International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 4, August 2020, pp. 4208~4216
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i4.pp4208-4216

Design an active verification mechanism for certificates revocation in OCSP for internet authentication

Khalid F. Mahmmod, Mohammed M. Azeez, Zeyad H. Ismael
Department of Electrical and Communication Engineering, Ninevah University, Iraq

Article Info

Article history:

Received Sep 11, 2019
Revised Jan 14, 2020
Accepted Feb 1, 2020

Keywords:

Online certificate status protocol (OCSP)
Certification authority (CA)
Field programmable gate array (FPGA)

ABSTRACT

No doubt that data security online is crucial. Therefore, great attention has been paid to that aspect by companies and organizations given its economic and social implications. Thus, online certificate status protocol (OCSP) is considered one of the most prominent protocol functioning in this field, which offers a prompt support for certificates online. In this research, a model designed based on field programable gate array (FPGA) using Merkel's tree has been proposed to overcome the delay that might have occurred in sorting and authentication of certificates. Having adopted this model and with the assistance of Hash function algorithm, more than 50% of certificates have been processed in comparison with standard protocol. Moreover, certificates have been provided with substantial storage space with high throughput. Basically, Hash function algorithm has been designed to arrange and specify a site of verified or denied certificates within time of validity to protect servers from intrusion and clients from using applications with harmful contents.

Corresponding Author:

Khalid F. Mahmmod,
Department of Electrical and Commnication Engineering,
Nineveh University,
Mosul, Iraq.
Email: Khalid.mhmoed@uoninevah.edu.iq

1. INTRODUCTION

The process of connecting clients' computers and sharing data among themselves would expose it to an external attack. Therefore, it's imperative to develop a system that limits such attacks in which a client identification is needed before making secure connection through saving some crucial information into separate entity known as digital certificate issuance [1]. It's a digitally signed document which links public key value either with a person ID, device, or service that with a matching private key where a client becomes able and authorized to access sensitive sites, companies' services, and license award. It also prevents attackers benefit from these features and affecting the system resources through any possible method. These services could be essential for users' communication security. Moreover, security certificates can also be used in many applications such as E-mail and web browsing.

Public key certificate or also known as digital certificate using criterion X.509 v3 [2] as shown in Table 1 is deemed as one of major features of digital certificate. Certificates often contain the following information: public key of service or subject, identifier data of service or subject, subject such as; name, e-mail address, period of validity and expiration, a valid certificate for specific period, a request for a new certificate according to the type of certificate expired, information on source identifier, digital signature of data issuing party that connects the fragmentation of public key with private key of user, and optional attachments can be added to the certificate. Host has no longer in need for saving pass words for services and subjects that need to be verified as an essential condition to access by users, instead, host needs to build trust with the source of certificate through safe channels [3].

The urgent need to verify the certificate status as well as the rise in the speed of network data processing have made the use of certification revocation lists (CRL) not possible in verifying these certificates online in public key infrastructure (PKI) for their inability to offer support in real time [2]. That would expose the entire system to attack. Therefore, OCSP [4, 5] has been adopted to gain digital certificate X.509. This protocol has been described first time in criterion 2560 RFC titled " Online Certificate Status Protocol of Public Key Via Internet X 509"to overcome CRL disadvantages. It immediately responds to revoke certificate in real time from periodically updated database called certification authority (CA). It also reduces network response time to both servers and clients through timestamp which renders the system operate more efficiently in safe environment. Request and response messages are sent through OCSP in which "http" is usually used in exchanging those messages [6]. To gain digital certificate status (good, denied- unknown), user or host or service should send certificate request, the ones authorized to use encryption private key linked to public key hash which would be a significant part of digital certificate whether for service or accessing to topic wanted [7].

Many algorithms are being utilized by OCSP in searching for certificate status, where Merkel's Hash Tree is deemed the most prominent one [8]. It's a method used in searching and matching revoked certificates values. These values should be arranged in parent node and only two children, this tree should contain revoked certificates represented by a serial number. Each node in the tree possesses serial sections of sub-nodes that to be represented by a group of certificates that arranged reasonably and sequentially. Finally, these certificates are entered into the tree where are deleted by a particular algorithm designed for adding and deleting nodes. Process of searching and finding certificate targeted using hashing tree would be sequentially implemented according to the type of searching algorithm.

The main aim of this research paper is to develop and verify the searching method certificate the current OCSP protocol by using Merkle trees, Where the required verification implemented on protocol OCSP via simulated VHDL and applied to the FPGA board. Moreover, the validity of the target certificate has been tested in terms of its effective date, besides distinguishing valid, fake and expired certificates from each other. The Enhanced OCSP, which is enhanced by Merkle trees, stores data for the three types of certificates in a database to facilitate the processing of these certificates by servers .The Merkle trees enabled to cut the protocol by almost half the time required to search for certificates, this is done By pairing certificates, the target certificate root obtained to provide more security for the servers in the future.

2. VERIFICATION OF CERTIFICATE REVOCATION BY USING MERKLE'S TREE

The use of hash tree like Merkel's tree considerably reduces the number of certificates that server should preserve to substantiate certificates validation and soundness as shown in Table 1, it also substantially specifies the size of network I/O bundles. Merkel's trees require small memory space, where evidence easily and quickly calculated as only require small and mixed part of certificates to substantiate their validity. Consistency verification, data verification, and data synchronization could be achieved by using Merkle's trees along with other systems, as shown in Figure 1. Merkle's trees are also known as audit proof because allow verifying including and removing certificates as well as consistency [9]. Server of CA which retains client record provides evidence on certificate existence within certificates record. It routinely operates to meet clients' requests through providing them with certificates needed using transport layer secure. If the Merkel's verification substantiation failed in constituting hash root which belongs to Merkel's hash consistency, this would either mean that certificate wasn't in CA record or was forged.

Table 1. Data arrangement in public key certificate using criterion X.509 v3

Field	Value
Version	V3
Serial number	03ba17c062511892710706466 35c6ea55eb4
Signature algorithm	Sha 256Rsa
Signature hash algorithm	Sha256
Issuer	let encrypt authority x3,let Encryptus
Valid from	12:38 2018/03/31
Valid to	12:38 2018/06/29

Merkel's outline in Figure 1 illustrates how consistent and documented the certificates have been from c1, c2...cn stored in node leaves. Let's assume that the tree height is H, i.e., the number of tree leaves is 2^H, and the number of nodes is 2^H-1, each unit of these nodes contains certification digits which represents h (ci), as (h) is hash function, (c) is certificate targeted. Each within node is derived from hash quotient for the sum

of two nodes e.g., R12=h (R1||R2) until tree is formed and nodes value is signed. Certification digits of nodes R12, R34 and root of R1234 are needed to document CA. Certification digits of h(c1) and digest values mentioned above can be sent to recipient for the purpose of verification. In this case, c1 certificate digest remains secure by server, thus, client might gain certificate root, generate private key derived from public key hash, and then access secure systems [10, 11]. Otherwise, certificate will be considered forged, not part of server and there will be no more future dealing with client, in case verification of generating private key derived from R1234 certificate root fails. Overwhelming speed in generating private key is attributed to very rapid mathematic algorithms that usually utilized by hash tree.

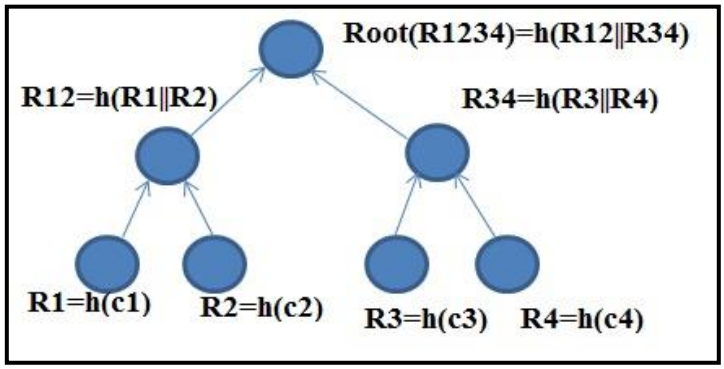

Figure 1. Merkel's hash tree from height of 2

2.1. Certificate arrangement by hash algorithm

Hash function [12, 13] is used to arrange and set positions of certificates within Merkel's tree through adopting (CRT position=Key Mod hash size). Let's assume that certificate key is C1=39, table size of certificates' hash=11, and by applying previous equation, certificate position will be in box (6) within Table 2(a). Then, certificates are rearranged in Merkel's tree based on values extracted from hash table in sequence. In case of collision in hash table as a result of producing two certificates at the same position within Merkel's tree, an equation of (CRT new position=(current position +1) Mod hash size) will be applied to disband collision status Table 2(b).

Table 2. Certificates keys arrangement within Hash table (a) in collision status (b) collision disbanding

KEY	23	18	29	28	39	13	16	42	17
Position	1	7	7	6	6	2	5	9	6

(a)

KEY	17	23	13			16	28	18	29	39	42
Position	0	1	2	3	4	5		6	7	8	9	10

(b)

2.2. Creation and deletion node in Merkel's tree for certification

Hash tree is characterized by consistency and arrangement in terms of adding and deleting certificates in its sections, that's why being effective in saving and arranging access to certificates [9]. Optimal number for this tree sections is logarithm as shown in (1), in which NO.p represents the optimal number of tree sections, while NO.c is the total number of revoked certificates in the system. Certificates are arranged in a new approach in these sections to make certificate search process in every section is less complicated in terms of time and performance. When any certificate is revoked, it should be incorporated in section concerned in the system within sequence of revoked certificates. Additionally, Hash value of sequential certificates in mother node is recalculated. Finally, searching process for certificate targeted necessitates finding and confirming the section where certificate is located using search algorithm, it is based on encoding Merkel's tree leaves to facilitate access to any section of the tree and calculate it according to the following equation [8].

$$NO.p=Log2(NO.c) \quad (1)$$

The most significant problems that servers confront are: finding certificate required whether being valid (within timeline set by timestamp without troubles with its own private key) or revoked for the same reason above. Another problem is time needed for gaining this certificate (within specified time for server response), it's essential factor to overcome some intrusions, as an intruder exploits a revoked certificate and the delay in server response to certificate status (which is a fundamental difference between OCSP and CRL that offers no immediate support for clients). Furthermore, adding new certificates and deleting revoked ones in server is an additional problem. This research has focused on some basics: improving server efficiency to function more effectively. Using extended Merkel's hash tree to substantiate whether certificate is valid or not [14]. It's been designed to resolve increasing number of certificates problem and classification of nodes in the tree Figure 2 to avoid having any problem with one of nodes during addition or deletion process [15]. Rapidity of response to identifying status of an electronic certificate required.

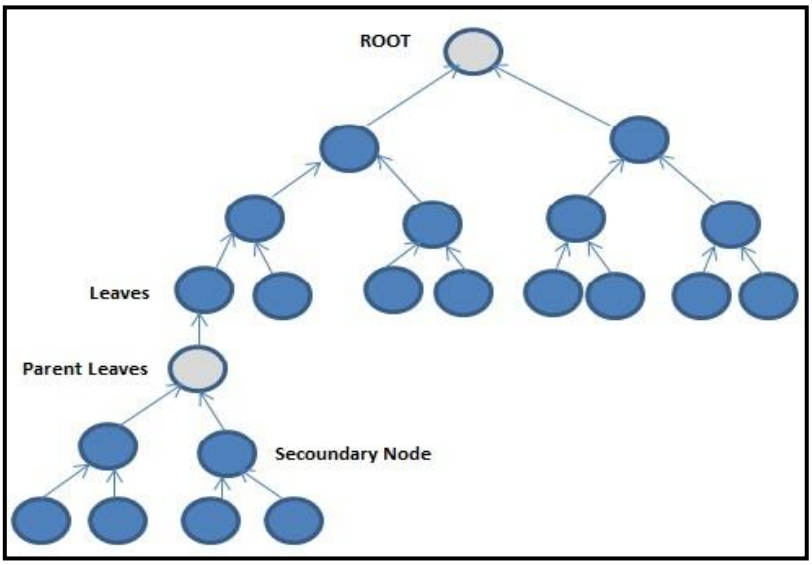

Figure 2. Extended Merkel's hash tree

In the world current systems, notary server [1] is used to ascertain certificates validity to obtain certificates' public key where client is authorized to access servers. Some information about user of private key derived from public key such timestamp is saved to provide further security features when verifying this key validation (key creation date and expiration). A client information when logging in is documented by notary server just like service protocol or service identifier, it provides communication information such as (host's name, service outlet). Notary server also documents a signature for each service obtained by a client using his private key in database [16-27]. Currently, certificates are audited using notary server to reveal whether are valid or revoked due to dysfunction in certificate's private key. Finally, certificate with validity time expired would be deemed as unknown.

3. PROPOSED SOLUTION

Proposed solution is an effective method designed to overcome problems in regard to searching and sorting revoked certificates out of valid ones. It also characterized by its rapidity in processing certificates, thus reducing server response time relying on the time of timestamp for each certificate (t_1, t_2,....t_n), as illustrated in Figure 3. Process of certificate validity searching is implemented in the solution based on searching method of Merkel's tree. Process of taking and placing certificates as a mass through cryptographic hash functions (CHF) for the purpose of obtaining hash is the first step in creating tree. Afterwards, all resulted hash processes of certificates are paired with each other to obtain a new hash. This process is continuously repeated until only one hash remains which is called Merkel's root. Through hash pairing process, Merkel's trees become very sensitive over any change in input parameters (certificates). If any change occurred on one of tree leaves, it would result in entire change on tree branch. Pursuing the tree branch to reach the root would help verify certificates.

According to the height of Merkel's tree H, the number of leaves in the tree can be identified which indicates certificates status that of 2^H, number of internal nodes 2^H -1 can also be identified. Certificate search process can be initiated to identify any section in the tree. Received certificate data can be matched to those in Merkel's tree within servers to verify it. This can only be achaived by obtaining the correct root of the tree through; pairing new certificate h (c1, t1) with the one to be verified h (c2, t2), extracting certificate digest and timestamp to obtain certificate root, Root= h (c1, t1||c2, t2). Certificate will be valid if the extracted root is matching, other than that, will be revoked and added to forged certificates. It's necessary to alert client, block thier access to the server, and no future dealing with him, Figure 4.

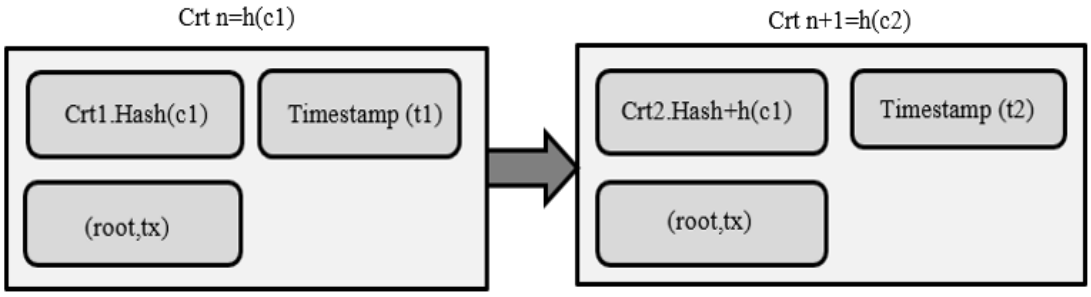

Figure 3. Searching for targeted certificates root in Merkel's tree within the time of timestamp in CA server

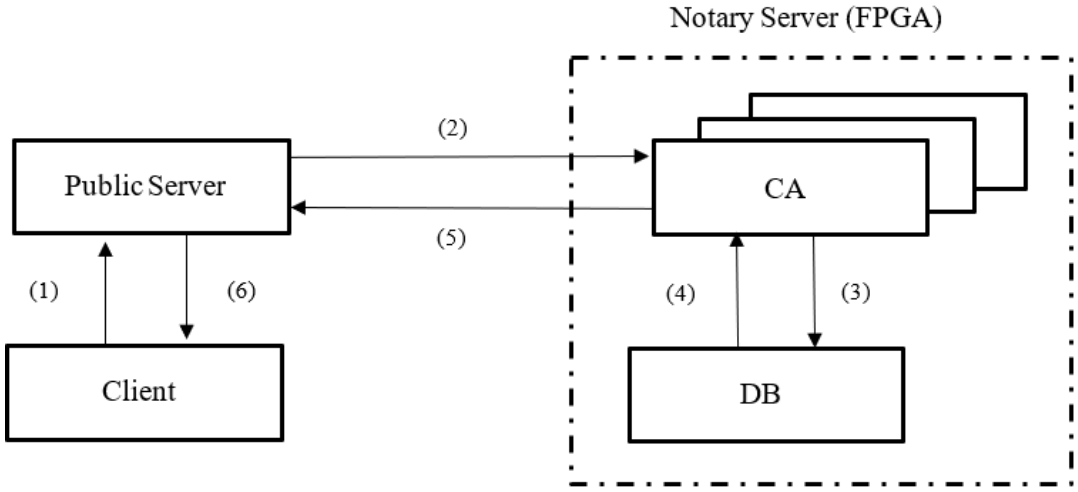

Figure 4. Connection process between client and server to verify digital certificate by CA

Certificates validity between clients and server can be verified through checking certificate of origin when submitted to create secure connection session SSL/TLS. This research has sought to develop an effective method to verify certificates depending on root extracted from certificate with assistance of Merkel's tree, as shown in flowchart Figure 5. Responder verifies certificates during certificate consistency assurance phase as previously mentioned in Figure 1. Then, sequential number in CA is checked to make sure that certificate is not existing in the system as documenting query, as to be noticed in certificate status phase. Having certificate input into proposed extension, each certification digits paired with another to obtain a node in the tree, this process continues until obtaining certificate root by Merkel's tree root. Thus, validity of targeted certificate has been confirmed, where responder can send a message to client and open SSL / TLS session. If certificate sequential number was fake, certificate would not be confirmed by CA. Hence, it would be deemed as denied and fake certificate and included in revoked certificates section in Merkel's tree. Moreover, it would send a warning message against future dealing with this person. Any problem with certificate's public key will result in increasing timestamp. In this case, certificate will be unknown and its root is not extracted. Client is required to resend it again using certificate's public key in case first verification attempt fails.

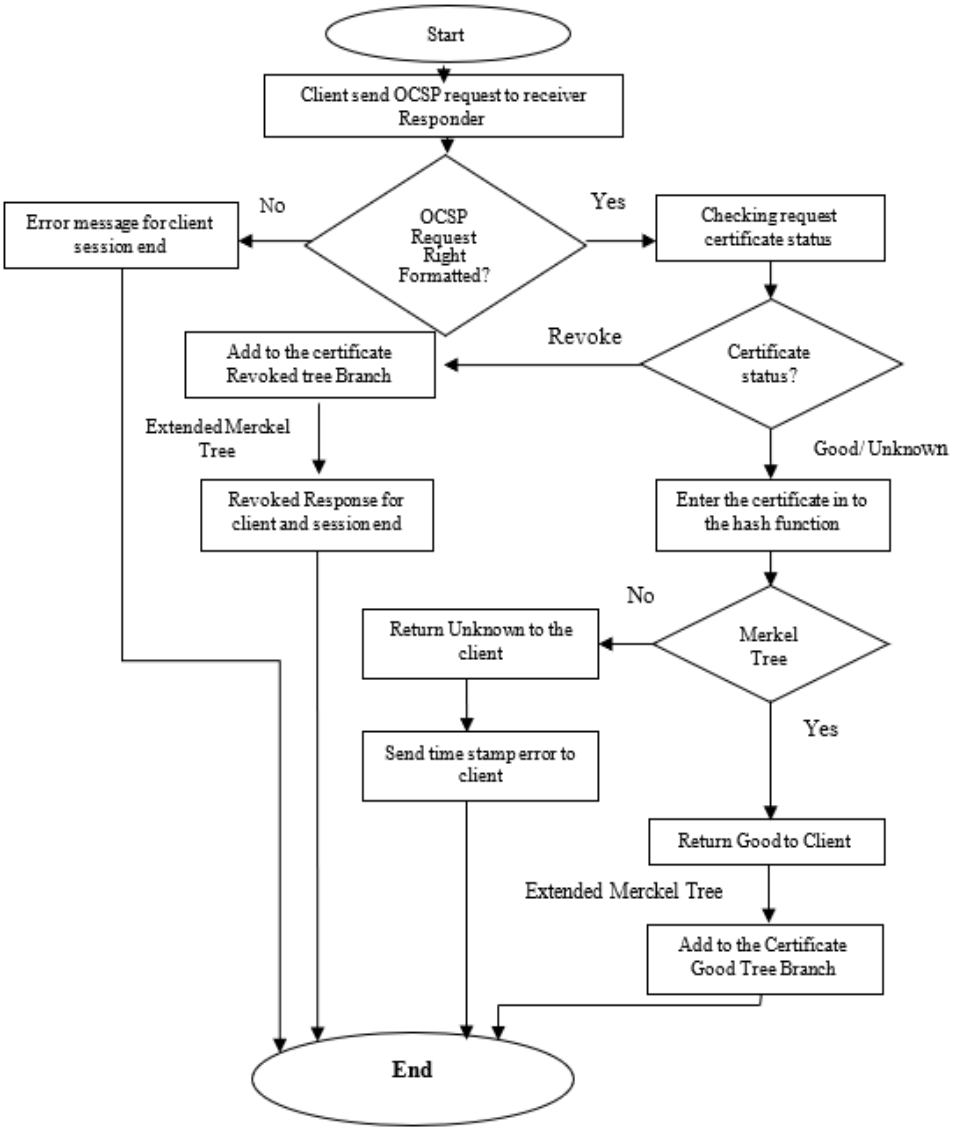

Figure 5. Proposed flowchart of OCSP responder to verify certificates using extended Merkel's tree, works on sorting certificates out in different cases

4. SIMULATION RESULTS AND DISCUSSION

In this section, certificates entered in the Enhanced OCSP where the latter designed on the FPGA board, as well as simulation results improved OCSP obtained, using extended Merkel trees, all three types of certificates classified and distinct. Regardless of the communication process. Here we will commit to focus on the way these certificates are handled by the extended Merkel trees, placing the target certificates to be verified in blocks as shown in Figure 7, Where the blocks of certificates placed in temporary stores designated for this purpose such as (First parente node,Second parente node…etc) and also these blocks are passed on to the hash algorithm (CFH) to get the message summary (Digest). a digest is a value that cannot be repeated for two different certificates, as the process of the arranging certificates in the form of pairs depends directly on the digest value as shown in Figure 8. Through the use of the algorithm (SHA1), the pairing process fragmented to obtain a new hash, this process continues until the correct root of the required certificate reached, and the time stamp of the certificate also regarded in case of the validity examined, As noted, the needed time to verify certificates halved by using extended Merkel trees. The storage space for these certificates reduced by obtaining the required root certificates, which made the Enhanced Protocol better in terms of speed and storage compared to the standard protocol currently in place.

Responder of OCSP has been modelled to verify certificates of origin using FPGA, as shown in Figures 6 and 7, and to make amendments to OCSP to ensure too rapidity in gaining best results that prevent attacker from analyzing certificate and reaching the root during the time of timestamp. Having confirmed

certificate sequential number by CA, "http" would send certificate. Certificate will be directly denied and recorded within fake certificates category in DB database when its sequential number is not existing within CA. But when the sequential number is correct and existing within CA database, client's request inputs into CHF to gain certification digits. Then, certification digits is paired with other peers within Merkel's tree to obtain certificate root. Having obtained certificate root, "http" would send client a signal over certificate acceptance. When certificate obtaining takes more time than timestamp, a signal is sent to client that certificate being unknown and he can resend the request again. Merkel's tree extends whenever new certificates are registered based on certificate status after checking and ascertaining the root of origin.

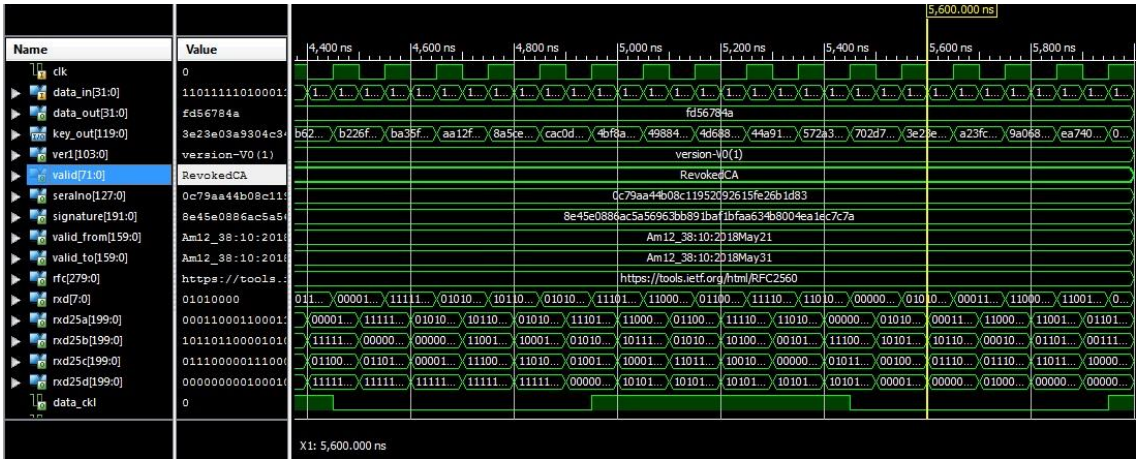

Figure 6. Certificate of origin X509 v3 within proposed OCSP responder illustrates period expiration-based denied certificates

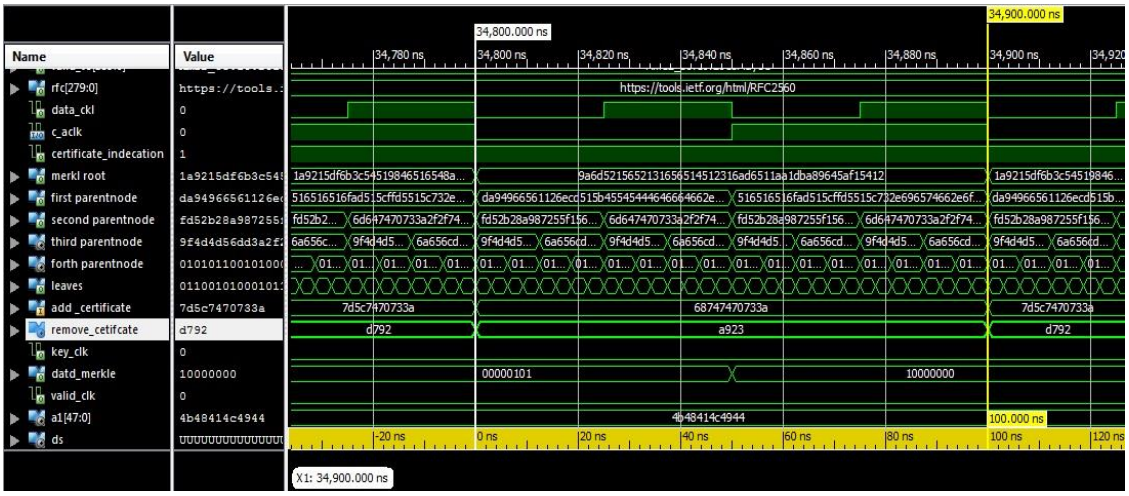

Figure 7. Certificate of origin (X509 v3) input into hash algorithm and arrange certificates as pairs within Merkel's tree in CA servers of OCSP responder

Responder of OCSP has been designed based on FPGA to respond to identifying certificate status as shown in Figure 6. Certificate is entirely displayed with all the details from copy number referred to "ver1", certificate status referred to "valid", sequential number and digital signature referred to "Serial No. & Signature", and finally the time of timestamp of certificate referred to "valid from, valid to". Figure 7 illustrates how to modelling certificates in pairs within Merkel's tree arranged in single vector so to pair with a neighboring certificate for producing nodes and then reaching tree root. Figure 8 demonstrates certificate status and extracting image of certification digits referred to "SHA1" which is image of certificate's electronic signature to ensure no certificate forgery.

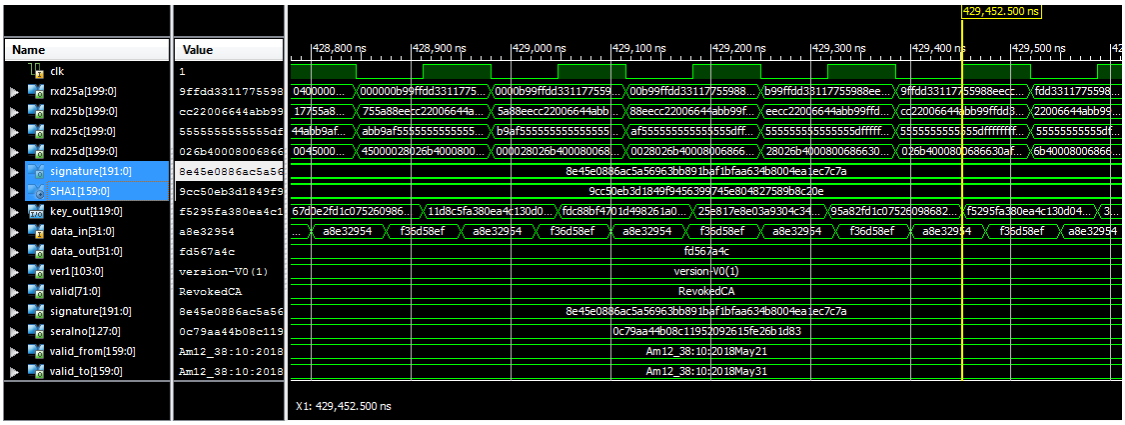

Figure 8. Certificate of origin during the formation of Merkel's tree, arrangement of certificates in pairs, and certification digits extraction by algorithm SHA1 used in electronic signature

5. PERFORMANCE ANALYSIS

This section provides comparative analysis for some coefficients adopted by networks for measuring efficiency and quality between std. OCSP [4] and OCSP responder proposed based on FPGA. Figure 9 demonstrates the difference in processing the number of certificates used in session. It's noted that the use of proposed model could process much greater number of certificates. This is attributed to the use of hash merlel technique which reduces the time almost by half through pairing among certificates to obtain nodes and eventually reach certificate's root. Process of pairing each two certificates aims at simultaneously verifying them which allows processing a greater number of certificates at the same time.

According to the comparison of throughput between OCSP responder with std. OCSP [7] in Figure 10, has showed that OCSP responder outperformed over std. OCSP because the certificate to be verified had been input into SHA1. As a result, a fixed-length image is generated which is a digest where pairing process between couples is occurred to produce Merkel's tree nodes. Given that SHA1 is responsible for arranging and specifying the length of certificates, thus, it allows processing much amount of data in each single certificate according to its length with less storage space when saved.

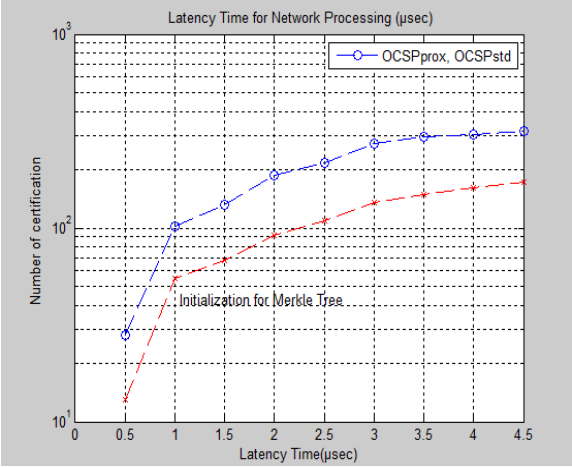

Figure 9. Diagram represents the number of certificates used by OCSP responder within unit of time

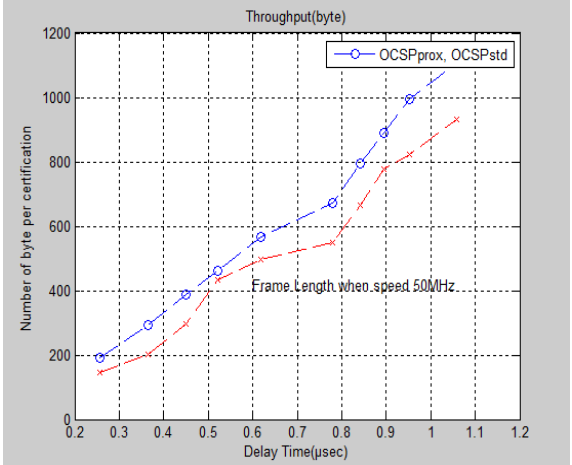

Figure 10. Diagram illustrates throughput and delay time of processing for OCSP responder

6. CONCLUSION

Verification of certificates revocation is crucial, and considered part of online certificate validity verification. There are several methods adopted in verifying certificates, but the most significant are certificates revocation lists (CRL) and online certificate status protocol (OCSP). Nevertheless, they suffer tardiness in dealing with certificate status in addition to lack of large storage space. This research has

provided a strategy to overcome the delay in verification as well as provided large storage space. Through using extended Merkle's tree, 50% of delay time has been saved. While, the use of Certificate Digest has reduced storage space in servers, that would reduce the probability of attack on servers by fake or revoked certificates due to their period of time expiration.

REFERENCES

[1] C. Ekechukwu, D. Lindskog and R. Ruhl, "A notary extension for the online certificate status protocol," *International Conference on Social Computing*, pp. 1016-1021, 2013.

[2] A. A. Chariton, E. Degkleri, P. Papadopoulos, P. Illia and E. P. Markatos, "CCSP: A compressed certificate status protocol," *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1-9, 2017.

[3] Y. Xiaoqing, and W. Ming, "Design of IKEv2 protocol based on the PKI/OCSP," *International Conference on Computer Science and Information Processing (CSIP)*, pp. 1357-1360, 2012.

[4] Q. Hu, M. R. Asghar, and N. Brownlee, "Certificate revocation guard (CRG): An efficient mechanism for checking certificate revocation," *IEEE 41st Conference on Local Computer Networks*, pp. 527-530. 2016.

[5] M. Myers, and H. Tschofenig, "Online certificate status protocol (OCSP) extensions to IKEv2," *RFC 4806*, pp. 1-9, 2007.

[6] D. Kouril, L. Matyska and M. Prochazka, "A robust and efficient mechanism to distribute certificate revocation information using the grid monitoring architecture," *21st International Conference on Advanced Information Networking and Applications Workshops*, vol. 1, pp. 614-619, 2007.

[7] P. Szalachowski, L. Chuat and A. Perrig, "PKI safety net (PKISN): Addressing the too-big-to-be-revoked problem of the TLS ecosystem," *IEEE European Symposium on Security and Privacy*, pp. 407-422, 2016.

[8] R. M. Annavajjala and V. Anand, "Partition based hash tree-An efficient certificate revocation system," *IEEE International Conference on Electro Information Technology*, pp. 551-556, 2017.

[9] J. L. Muñoz, *et al*., "Certificate revocation system implementation based on the Merkle hash tree," *International Journal of Information Security*, vol. 2, no. 2, pp. 110-124, 2004.

[10] A. Hülsing, *et al*., "XMSS: Extended hash-based signatures," *Crypto forum research group internet-draft, draft-irtf-cfrg-xmss-hash-based-signatures-01*, 2015.

[11] E. Mykletun, M. Narasimha and G. Tsudik, "Providing authentication and integrity in outsourced databases using Merkle hash trees," *UCI-SCONCE Technical Report*, 2003.

[12] H. Handschuh and B. Preneel, "Key-recovery attacks on universal hash function based MAC algorithms," *Annual International Cryptology Conference, Springer*, pp. 144-161, 2008.

[13] Quisquater, Jean-Jacques, and Marc Girault, "2n-bit hash-functions using n-bit symmetric block cipher algorithms," *Workshop on the Theory and Application of of Cryptographic Techniques, Springer*, pp. 102-109, 1989.

[14] A. Huelsing, *et al*. "Xmss: extended merkle signature scheme," *RFC 8391*, pp. 1-7, 2018.

[15] D. Williams and E. G. Sirer, "Optimal parameter selection for efficient memory integrity verification using merkle hash trees," *Proceedings Third IEEE International Symposium on Network Computing and Applications*, pp. 383-388, 2004.

[16] D. Butin, "Hash-based signatures: State of play," *IEEE Security & Privacy*, vol. 15, no. 4, pp. 37-43, 2017.

[17] A.U. Schmidt, *et al*., "Tree-formed verification data for trusted platforms," *Computers & Security*, vol. 32, pp. 19-35, 2013.

[18] C. Liu *et al*., "MuR-DPA: Top-down levelled multi-replica merkle hash tree based secure public auditing for dynamic big data storage on cloud," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2609-2622, 2015.

[19] C. Papamanthou, R. Tamassia and N. Triandopoulos, "Authenticated hash tables," *Proceedings of the 15th ACM conference on Computer and communications security*, pp. 437-448, 2008.

[20] R. J. Bayardo and J. Sorensen, "Merkle tree authentication of "http" responses," *Special interest tracks and posters of the 14th international conference on World Wide Web*, pp. 1182-1183, 2005.

[21] K. Ivanov, "Autonomous collision attack on OCSP services," *Cryptography and Security*, 2016.

[22] S. Koga and K. Sakurai, "A distributed online certificate status protocol with a single public key," *International Workshop on Public Key Cryptography. Springer*, pp. 389-401, 2004.

[23] Authority, "Federal PKI Policy," *Shared Service provider roadmap: Navigating the process to acceptance*, pp. 1-5, 2007.

[24] Jan, Štefkovič, "Network Traffic Analysis of OCSP Protocol," *Masaryk University Faculty of Informatics*, 2017.

[25] M. Myers, "Internet Public Key Infrastructure-Online Certificate Status Protocol-OCSP," RFC 2560-X. 509, 1999.

[26] A. A. Jaber, F. K. I, Al-Mousawi and H. S. Jasem, "Internet of things based industrial environment monitoring and control: a design approach," *International Journal of Electrical & Computer Engineering*, vol. 9, no. 6, pp. 4657-4667 , 2019.

[27] H. T. Zaw, A. H. Maw, "Traffic management with elephant flow detection in software defined networks (SDN)," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. 3203-3211, 2019.