A Study on Batch Verification Scheme in Outsourced Encrypted Database

Giau Ho Kim
Le Quy Don Technical University
Hanoi, Vietnam
hkgiau@gmail.com

Manh Tran Cong
Le Quy Don Technical University
Hanoi, Vietnam
manhtc@gmail.com

Minh Nguyen Hieu
Institute of Cryptography Science and Technology
Hanoi, Vietnam
hieuminhmta@gmail.com

ABSTRACT

Outsourced database service helps data owners save initial investment in hardware, software and they are always supported by experienced staff. However, because the database is stored in the service provider's server and data owner(DO) have to retrieve data through the Internet environment, the owner's data may not be secure. A good way to protect their data is that data always are encrypted before storing. This leads to when data is queried, the owner must make sure returned data is valid. To do that, the returned results have to be verified. In this paper, we propose a batch verification scheme based on multiple hard problems, and offer that scheme to verify the outsourced encrypted database. Analysis and experimental results present the effectiveness of the proposed method in the case of dynamic database.

CCS CONCEPTS

• **Security and privacy** → *Management and querying of encrypted data.*

KEYWORDS

Outsourced database, Encrypted data, Batch verification, Encrypted database verification

ACM Reference Format:
Giau Ho Kim, Manh Tran Cong, and Minh Nguyen Hieu. 2019. A Study on Batch Verification Scheme in Outsourced Encrypted Database. In *The Tenth International Symposium on Information and Communication Technology (SoICT 2019), December 4–6, 2019, Hanoi - Ha Long Bay, Vietnam, Viet Nam.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3368926.3369732

1 INTRODUCTION

Cloud computing provides many utility services to it's client. In which, outsourced database service (ODBS) provides storage, retrieval and maintenance database. Customers take the advantage of strong capabilities from hardware, professional staff... from service providers. However, they also face many risks because service providers may have direct access or attack to their customers data or illegal intrusions from the Internet environment. Since then, the issue of database protection on the outsourced environment has been concerned and sought to resolve.

In the outsourced model, Data Owners (DO) need a Database Service Provider (DSP). DO shares his data with users via the DSP server. To protect its data against data leakage, DO can encrypt the data before storing it on the DSP server. Concurrent with that, the DO must protect the integrity of the data. This means that when the data is illegally tampered (updated, deleted...), the DO must detect and process it to ensure that when the DSP returns the data to the user, the DO must verify that data is original version which is put on DSP by DO.

Related work. To determine if the data is valid, when returning results, DO must verify the results and they must be original with the data that DO created. The DO(s) must detect and process any illegally intervened (insert, update, deleted...) to there data. There are many studies on data validation on ODBS, including suggestions such as:

- Use the Authenticated Data Structure (ADS) [6, 8, 14]. The ADS structure will take a long time to recalculate if data is changed. Thus, ADS is not effective for dynamic database due to the time of structural change of ADS is very large. As long as a record is inserted, updated or deleted, ADS is required to rebuild its authentication structure.

- Mykletun [7] introduces a digital signature (Condensed-RSA) to reduce computation time, but does not recommend a query processing method. Then, Yuan [13] proposed a validable query query scheme for the outsourced database. Each record is assigned an authentication tag to check the integrity of the aggregated query results. However, this method only validates on plain data and must check the tags of records related to the query. This means that the system must recalculate the signature of the record even though the query only takes a few attributes from the record.

Outsourcing database research often build support structures for authentication, so if dynamic databases are costly to recalculate. In addition, some studies only support specific types of queries that do not support queries across multiple tables; Or the studies only propose a separate authentication method with the decryption process, then, after authentication, DO takes more time to conduct the decoding, returning plain data to the user. Within the scope of this study, our targets focus on the authentication of returned data without suggesting a method of querying on encrypted data. DSP returns encoded data for each query. The proposed method will check the returned results, if the returned data is valid, they will

SoICT 2019, December 4–6, 2019, Hanoi - Ha Long Bay, Vietnam, Viet Nam
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-7245-9/19/12…$15.00
https://doi.org/10.1145/3368926.3369732

be decrypted and return the plain results to the user. Our method is very effective in the case of dynamic database because there is no need to rebuild authentication structures. In addition, the proposed algorithms are concurrently verification with decrytion so the verification time is not much different from non-authenticating (without authentication, it still has to decrypt the data).

Organization. The rest of the paper is organized as follows: In Section 2, we introduce digital signature and batch verification. In Section 3, we propose batch verification scheme based on the multiple hard problems. In Section 4, we propose verification algorithms to verify outsourced encrypted database. In Section 5, we analyze time complexity and performance of algorithms and Section 6 gives our conclusions.

2 BATCH VERIFICATION OVERVIEW

2.1 Digital signature

Digital signatures are used to verify the origin and integrity of information. There are some common digital signatures such as: RSA, DSA, GOST.... RSA digital signatures are based on the hard of the prime factorization analysis problem, while the remaining digital signatures are based on the hard of the discrete logarithm problem.

RSA digital signature algorithm. R.L.Rivest, A.Shamir and L.Adleman proposed a digital signature scheme named RSA [9]. The parameters in the schema are generated by randomly choosing large prime numbers $p, q, p \neq q$ and computing $n = pq$.

- Key generation: private key is d, public key is (e, n).
  1. Let $\phi(n) = (p-1)(q-1)$
  2. Choose e randomly where $gcd(e, \phi(n)) = 1$.
  3. Compute $d = e^{-1} \bmod \phi(n)$.
- Signature generation: generate signature σ for message $m \in \{0,1\}^*$
  Compute $\sigma = m^d \bmod n$.
- Verification: check signature σ true of false.
  1. Calculate $m' = \sigma^e \bmod n$.
  2. If $m' = m$, return 1 (true), else return 0 (false).

Schnorr digital signature algorithm. Schnorr proposed a digital signature algorithm [10]. Given the parameters p, q, g on the discrete logarithm problem (DLP), p is a large prime number in the range from 512 bits to 1024 bits, q is a prime divisor of $p-1$ is 160 bits in size, a generator element g has order q in $\mathbb{Z}_p$.

- Key generation:
  Choose randomly $x \in 1, ..., q-1$, compute $y = g^x \bmod p$. Private key is x, Public key is y
- Signature generation: generate signature σ for message $m \in \{0,1\}^*$
  1. Choose randomly $t \in \mathbb{Z}_q$, calculate $r = g^t \bmod p$.
  2. Compute $h = H(m||r)$.
  3. Compute $s = t - hx \bmod q$.
  4. Output $\sigma \leftarrow (h, s)$.
- Verification: check signature σ true of false.
  1. Compute $r' = g^s y^h (\bmod p)$.
  2. Compute $h' = H(m||r')$.
  3. If $h' = h$, return 1 (true), else return 0 (false).

2.2 Batch verification

Batch verification is a form of verification multiple digital signatures at the same time. The purpose of batch verification is to reduce the cost of computation compared to each signature verification, and it is suitable for systems that require a large number of signatures at the same time [1, 3]. Batch verification is applied in many range of authentication such as: automatic monitoring system, wireless transmission control, Internet of things...

Harn et al., proposed batch verification scheme based on RSA [4]. The parameters are two large primes p and q, compute $n = pq$. e is a private key and d is a public key where $ed = 1 \bmod \phi(n)$. Assume that k signatures $(\sigma_1, \sigma_2, ..., \sigma_k)$ corresponding to k messages m_i, where $\sigma_i = H(m_i)^d \bmod n$ $(1 \leq i \leq k)$. To verify k signature at the same time, Harn et al., checks the equation: $(\prod_{i=1}^{k} \sigma_i)^e = \prod_{i=1}^{k} H(m_i) \bmod n$, if that true then k signatures σ_i are valid.

Shao proposed batch verification scheme based on Schnorr signature scheme [11]. Assume that k messages were signed by Schnorr's Signature generation, that are k signatures $\sigma_i(r_i, s_i), i = 1, 2, ..., k$, $i = 1, 2..., k$ were signed by the same private key x. To verify k at the same time, Shao chooses randomly k integers $u_i \in (1, 2^{32})$, $i = 1, 2, ..., k$ and checks an equation:
$\prod_{i=1}^{k} r_i^{u_i} = (g^{\sum_{i=1}^{k} u_i s_i} y^{\sum_{i=1}^{k} u_i H(m_i||r_i)}) \bmod p$, if it true, the k signatures are valid.

The previous proposed batch verification algorithms are often based on the difficult of integer factorization or discrete logarithm problems. The advantage of these algorithms is the fast signature authentication time. To increase the security of signatures, scientists proposed the idea of combining multiple hard into the digital signature scheme [2, 5, 12]. However, its disadvantages will increase computational complexity. In many specific cases, the system needs to be considered between improving security and processing speed to choose the appropriate authentication algorithm. There are many digital signature schemes based on multiple hard problems, but according to the our survey, there is not yet a scheme to validate lots based on multiple hard problems. Proposing a lot verification scheme based on multiple hard problems will get the safety from the multiple hard problems and fast verification time from the batch verification.

3 BATCH VERIFICATION SCHEME PROPOSAL

In this section, we propose the batch verification schemes based on the multiple hard problems: factorization and discrete logarithm. The main purpose of proposing is to increase the security of the signature. Solving multiple hard problems simultaneously is more difficult than solving one hard problem. With our method, in case of solving a hard problem, the signature scheme is still safe. The batch verification schema bases on RSA-Schnorr scheme. To break this signature scheme requires two simultaneous hard problems to solve: discrete logarithm calculation on the $\mathbb{Z}_p$ field and the factorization analysis n. Parameters used in the schema: Choose two large prime numbers q, q', compute $n = qq'$ so that $p = 2n + 1$ is prime, the generator g has order n, value $\phi(n) = (q-1)(q'-1)$.

Algorithm for keys generation, signature generation, signature verification and batch verification as follows:

- Keys generation:

Algorithm 1: Public key, private key generation

```
1 Choose $x$ randomly: $x \in \mathbb{Z}_p^*$
2 Compute $y = g^x \bmod p$, if $gcd(y, \phi(n)) \neq 1$, go back step 1
3 Compute $d$: $yd = 1 \bmod \phi(n)$
4 $y$ is a public key. $(d, x)$ are private key
```

- Signature generation $S(m)$:

Algorithm 2: Generate signature σ for message $m \in \{0, 1\}^*$

```
  Input: Message $m$
  Output: Signature $\sigma$
1 Choose randomly $t \in \mathbb{Z}_n^*$, compute $r = g^t \bmod p$
2 Compute $s = (t - H(m||r)x)^d \bmod n$
3 Signature $\sigma \longleftarrow (r, s)$
```

- Signature verification:

Algorithm 3: Verify signature $\sigma(r, s)$ for message m

```
  Input: Message $m$, signature $\sigma(r, s)$
  Output: 1 (true) or 0 (false)
1 Compute $s' = s^y \bmod n$
2 Compute $r' = g^{s'} y^{H(m||r)} \bmod p$
3 If $r' = r$ then return 1 (true), else return 0 (false)
```

- Batch verification $V(\sigma_i)$:

Algorithm 4: Verify k signatures $\sigma_i(r_i, s_i)$ for k messages $m_i, i = 1, 2, ..., k$, that were signed by the same private key

```
  Input: Messages $m_i$, $k$ signatures $\sigma_i(r_i, s_i)$, $1 \leq i \leq k$
  Output: 1 (true) or 0 (false)
1 $s'_i = s_i^y \bmod n$
2 $u = \sum_{i=1}^k s'_i$
3 $v = \sum_{i=1}^k H(m_i||r_i)$
4 if ($\prod_{i=1}^k r_i = g^u y^v \bmod p$) then return 1
5 else return 0
```

Proof. According to algorithm **??**, if k signatures $\sigma_i(r_i, s_i)$ are valid then each signature is satisfied: $r = g^{s'} y^{H(m||r)} \bmod p$ where $s' = s^y \bmod n$.

$r_1 = g^{s'_1} y^{H(m_1||r_1)} \bmod p$

$r_2 = g^{s'_2} y^{H(m_2||r_2)} \bmod p$

$\vdots$

$r_k = g^{s'_k} y^{H(m_k||r_k)} \bmod p$

$\Rightarrow \prod_{i=1}^k r_i = g^{s'_1} y^{H(m_1||r_1)} \dots g^{s'_k} y^{H(m_k||r_k)} \bmod p$

$= g^{s'_1 + \dots + s'_k} y^{H(m_1||r_1) + \dots + H(m_k||r_k)} \bmod p$

$= g^{\sum_{i=1}^k s'_i} y^{\sum_{i=1}^k H(m_i||r_i)} \bmod p$

Security analyses. The proposed scheme will be analyzed the possible attacks. We show that if the adversary (adv) want to break the signature scheme, they have to solve the factorization problem and the discrete logarithm problem simultaneously.

- *Attack 1.* Adv intends to retrieve the private key from the public key. To derive private key x from y, adv has to solve the discrete logarithm $y = g^x \bmod p$. To reveal the private key d from y, adv needs to factorize n into two large primes q and q'.
- *Attack 2.* In case adv solve the discrete logarithm $y = g^x \bmod p$ and retrieve a private key x, but Adv cannot compute s to have a valid signature, because adv don't have a private key d.
- *Attack 3.* In case adv factorize n into two large primes q and q' to reveal private key d, but adv want to have a valid signature, adv must have a private key x.

4 OUTSOURCED ENCRYPTED DATABASE VERIFICATION PROPOSAL

In this section, we propose algorithms to verify the encrypted data returned from the DSP is valid. In order to validate the data returned when querying ODBS, the proposed model with four components: data owner, service provider, user (Querier) and proxy server. In it, DO is the one who creates the database and signs on the data with his secret key. DO shares data with users. Proxy servers are servers owned by DO, outside the scope of DSP. Assuming the proxy server is trusted. The proxy server does not store the database, it only performs the task of processing authentication and decoding the results returned to the user. The outsourced encrypted database verification model is as shown in figure 1.

The process of authenticating an encrypted database outsides through three stages:

- The key generation: The data owner chooses the key to use in the entire process of creating and validating the data. Key generation is handled outside of DSP and managed by DO;
- Database storing: Create a signature for data, then encrypt the data before archiving on DSP.
- Query data validation: authentication and decryption, return plain results to users.

Given a database D containing many tables. Suppose table T has m columns and n records $r = (r_{i1}, r_{i2}..., r_{im})$, where r_{ij} is the data at the i-th row and the j-th column $(1 \leq i \leq n, 1 \leq j \leq m)$. To avoid the case of Adv swapping values between records, we adds a *AutoNum* data field (automatically increasing number) to the data tables. Adding the *AutoNum* field does not affect the table structure of the original database. When querying data, the SQL statement will be rewritten by the proxy server to force DSP to return the *AutoNum* field. The proxy server verify and decrypt the results. Choose randomly and different prime numbers q, q', compute $n = qq'$ so that $p = 2n + 1$ is a prime number. The element generator g has order n, value $\phi(n) = (q - 1)(q' - 1)$. The stages of outsourcing database authentication activities are as follows:

- Key generation as describes in algorithm 5
- Database storing: each table $T \in D$, the processing to store encrypted database to DSP server as algorithm 6

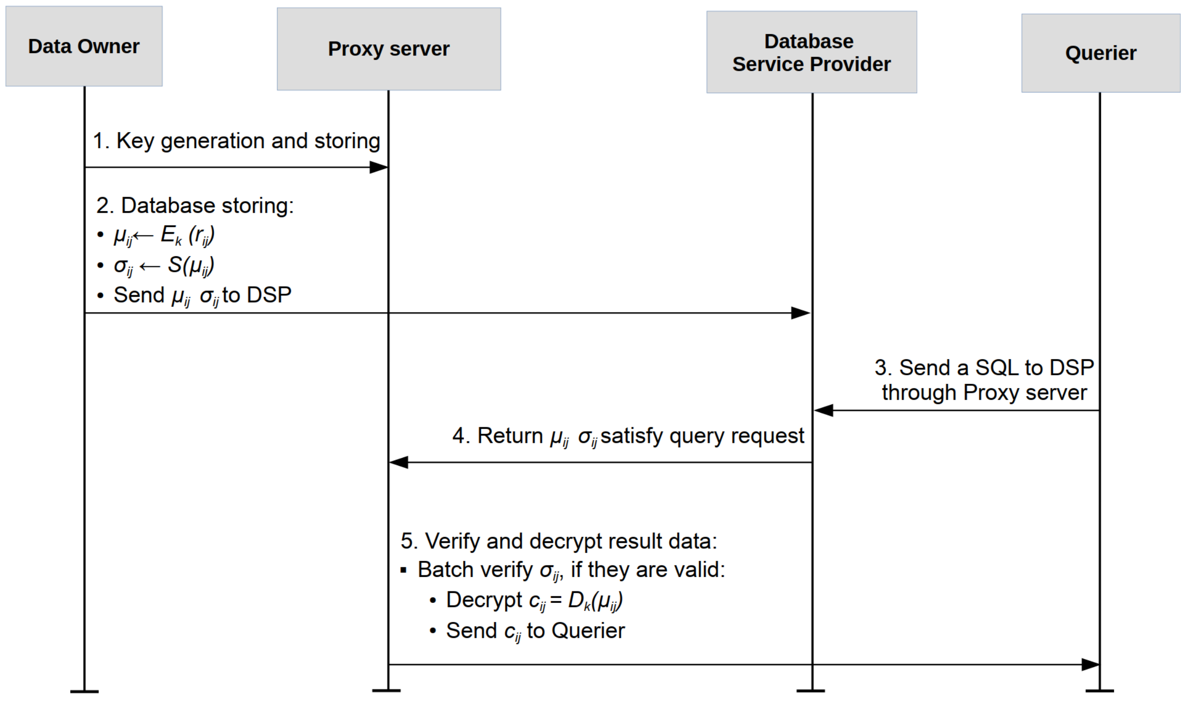
Data Owner
Proxy server
Database Service Provider
Querier
1. Key generation and storing
2. Database storing:
• $\mu_{ij} \leftarrow E_k(r_{ij})$
• $\sigma_{ij} \leftarrow S(\mu_{ij})$
• Send μ_{ij} σ_{ij} to DSP
3. Send a SQL to DSP through Proxy server
4. Return μ_{ij} σ_{ij} satisfy query request
5. Verify and decrypt result data:
▪ Batch verify σ_{ij}, if they are valid:
• Decrypt $c_{ij} = D_k(\mu_{ij})$
• Send c_{ij} to Querier

Figure 1: Outsourced encrypted database verification model.

- Query results verification as describes in algorithm 7: When a user sends a query to the DSP through a proxy server, the DSP server returns the data $T_r = \{AutoNum, \mu_{ij}, \sigma_{ij} | i = 1, 2, ...h_T; j = 1, 2, \ldots, k_T\}$ to proxy server. The proxy server verifies the results, if they are valid, the proxy server decrypts results before returning to user.

Algorithm 5: Public key, private key generation

```
1 Choose randomly $k$ to encrypt/decrypt data
2 Choose randomly $x$ where $x \in \mathbb{Z}_n^*$
3 Compute $y = g^x \bmod p$, if $gcd(y, \phi(n)) \neq 1$, go back step 2
4 Compute $d$ where $yd = 1 \bmod \phi(n)$
5 Public key is $y$. Private key are $(k, d, x)$
```

Algorithms 7 show that the returned results are validated based on signature values, so the proposed algorithm does not need supporting objects. Therefore, the proposed method is effective for the dynamic database (inserted, updated...)

5 PERFORMANCE ANALYSIS

The time taken for key generation, signature generation, and ODBS authentication stages depends on the number of calculations. Which focuses on the time to perform exponential operations, modulo multiplication, time to calculate hash functions, coding time, decryption and ignores modulo addition operations because the execution time is insignificant.

Algorithm 6: Storing encrypted database to DSP server

```
Input: Table $T$, private key
Output: Store encrypted database to DSP server
1 $n = qq'$
2 for $i = 1$ to $n_T$ do
3     for $j = 1$ to $m_T$ do
4         Choose randomly $t \in \mathbb{Z}_n^*$, $r = g^t \bmod p$
5         $s = (t - H(AutoNum||r_{ij}||r)x)^d \bmod n$
6         $\sigma_{ij} \leftarrow (r, s)$
7         $\mu_{ij} = E_k(r_{ij})$
8         $d_i \leftarrow \{\mu_{ij}, \sigma_{ij}\}$
9     end
10    $T_i' \leftarrow \{d_i\}$
11 end
12 Store table $T'$ to DSP server.
```

The execution time of the proposed algorithm stages is as table 2. In which, k is the number of signatures to verify at the same time.

The implementation phase of database authentication on ODBS, in addition to creating and validating signatures, it also adds the process of encoding - decoding data. Assuming the database has t_1 tables, each table has up to n_T rows and m_T columns. When querying data, the DSP server returns t_2 tables, each with up to h_T rows, k_T columns. The time complexity for proposed outsourced encrypted database verification algorithms is described as table 3.

Algorithm 7: Results verification and decryption

```
Input: Table $T_r$, private key $k$, public key
Output: Plain results
for $i = 1$ to $h_T$ do
    $u = 0; v = 0; r = 1;$
    for $j = 1$ to $k_T$ do
        $a_{ij} = D_k(\mu_{ij})$
        $s' = s_{ij}^{y} \bmod n$
        $u += s'$
        $v += H(AutoNum||a_{ij}||r_{ij})$
        $r* = r_{ij}$
        $d_i \leftarrow \{a_{ij}\}$
    end
    if $(r \bmod p == g^u y^v \bmod p)$ then $T_i \leftarrow \{d_i\}$
end
Return $T$ to user
```

Table 1: Definition of notations

T_{mul}	Time for executing the modular multiplication
T_{exp}	Time for executing the modular exponentiation
T_{srt}	Time for executing the modular square root computation
T_H	Time for performing hash function
T_E	Time for performing encryption
T_D	Time for performing decryption

Table 2: Time complexity comparison of the signature scheme based on multiple hard problem

Items	Proposed scheme	[5] scheme
Signature generation	$2T_{exp} + T_{mul} + T_H$	$3T_{exp} + 3T_{mul} + 2T_{srt} + T_H$
Signature verification	$3T_{exp} + T_{mul} + T_H$	$4T_{exp} + 2T_{mul} + T_H$
Batch verification	$k(T_{exp} + T_{mul} + T_H) + 2T_{exp} + T_{mul}$	-

Table 3: Time complexity for outsourced encrypted database verification

Items	Time complexity
Database storing	$t_1 n_T m_T (2T_{exp} + T_{mul} + T_E + T_H)$
Results verification	$t_2 h_T (k_T (T_{exp} + T_{mul} + T_D + T_H) + 2T_{exp} + T_{mul})$

For the purpose of experimentation, the dissertation installed the algorithms in the Python programming language and performed them on the Core ™ i3-2375M CPU@1.50GHz×4, 8GB RAM, Ubuntu 18.04 operating system.

Time processing to create the signatures of the proposed algorithms on the corresponding number of messages such as tables 4.

Table 4: Time processing of propose algorithm

Messages	Signature generation
10	0,03501
50	0,158229
100	0,311561
300	0,905676
500	1,473525

When verifying data, the data in returned record is verified one by one. However, this takes longer than verifying the data in the record at the same time. To clarify the effectiveness between batch and serial verification methods, we conducts time-tested on two methods: serial verification of k signatures based on algorithm 3 and batch verification based on algorithm 4, the result is as shown in figure 2.

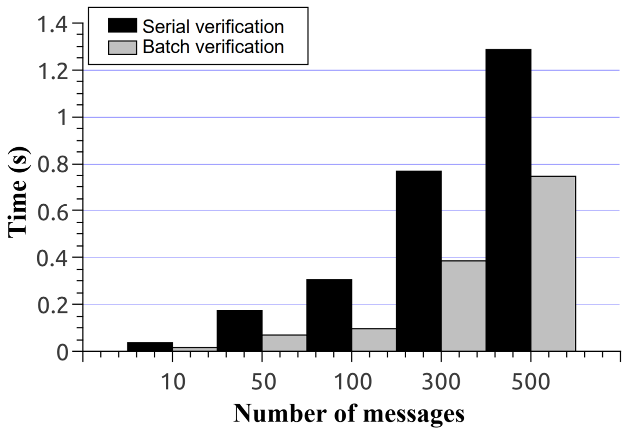

Figure 2: Time processing of serial and batch verification.

It is easy to see that batch validation time is always lower than the serial verification time. When the number of messages is 10 (equivalent to 10 attributes of the database table), in the proposed algorithm, the serial verification time is 0.036749s, nearly 2.5 times more than the batch verification (0.015853s); When the number of messages is 100, the serial verification time is nearly 3.2 times more than the batch verification (0.305106/0.096144).

Using two hard problems in an authentication scheme increases security for long-term data storage. However, this is synonymous with increasing computational complexity. But in the proposed model, verification is processed in the proxy server, independent of the DSP server. Therefore, the number of records returned is also reduced compared to the original stored data (depending on SQL query conditions). In addition, DO can use many other techniques such as parallel processing, load balancing... to increase the computing ability of the proxy server. In fact, the number of tables in the database is not many (usually no more than 100 tables), and the number of columns is not large (the number of columns representing the object's attributes should usually not exceed 300 attributes). Therefore, the encrypted verification depends on the number of returned records. On the other hand, the encrypted verification is performed simultaneously with the operation of decryption the data, so the verification is considered to be negligible compared to without a signature (because it still has to decrypt the data, return the plain data to the user).

6 CONCLUSIONS

In this paper, we introduce digital signatures method for batch verification algorithms. The contribution of the paper is to propose a batch verification scheme based on multiple hard problems and a new model that can verify the outsourced encrypted database, returning plain results to the user. This helps when the record is corrupted or updated, the DO knows the incomplete data from which has appropriate handling policy. Moreover, when validating data in combination with data decryption, the processing speed is almost no different than without checking the signature (due to the need to decrypt the data). In addition, the proposed model combined with parallel processing will greatly reduce query time. The future works direction is to propose a model to check the completeness of the data when validating outsourced data queries.

REFERENCES

[1] Jan Camenisch, Susan Hohenberger, and Michael Østergaard Pedersen. 2012. Batch verification of short signatures. *Journal of cryptology* 25, 4 (2012), 723–747.

[2] Shin-Yan Chiou. 2016. Novel digital signature schemes based on factoring and discrete logarithms. *International Journal of Security and Its Applications* 10, 3 (2016), 295–310.

[3] Keisuke Hakuta, Yosuke Katoh, Hisayoshi Sato, and Tsuyoshi Takagi. 2013. Batch Verification Suitable for Efficiently Verifying a Limited Number of Signatures. In *Information Security and Cryptology - ICISC 2012*, Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 425–440.

[4] Lein Harn. 1998. Batch verifying multiple RSA digital signatures. *Electronics Letters* 34, 12 (1998), 1219–1220.

[5] ES Ismail and NMF Tahat. 2011. A new signature scheme based on multiple hard number theoretic problems. *ISRN Communications and Networking* 2011 (2011).

[6] Rohit Jain and Sunil Prabhakar. 2013. Trustworthy data from untrusted databases. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on.* IEEE, 529–540.

[7] Einar Mykletun, Maithili Narasimha, and Gene Tsudik. 2006. Authentication and integrity in outsourced databases. *ACM Transactions on Storage (TOS)* 2, 2 (2006), 107–138.

[8] Muhammad Saqib Niaz and Gunter Saake. 2015. Merkle Hash Tree based Techniques for Data Integrity of Outsourced Data.. In *GvD.* 66–71.

[9] Ronald L. Rivest, Adi Shamir, and Leonard Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (1978), 120–126.

[10] Claus-Peter Schnorr. 1989. Efficient identification and signatures for smart cards. In *Conference on the Theory and Application of Cryptology.* Springer, 239–252.

[11] Zuhua Shao. 2001. Batch verifying multiple DSA-type digital signatures. *Computer Networks* 37, 3-4 (2001), 383–389.

[12] Ching-Te Wang, Chu-Hsing Lin, and Chin-Chen Chang. 2003. Signature schemes based on two hard problems simultaneously. In *17th International Conference on Advanced Information Networking and Applications, 2003. AINA 2003.* IEEE, 557–560.

[13] Jiawei Yuan and Shucheng Yu. 2013. Flexible and publicly verifiable aggregation query for outsourced databases in cloud. In *Communications and network security (cns), 2013 ieee conference on.* IEEE, 520–524.

[14] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. 2015. Integridb: Verifiable SQL for outsourced databases. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security.* ACM, 1480–1491.