as API key is unique to you.

The signing procedure utilizes an encryption algorithm to merge the URL and your shared secret. The resulting shared secret key in our proposed model is being implemented by means of a HMAC Algorithm.

Here we are going to analyze the various algorithms that are best suited for the process of verification of the digital certificate.

Digital signature is created automatically

To automatically create a digital signature for utilize with an API key (using the Google Cloud Platform console):
Step 1: Build an unsigned request URL with an API key.
Step 2: Visit the URL Signing Secret page.
Step 3: Get a digitally signed URL.

May-June 2020
ISSN: 0193-4120 Page No. 4788-4792

Step 4: Verify the digiSign certificate by generating a shared key to the cloud user by means of using the appropriate algorithm for verification.

This way we ensure that along with a digital signed certificate

We are undergoing a two way authentication to ensure that the certificate generate has not be hacked or breached.

The URL signing secret code can be generated by using the node.js, python, and java or c #coding framework

5. Experimental Results

The proposed model can be evaluated using the OpenSSL. Here the Signature of the attestation can be verified bundle leading up to the root certificates for Google and the HSM manufacturer, and signed by the certificate authorities (CAs) for both Google and the HSM manufacturer. Although Key management also offers less data security as compare to User Roles as well as two factor Authentication. In two factor user authentication perform good role and have more impact on security as compared to generation of the digital certificate. Securing the data, security of data

Increases by respect to values of data, however after some point it started degrading. But categorization of data plays superior role in security of data, as categorization of data could depict that data must be more secured.

HMAC have enhanced functionality to secure the data; however in compared to the encryption as well as decryption complexity it is less secure. The length of the key or the type of the algorithm that is being used plays a main role in the process of the authentication or the security of the cloud.

The Google KMS uses the algorithm and the HMAC equation to generate a certificate and use the key generation process using the OPENSSL[11].The results of the graph depict how the type of HMAC algorithm affects the security key generation.

In other hand Encryption/decryption complexity creates data safer than any other factor as well as technique. It is very secured than key management. Two factor authentication as well as certificate generation using the Google KMS may increase the process of security in the cloud to a certain extent while in contrast to the process of only digitally signing the certificate by the cloud user.

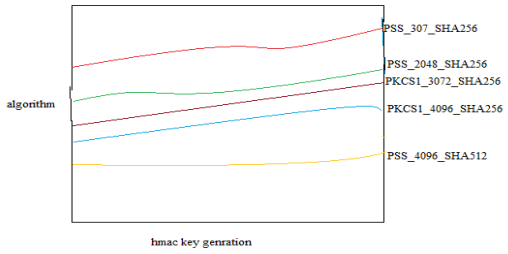

Figure 1: Graph depicting best HMAC algorithm for Google KMS

6. Conclusion

EC_SIGN_P256_SHA256 is the suggested elliptic curve algorithm for digital signing. In case you are planning to utilize RSA signing algorithms, SA_SIGN_PSS_3072_SHA256 issuggested.RSA_DECRYPT_OAEP_3072_SHA256 is the suggested algorithm, for asymmetric encryption.

References

[1] Krotsiani, M., Kloukinas, C. &Spanoudakis, G. (2017). Cloud Certification Process Validation using Formal Methods. Paper presented at the 15th International Conference on Service Oriented Computing (ICSOC 2017), 13-16 Nov 2017, Malaga, Spain

[2] Language support for modular autonomic managers in reconfigurable software components. In Proceedings of the 2017 IEEE International Conference on Autonomic Computing. IEEE, 271–278. Paolo Arcaini, ElviniaRiccobene, and PatriziaScandurra. 2015.

[3] Modeling and analyzing MAPE-K feedback loops for self-adaptation. In Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. 13–23..

[4] DananThilakanatha ,ShipingChen,Surya Nepal, Rafael A.Calvo and Leila Alem,"A platform for secure monitoring and sharing of generic health data in the Cloud", Elsevier Ltd, 2013

[5] Supreeth,Dr.Vinay Chopra ,India.Data Security approach using HMAC algorithm for cloud environment

[6] https://www.dhsinformatics.com/pf/ieee-cloud-computing-projects-bangalore/.

[7] E. H. Miller, "A note on reflector arrays," *IEEE Trans.Antennas*https://www.math.stonybrook.edu/~rtanase/files/DigitalSignature-net.pdf

[8] https://www.academia.edu/RegisterToDownload#RelatedPapers

[9] Appcara, http://www.appcara.com/products/appstack-r3, Accessed in March 2016.

[10] https://www.openssl.org/

[11] https://owncloud.org/

[12] https://cloud.google.com/kms/docs/attest-key