International Journal of Mathematics and
Computer Science, **15**(2020), no. 4, 1285–1293

Performance of Merkle Signature Scheme in Data Integrity Verification at Client

D. Sophia Navis Mary, S. Gopinathan

Ethiraj College for Women (Autonomous)
Department of Computer Science
University of Madras
Chennai, India

email: sophianavis@gmail.com

(Received July 9, 2020, Accepted September 1, 2020)

Abstract

To store voluminous data of the enterprises and organizations opt for cloud storage. Outsourcing data in the cloud allows many authorized users to access the data from different geographical locations. The service provider stores the users data in data centers at remote locations. Cloud Service provides infrastructure and web services to the users retrieving an unlimited data and paying the service consumed by the user. The cryptographic primitives use digital signatures for data integrity checking on the remote data. The client is unable to download the whole data stored in data center to validate its integrity. Therefore, efficient techniques required to prove the integrity of their outsourced data with least amount of communication, and computation cost. This paper describes the digital signature schemes used in auditing techniques at client and its performance.

1 Introduction

This cloud computing offers many services on demand to the users. Cloud service providers deliver the services on demand in the forms of Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service

Key words and phrases::Cloud storage, Data integrity, Digital Signatures, Auditing techniques, Cloud service.
AMS (MOS) Subject Classifications: 11Z05, 94D99.
ISSN 1814-0432, 2020, http://ijmcs.future-in-tech.net

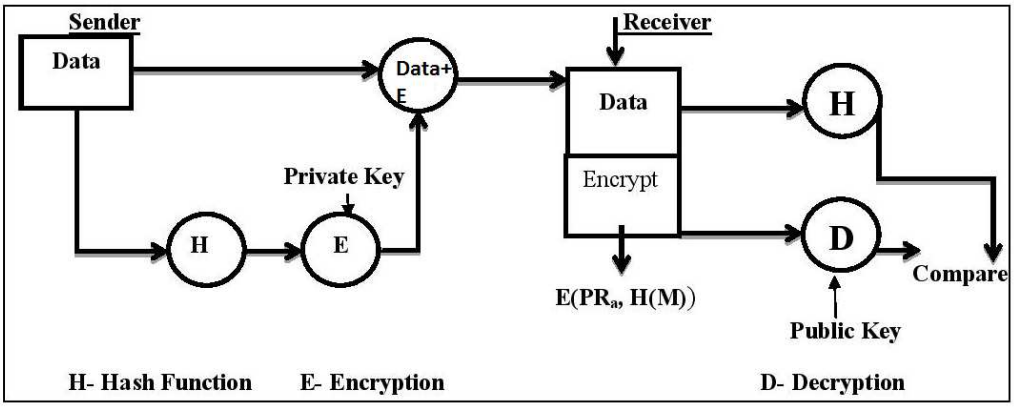

Figure 1: The working of Digital Signature

(Iaas). Organizations of any size, type and industry are using the cloud for data storage, virtual desktop, big data analytics, and software development and testing. Financial companies use the cloud services to identify and prevent fraud. Cloud computing equally brought the challenges on the data stored in cloud. The cloud user can access the resources with rapid elasticity from a broad network. To secure the client data in the cloud is the major concern of the Service provider. To ensure the data security digital signature schemes plays a vital role.

2 Digital Signature

Digital signature is a cryptographic technique [1] used to prove the authenticity of digital messages or documents. A valid digital signature ensures that the message is changed or not, provides authentication and sender cannot deny the message sent by him (non-repudiation). A digital signature scheme [2] is a tuple $(D, A, K, Sign, ver)$, where:
D stands for a finite number of input messages
A stands for a finite number of signatures
K represents a finite set of probable keys
For all k, signature algorithm is depicted as

$$Sign_k : D \rightarrow A \text{.....} (1)$$

a verification algorithm

$$\text{verify}_k(x,y) = \begin{cases} \text{true} & \text{if } y = sign_k(x) \\ \text{false} & \text{if } y \neq sign_k(x) \end{cases} \quad \text{.....(2)}$$

$(x, y) \in DA$ is called a signed message. The signature generation is based on asymmetric cryptography.

The first step is signature creation, for the input data the digest is formed and encrypted with the private key and transmitted to the receiver. To validate the signature, the recipient separates the encrypted digest from the message and decrypts it using the public key. The recipient generates a digest from the data which is received and compares it with unencrypted data.

2.1 Homomorphic Signature Scheme

In homomorphic signature schemes, the signature is computed by functions known public over the data or file. To evaluate the signature genuineness, the receiver generates the signature using the public key. The signature scheme involves key generation, signature generation and signature verification tasks.

The homomorphic signature scheme is defining as [3] a message space D, a limited number of private keys N, a probable number of public keys P' then the signature generation is

SigGen:$N * D \rightarrow Y$
Verify:$(K^{'}, x, Sig(N, x)) =$ yes,no $\forall x \in D$ and a verification algorithm N, P' are matching private, public keys. RSA signature is a homomorphic signature. The working of RSA is depicted in Figure 2

3 Merkle Signature Scheme

Merkle signature scheme consists of key generation; signature generation and signature verification for validate the data integrity. It is based on one time signature scheme. Merkle signature scheme uses a public key which is a concatenated hash value of the data blocks. This scheme first generates the public key by construct a hash tree of data blocks of fixed size.

3.1 Merkle Tree Construction

Merkle signature requires the initial step of generating the public key Ai and private key Yi of $2n$ message blocks. For each key Yi a hash value$Vi = H(Yi)$

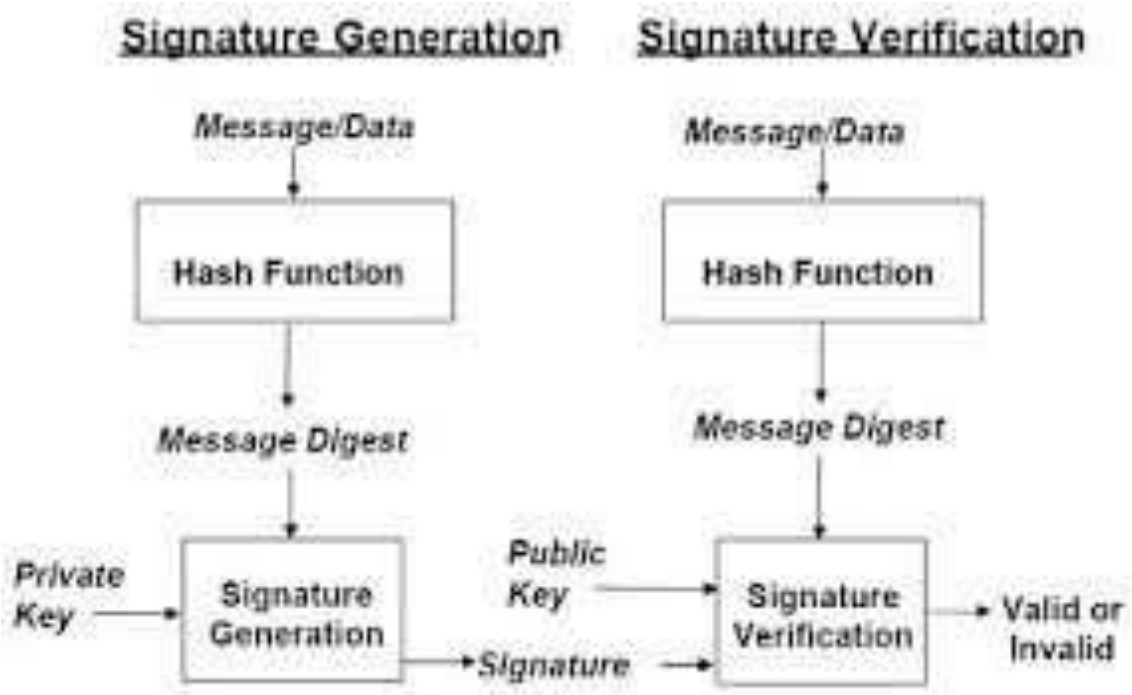

Figure 2: The working of RSA Signature

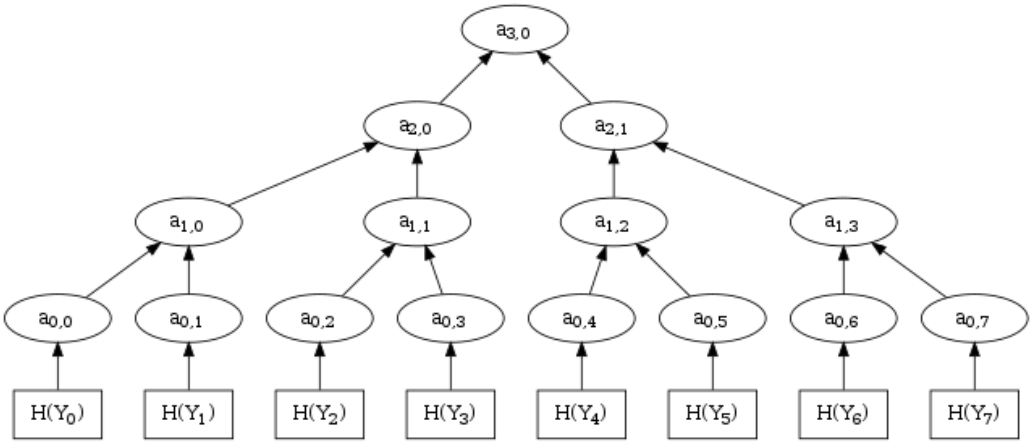

Figure 3: The Merkle Tree structure

is computed where $1 < i <= 2n$. The Hash values treated as a node of the hash tree. This Figure 3 shows a Merkle tree with eight leaf nodes. Any one of the data elements were changed, the hash value of root $a3,0$ would also change. The data segments $(Y0..Y7)$ determine the root together.
The signature cannot be used for a more than one time. Key generation for Merkle Scheme, one selects a positive number n and it decides that $2n$ signatures should be verifiable with the generated public key. The $2n$ One Time signature key pairs $(Ai, Yi), i = 1, ..., 2n$ are generated where Ai are signature key, Yi is a verification key. The Merkle Signature public key is concatenation of all these onetime signature keys. Each inner node of the tree (including root) is the hash value of the concatenation of its two children (Fig.3).

To determine the public Merkle Signature key, a binary authentication tree is constructed as follows: Each verification key Yi is written as a bit string. The leaves of the authentication tree are the hash values H (Yi) of the verification keys.

4 Experimental Results

The experiment implements the digital signatures schemes using RSA with different hash functions and Merkle signature for the given input data files of varying size from 20 MB to 1GB. The time to generate the signature is obtained and recorded for file size of 20 to 1024 MB in Table 1. The results are executed on Intelcore i5 CPU (2.4MHz) with 4GB RAM running

Windows 8 32-bit release. The signature schemes are implemented in Java using Eclipse. Table 1 gives the signature generation cost of various methods are executed and recorded.

Table 1: Signature Generation Cost.

Key Size Bytes	**Data Blocks MB**	**RSA 512 MilliSec**	**RSA 256 MilliSec**	**RSA SHA1 MilliSec**	**Merkle MilliSec**
256	20	764	468	321	795
256	58	2153	1279	858	1886
256	226	8221	4899	4415	8446
256	355	12854	7644	7566	11685
256	1024	39487	22651	18674	54756

The graph shows the results of the time taken by various signature schemes to generate the signature for various files varying in size from 20MB to 1GB. Figure 5 shows the verification time taken by the various schemes the time

Table 2: Signature Verification Cost.

Key Size Bytes	**Data Blocks MB**	**RSA 512 MilliSec**	**RSA 256 MilliSec**	**RSA SHA1 MilliSec**	**Merkle MilliSec**
256	20	422	749	296	640
256	58	1186	2137	858	1186
256	226	4633	8658	3276	8190
256	355	7161	13228	5070	12675
256	1024	21528	37705	17581	40794

consumption increase if the file size is large.

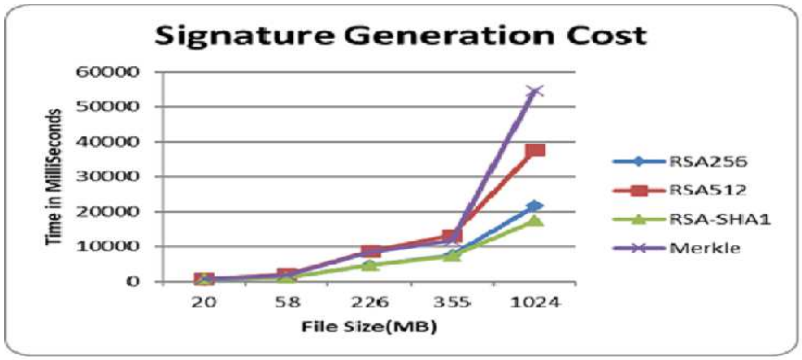

Figure 4: A Signature Generation cost

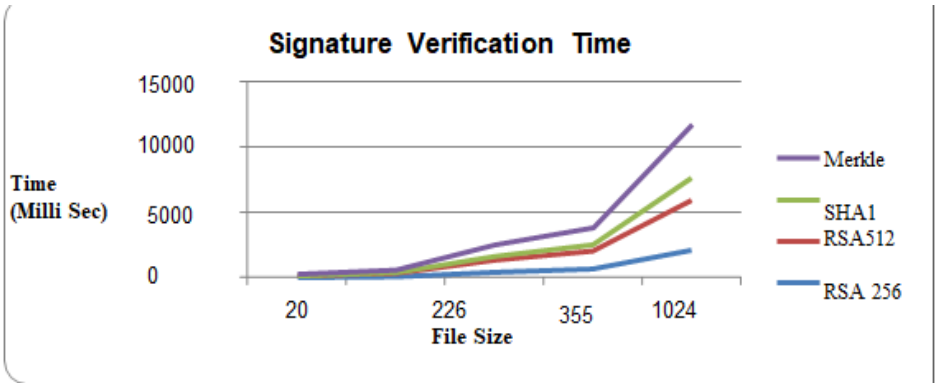

Figure 5: Signature Verification cost

5 Conclusion

This paper presents the performance of the data integrity checking schemes using digital signatures and analyses the performance in terms of signature generation time consumption and verification time of RSA based signatures with different key size and Merkle signature scheme. The study helps to understand the cost of signature schemes used in data integrity verification.

References

[1] Wiki:Digital Signatures, *https://en.wikipedia.org/wiki/Digitalsignature.*

[2] *http://www.emptrust.com/blog/benefits-of-using-digital-signatures.*

[3] Dan Boneh, David Mandell Freeman, *"Homomorphic signatures for polynomial functions." Annual international conference on the theory and applications of cryptographic techniques.* Springer, Berlin, Heidelberg, 2011.

[4] Dan Boneh, David Mandell Freeman, *"Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures.International Workshop on Public KeyCryptography,* Taormina, Italy, 2011.

[5] Robert Johnson et al., *"Homomorphic signature schemes," cryptographers track at the RSA conference.* Springer, Berlin, Heidelberg, 2002

[6] Michels, Markus, Markus Stadler, Hung-Min Sun, *"On the security of some variants of the RSA signature scheme". European Symposium on Research in Computer Security,* Springer, Berlin, Heidelberg, 1998.

[7] Chu-Hsing Lin, Chen-Yu Lee, Tang-Wei Wu, *."A cloud-aided RSA signature scheme for sealing and storing the digital evidences in computer forensics." International Journal of Security and its Applications,* **6,** no. 2, (2012), 241–244.

[8] Johannes Buchmann et al., *"On the security of the Winternitz one-time signature scheme," International Conference on Cryptology in Africa,* AFRICACRYPT 2011: Progress in Cryptology AFRICACRYPT, (2011), 363–378.

[9] Johannes Buchmann, Erik Dahmen, Andreas Hlsing, *"XMSS-a practical forward secure signature scheme based on minimal security assumptions," International Workshop on Post-Quantum Cryptography,* Springer, Berlin, Heidelberg, 2011..

[10] Georgö Becker, *"Merkle signature schemes, merkle trees and their cryptanalysis,* University Bochum, Tech. Rep., 2008.

[11] LC Coronadoö Garca, *On the security and the efficiency of the Merkle signature scheme,* Technical Report 2005/192, Cryptology ePrint Archive, 2005. Available at http://eprint.iacr. org/2005/192, 2005.

[12] Ralph C. Merkle, *"Advances in Cryptology",* CRYPTO89, Lecture Notes in Computer Science, **435,** Springer-Verlag, Berlin, Heidelberg, (1998).

[13] Michael Szydl, *"Merkle tree traversal in log space and time,"* In Eurocrypt 2004, Lecture Notes in Computer Science, **3027,** (2003), 541–554.