Can Fully Homomorphic Encryption on Public Blockchains Protect Sensitive Medical Consultations? A Security and Privacy Assessment

Shahruddeen Nizamuddeen Muhammad
The University of Law Business School
Birmingham, United Kingdom
nizaushahru@gmail.com

***Abstract*—People seeking care for stigmatised conditions often delay treatment because they fear exposure, yet digital health services routinely expose user data and metadata. Fully homomorphic encryption (FHE) now allows smart contracts on public blockchains to hold encrypted data whose decryption is governed by contract rules. This paper assesses whether this approach can protect sensitive medical consultations. We present ConfidentialCare, a prototype built on Zama's fhEVM and the Ethereum Sepolia testnet, in which patients' symptoms are encrypted in the browser and only an authorised doctor may decrypt them. We evaluated three versions of the system through metadata inspection, access-control tests and attack experiments, asking what FHE protects and whether access to patient data can be controlled. Consultation content remained confidential in all tested cases, and the design collects no names, emails or accounts. However, public on-chain metadata revealed which wallet consulted which doctor, the doctor's specialty, when, and the approximate length of each message, and this history would be exposed permanently if the wallet were ever linked to the patient. We also found that a doctor granted access could re-share a patient's encrypted data, both through a confused-deputy flaw in the application contract, which we fixed, and directly through fhEVM's access control list, and that neither withdrawing access nor closing a consultation removed it. A redesign based on time-limited, revocable delegation prevented re-sharing, and Zama's live service enforced its revocation and expiry immediately. On Sepolia, Zama's key-management service refused decryption without a valid delegation, and revocation took effect within the same block in all six trials. We conclude that FHE protects what patients disclose, but not the fact or context of seeking care, and that in fhEVM a permission to decrypt is also a permission to share.**

***Index Terms*—fully homomorphic encryption, blockchain, smart contracts, healthcare privacy, telemedicine, access control, metadata**

I. Introduction

Stigma remains one of the most persistent barriers to healthcare for people living with conditions such as HIV, sexually transmitted infections and mental illness. Where a diagnosis carries social, legal or economic consequences, many people delay seeking help, or avoid it altogether, because they fear that their condition will become known to others. Nigeria illustrates the scale of this problem. It has the highest population of people living with HIV in West Africa, and stigma and discrimination have remained a major challenge to prevention, treatment and care [1]. The Nigeria PLHIV Stigma Index Survey 2.0, which interviewed 1,240 people living with HIV across 16 states and the Federal Capital Territory in 2021, found that 22% of respondents had experienced HIV-related stigma or discrimination in the previous 12 months [1]. Fear of exposure also shaped decisions about care: 38.1% of respondents delayed starting treatment because they worried that partners, family or friends would find out, and over half of those who chose to be tested said that fear of how others would react delayed that decision [1]. In addition, 19.5% reported that their HIV status had been disclosed without their consent [1].

When formal services feel unsafe, people turn to informal alternatives. In Nigerian cities, for example, handwritten messages and phone numbers painted on public walls have been used to connect people living with HIV to one another [2]. Such channels show both the need for discreet support and the absence of a trusted way to obtain it. They offer no verification of who is on the other end, no protection of the information shared, and no link to qualified medical advice.

Digital health services could, in principle, offer a safer route. Online consultations allow patients to seek advice without visiting a clinic in person, which may reduce the risk of being seen or recognised. However, these services introduce privacy risks of their own. A traffic analysis of popular medicines-related apps found that 19 of the 24 apps studied (79%) shared user data with third parties, involving 55 different companies and a wider network of further recipients [3]. Telehealth platforms also typically require patients to create accounts linked to names, email addresses and phone numbers, so a record of who consulted whom, and when, is held by the service provider. For a patient whose main concern is exposure, this is the very risk they are trying to avoid.

Blockchain technology has been proposed as a way to give patients greater control over their health data, removing the need for a single organisation to hold and manage records [4], [5]. Yet public blockchains are transparent by design: any data written to them can be read by anyone. Systematic reviews of blockchain in healthcare show that most designs therefore store sensitive data off-chain and record only references or hashes on the ledger [5], which leaves confidentiality to conventional storage systems rather than solving it.

Fully homomorphic encryption (FHE) offers a different approach. FHE allows computation on encrypted data without

first decrypting it [7], and recent platforms such as Zama's fhEVM bring this capability to smart contracts on public blockchains [14]. In this model, sensitive information can remain encrypted on-chain, while smart contract rules determine who is allowed to decrypt it. This raises the possibility of an online consultation service in which neither the platform operator nor the public can read what patients disclose, and in which patients need provide no name, email address or account. However, FHE-based smart contracts are a recent development, and independent evaluations of their use in sensitive, real-world applications remain scarce [13].

This paper assesses whether FHE on a public blockchain can protect sensitive medical consultations. We present ConfidentialCare, a prototype platform that allows pseudonymous patients to consult verified doctors. Patients' symptoms are encrypted in the browser before reaching the blockchain, and only an authorised doctor can decrypt them. Using this prototype, we address two research questions:

- **RQ1 (confidentiality):** What does FHE protect in this setting, and what information still leaks?
- **RQ2 (access control):** Can patients control who can decrypt their data, including preventing re-sharing and withdrawing access?

This paper makes four contributions. First, we design and implement ConfidentialCare, and define a threat model for FHE-based consultation systems. Second, we show that while consultation content remains confidential, on-chain metadata allows an observer to reconstruct a wallet's consultation history, including the specialties consulted. Third, we identify and fix a confused-deputy flaw through which encrypted data could be re-shared without the patient's consent, and show that fhEVM's permission model allows any party granted access to re-share data independently of the application. Fourth, we evaluate a redesign based on time-limited, revocable delegation and per-doctor patient wallets, and identify pitfalls that developers of fhEVM applications can easily overlook. We also report a preliminary observation on how the cost of FHE limits message length, and leave a full performance and usability evaluation to future work.

The rest of this paper is organised as follows. Section II reviews background and related work. Section III describes the design of ConfidentialCare and its three versions, and Section IV sets out the threat model. Section V explains our evaluation method, and Section VI presents the results. Section VII discusses their implications and limitations, and Section VIII concludes.

II. Background and Related Work

A. Fully Homomorphic Encryption and TFHE

Homomorphic encryption allows computations to be performed on ciphertexts, producing an encrypted result that, when decrypted, matches the result of the same computation on the plaintext. TFHE is a fully homomorphic scheme whose security rests on the Learning With Errors (LWE) problem: a value is combined with a secret key and a small amount of random noise, and recovering it without the key requires solving a hard problem on high-dimensional lattices [7]. Encryption is randomised, so the same value encrypted twice yields different ciphertexts, and lattice problems are believed to resist attacks by quantum computers. TFHE introduced fast bootstrapping, reducing its cost from 690 ms to 13 ms on a single core [7]. Later work has continued to optimise it [8], and Zama reports bootstrapping below one millisecond on GPUs at 128-bit security [9].

B. Confidential Smart Contracts and fhEVM

Public smart contracts expose all of their data, and three main approaches to confidentiality have emerged. Zero-knowledge systems such as Hawk keep transaction data off the public chain and prove its correctness [10]. Systems based on trusted execution environments (TEEs), such as Ekiden, decrypt data inside secure hardware [11], but more than 20 attacks on Intel SGX have been published, mostly through side channels [12]. FHE-based systems keep data encrypted even during computation [13].

Zama's fhEVM follows the FHE approach. The blockchain does not store ciphertexts; it stores references to them, called *handles*, while an off-chain coprocessor holds the ciphertexts and performs computation on them without the decryption key [14]. Contracts control who may decrypt a value by adding addresses to an access control list (ACL). The decryption key is split among several key-management operators using threshold multi-party computation [15]; on Sepolia, Zama's key-management registry listed 13 signers with a threshold of 7 when we checked on 3 November 2026. On Zama's mainnet, keys were generated by 13 operators, with the protocol designed to tolerate up to four faulty or malicious operators [16]. To decrypt, a user signs a request with their wallet; the key-management service checks the ACL and returns the result re-encrypted under a temporary key held only by the user's browser.

C. Blockchain in Healthcare

Surveys of blockchain in healthcare report benefits for data integrity, traceability and access control [4]–[6]. A review of 51 studies found that public blockchains were the most common choice (43%), with privacy typically achieved by storing data off-chain and placing only hashes on-chain [5]. Immutability also conflicts with the right to erasure under data protection law, and pseudonymous on-chain records may reveal relationships between patients and providers [17].

D. Homomorphic Encryption in Healthcare

Healthcare research on homomorphic encryption focuses mainly on computation in the cloud, such as private machine-learning prediction and genomic analysis [18]. A systematic review of 69 studies concluded that homomorphic encryption is feasible for protecting patient data, but that bootstrapping, noise growth and ciphertext size limit performance [19]. A comparison of TFHE-rs and OpenFHE on medical laboratory rules found that TFHE-rs performed better on binary

operations and key generation, while neural-network inference remained slow [20]. Little of this work addresses FHE-based communication between patients and doctors.

E. Metadata and Deanonymisation

Encrypting content does not hide metadata. Telephone metadata has been shown to be easily re-identified and to support sensitive inferences [21]. On blockchains, pseudonymous addresses can be linked to real users: profiles of almost 40% of Bitcoin users could be largely recovered [22], clustering heuristics have mapped much of the Bitcoin economy [23], and Ethereum accounts can be linked through transaction timing, gas prices and graph structure [24], [25].

F. Research Gap

Blockchain health systems typically keep data off-chain, FHE health research targets cloud analytics rather than consultation, and evaluations of fhEVM come mainly from its developers. To our knowledge, no independent study has measured what FHE-based smart contracts protect, and fail to protect, in a sensitive health application. Zama's developer documentation describes how contracts grant permanent, transient and public access through the ACL, and recommends checking that a sender is authorised before using a ciphertext [26]. It does not, however, discuss the consequence that any address granted permanent access can extend that access to others, or make the ciphertext publicly decryptable, independently of the application; this behaviour is visible only in the ACL contract's code.

III. System Design

A. Overview and Roles

ConfidentialCare involves three roles: an *administrator*, who maintains a registry of verified doctors; *patients*, identified only by a wallet address, with no name, email or account; and *doctors*, each registered with a wallet, name, specialty and credentials. The system consists of a Solidity smart contract built on Zama's fhEVM, deployed on the Ethereum Sepolia testnet, and a Next.js web application. Encryption and decryption take place in the browser using Zama's SDK.

B. Consultation Flow (v1 and v2)

1) The patient selects a doctor and types their symptoms.
2) The browser splits the text into 64-bit chunks of eight bytes and encrypts each chunk, producing a list of handles and an input proof.
3) The patient submits the handles, the proof and the chosen doctor's identifier to the contract, which stores the handles and grants itself permission to use them.
4) The patient grants the doctor access, adding the doctor's wallet to the ACL for each handle.
5) The doctor requests decryption by signing an EIP-712 message; the key-management service checks the ACL and returns the plaintext re-encrypted to the doctor's temporary key.
6) The doctor's response is encrypted in the same way and made decryptable by the patient.
7) The patient may close the consultation.

TABLE I
Contract Versions

Ver.	Sepolia address	Design
v1	`0x52e7...83AA`	Original prototype; doctors added to the ACL
v2	`0x708d...7D59`	Fresh input proof required for all encrypted inputs; "revoke" renamed `markAccessWithdrawn`
v3	`0x9cF0...427B`	Doctors not added to the ACL; patient-issued delegation with 14-day expiry; one patient wallet per doctor; closing requires the delegation to end

C. Versions Evaluated

We evaluated three versions of the contract (Table I). Version 1 is the original prototype. Version 2 fixes a flaw found during testing (Section VI). Version 3 is a redesign based on delegated decryption.

D. Version 3: Delegated Decryption

In v3, doctors are never added to the ACL for patient data. Instead, the patient delegates decryption rights to the assigned doctor through Zama's ACL, with a default expiry of 14 days. The doctor decrypts by signing a delegated decryption request; the key-management service checks both that the patient holds permission on the handles and that a valid, unexpired delegation exists. Because a delegation covers all of the delegating wallet's data in the contract, each patient uses a separate wallet for each doctor, derived deterministically in the browser from a single signature by the patient's main wallet, so that it can be recovered without storing keys. The contract enforces that a patient wallet consults only one doctor, and `closeConsultation` succeeds only once the delegation has been revoked or has expired; the interface performs revocation and closing as a single action. The patient's main wallet funds each per-doctor wallet with gas.

IV. Threat Model

A. Assets

The primary asset is the content of consultations: the patient's symptoms and the doctor's response. Secondary assets are the facts surrounding a consultation: which wallet sought care, from which doctor or specialty, when, how often and in how much detail, and ultimately the patient's real identity.

B. Adversaries

We consider six adversaries: (A1) a *public observer* who reads the blockchain, its events and contract state; (A2) an *unassigned doctor* registered on the platform; (A3) the *administrator*; (A4) an *assigned doctor* who misuses legitimately

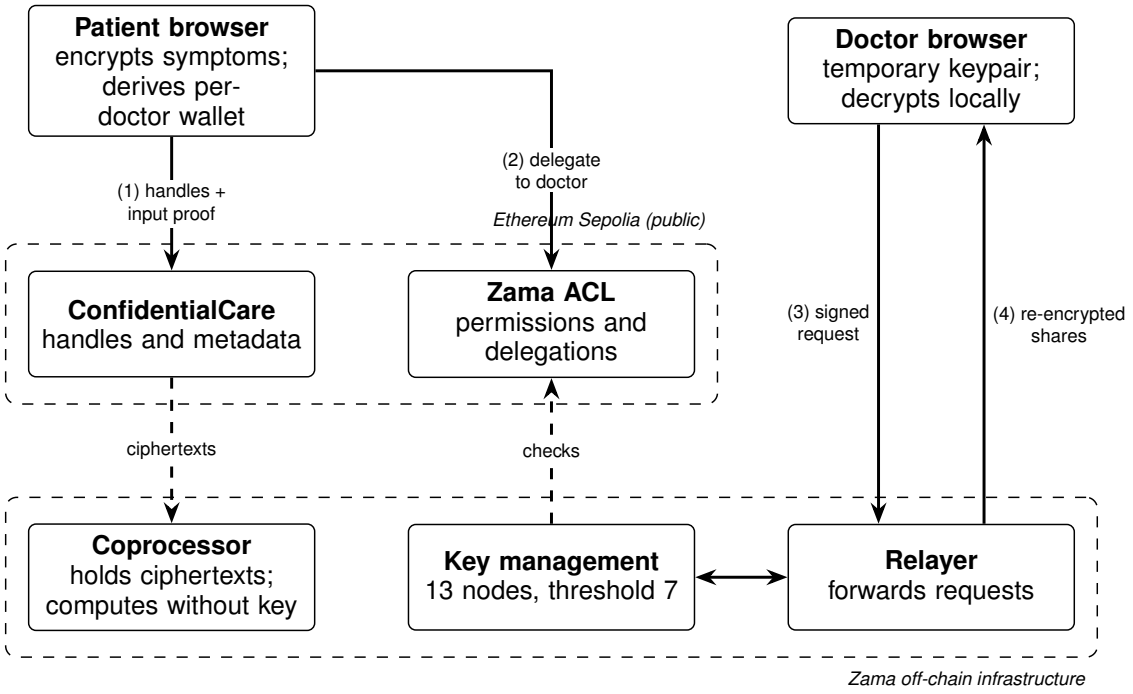


Fig. 1. Architecture of ConfidentialCare (v3). The patient's browser encrypts symptoms and submits handles with an input proof (1), then delegates decryption to the doctor through Zama's ACL (2). Ciphertexts are held by Zama's coprocessor. The doctor signs a decryption request (3); the key-management service checks the ACL and returns shares re-encrypted to the doctor's temporary key (4). Everything on Sepolia, including handles, metadata, permissions and delegations, is publicly readable.

granted access; (A5) an *unrelated wallet* attempting to decrypt data; and (A6) a *linking adversary*, such as an exchange, network provider or person close to the patient, who can connect a wallet to a real identity.

C. Trust Assumptions

We assume that the TFHE scheme and its implementation are secure, that fewer than the threshold of key-management operators collude, and that Zama's coprocessor, relayer and key-management service behave correctly and remain available. We also assume that the patient's and doctor's devices, and the web application delivered to them, are not compromised.

D. Out of Scope

We do not test compromised devices, malicious front-end code, attacks on Zama's infrastructure or surveillance of users' internet connections. We note, however, that plaintext exists in the patient's browser before encryption and in the doctor's browser after decryption, so a compromised client would defeat the system's confidentiality, and that an authorised doctor can always copy what they read.

V. Evaluation Method

A. Environments and Evidence Levels

We used four kinds of evidence, which differ in strength (Table II). Each result in Section VI states which kind it rests on.

TABLE II
Kinds of Evidence

Evidence	Description
Sepolia, real	Transactions sent and data read on the public testnet
Sepolia, simulated	Calls simulated against the live chain without sending a transaction (`staticCall`)
Local mock	Hardhat with Zama's mock fhEVM, used for attacks that would otherwise change public state
Code and documentation	Analysis of Zama's contract and SDK source code and official documentation

B. Tests

We inspected all publicly readable data for test consultations, including transaction data, event logs and the contract's public view functions, through Etherscan. We then ran automated access-control tests with Hardhat scripts. Decryption tests attempted to decrypt a consultation from the assigned doctor (as a control), an unassigned doctor, the administrator and a newly generated wallet. Permission tests simulated each restricted function from an unauthorised wallet, together with control calls from the authorised role, so that refusals could not be trivially true. Each role used a separate wallet; where roles overlapped in v1, the affected tests were labelled and excluded from the pass count. Because a refusal by Zama's

TABLE III
SOFTWARE VERSIONS

Ver.	Main packages
v1	`@zama-fhe/relayer-sdk` 0.4.1, `@fhevm/solidity` 0.11.1
v2	`@zama-fhe/relayer-sdk` 0.4.1, `@fhevm/solidity` 0.11.1; key results re-checked with `@zama-fhe/sdk` 3.7.1
v3	`@zama-fhe/sdk` 3.7.1, `@fhevm/solidity` 0.13.4, `@fhevm/sdk` 0.13.8, Hardhat plugin 0.13.0-1 (alpha)

TABLE IV
ACCESS-CONTROL TEST OUTCOMES

	v1	v2	v3 local	v3 Sepolia
Tests	46	48	23	215
Secure behaviour	35	41	23	213
Vulnerability found	3	4	0	0
Skipped or informational	8	3	0	0

service may be indistinguishable from a temporary outage, each refusal on Sepolia was paired with a successful request sent within 60 seconds. One v3 check affected by the order of tests was repeated with a fresh delegation.

C. Software Versions and Reproducibility

Zama's software changed during the study: the original relayer SDK was deprecated in November 2026, and Sepolia moved to FHEVM v0.13. Table III lists the versions used. The v2 source was committed after deployment, and its hash was checked against the deployment record. A pre-release (alpha) version of Zama's Hardhat plugin was used for v3 local tests, as no stable version matching FHEVM v0.13 was available. Code, test scripts and results are available at https://github.com/Nizam-shahr/confidential-medical-care; v3 was deployed from commit `01f4c65`, and the full results are on branch `v3-delegation-sdk-migration` at `a8fd261`.

VI. RESULTS

A. RQ1: What FHE Protects

Consultation content remained confidential in all tested cases *(Sepolia, real)*. Complaints were stored on-chain only as handles, and the public view functions returned no plaintext. Only the assigned doctor and the patient could decrypt consultation data. Requests from an unassigned doctor, the administrator (tested in v2, where it used a separate wallet) and an unrelated wallet were refused, and access became available only after the patient's grant. All attempts to call restricted functions from unauthorised wallets reverted, while the corresponding control calls succeeded. Table IV summarises the outcomes. In v1 and v2, refusals were enforced by the SDK's client-side permission check before requests reached Zama's key-management service.

In v1, the vulnerabilities were re-sharing through the contract (B30) and direct re-sharing through the ACL (E1, E2). In v2, B30 was fixed, while E1 and E2 remained, and two further tests showed that withdrawing access (C1) and closing (C2) did not end access; v1 could not run these tests, as no transactions were sent on it. For v3 on Sepolia, each row is a relayer request or simulation; two control rows failed because their delegation had expired during an earlier test, and both checks passed when repeated with a renewed delegation. Skipped tests in v1 reflect a role overlap (the administrator was also the patient) and the absence of transactions; informational tests recorded metadata rather than pass or fail outcomes.

B. RQ1: What Still Leaks

Consultation metadata. The function `getConsultationMeta` returned, in plaintext, the patient's wallet address, the identifier of the consulted doctor, the consultation status, its creation time and the number of encrypted chunks *(Sepolia, real)*. For one test consultation, it showed that the consultation was created on 14 April 2026 at 13:24 UTC and contained 19 chunks, indicating a message of roughly 150 characters. The handles themselves exposed the same count, since each records its position in the input, together with the chain identifier and the encrypted data type.

Doctor identity and specialty. The public doctor registry returned each doctor's wallet, name, specialty, credentials and number of consultations. Combined with the consultation metadata, this revealed that the test patient's wallet had consulted a psychiatrist, without any decryption.

Consultation history. The function `getPatientConsultations` returned every consultation linked to a patient wallet, so an observer could reconstruct a wallet's history of consultations and specialties using only public read calls.

Events. The contract's `ConsultationCreated` event names the patient, and the `Allowed` event emitted by Zama's ACL contract names the doctor's wallet when access is granted. Removing the contract's view functions would therefore not prevent linkage, since the same information can be read from storage and events.

Wallet funding (v3). Each per-doctor wallet is funded by a public transfer from the patient's main wallet. These transfers, together with consultation and delegation events, link the main wallet to each per-doctor wallet and to each doctor. Data remains separated between doctors, but the wallets are not unlinkable.

Table V summarises what a public observer can learn.

C. RQ2: Unauthorised Re-sharing

Through the application contract. In v1, a doctor who had been granted access could resubmit the patient's encrypted handles, with an empty input proof, as a new consultation addressed to a different doctor, and then grant that doctor access. In a local mock fhEVM environment, we reproduced the full chain: the second doctor decrypted the patient's

TABLE V
DATA VISIBLE TO A PUBLIC OBSERVER

Data	Visible?	What it reveals
Symptoms and response	No	Nothing without permission
Patient wallet	Yes	Links a wallet's consultations
Doctor and specialty	Yes	Likely condition
Creation time	Yes	When care was sought
Chunk count	Yes	Approximate message length
Response status	Yes	Whether the doctor replied
Doctor wallet and caseload	Yes	Doctors' identities and activity
Funding transfers (v3)	Yes	Links per-doctor wallets to the main wallet

complaint, and the result matched the original exactly *(local mock)*. On Sepolia, the first step was confirmed by simulation.

The root cause lies in how fhEVM treats inputs without a proof. When the proof is empty, a handle is accepted if the sender holds ACL permission on it, with no check of where the ciphertext came from or whose data it is. ACL permissions are attached to ciphertexts rather than to a context or purpose, so once the patient granted access, the doctor could use the ciphertext in any contract call. Because ConfidentialCare held its own permission on every submitted ciphertext, it then granted access to the second doctor using that permission. This is an instance of the confused deputy problem [27], in which a privileged program is tricked into misusing its authority on behalf of a less privileged caller.

In v2, we required a non-empty input proof for all encrypted inputs. A valid proof is signed for a specific user, contract and chain, and lists the exact handles it covers, each identified by its index. The patched contract rejected the re-sharing attack, replay of the patient's proof, a valid proof submitted together with another user's handles, and an empty response that could previously fill the consultation's single response slot *(local mock and Sepolia)*.

Directly through the ACL. The fix does not address a second route. Any address granted ACL permission on a ciphertext can call Zama's ACL contract directly to grant access to further addresses, or to make the ciphertext publicly decryptable. We confirmed this by simulation on v1 and v2 *(Sepolia, simulated)*; no data was actually shared or published. In fhEVM, a permission to decrypt is therefore also a permission to share.

D. RQ2: Withdrawing Access and Closing Consultations

Neither withdrawing access nor closing a consultation ended a doctor's ability to decrypt in v2 *(Sepolia, real)*. After the patient called `markAccessWithdrawn`, and after the consultation was closed, the doctor still decrypted the complaint using handles decoded from the original transaction data. The contract correctly rejected a response to a closed consultation, but because ACL permissions cannot be withdrawn once granted, and ciphertexts cannot be deleted, any access granted in v1 and v2 is effectively permanent.

E. RQ2: Delegated Decryption (v3)

Analysis of Zama's ACL contract showed that `allow` and `allowForDecryption` require the caller itself to hold permission on the handle, and that a delegation does not count as such permission *(code)*. A delegated doctor therefore cannot re-share the data, and a delegation passed onwards has no effect. In local tests, the delegated doctor could decrypt the complaint, but attempts to call `allow`, delegate onwards or make the data public all failed; the doctor was refused after revocation and after expiry; closing failed while a delegation was active; and a delegation from the wallet used with one doctor gave another doctor no access *(local mock)*. All v2 input-proof protections continued to hold. While a delegation is active, the assigned doctor can also decrypt their own response, since a delegation covers all of the patient wallet's data in the contract; this exposes nothing the doctor does not already hold.

On Sepolia *(Sepolia, real)*, the v3 contract (`0x9cF0...427B`) confirmed these results against Zama's live key-management service. Because the SDK does not pre-check delegations, we sent delegated decryption requests directly to Zama's relayer. With an active delegation, 11 or 12 of the 13 key-management servers returned their shares, above the threshold of 7, and the plaintext was correct in all 44 successful decryptions. Requests with no delegation, a revoked delegation or an expired delegation were refused by Zama's service in every case, with the error `not_allowed_on_host_acl`; each refusal was immediately followed by a successful request, ruling out outages. Requests signed by an attacker, or with a substituted key, returned no shares. Revocation took effect immediately: in all six trials, the first request after the revocation was confirmed was refused, within the same block. This contrasts with Zama's SDK documentation, which suggests a delay of about 10 blocks. A five-minute delegation ended at the first block after its expiry in both trials. Simulated re-sharing attempts by the delegated doctor reverted with `SenderNotAllowed`, delegations were isolated between doctors, and after revoke-then-close the doctor was refused.

VII. DISCUSSION

A. Confidentiality Is Not Anonymity

FHE protected what patients disclosed, and the design collects no names, emails or accounts. Patients are therefore pseudonymous, but not anonymous. A wallet remains unlinked to its owner only until something connects the two, and several routes can do so: obtaining ETH from an exchange that verifies identity, reusing the wallet elsewhere, network data such as IP addresses seen by the website host, RPC provider or relayer, matching timestamps, or someone close to the patient seeing the wallet address. Studies of Bitcoin and Ethereum show that such linkage is achievable in practice [22], [24]. Because blockchain records are public and permanent,

a link made even years later would expose the patient's entire consultation history retroactively, and it could never be erased. For stigmatised conditions, the specialty of the doctor consulted can be as revealing as the symptoms themselves.

B. Permission Means Ownership

Access control in FHE-based smart contracts depends not only on the cryptography, but on how contracts accept encrypted inputs and on how permissions can be passed on. Because any holder of an ACL permission can extend it, granting a third party ACL access to a patient's ciphertext is equivalent to giving them a copy that they may share. Delegated decryption avoids this, since a delegate can decrypt but cannot pass access on, and delegations can expire and be revoked.

C. Pitfalls for fhEVM Developers

Our work identified three pitfalls that developers can easily overlook. First, accepting encrypted inputs without a fresh input proof allows any party holding permission on a ciphertext to reuse it as their own. Zama's documentation recommends checking that the sender is authorised to use a ciphertext [26], but this check would not have prevented the re-sharing attack, because the doctor was legitimately authorised: the problem is not whether the sender may use the data, but whether it is theirs to reuse. Second, the current SDK creates a permanent delegation if no expiry is given, so a single missing parameter can leave a patient's data readable indefinitely. Third, Zama's ACL supports delegation across all contracts through a special address, which would widen a delegation far beyond its intended scope. Applications handling sensitive data should require fresh input proofs, always set an explicit expiry, never use all-contract delegation, and avoid granting ACL permissions to third parties.

D. Trade-offs of Delegation

Delegation solves re-sharing and revocation only at a cost. Because a delegation covers all of a wallet's data in a contract, isolating consultations requires one patient wallet per doctor, and each wallet must be funded with gas. Funding them from the patient's main wallet links them on-chain. Gas sponsorship could remove this link, for example using EIP-7702, which lets an ordinary wallet have its fees paid by a sponsor while keeping its address, together with a paymaster contract. However, a sponsor would then learn which wallets it pays for, moving trust rather than removing it. EIP-7702's use of the term "delegation" refers to delegating a wallet to contract code, and is unrelated to Zama's decryption delegation. Revocation itself was effective: in our Sepolia trials it applied within the same block, so once revoked, a doctor could not decrypt further.

E. Preliminary Cost Observation

Although performance was not the focus of this study, gas measurements on Sepolia indicate that the cost of FHE grows steeply with message length. On 4 November 2026, submitting a consultation required about 3.29 million gas for 6 chunks, 7.15 million for 16 chunks and 13.52 million for 32 chunks, the contract's maximum of 256 bytes, or roughly 0.39 million gas per additional chunk. Sepolia caps a transaction at 16,777,216 gas, so the cost of FHE directly limits how long a patient's message can be. Deploying v3 used 15.8 million gas, and the node's gas estimate of 19.1 million exceeded the cap, so an explicit gas limit was required. Because the limit is enforced in bytes while the interface counts characters, messages in other scripts would hold fewer characters: about 128 accented Latin characters or 64 emoji, by our reading of the code. This is untested, but matters for users who write in languages that use non-ASCII characters. Gas figures from April and November 2026 are not comparable, since Sepolia's gas pricing changed in between. A systematic evaluation of performance and cost against conventional baselines is left to future work.

F. Mitigating Metadata Leakage

Possible mitigations include encrypting doctor identifiers, avoiding on-chain patient indexes, using per-doctor wallets, sponsoring gas and submitting transactions through a relayer. Our findings show the limits of the first: even with encrypted doctor identifiers, the ACL's own events reveal which addresses are granted access, and the same holds for delegation: in v3, Zama's `DelegatedForUserDecryption` and `RevokedDelegationForUserDecryption` events record both the patient's per-doctor wallet and the doctor's wallet as indexed, searchable fields, together with the expiry date and, through the block time, when access was granted or revoked.

G. AI-Assisted Routing

An AI agent could help route patients to a suitable specialist. A server-side agent would, however, need to decrypt symptoms to interpret them, since current language models cannot practically operate on FHE-encrypted text, exposing data to the agent's operator or a model provider. A lightweight model running on the patient's device could recommend a specialty before the symptoms are encrypted, so that plaintext never leaves the device, and routing over structured encrypted inputs could be computed homomorphically. In either case, the selected doctor would remain visible on-chain, and automated triage raises clinical safety and regulatory questions.

H. Limitations

Our evaluation has several limitations. Refusals in v1 and v2 were enforced by the SDK's client-side check, so only the v3 tests, which bypassed the SDK, confirm enforcement by Zama's key-management service. Experiments ran on a testnet with synthetic data and a single implementation, and some attacks were demonstrated only in a local mock environment or by simulation. The v3 local tests used a pre-release Hardhat plugin. Zama's software changed during the study, so results apply to the versions listed in Table III. Sepolia's gas pricing also changed between deployments, so gas figures from

different dates are not comparable. We did not systematically measure performance or cost, evaluate usability with patients and doctors, or assess regulatory compliance in detail; the right to erasure and the treatment of health data as special category data under UK GDPR are particularly relevant to permanent on-chain records.

VIII. Conclusion and Future Work

This paper assessed whether FHE on a public blockchain can protect sensitive medical consultations. Using ConfidentialCare, we found that FHE kept consultation content confidential, and that patients need provide no personal details. However, on-chain metadata allowed an observer to reconstruct a wallet's consultation history, including the specialty of each doctor consulted, and this history would be exposed permanently if the wallet were ever linked to the patient. We identified and fixed a confused-deputy flaw that allowed encrypted data to be re-shared without consent, and showed that fhEVM's permission model allows any party granted access to re-share data independently of the application. Granted access could not be withdrawn. A redesign based on time-limited, revocable delegation prevented re-sharing, and on Sepolia revocation and expiry took effect immediately, at the cost of additional wallets, gas and trust assumptions.

Future work will measure performance and cost against conventional baselines, evaluate usability with patients and doctors, assess compliance with UK GDPR and medical device regulation, explore gas sponsorship to reduce wallet linkage, and investigate privacy-preserving AI-assisted routing, including the security and robustness of such agents.

Ethics Statement

All consultations were created by the author with synthetic data, and all doctor records were fictitious. No real patient data was collected, and no data belonging to other users of the deployed contracts was decrypted or analysed. Attacks that would have exposed data on a public network were demonstrated only in a local environment or by simulation.

Use of AI Tools

Claude Code (Anthropic) was used to assist with developing and running the security tests. Claude was also used to help draft and edit the text. All results and interpretations were reviewed and verified by the author.

Data and Code Availability

All code, test scripts and results are publicly available at https://github.com/Nizam-shahr/confidential-medical-care. Contract versions correspond to commits `f3bbdcd` (v1), `aa0e9aa`, tagged `v2-sepolia` (v2), and `01f4c65` (v3); results are under `results/` on branch `v3-delegation-sdk-migration`. The contracts are verified on Etherscan at `0x52e7a7c4414921d7a68b00d5abae51e2861e83aa` (v1), `0x708de98F102B3E99071b92c851dB1C832e197D59` (v2) and `0x9cF0D8f6975C4779C40947D9fC10BCd43fB4427B` (v3) on Sepolia.

References

[1] NEPWHAN and GNP+, "Nigeria PLHIV Stigma Index Survey 2.0 Report," National Agency for the Control of AIDS, Abuja, 2021. [Online]. Available: https://naca.gov.ng/wp-content/uploads/2023/03/NIGERIA-PLHIV-STIGMA-INDEX-2.0-REPORT_2021.pdf

[2] K. Gänsler, "Nigerian dating agency for people with HIV-AIDS," *Deutsche Welle*, Dec. 2016. [Online]. Available: https://www.dw.com/en/nigerian-dating-agency-for-people-with-hiv-aids/a-36261864

[3] Q. Grundy *et al.*, "Data sharing practices of medicines related apps and the mobile ecosystem: traffic, content, and network analysis," *BMJ*, vol. 364, l920, 2019.

[4] S. Shi, D. He, L. Li, N. Kumar, M. K. Khan and K.-K. R. Choo, "Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey," *Computers & Security*, 2020.

[5] K. Kiania, S. M. Jameii and A. M. Rahmani, "Blockchain-based privacy and security preserving in electronic health: A systematic review," *Multimedia Tools and Applications*, 2023, doi: 10.1007/s11042-023-14488-w.

[6] A. L. A. Fonsêca, I. M. P. Barbalho, F. Fernandes *et al.*, "Blockchain in health information systems: A systematic review," *Int. J. Environ. Res. Public Health*, vol. 21, no. 11, p. 1512, 2024, doi: 10.3390/ijerph21111512.

[7] I. Chillotti, N. Gama, M. Georgieva and M. Izabachène, "TFHE: Fast fully homomorphic encryption over the torus," *Journal of Cryptology*, vol. 33, no. 1, pp. 34–91, 2020, doi: 10.1007/s00145-019-09319-x.

[8] A. Guimarães, E. Borin and D. F. Aranha, "MOSFHET: Optimized software for FHE over the torus," *Journal of Cryptographic Engineering*, vol. 14, no. 3, pp. 577–593, 2024, doi: 10.1007/s13389-024-00359-z.

[9] Zama, "TFHE-rs v1.4: GPU performance breakthrough and more," Oct. 2025. [Online]. Available: https://www.zama.org/post/tfhe-rs-v1-4-gpu-performance-breakthrough-and-more

[10] A. Kosba, A. Miller, E. Shi, Z. Wen and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Security and Privacy*, 2016.

[11] R. Cheng *et al.*, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *Proc. IEEE EuroS&P*, 2019.

[12] A. Nilsson, P. N. Bideh and J. Brorsson, "A survey of published attacks on Intel SGX," arXiv:2006.13598, 2020.

[13] D. Aronoff *et al.*, "SoK: Fully-homomorphic encryption in smart contracts," Cryptology ePrint Archive, Paper 2025/527, 2025.

[14] Zama, "Introducing the fhEVM coprocessor," Dec. 2024. [Online]. Available: https://www.zama.org/post/fhevm-coprocessor

[15] C. Bootland *et al.*, "Threshold (fully) homomorphic encryption," Cryptology ePrint Archive, Paper 2025/699, 2025.

[16] Zama, "Decentralized key generation for Zama mainnet has successfully completed," Dec. 2025. [Online]. Available: https://www.zama.org/post/decentralized-key-generation-for-zama-mainnet-has-successfully-completed

[17] A. Hasselgren, P. K. Wan, M. Horn, K. Kralevska, D. Gligoroski and A. Faxvaag, "GDPR compliance for blockchain applications in healthcare," arXiv:2009.12913, 2020.

[18] D. Kahrobaei, A. Wood and K. Najarian, "Homomorphic encryption for machine learning in medicine and bioinformatics," *ACM Computing Surveys*, 2020.

[19] K. Munjal and R. Bhatia, "A systematic review of homomorphic encryption and its contributions in healthcare industry," *Complex & Intelligent Systems*, vol. 9, pp. 3759–3786, 2023.

[20] J. S. Rauthan, "Homomorphic encryption in healthcare industry applications for protecting data privacy," arXiv:2501.04058, 2025.

[21] J. Mayer, P. Mutchler and J. C. Mitchell, "Evaluating the privacy properties of telephone metadata," *Proc. Natl. Acad. Sci. USA*, vol. 113, no. 20, pp. 5536–5541, 2016.

[22] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer and S. Capkun, "Evaluating user privacy in Bitcoin," in *Financial Cryptography and Data Security*, LNCS vol. 7859, 2013, pp. 34–51.

[23] S. Meiklejohn *et al.*, "A fistful of bitcoins: Characterizing payments among men with no names," in *Proc. ACM Internet Measurement Conf.*, 2013.

[24] F. Béres, I. A. Seres, A. A. Benczúr and M. Quintyne-Collins, "Blockchain is watching you: Profiling and deanonymizing Ethereum users," arXiv:2005.14051, 2020.

[25] J. Zhou, C. Hu, J. Chi, J. Wu, M. Shen and Q. Xuan, “Behavior-aware account de-anonymization on Ethereum interaction graph,” arXiv:2203.09360, 2022.
[26] Zama, “Access control list” and “ACL examples,” FHEVM Solidity guides, v0.10. [Online]. Available: https://docs.zama.ai (accessed 4 Nov. 2026).
[27] N. Hardy, “The confused deputy (or why capabilities might have been invented),” *ACM SIGOPS Operating Systems Review*, vol. 22, no. 4, pp. 36–38, 1988.