Journal of Computer Science and Frontier Technologies
ISSN Print: 3104-4204
ISSN Online: 3104-4212
CD ISSN: 3104-4220
https://doi.org/10.63313/JCSFT.9054

A Verifiable Epidemic Monitoring and Early Warning Scheme for Privacy Protection under Malicious Cloud Server Threat Model

XinMiao Ma
Qingdao University, Qingdao 266071, China
Email: maxinmiao@qdu.edu.cn

How to cite this paper: Ma, X. M. (2026). A Verifiable Epidemic Monitoring and Early Warning Scheme for Privacy Protection under Malicious Cloud Server Threat Model. *Journal of Computer Science and Frontier Technologies*, 2(3), 133-143. ISSN Print: 3104-4204; ISSN Online: 3104-4212.
https://doi.org/10.63313/JCSFT.9054
Published: 2026-03-20

Copyright © 2026 by author(s) and Erytis Publishing Limited.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/

Abstract

The current epidemic monitoring and early warning systems usually directly upload raw disease data to the cloud server without verifying the aggregated results returned by the cloud server, which brings about the dual risks of data leakage and result tampering. To address these issues, this paper proposes a cloud-assisted privacy-preserving and verifiable epidemic monitoring and early warning scheme. This scheme encrypts the disease data by integrating technol-ogies such as convergent encryption(CE), hash-based indexing, and oblivious pseudorandom function (OPRF), and verifies the results returned by the cloud server to protect sensitive information during cloud storage and computing processes, while preventing malicious cloud servers from forging or tampering with data. The analysis shows that our solution can provide privacy protection and result verifiability, making it a feasible solution for epidemic monitoring and early warning.

Keywords

Cloud Computing; Epidemic diseases; Monitoring and early warning; Privacy-preserving; Verifiability of results

1. Introduction

With the continuous progress of society and the development of the nation, people's living standards have steadily improved, and people are also becoming increasingly concerned with their own physical health. At the same time, they are also increasingly concerned with the issue of epidemic early warning.The most typical is a major infectious disease event. Infectious diseases have been a disease that severely threatens human life and health and seriously affects the development of social economy. For example, the COVID-19 outbreak in 2019 spread rapidly from Wuhan in Hubei province to other parts of the world, causing significant disruption

to people's work and lives due to its high transmissibility, rapid spread, and diverse transmission pathways. Many businesses faced a crisis of closure, prompting deep reflection [1]. In the context of current globalization, the rapid spread and potential threats of epidemics have become increasingly prominent. Therefore, establishing an efficient and accurate epidemic early warning system is of paramount importance.

Cloud computing, commonly referred to as "the cloud", is a computing model that relies on the internet to provide flexible and scalable computing resources and services according to demand. Users can directly access computing power, storage space, and data processing capabilities via the network without needing to manage or maintain the underlying hardware infrastructure [2]. However, the application of any technology has two sides, and cloud computing is no exception. It faces numerous challenges and risks. Although outsourcing computing has many advantages, it also has obvious disadvantages and problems, such as the leakage of sensitive information, and cloud service providers (CSPs) may tamper with the calculation results for their own interests [3].

2. Related Work

The earliest epidemic early warning systems primarily relied on collecting data through paper forms, combined with case statistics, manual reporting, and analysis by public health institutions for monitoring. However, these traditional methods generally suffered from issues such as information lag and poor timeliness. In 1994, Dr. Marc R. L. at Massachusetts General Hospital in the United States proposed and established the ProMED mail system, a global digital platform for disease surveillance. By promoting effective communication, this system has made significant contributions to global public health surveillance [4]. Following the SARS outbreak in 2003, the Chinese government accelerated the development of public health information systems, gradually exploring ways to leverage the internet to mitigate the severe impact of infectious diseases on society. In 2004, the National Notifiable Disease Reporting System(NIDRIS) was officially launched. As an online platform, it enables medical institutions to report cases of notifiable infectious diseases directly and in real-time via the internet to the national infectious disease surveillance center database [5]. Subsequently, in 2008, the Chinese Center for Disease Control and Prevention initiated a nationwide epidemic early warning system — the China Infectious Disease Automated-alert and Response System(CIDARS) [6]. In 2020, Bashir et al. proposed MedCloud, a cloud-based disease surveillance and information management platform, aimed at addressing challenges in existing systems such as information delays, management complexity, and uneven resource distribution [7].

In 2003, the U.S. Centers for Disease Control and Prevention launched the National Syndromic Surveillance Program(NSSP), designed to enable cross-agency statistical

tracking of early epidemic warning symptoms [8]. Subsequently, in 2009, the CDC introduced the National Outbreak Reporting System(NORS), which has played a key role in epidemic monitoring and public health emergency response [9]. However, none of the aforementioned systems and solutions have implemented effective privacy protection for disease information, leading to risks of sensitive patient health data being exposed during collection, transmission, and sharing, potentially infringing on personal privacy.

In 2025, Li et al. proposed EpiOracle, a cross-institutional privacy-preserving epidemic monitoring and early warning scheme. This scheme is based on key encryption primitives and supports functions such as fuzzy symptom detection, symptom frequency statistics, and privacy protection [10]. Its system assumes that the cloud server operates under a semi-honest model, meaning the server may attempt to infer private information but will not maliciously falsify data or disrupt the protocol. However, this scheme does not account for scenarios where the server might deviate from the semi-honest model by forging disease symptom counts to launch attacks. Similarly, none of the aforementioned systems have implemented functionality to verify the results returned by the cloud server.

3. Our Main Design

3.1. System Model

The system model diagram is shown in Figure 1, which consists of three entities:Hospitals (H_i), a Public Health Agency (PHA), and a Cloud Server (CS).

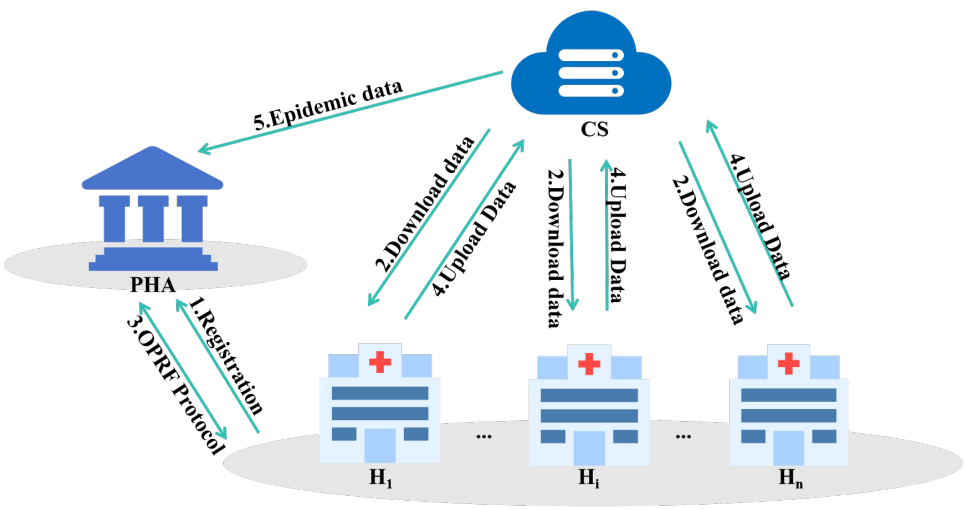


Figure 1. system model.

3.2. Registration Phase: $(\mathcal{H}_i, PHA) - Reg$

In this stage, each Hospital H_i needs to complete identity registration with the public health agency (PHA). The specific process is as follows:

(1) Submit registration application: H_i provides the legal entity certification documents required for registration to the PHA.

(2) Verification and certificate issuance: The PHA reviews the qualifications of H_i to confirm its legitimacy. After the review is approved, the PHA issues a digital certificate for it:

$$CA_i = \langle ID_{H_i}, Sig_{sk_{PHA}}(ID_{H_i}) \rangle, \quad (1)$$

Where ID_{H_i} is the unique identity identifier of H_i, and $\mathrm{Sig}_{\mathrm{sk_{PHA}}}(ID_{H_i})$ represents the result of performing ECDSA-SHA256 signature operation on the identity identifier information ID_{H_i} of the Hospital using the public health agency's private key $\mathrm{sk_{PHA}}$.

3.3. Disease Data Encryption Stage: $\mathcal{H}_i - DataEnc$

In this stage, for the patient's disease information data (id_p, m), each Hospital H_i performs the following encryption operations:

(1) Perform convergent encryption on the disease information to generate ciphertext $cc_m = \mathrm{Enc}(ck_m, m)$, where the encryption key is obtained by hashing the disease information, i.e., $ck_m = H_2(m)$;

(2) Calculate the short hash value $sh = SH(cc_m)$ of the convergent encryption ciphertext of the disease information;

(3) Send the tuple (sh, cc_m) to the cloud server (CS).

3.4. Cloud Database Updating Phase: $(CS, \mathcal{H}_i, PHA) - DBUpdate$

After receiving the tuple (sh, cc_m), the CS first performs preliminary collision detection by comparing the received short hash value sh with the short hash values stored in the database $\mathcal{DB}$. The specific index structure of the cloud database is shown in Table 1.

Table 1. Index Structure of Cloud Database

Short hash value	Converged ciphertext	Number	Hash value	The current uploader	Outsourcing proof
sh_1	cc_{m_1}	cnt_{m_1}	$H_2(m_1)$	$CurU_{m_1}$	op_{m_1}
	cc_{m_2}	cnt_{m_2}	$H_2(m_2)$	$CurU_{m_2}$	op_{m_2}
	...	...	...	...	...
...	...	...	...	...	...

If the CS does not find a matching short hash value in the database $\mathcal{DB}$, it returns a prompt message "No matching short hash value found" to the Hospital H_i. Upon receiving this message, H_i first verifies whether the result returned by the CS is correct by interacting with the PHA to obtain relevant information for verification. The specific process is as follows: H_i sends $Hash = H_2(m)$ to the public health agency (PHA). After receiving the hash value $Hash$, the PHA first searches for this hash value $Hash$ in its locally maintained Hash Value - Private Key Table (as shown in Table 2). If the PHA does not find an equal hash value, it indicates that this disease

information m is being uploaded for the first time. The PHA records $Ver = NULL$ and sends Ver to H_i. At this time, the PHA generates the corresponding private key information x_i for the hash value Hash based on the current time and inserts it into the Hash Value - Private Key Table. Afterwards, H_i determines whether the result returned by the CS is correct by judging whether Ver is $NULL$: If Ver is not $NULL$, it indicates that the CS has forged or tampered with the disease data, and the CS is malicious. In this case, no subsequent processes are executed, and the protocol terminates; if Ver is $NULL$, it indicates that the disease information m is indeed not stored in the database $D\mathcal{B}$, and the CS is honest. At this time, H_i sets the disease information count cnt_m to 1, sets $CurU_m$=ID_{H_i}, and generates the outsourcing proof op_m for this disease information m according to the following steps:

(1) H_i first calculates the concatenated message $T_m = sh \parallel cc_m \parallel cnt_m \parallel H_2(m) \parallel CurU_m$, randomly selects $s \in \mathbb{Z}_{p-1}$, generates a $= H_1(T_m)^s$, and sends a to the PHA.

(2) After receiving a, the PHA uses the updated private key x_i to calculate $b = a^{x_i}$, and sends b to H_i.

(3) After receiving b, H_i calculates the final outsourcing proof output $op_m = H_2(T_m \parallel b^{1/s})$.

Subsequently, H_i sends the six-tupl$< sh, cc_m, cnt_m = 1, H_2(m), CurU_m = ID_{H_i}, op_m >$ to the CS, and the CS updates it into the database $D\mathcal{B}$, namely:

$$\mathcal{DB} = \mathcal{DB} \cup \{\langle sh, cc_m, cnt_m = 1, H_2(m), CurU_m = ID_{H_i}, op_m \rangle\} \quad (2)$$

Table 2. Hash Value - Private Key Table

Hash value	Private key information
$H(m_1)$	x_1
$H(m_2)$	x_2
$H(m_3)$	x_3
...	...

If the CS finds that the short hash value sh already exists in $D\mathcal{B}$ during the preliminary collision detection stage, it indicates that this disease information m may have been recorded in the database. At this time, the CS will further determine whether this disease m actually exists in the database $D\mathcal{B}$ based on the convergent encryption ciphertext cc_m. The CS extracts all five-tuples starting with this short hash value sh from the database $D\mathcal{B}$ to form a subset:

$$\mathcal{DB}[sh] = \{\langle sh, cc_{m'}, cnt_{m'}, H_2(m'), CurU_{m'}, op_{m'} \rangle\} \subseteq \mathcal{DB} \quad (3)$$

Then, for each $\langle sh, cc_{m'}, cnt_{m'}, H_2(m'), CurU_{m'}, op_{m'} \rangle \in \mathcal{DB}[sh]$, the CS compares the newly received disease information converged ciphertext cc_m with the existing converged ciphertext in the database. If for all six-tuples in $D\mathcal{B}[sh]$, the convergent encryption ciphertext of the disease information satisfies $cc_m \neq cc_{m'}$, it indicates that this disease information m is being uploaded for the first time. The CS returns a prompt message "No matching convergent ciphertext" to the Hospital H_i. The

subsequent operation process is the same as the operation described above when no matching short hash value sh is found, and the CS will update the dataset $D\mathcal{B}$ according to formula (2).

If there exists a six-tuple $\langle sh, cc_{m'}, cnt_{m'}, H_2(m'), CurU_{m'}, op_{m'}\rangle \in \mathcal{DB}[sh]$ such that the convergent encryption ciphertext of the disease information satisfies $cc_m = cc_{m'}$, it indicates that this disease information m already exists in $D\mathcal{B}$. At this time, the cloud server (CS) sends this six-tuple $< sh, cc_{m'}, cnt_{m'}, H_2(m'), CurU_{m'}, op_{m'} >$ to the Hospital H_i. Upon receiving this six-tuple, H_i first verifies whether the result returned by the CS is correct by interacting with the PHA to obtain relevant information for verification. The specific process is as follows:

1) H_i calculates the concatenated message $T_{m'} = sh \parallel cc_{m'} \parallel cnt_{m'} \parallel H_2(m') \parallel CurU_{m'}$, randomly selects $s \in \mathbb{Z}_{p-1}$, generates $a = H_1(T_{m'})^s$, and sends a and $Hash = H_2(m)$ to the public health agency (PHA).

2) After receiving a and $Hash$, the PHA first retrieves the private key information x_{i-1} corresponding to the hash value $Hash$ from its maintained Hash Value - Private Key Table, then uses the private key x_{i-1} to calculate $b = a^{x_{i-1}}$, and returns the result$b$to H_i. At this time, the PHA regenerates the corresponding private key information x_i for the original hash value $Hash$ based on the current time and updates it into the Hash Value - Private Key Table.

3) After receiving b, H_i calculates the verification proof $Ver = H_2\left(T_{m'} \parallel b^{1/s}\right)$.

4) Verification: If $Ver = op_{m'}$, it indicates that the six-tuple information returned by the CS is accurate; otherwise, it indicates that the returned six-tuple information may have been forged or tampered with, and the protocol terminates at this point.

After verification is passed, H_i performs the following update operations: update the count to $cnt'_{m'} = cnt_{m'} + 1$, update the current uploader identifier to $CurU'_{m'} = ID_{H_i}$. H_i generates the outsourcing proof $op'_{m'}$ by interacting with the PHA. The specific process is as follows:

1) H_i first concatenates the above six-tuple information to form $T'_{m'} = sh \parallel cc_{m'} \parallel cnt'_{m'} \parallel H_2(m') \parallel CurU'_{m'}$, randomly selects $s \in \mathbb{Z}_{p-1}$, generates $a = H_1(T'_{m'})^s$, and sends a and $Hash = H_2(m)$ to the public health agency (PHA).

2) After receiving a and $Hash$, the PHA first retrieves the private key information x_i corresponding to the hash value $Hash$ from its maintained hash value-private key table, then uses the private key x_i to calculate $b = a^{x_i}$,, and returns the result b to H_i.

3) After receiving b, Hi calculates the outsourcing proof for the disease information $op'_{m'} = H_2\left(T'_{m'} \parallel b^{1/s}\right)$.

Subsequently, H_i sends the updated six-tuple $< sh, cc_{m'}, cnt'_{m'}, H_2(m'), CurU'_{m'}, op'_{m'} >$ to the CS. Finally, the CS updates the corresponding entry in the database: replace the count $cnt_{m'}$ with $cnt'_{m'}$, replace the current uploader $CurU_{m'}$ with $CurU'_{m'}$, and replace the outsourcing proof $op_{m'}$ with $op'_{m'}$. If the updated count $cnt'_{m'}$ reaches the preset threshold t, the CS sends

the following six-tuple information to the PHA:

$$< sh, cc_{m'}, cnt'_{m'}, H_2(m'), CurU'_{m'}, op'_{m'} > \quad (4)$$

3.5. Verifiable Epidemic Prediction Phase: $PHA - Ver\ Pr\ e$

After receiving the six-tuple $< sh, cc_{m'}, cnt'_{m'}, H_2(m'), CurU'_{m'}, op'_{m'} >$ sent by the CS, the PHA first looks up the corresponding private key information x_i in its locally maintained Hash Value - Private Key Table based on $H_2(m')$. Subsequently, the PHA generates the OP for verification based on this private key x_i, and the calculation formula is as follows:

$$OP = H_2(T''_{m'} \| H_1(T''_{m'})^{x_i}), \quad (5)$$

where $T''_{m'} = sh \| cc_{m'} \| cnt'_{m'} \| H_2(m') \| CurU'_{m'}$. Next, the PHA compares the locally generated OP with the $op'_{m'}$ in the received six-tuple information, and determines whether $cnt'_{m'}$ is greater than the preset threshold t, that is, determines whether the following two conditions are simultaneously satisfied:

$$\begin{cases} cnt'_{m'} \stackrel{?}{=} t \\ OP \stackrel{?}{=} op'_{m'} \end{cases} \quad (6)$$

If both conditions in formula (6) are satisfied, it indicates that the result returned by the CS is authentic and valid, and the subsequent process continues; otherwise, the returned result may have been forged or tampered with by the CS, and the protocol terminates at this point.

After the above verification is passed, the PHA confirms that at least t patients have been diagnosed with disease m'. At this time, the PHA uses $H_2(m')$ and the ciphertext $cc_{m'}$ obtained from the CS to decrypt the ciphertext and recover the original disease information m'. Finally, based on the disease information m', the PHA promptly issues an early warning.

4. Efficiency Analysis

4.1. Theoretical Analysis

Table 3. Theoretical calculation costs for each participant at different stages

Participants / Phases	$\mathcal{H}_i$	CS	PHA
$(\mathcal{H}_i, PHA) - Re\ g$	-	-	$n \cdot t_{Sig}$
$\mathcal{H}_i - DataEnc$	$t_{Enc} + t_{Hash} + t_{SH}$	-	-
$(CS, \mathcal{H}_i, PHA) - DBUpdate$	$4t_{Hash} + 4t_{Exp} + 2t_{Inv}$	$O(\log n_{SH} + \log(\#DB/n_{SH}))$	$2O(\log n) + 2t_{Exp}$
$PHA - Ver\ Pr\ e$	-	-	$t_{Ver} + O(\log n) + 2t_{Hash} + t_{Exp} + t_{Dec}$

Let t_{Enc} be the time required for one convergent encryption operation (e.g., one

AES-256 encryption), t_{Dec} be the time required for one convergent decryption operation (e.g., one AES-256 decryption), t_{Hash} be the time overhead for performing one hash operation, t_{sh} be the time overhead for performing one short hash operation, t_{sig} denote the time overhead for generating an ECDSA signature for a message, t_{Ver} be the time overhead for verifying a message-signature pair, t_{Exp} denote the time overhead for one exponentiation operation, and t_{Inv} denote the time overhead for one modular $p-1$ inversion operation. According to the above symbol representation, the theoretical computational cost analysis of the proposed scheme is shown in Table 3:

4.2. Experimental Analysis

The proposed scheme in this chapter was implemented in the C language, with cryptographic operations carried out using the OpenSSL library. Specifically, the convergent encryption key was generated using the SHA-256 hash algorithm, and the symmetric encryption part employed the AES-256 algorithm. The short hash length was set to 8 bits, and the epidemic warning threshold $t$was configured as 10% of the total dataset size. For data management, MySQL was used. The relational database stored disease records and their corresponding verification proofs to support efficient query processing and system validation.

(1) Computational Overhead

In Scenario 1, the parameter |p| was set to 256 bits, and experiments were performed using synthetic medical datasets of varying sizes to evaluate computational overhead. The total time overhead (in seconds) for each phase under different data sizes is summarized in Table 4, with each data point representing the average of ten experimental runs. As shown in Table 4, the time overhead for the disease data encryption phase and the cloud database update phase exhibits an increasing trend as the dataset size grows. In the verifiable epidemic prediction phase, because in each dataset, the number of records for exactly two diseases reaches the preset threshold t, the execution time for this phase remains relatively stable, with minor fluctuations observed falling within a reasonable margin of error. Subsequently, the parameter |p| was set to 512 bits. As shown in Table 5, the general trend of the data is the same as that for |p| = 256 bits. In conclusion, the proposed scheme has low time overhead.

Table 4. The time consumption (unit: s) of each stage for different dataset sizes when |p| = 256 bits

Phases	$\mathcal{H}_i - DataEnc$	$(CS, \mathcal{H}_i, PHA) - DBUpdate$			$PHA - Ver\ Pr\ e$
Participants / Dataset Size	$\mathcal{H}_i$	CS	$\mathcal{H}_i$	PHA	PHA
1000	0.007125	3.018339	0.302942	0.518196	0.000281
3000	0.022771	8.703808	0.917349	1.529064	0.000277

7000	0.056211	21.113223	2.194944	3.755217	0.000293
10000	0.078614	31.181597	3.199219	5.594381	0.000285
20000	0.163387	60.482723	6.384536	12.002944	0.000274
50000	0.401534	168.611845	16.164026	31.757637	0.000309

Table 5. The time consumption (unit: s) of each stage for different dataset sizes when |p| = 512 bits

Phases	$\mathcal{H}_i - DataEnc$	$(CS, \mathcal{H}_i, PHA) - DBUpdate$			$PHA - Ver\ Pr\ e$
Participants / Dataset Size	$\mathcal{H}_i$	CS	$\mathcal{H}_i$	PHA	PHA
1000	0.007359	2.968681	0.332286	0.535069	0.000298
3000	0.022353	8.715446	0.984424	1.604030	0.000280
7000	0.052255	20.580753	2.279431	3.934709	0.000290
10000	0.076648	30.017801	3.337219	6.048310	0.000287
20000	0.155169	60.851971	6.701405	13.204054	0.000295
50000	0.387800	167.879924	16.474025	35.199068	0.000318

(2) Communication Overhead

For the evaluation of communication overhead, the parameter |p| was set to 256 bits, and tests were conducted on the six different dataset sizes mentioned above. As shown in Table 6, the communication overhead for both the disease data encryption phase and the cloud database update phase (involving communication between the cloud server CS and the medical institution Hi, as well as between the medical institution Hi and the public health authority PHA) increases with the dataset size. It is worth noting that during the cloud database update phase, because exactly two diseases in each dataset reach the threshold t, the communication overhead between the cloud server (CS) and the public health authority (PHA) remains stable for dataset sizes ranging from 1000 to 20000, and only shows a significant increase when the size reaches 50000. This phenomenon is primarily attributed to differences in the length of individual disease record data.

Subsequently, the parameter |p| was set to 512 bits. As shown in Table 7, the general trend pattern is the same as |p| = 256 bits. In conclusion, the proposed scheme has low communication overhead.

Table 6. The communication overhead (unit: KB) of each stage for different dataset sizes when |p| = 256 bits

Phases	$\mathcal{H}_i - DataEnc$	$(CS, \mathcal{H}_i, PHA) - DBUpdate$				
Participants / Dataset Size	$\mathcal{H}_i \rightarrow CS$	$CS \rightarrow \mathcal{H}_i$	$\mathcal{H}_i \rightarrow CS$	$\mathcal{H}_i \rightarrow PHA$	$PHA \rightarrow \mathcal{H}_i$	$CS \rightarrow PHA$
1000	16.6016	84.3408	90.8203	92.7734	61.5254	0.1816
3000	49.8047	256.2598	272.4609	279.2559	185.5039	0.1816

7000	116.2109	601.5088	635.7422	653.1563	434.4141	0.1816
10000	166.0156	860.8398	908.2031	933.8330	621.3594	0.1816
20000	332.0313	1726.5771	1816.4063	1869.5527	1244.5625	0.1816
50000	1611.3281	5104.9580	5322.2656	4677.7832	3115.2529	0.2129

Table 7. The communication overhead (unit: KB) of each stage for different dataset sizes when |p| = 512 bits

Phases	$\mathcal{H}_i - DataEnc$	$(CS, \mathcal{H}_i, PHA) - DBUpdate$				
Participants / Dataset Size	$\mathcal{H}_i \rightarrow CS$	$CS \rightarrow \mathcal{H}_i$	$\mathcal{H}_i \rightarrow CS$	$\mathcal{H}_i \rightarrow PHA$	$PHA \rightarrow \mathcal{H}_i$	$CS \rightarrow PHA$
1000	16.6016	84.3408	90.8203	154.3027	123.0605	0.1816
3000	49.8047	256.2598	272.4609	464.7900	371.0371	0.1816
7000	116.2109	601.5088	635.7422	1087.6328	868.8867	0.1816
10000	166.0156	860.8398	908.2031	1555.2988	1242.7910	0.1816
20000	332.0313	1726.5771	1816.4063	3114.2676	2489.2920	0.1816
50000	1611.3281	5104.9580	5322.2656	7793.4678	6230.9424	0.2129

5. Conclusion

This paper proposes an epidemic surveillance and early warning scheme that ensures both privacy protection and result verifiability under the malicious cloud server threat model. The scheme not only safeguards patients' private medical data but also enables healthcare institutions and public health agencies to verify the authenticity and accuracy of the results returned by the cloud server, preventing malicious cloud servers from forging or tampering with data.

References

[1] Ciotti M, Ciccozzi M, Terrinoni A, et al. The COVID-19 pandemic[J]. Critical reviews in clinical laboratory sciences, 2020, 57(6): 365-388.

[2] Kim W. Cloud computing: Today and tomorrow[J]. J. Object Technol., 2009, 8(1): 65-72.

[3] Motahari-Nezhad H R, Stephenson B, Singhal S. Outsourcing business to cloud computing services: Opportunities and challenges[J]. IEEE Internet Computing, 2009, 10(4): 1-17.

[4] V. L. Yu and L. C. Madoff. “ProMED-mail: an early warning system for emerging diseases”. In: Clinical infectious diseases 39.2 (2004), pp. 227 – 232.

[5] W. L. Vlieg et al. “Comparing national infectious disease surveillance systems: China and the Netherlands”. In: BMC public health 17(2017), pp. 1 – 9.

[6] W. Yang et al. “China infectious diseases automated-alert and response system (cidars)”. In: Early Warning for Infectious Disease Outbreak. Elsevier, 2017, pp. 133 – 161.

[7] A. Bashir, A. W. Malik, A. U. Rahman, S. Iqbal, P. R. Cleary, and A. Ikram. “MedCloud: cloud-based disease surveillance and information management system”. In: IEEE Access 8 (2020), pp. 81271 – 81282.

[8] D. W. Gould, D. Walker, and P. W. Yoon. “The evolution of BioSense: lessons learned

and future directions”. In: Public Health Reports 132.1 suppl (2017), 7S–11S.

[9] M. E. Wikswo et al. “Enteric illness outbreaks reported through the national outbreak reporting system—United States, 2009–2019”. In: Clinical Infectious Diseases 74.11 (2022), pp. 1906–1913.

[10] Li, S., Zhang, Y., Song, Y., Wu, F., Lyu, F., Yang, K., & Tang, Q. (2025). EpiOracle: Privacy-Preserving Cross-Facility Early Warning for Unknown Epidemics. Proceedings on Privacy Enhancing Technologies.