AWARE: A Non-Interactive Anonymous Whistleblowing System against Recipient Corruption

Ge Gao, Haining Yu, Jinbo Yang, Dongyang Zhan, Xiaohua Jia

***Abstract*—Anonymous whistleblowing systems allow insiders to report organizational misconduct while preserving confidentiality and anonymity. Existing systems often use Tor-based submission and encryption under a recipient public key, so the mailbox does not learn the submitter's network address or report contents. This protection does not remove reliance on the parties that authenticate insiders or open reports. Two corruption risks remain: (1) token-issuance records held by an authentication server can be correlated with a submission to link the report to a registered insider; and (2) a recipient with unilateral plaintext access may suppress, selectively disclose, or influence the report before any accountable opening process. We present *AWARE*, a non-interactive whistleblowing protocol for anonymous report submission by eligible insiders and threshold opening of reports. AWARE separates identity authentication from submission: after scheduled token refresh, an eligible insider can submit a report with a locally finalized token, without contacting the authentication server at submission time. AWARE also replaces single-recipient opening with committee threshold opening: report recovery requires a threshold of committee decryption shares that pass verification. Under the stated corruption and leakage assumptions, we prove the corresponding anonymity, token unforgeability, report confidentiality before opening, and opening integrity properties. A Java implementation shows practical costs in the tested settings.**

***Index Terms*—whistleblowing system, anonymous reporting, anonymous authentication token, threshold decryption**

I. Introduction

WHISTLEBLOWING systems [1]–[4] are channels through which employees, contractors, and other insiders report suspected fraud, abuse, safety violations, or other organizational misconduct to designated recipients. Such channels must preserve confidentiality and anonymity. They must also remain reliable when a report concerns the organization that operates, funds, or can influence the channel. This is a practical requirement. The U.S. Securities and Exchange Commission's Whistleblower Program received approximately 27,000 tips in fiscal year 2025,[1] and the Association of Certified Fraud Examiners reports in its 2026 occupational-fraud study that tips were the initial detection method in 43% of fraud cases.[2] These data show that whistleblowing systems are part of ordinary organizational reporting. They also explain why trust in the channel matters: an organization that benefits from suppressing a report may pressure or corrupt the parties that authenticate insiders, store submissions, or open reports.

Prior work [5] identifies several requirements for anonymous whistleblowing: whistleblower anonymity, confidentiality and integrity of disclosures, plausible deniability for the whistleblower, and recipient authenticity. Surveys [6] and policy guidance [7] also stress protected reporting paths and identity protection in organizations. Existing systems usually combine Tor-based submission with encryption under a recipient public key. SecureDrop [1] and GlobaLeaks [2], for example, use a Tor onion-service submission server as a mailbox. A whistleblower may obtain a pseudonymous return credential, encrypt a report under a recipient public key, and submit the ciphertext through the mailbox. The recipient later retrieves the ciphertext and decrypts it locally. Formal work [8], [9] also captures the need to authenticate eligible insiders before an insider-restricted system accepts reports.

Tor-based submission hides the submitter's network address from the mailbox, and encryption hides the report contents while the report is stored there. SecureDrop, GlobaLeaks, and WikiLeaks [3] rely on this separation, and the newer SecureDrop Protocol reduces server-visible metadata by removing user accounts and minimizing message metadata.[3] This protects the submission mailbox, but it does not remove reliance on parties that may be corrupted. If the authentication server or the recipient is corrupt, two risks remain. First, in systems that enforce insider eligibility, token-issuance records held by an authentication server may be correlated with a submission to link the report to a registered insider, if logs are retained or later compromised. Second, encryption under a single recipient public key gives one recipient or workstation unilateral plaintext access. A corrupted party can then suppress, selectively disclose, or influence a report before any accountable opening process. We study these risks without replacing the underlying anonymous communication channel.

Challenge 1: authentication without token issuance at submission time. An insider-restricted reporting channel is not open to arbitrary external senders. It must check that a reporter is eligible, but this check must not identify which eligible insider later submits a report. A direct design lets an authentication server issue anonymous authentication tokens. If issuance happens during the reporting interaction, the issuance event gives that server a timing signal tied to the submission. Even when the report is sent through Tor or a mix network,

This work was supported in part by the National Natural Science Foundation of China under Grant 62172123 and Grant 62302122 and Heilongjiang Provincial Natural Science Foundation of China under Grant JQ2024F001. (Corresponding author: Haining Yu.)

G. Gao, H. Yu, J. Yang, and D. Zhan are with the School of Cyberspace Science, Harbin Institute of Technology, Harbin 150001, China (e-mail: yuhaining@hit.edu.cn).

Xiaohua Jia is with the Department of Computer Science, City University of Hong Kong, Hong Kong, SAR, China.

[1]https://www.sec.gov/files/fy25-annual-whistleblower-report.pdf

[2]https://www.acfe.com/about-the-acfe/newsroom-for-media/press-releases/press-release-detail?s=occupational-fraud-2026-a-report-to-the-nations-pr

[3]https://freedom.press/tech/news/introducing-securedrop-protocol/

an honest-but-curious authentication server can compare token-issuance times with bulletin-board upload patterns. Scheduled token refresh removes this submission-time signal. Each eligible user receives the same number of tagged pre-tokens during fixed refresh epochs. A later whistleblower finalizes an unused pre-token locally and posts the report without contacting the authentication server. Timing, eligibility-set size, token-use patterns, and network-path metadata remain leakage sources addressed by traffic- and metadata-hiding techniques.

Challenge 2: report opening without unilateral plaintext control. Encryption under a single recipient public key protects a report while it is stored by the mailbox. After retrieval, however, the recipient side controls the plaintext. A corrupted recipient, administrator, or workstation can read the report before any collective process and decide whether to suppress, selectively disclose, or influence it. This risk is realistic in adversarial organizations: insider-threat guidance identifies collusion between internal personnel and external actors as a recurring pattern of malicious activity,[4] and bid-rigging enforcement actions show the financial incentives behind coordinated misconduct.[5] A direct response is to distribute opening authority among a committee. A threshold rule alone is not enough; it mainly changes the trust assumption. If corrupted committee members can meet the opening threshold, confidentiality is outside the claim, and recent work on threshold-based whistleblowing makes this limitation explicit for incentive-driven collusion [10]. A threshold design must state how many corrupted members it tolerates and must reject malformed or replayed shares before reconstruction. The opening protocol should therefore encrypt each report under a committee public key, require committee members to release decryption shares only after the public opening condition holds, and reconstruct only from shares that pass verification.

We present *AWARE*, a non-interactive whistleblowing protocol for anonymous report submission by eligible insiders and threshold opening of reports. AWARE separates identity authentication, report submission, public posting, and report opening. Corruption of one party therefore does not by itself give both submission-linkage evidence and plaintext-opening power. After setup and scheduled refresh, an eligible insider generates the reporting token and ciphertext locally and posts one report entry, without an online exchange with the authentication server or the opening committee. After the public opening condition holds, committee members post share records with verification material. Reconstruction uses only shares that pass verification. This paper makes three contributions:

1) We introduce a scheduled-refresh anonymous authentication mechanism for insider-restricted reporting. A registered insider can later submit a report with a locally finalized token, without contacting the authentication server.
2) We design a verifiable threshold-opening mechanism for reports. Report recovery requires a threshold of committee decryption shares that pass verification after the public opening condition holds.
3) We formalize the corruption and leakage model and prove token-use unlinkability, authentication-token unforgeability, confidentiality of reports before opening, and opening integrity. We also give an optional CCA-secure strengthening, implement AWARE in Java, and measure the setup, refresh, submission, and opening phases. The measurements show practical costs in the tested settings.

The rest of this paper is organized as follows. Section II reviews related work. Section III introduces the technical preliminaries. Section IV states the system model, threat model, and design goals. Section V gives the AWARE construction. Section VI provides the security analysis. Section VII discusses an optional CCA strengthening for report encryption. Section VIII evaluates performance, and Section IX concludes.

II. Related Work

Related work covers several parts of anonymous reporting. Network systems hide the path to a submission service; token and credential systems support anonymous authentication; voting and reputation systems study accountable anonymous participation. AWARE has a different combination of requirements: registered insiders, no identity-authentication event at submission time, and report opening from committee decryption shares that pass verification. We discuss Tor/onion-service deployments [1]–[4], [11], [12], metadata-hiding communication systems [13]–[16], anonymous tokens, credentials, and reporting schemes [17]–[23], [36], electronic voting [24]–[29], and anonymous reputation [30]–[35].

Tor-based systems. SecureDrop, GlobaLeaks, and related deployments use Tor onion services as submission mailboxes. Onion services remove Tor exit relays, hide the service location, and hide the submitter's network address from the submission service. This is network-level protection. A compromised or misconfigured submission server may still observe reply-box state, message arrival times, and recipient retrieval behavior. CenTor [12] improves onion-service performance and security, and the proposed SecureDrop Protocol reduces account-creation and message-flow metadata in a proof-of-concept design [4]. These systems are compatible with AWARE, but they address a different layer. AWARE addresses identity authentication and report opening when reporting infrastructure may be corrupted.

Anonymous communication systems. Mix-net and private-information-retrieval (PIR)-based systems [13]–[16] hide communication metadata beyond source IP addresses. Express lowers the cost of metadata-hiding communication [14], XRD scales private messaging [15], and Aimless Onions reduces topology-distribution overhead for mix networks [16]. These systems can carry report traffic with stronger traffic-analysis resistance than ordinary client-server submission. They do not authenticate registered insiders, bind reports to anonymous authentication tokens, or distribute report-opening authority among a committee.

Whistleblowing and token systems. Anonymous whistleblowing over authenticated channels is formalized by anonymous transfer [8]; lower bounds show limits on non-trivial anonymity in that model [9]. Other reporting schemes use ring signatures,

[4]https://www.cisa.gov/topics/physical-security/insider-threat-mitigation/defining-insider-threats

[5]https://www.justice.gov/opa/pr/four-defendants-plead-guilty-ongoing-bid-rigging-fraud-and-bribery-investigation-related-us

TABLE I
COMPARISON OF REPRESENTATIVE APPROACHES FOR ANONYMOUS REPORTING. THE COMPARISON FOLLOWS EACH APPROACH'S STATED MODEL AND SEPARATES IDENTITY HIDING, CONTROLLED OPENING, TOKEN-USE UNLINKABILITY, REPORT INTEGRITY, AND THRESHOLD OPENING.

Scheme	Properties				
	Reporter Anonymity[1]	Controlled Opening[2]	Unlinkable Reporting[3]	Report Integrity[4]	Threshold Opening[5]
Tor / Onion Services [1], [2], [11], [12]	◐	◐	○	○	○
Mix / PIR [13]–[16]	◐	◐	○	○	○
Credential / Token [17]–[23]	◐	○	◐	●	○
Electronic Voting [24]–[29]	◐	●	○	●	●
Anonymous Reputation [30]–[35]	◐	○	○	●	○
AWARE (Ours)	◐	●	◐	●	●

Legend: ● Supported under the stated model ◐ Partial / conditional ○ Not supported

[1] **Reporter Anonymity:** The reporter's identity is hidden from the parties covered by the scheme's threat model. ◐ marks protection limited to network observers, verifiers, or selected servers. AWARE's claim is token-layer anonymity under scheduled refresh and excludes network metadata.

[2] **Controlled Opening:** Plaintext recovery is not left to one recipient before the opening condition. AWARE requires t valid committee shares; Tor/Mix/PIR deployments usually leave final decryption to the designated recipient.

[3] **Unlinkable Reporting:** Identity checking and report submission are unlinkable under scheduled-refresh leakage. Timing, small groups, fresh-token patterns, and network metadata are deployment-layer leakage. Privacy Pass gives unlinkable tokens outside the registered-insider setting.

[4] **Report Integrity:** The system controls acceptance of well-formed submissions, for example through authentication-token checks and append-only report records. It does not decide report truthfulness.

[5] **Threshold Opening:** Recovery requires t verified committee shares and rejects malformed or falsely attributed shares. Liveness still depends on enough valid shares being released.

group signatures, proxy signatures, or blockchains [19], [20], [23], [36] to support anonymous reporting or rewards. Their opening model is different: they do not give threshold-verified opening of encrypted reports against recipient corruption. Privacy Pass separates token issuance from token use [17], [18]. Anonymous credentials and keyed-verification credentials [21], [22], [37]–[39] support selective disclosure or anonymous authentication. These primitives do not by themselves give scheduled insider-token issuance, encrypted report submission, share verification, and threshold opening.

Electronic voting. Electronic-voting protocols [24]–[29] combine privacy with public verifiability through tools such as bulletin boards, threshold tallying, and verifiable decryption. Their goals are election-specific, including ballot privacy, receipt-freeness, coercion resistance, tally hiding, and cast-as-intended verification. AWARE uses public records and threshold opening, but the object being opened is different. A report must remain confidential until the opening condition holds, and no single recipient should control plaintext recovery.

Anonymous reputation. Anonymous reputation systems [30]–[35] allow users to contribute feedback while limiting identity exposure. Their main goals are reputation aggregation, Sybil resistance, and, in some systems, conditional deanonymization. They do not separate scheduled token issuance from later report submission for registered insiders, and they do not provide threshold opening of encrypted reports.

Summary. Table I compares representative systems along the five properties used above. AWARE is marked partial for reporter anonymity and unlinkable reporting because these claims are token-layer and scheduled-refresh claims, not traffic-anonymity claims. Under the stated corruption model, AWARE provides controlled and threshold opening through verified committee shares.

III. PRELIMINARIES

A. Notation

Let λ be the security parameter, and let $\mathrm{negl}(\lambda)$ denote a negligible function. All algorithms and adversaries are probabilistic polynomial-time (PPT) unless stated otherwise. For $n \in \mathbb{N}$, write $[n] = \{1, \ldots, n\}$. We write $x \xleftarrow{\$} S$ for uniform sampling from a finite set S. Group exponents are taken modulo the relevant group order. Structured values are encoded canonically before they are hashed, signed, or used as associated data. We write $\|$ for byte-string concatenation.

B. Bilinear Groups

We use asymmetric bilinear groups. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic groups of prime order p, with generators g and $\widetilde{g}$, respectively, and let $\mathbb{G}_T$ be a group of the same order. A pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is efficiently computable, non-degenerate, and bilinear: $e(X^a, Y^b) = e(X, Y)^{ab}$ for all $X \in \mathbb{G}_1$, $Y \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$. Type-3 means that no efficient isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$ is available, and no pairing is available inside either source group.

C. Dealerless Distributed Key Generation

A dealerless distributed key-generation (DDKG) protocol lets n parties generate a threshold key without assigning the secret to a dealer. We use the abstraction

$$(pk_{\mathsf{th}}, pk_c, (sk_i^{\mathsf{th}})_{i\in[n]}) \leftarrow \mathsf{DDKG}.\mathtt{KeyGen}(1^\lambda, n, t)$$

over authenticated private channels and authenticated broadcast. If the protocol does not abort, pk_{th} is the threshold public

key, sk_i^{th} is party i's local share, fewer than t shares do not determine the secret, and pk_c is public material for verifying attributed decryption shares. Such protocols can be built from verifiable secret sharing [40].

D. Threshold Encryption with Verifiable Decryption Shares

Let TGS be a threshold encryption scheme with associated data and verifiable decryption shares. It encrypts a message under a threshold public key and reconstructs the plaintext only from decryption shares that pass public verification. Associated data is public and authenticated but not encrypted. Our instantiation follows Boneh and Shoup's associated-data threshold encryption construction [41]. The interface consists of `DKeyGen`, `Enc`, `CtVrfy`, `ShareDec`, `ShareVrfy`, and `Combine`.

- $(pk_{\mathsf{th}}, pk_c, (sk_i^{\mathsf{th}})_{i\in[n]}) \leftarrow \texttt{DKeyGen}(1^\lambda, n, t)$: generate the threshold public key, public verification material, and one local decryption state for each party.
- $\mathsf{ct} \leftarrow \texttt{Enc}(pk_{\mathsf{th}}, M, \mathsf{ad})$: encrypt M under pk_{th} with associated data ad.
- $b \leftarrow \texttt{CtVrfy}(pk_{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$: publicly verify ciphertext well-formedness.
- $d_i \leftarrow \texttt{ShareDec}(sk_i^{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$: compute party i's decryption share, or reject if the ciphertext is invalid.
- $b \leftarrow \texttt{ShareVrfy}(pk_c, i, \mathsf{ct}, \mathsf{ad}, d_i)$: publicly verify that d_i is a valid share attributed to party i.
- $(M/\perp, \mathsf{blame}) \leftarrow \texttt{Combine}(pk_c, \mathsf{ct}, \mathsf{ad}, \{d_j\}_{j\in\mathcal{J}})$: combine the input shares and output the plaintext or $\perp$, together with blame information for invalid shares.

We require three properties: (i) correctness, namely `Combine` outputs M for every honestly generated ciphertext and any set of at least t honestly generated valid shares; (ii) AD-only CCA confidentiality against fewer than t corrupted decryption parties with admissible share-decryption access; and (iii) ciphertext and share soundness: except with negligible probability, values accepted by `CtVrfy` and `ShareVrfy` satisfy the validity relations of the Boneh–Shoup threshold instantiation.

E. Tagged Non-interactive Blind Signatures (TNIBS)

A tagged non-interactive blind signature (TNIBS) scheme [42] binds issuance to a public tag η. A signer issues a presignature for $(\mathsf{pk}_R, \eta, \mu)$. The recipient uses sk_R to finalize the presignature into a signature on a randomized message m, without contacting the signer. We use the interface $\mathsf{TNIBS} = (\texttt{KeyGen}, \texttt{RKeyGen}, \texttt{Issue}, \texttt{Obtain}, \texttt{Verify})$.

- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \texttt{KeyGen}(1^\lambda)$: generate the signer key pair.
- $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \texttt{RKeyGen}(1^\lambda)$: generate the recipient key pair.
- $\mathsf{psig} \leftarrow \texttt{Issue}(\mathsf{sk}, \mathsf{pk}_R, \eta, \mu)$: generate a presignature bound to pk_R, public tag η, and public message μ.
- $(m, \sigma) \leftarrow \texttt{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{pk}_R, \eta, \mu)$: finalize the presignature into a signature σ on message m.
- $b \leftarrow \texttt{Verify}(\mathsf{pk}, \eta, (m, \sigma))$: output $b = 1$ if the signature is valid, and $b = 0$ otherwise.

We use Hanzlik's tagged construction [42] as a black-box TNIBS primitive with public tags. We need three properties: (i) one-more unforgeability within each public tag, where multiple randomized signatures on the same (η, m) pair count once; (ii) tagged recipient blindness, which hides which recipient public key was used to finalize a valid signature under a fixed public tag; and (iii) tagged nonce blindness, which hides which nonce and presignature were used to finalize a valid signature under a fixed recipient public key and public tag.

IV. Models and Problem Definition

A. System Model

The system model consists of four entities (Fig. 1).

- **Authentication Server ($\mathcal{AS}$):** The organization uses $\mathcal{AS}$ to authenticate eligible insiders. The server maintains the issuer signing key and a registration database that binds each eligible insider's real identity to a registered public key. At each refresh epoch, $\mathcal{AS}$ authenticates every registered user in the relevant eligibility class and delivers m tagged pre-tokens bound to that user's registered key. $\mathcal{AS}$ is not contacted during report submission.
- **User ($\mathcal{U}$) / Whistleblower ($\mathcal{W}$):** A user $\mathcal{U}$ is an eligible insider who holds secret key usk and a local pool of unused pre-tokens or finalized tokens. The same party is denoted by $\mathcal{W}$ when it submits a report M. To submit M, $\mathcal{W}$ locally finalizes one unused pre-token into an authentication token, encrypts M, and submits the token-bound ciphertext to $\mathcal{BB}$. The real-identity binding for $\mathcal{U}$ is established by institutional checks, such as employment, membership, or compliance verification.
- **Bulletin Board ($\mathcal{BB}$):** The bulletin board is a public append-only record. Once accepted, an entry cannot be modified, deleted, reordered, or falsely attributed. $\mathcal{BB}$ provides attribution for committee entries but does not decide token, ciphertext, or opening-share validity. Committee members perform these checks and use the append-only order for replay.
- **Committee Members ($\{\mathcal{CM}_i\}_{i=1}^n$):** The committee consists of n designated members. A dealerless DKG generates the threshold public key and local threshold-decryption states, so no party learns the full threshold secret. Each $\mathcal{CM}_i$ also holds the committee-only share-encryption key $k_{\mathcal{BB}}$. Committee members read $\mathcal{BB}$, verify accepted ciphertext records, and, after the public opening condition holds, publish encrypted decryption shares under $k_{\mathcal{BB}}$. A report is reconstructed from t shares that pass committee verification.

AWARE has three phases. During scheduled refresh, $\mathcal{AS}$ issues pre-tokens independently of later reports. During report submission, a whistleblower submits a token-bound ciphertext to $\mathcal{BB}$ without contacting $\mathcal{AS}$. During opening, committee members release decryption shares only after the public opening condition holds. Registration, refresh delivery, DKG, and committee-only key establishment use authenticated channels; $\mathcal{BB}$ is publicly readable and provides the order used for replay handling.

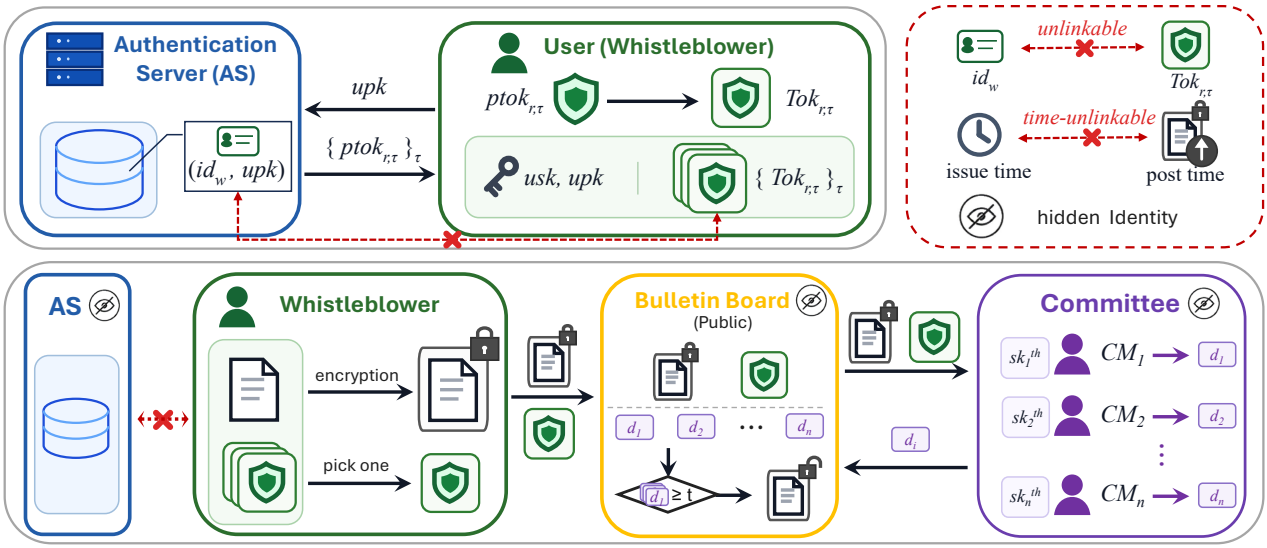


Fig. 1. System model of AWARE. The authentication server delivers pre-tokens through scheduled refresh delivery, the whistleblower submits a token-bound ciphertext to the bulletin board, and the committee opens accepted reports only after the public opening condition holds. Here, r indexes refresh epochs and $\tau \in [m]$ indexes the pre-token slots assigned to each eligible user in an epoch.

B. Threat Model

We consider adversaries after honest initialization of public parameters, registered user keys, and committee setup. They attempt to (i) link accepted reports to registered identities, (ii) obtain acceptance for reports not backed by issued tokens, (iii) learn report contents before opening, or (iv) influence the opening result through malformed or replayed shares.

Authentication server. The authentication server is semi-honest during identity authentication and pre-token issuance. It follows the refresh protocol and the registered eligibility policy, but the adversary sees its local view. This view includes the registration database, issuer signing key, tag policy, and issuance history. It may be combined with the public record and the states of fewer than t corrupted committee members. The server does not observe local token finalization.

Users and registration. Registration is honest: each eligible insider is bound to a registered public key and receives the corresponding secret key. False enrollment and registration coercion are outside the model. After registration, users and external parties may be malicious; they may submit malformed ciphertexts, reuse or transfer tokens, or seek acceptance with tokens outside their own issuance history. A finalized token is a bearer credential once it leaves the user's local state.

Bulletin board. The bulletin board is an honest append-only public record for ordering and committee attribution. It provides no privacy and does not decide token, ciphertext, or share validity. The adversary reads all records, submits records for corrupted parties, and may attempt replay or ordering attacks; validity is determined by the verification procedures.

Committee. The adversary may corrupt fewer than t committee members; corrupted members are malicious and reveal their local opening state. They may withhold shares, submit malformed shares, replay prior shares, attempt early decryption, or collude with the authentication server. Honest committee members keep their local state secret and release decryption shares only after the public opening condition holds for an accepted report. Share withholding affects availability.

C. Security Goals

Under this threat model, AWARE has the following security goals. The formal definitions are given in Section VI.

Whistleblower anonymity. The authentication server knows registration identities and issuance records. Under the scheduled-refresh leakage model, an accepted authentication token must be unlinkable from the pre-token issuance record from which it was finalized, including the registered public key, issuer-chosen nonce, and pre-token. This claim is limited to the token layer. Submission time, bulletin-board order, epoch information, anonymity-set size, report content, and the network path are outside this property.

Authenticated report submission. Every accepted report must be backed by a valid unused authentication token. The adversary may spend legitimately obtained tokens, but the number of accepted reports with distinct replay tags must not exceed the number of tokens it obtained. Replay handling accepts only the first valid ciphertext for each tag. As with other bearer anonymous tokens, voluntary disclosure or loss of a finalized token outside the protocol is not covered.

Threshold report confidentiality. Before the opening condition holds, an accepted ciphertext must hide the report content, except for length and public metadata, from outsiders and from any adversary corrupting fewer than t committee members. The adversary may learn corrupted members' local threshold states and, through a corrupted committee member, the committee-only share-encryption key $k_{\mathcal{BB}}$. Honest members release no target decryption share before opening. The confidentiality claim ends once the opening condition holds and t valid shares are available.

Committee-internal opening integrity. Opening verification is internal to the committee. Committee members decrypt share payloads under $k_{\mathcal{BB}}$ and verify each share against the target ciphertext and the sender's public verification key. Fewer than t corrupted members cannot cause an honest member to accept a plaintext inconsistent with the accepted ciphertext and verified shares. The property does not provide public

verifiability or liveness. Reconstruction requires at least t valid shares; $n \geq 2t-1$ is sufficient against up to $t-1$ withholding members when honest members can complete opening.

V. Overview of AWARE

A. Technical Overview

AWARE uses two separations. First, identity authentication is separated from report submission: an eligible insider can later submit a report without creating a fresh authentication event visible to $\mathcal{AS}$. Second, report submission is separated from report opening: no single recipient can decrypt before the public opening condition holds.

If $\mathcal{AS}$ issued an anonymous token when a user is ready to report, the issuance time itself would be visible to $\mathcal{AS}$. AWARE avoids this submission-time signal by scheduled refresh delivery. In each refresh epoch, $\mathcal{AS}$ issues m tagged pre-tokens to every eligible user, using a fixed schedule, a uniform response format, and delivery records independent of later reports. Most pre-tokens are unused. When a user later reports, the user runs `RegObt` locally on one unused pre-token, obtains a finalized authentication token, and posts the report record to $\mathcal{BB}$ without contacting $\mathcal{AS}$. The ciphertext uses $\mathsf{ad} = \mathsf{Tok}_{r,\tau}$, so committee checks are tied to the accepted token and its first ledger-ordered ciphertext. The anonymity proof uses the blindness properties of the TNIBS construction under the scheduled-refresh leakage model.

Encrypting a report separately to several recipients does not solve the opening problem. Each recipient still receives a ciphertext that it can decrypt alone, so one compromised or colluding recipient can learn the report before any verifiable opening record exists. AWARE encrypts each report once under a committee public key generated by dealerless DKG. Honest committee members release no decryption share before the opening condition holds. After opening is authorized, each member posts an encrypted share to $\mathcal{BB}$. Committee members decrypt the share payloads, verify them against the ciphertext and the sender's verification key, and reconstruct only after t valid shares are available. Report confidentiality reduces to AD-only CCA security of the Boneh–Shoup threshold-decryption scheme, and opening integrity reduces to ciphertext and share-verification soundness.

AWARE can be used with Tor, mix networks, or other network-anonymity mechanisms. The protocol itself removes token issuance at submission time and replaces single-recipient opening with threshold committee opening. The analysis does not model the network path. Login behavior, traffic volume, token freshness, and anonymity-set size remain deployment leakage.

B. Details of AWARE

Figure 2 gives the protocol syntax. We record below the main state changes and the checks that matter for the security analysis.

Initialization. Setup creates the authentication and opening keys. $\mathcal{AS}$ generates the TNIBS signing key $(\mathsf{ask}, \mathsf{apk})$ used for pre-token issuance. Committee members run dealerless DKG for the Boneh–Shoup threshold-decryption key: each $\mathcal{CM}_i$ receives sk_i^{th}, and pk_{th} and pk_c are public. The committee also establishes $k_{\mathcal{BB}}$ for share encryption. No single party receives enough secret material to decrypt reports.

Registration and token issuance. `RegU` generates an honest-key TNIBS recipient key $(\mathsf{usk} = x, \mathsf{upk} = g^x)$, and upk is registered under a real identity via institutional checks. At each refresh epoch, `RegIss` delivers m tagged pre-tokens $\mathsf{ptok}_{r,\tau}$ per user through scheduled refresh delivery. `RegObt` is local to the user: the user verifies $h_{r,\tau} = H_1(\mathsf{nonce}_{r,\tau})$, checks the TNIBS pre-signature, and finalizes $\mathsf{Tok}_{r,\tau} = (\eta_r, \rho_{r,\tau}, \sigma_{r,\tau})$, where $\rho_{r,\tau} = h_{r,\tau}^{x^{-1}}$. This step is invisible to $\mathcal{AS}$: the server sees issuance, but not which pre-token becomes a report token. The well-formedness checks prevent $\mathcal{AS}$ from embedding user-specific malformed group elements into pre-tokens.

Reporting and authentication. The whistleblower selects an unused $\mathsf{Tok}_{r,\tau}$, runs `RepGen` to encrypt M under pk_{th} with associated data $\mathsf{Tok}_{r,\tau}$, and posts $(\mathsf{Tok}_{r,\tau}, \mathsf{ct})$ to $\mathcal{BB}$. Committee members run `WAuth` and TGS.`CtVrfy` to validate the token and ciphertext. Replay detection uses ledger order: the first valid posting for a replay tag $\rho_{r,\tau}$ consumes that tag; later ones are ignored. Authentication is derived from token validity, not from hidden issuer-side state.

Threshold opening. Threshold opening is a separate phase governed by the public opening condition. Before that condition holds, honest committee members do not release decryption shares for an accepted report. Once opening is authorized, each committee member runs `RepDec`: it verifies ciphertext well-formedness, computes a decryption share $w_i = v^{\alpha_i}$, encrypts it under $k_{\mathcal{BB}}$, and posts $\hat{d}_i$ to $\mathcal{BB}$ over an authenticated channel. `RepCom` decrypts the shares, verifies each via TGS.`ShareVrfy`, and runs `Combine` to recover M once t valid shares are collected. Opening requires a threshold of verified shares and is committee-internal: outsiders without $k_{\mathcal{BB}}$ cannot read the shares.

Definition 1 (Correctness)**.** *AWARE is correct if, for every security parameter λ, whenever the reporting user, the authentication server, and at least t committee members behave honestly, the following conditions hold with probability at least $1 - \mathrm{negl}(\lambda)$ over the randomness of `Setup`, `RegIss`, `RegObt`, `RepGen`, `RepDec`, and the hash functions:*

1) *every honestly generated fresh authentication token $\mathsf{Tok}_{r,\tau}$ is accepted by `WAuth`; and*
2) *after the public opening condition holds, once at least t honestly generated valid decryption-share entries are posted on $\mathcal{BB}$, `RepCom` outputs the original report M.*

C. Discussion

Issuance-side leakage. $\mathcal{AS}$ runs `RegIss` on a fixed schedule for all eligible users; every user receives m pre-tokens per epoch. This removes token issuance at submission time: the issuer-side record is generated independently of whether a user later submits a report. $\mathcal{AS}$ still learns when each user receives a pre-token bundle. If only a small anonymity set received usable tokens in an epoch, or if a report uses a very recent token, the candidate set shrinks. This leakage is outside TNIBS and must be controlled by deployment policy: coarse refresh epochs,

Convention. Inputs use fixed canonical type-separated encodings. Let $H_1 : \{0,1\}^* \to \mathbb{G}_1$ and $H_2 : \{0,1\}^* \to \mathbb{G}_2$. The symmetric encryption inside Boneh–Shoup threshold decryption is denoted by E_s, D_s, and H_K is the key-derivation hash. Parsed group elements must be non-identity group elements. After submission, $\mathcal{U}$ acts as $\mathcal{W}$. Replay follows $\mathcal{BB}$'s ledger order. Honest committee members run `RepDec` only after the public opening condition holds.

Phase I: Initialization

`Setup`$(1^\lambda, n, t)$**.**

S1. Run `TNIBS.KeyGen` for message length 2: sample $a_1, a_2 \leftarrow \mathbb{Z}_p^*$, set $\mathsf{ask} = (a_1, a_2)$, and set $\mathsf{apk} = (A_1, A_2) = (\widetilde{g}^{a_1}, \widetilde{g}^{a_2})$.

S2. Run TGS.`DKeyGen` using DDKG.`KeyGen`$(1^\lambda, n, t)$. Each $\mathcal{CM}_j$ samples $f_j(X) = \sum_{k<t} a_{j,k} X^k$, broadcasts $C_{j,k} = g^{a_{j,k}}$ and $\widetilde{C}_{j,k} = \widetilde{g}^{a_{j,k}}$, and privately sends $s_{j\to i} = f_j(i)$ to $\mathcal{CM}_i$. Each receiver checks:

$$g^{s_{j\to i}} = \prod_{k<t} C_{j,k}^{i^k},$$

$$\widetilde{g}^{s_{j\to i}} = \prod_{k<t} \widetilde{C}_{j,k}^{i^k},$$

$$e(C_{j,k}, \widetilde{g}) = e(g, \widetilde{C}_{j,k}).$$

Abort if any check fails. Otherwise $\mathcal{CM}_i$ computes:

$$\alpha_i = \sum_{j\in[n]} s_{j\to i}, \qquad \Theta = \prod_{j\in[n]} C_{j,0},$$

$$\Theta_i = \prod_{\substack{j\in[n]\\k<t}} C_{j,k}^{i^k}, \qquad \widetilde{\Theta}_i = \prod_{\substack{j\in[n]\\k<t}} \widetilde{C}_{j,k}^{i^k}.$$

Set $pk_{\mathsf{th}} = \Theta$, $pk_c = ((\Theta_i, \widetilde{\Theta}_i))_{i\in[n]}$, and $sk_i^{\mathsf{th}} = (i, \alpha_i, \Theta_i, \widetilde{\Theta}_i)$. Give sk_i^{th} only to $\mathcal{CM}_i$, and publish pk_{th} and pk_c.

S3. Establish $k_{\mathcal{BB}}$ for committee-only share encryption, and initialize blame_i lists and $\mathcal{BB}$'s record.

Phase II: Registration and Token Issuance

`RegU`.

R1. On honest registration, run `TNIBS.RKeyGen`: sample $x \leftarrow \mathbb{Z}_p^*$, set $\mathsf{usk} = x$, $\mathsf{upk} = g^x$, and register upk after institutional eligibility verification.

`RegIss`.

I1. For each (upk, τ) in epoch r, run `TNIBS.Issue`: sample $\mathsf{nonce}_{r,\tau}$, $y_{r,\tau} \leftarrow \mathbb{Z}_p^*$, and compute:

$$h_{r,\tau} = H_1(\mathsf{nonce}_{r,\tau}),$$

$$Z_{r,\tau} = \left(\mathsf{upk}^{a_1} h_{r,\tau}^{a_2}\right)^{y_{r,\tau}},$$

$$Y_{r,\tau} = g^{y_{r,\tau}^{-1}}, \quad \widetilde{Y}_{r,\tau} = \widetilde{g}^{y_{r,\tau}^{-1}},$$

$$T_{r,\tau} = H_2(\eta_r)^{y_{r,\tau}^{-1}},$$

$$\mathsf{psig}_{r,\tau} = (Z_{r,\tau}, Y_{r,\tau}, \widetilde{Y}_{r,\tau}, T_{r,\tau}).$$

I2. Deliver m padded pre-tokens to each eligible user, independent of reporting: $\mathsf{ptok}_{r,\tau} = (\eta_r, \mathsf{nonce}_{r,\tau}, h_{r,\tau}, \mathsf{psig}_{r,\tau})$ for $\tau \in [m]$.

`RegObt`.

O1. For each $\mathsf{ptok}_{r,\tau}$, check $h_{r,\tau} \stackrel{?}{=} H_1(\mathsf{nonce}_{r,\tau})$; reject otherwise.

O2. Run `TNIBS.Obtain`. Parse $\mathsf{psig}_{r,\tau} = (Z_{r,\tau}, Y_{r,\tau}, \widetilde{Y}_{r,\tau}, T_{r,\tau})$ and verify:

$$e(Z_{r,\tau}, \widetilde{Y}_{r,\tau}) \stackrel{?}{=} e(\mathsf{upk}, A_1) e(h_{r,\tau}, A_2),$$

$$e(Y_{r,\tau}, \widetilde{g}) \stackrel{?}{=} e(g, \widetilde{Y}_{r,\tau}),$$

$$e(g, T_{r,\tau}) \stackrel{?}{=} e(Y_{r,\tau}, H_2(\eta_r)).$$

Reject if any check fails.

O3. Set $\mu = \mathsf{usk}^{-1} = x^{-1}$, sample $\omega_{r,\tau} \leftarrow \mathbb{Z}_p^*$, and finalize:

$$\rho_{r,\tau} = h_{r,\tau}^{\mu}, \qquad Z'_{r,\tau} = Z_{r,\tau}^{\mu\omega_{r,\tau}},$$

$$Y'_{r,\tau} = Y_{r,\tau}^{\omega_{r,\tau}^{-1}}, \quad \widetilde{Y}'_{r,\tau} = \widetilde{Y}_{r,\tau}^{\omega_{r,\tau}^{-1}},$$

$$T'_{r,\tau} = T_{r,\tau}^{\omega_{r,\tau}^{-1}}, \quad \sigma_{r,\tau} = (Z'_{r,\tau}, Y'_{r,\tau}, \widetilde{Y}'_{r,\tau}, T'_{r,\tau}).$$

O4. Store $\mathsf{Tok}_{r,\tau} = (\eta_r, \rho_{r,\tau}, \sigma_{r,\tau})$, where $\rho_{r,\tau}$ is the signed random token message and η_r is the public TNIBS tag.

Phase III: Reporting and Authentication

`RepGen`$(M, \mathsf{Tok}_{r,\tau})$**.**

G1. Choose an unused accepted $\mathsf{Tok}_{r,\tau}$; set $\mathsf{ad} = \mathsf{Tok}_{r,\tau}$.

G2. Run TGS.`Enc`$(pk_{\mathsf{th}}, M, \mathsf{ad})$: sample $\beta \leftarrow \mathbb{Z}_p^*$ and compute:

$$v = g^\beta, \qquad \widetilde{v} = \widetilde{g}^\beta,$$

$$w = \Theta^\beta, \qquad k = H_K(v, w),$$

$$c_M \leftarrow E_s(k, M), \qquad \bar{u} = H_1(v, \mathsf{ad}),$$

$$\bar{w} = \bar{u}^\beta, \qquad \mathsf{ct} = (v, \widetilde{v}, c_M, \bar{w}).$$

G3. Append $(\mathsf{Tok}_{r,\tau}, \mathsf{ct})$ to $\mathcal{BB}$.

`WAuth`$(\mathsf{Tok}_{r,\tau})$**.**

A1. Parse $\mathsf{Tok}_{r,\tau} = (\eta_r, \rho_{r,\tau}, \sigma_{r,\tau})$; reject if η_r is outside the public acceptance window.

A2. Run `TNIBS.Verify`. Parse $\sigma_{r,\tau} = (Z'_{r,\tau}, Y'_{r,\tau}, \widetilde{Y}'_{r,\tau}, T'_{r,\tau})$; accept iff:

$$e(Z'_{r,\tau}, \widetilde{Y}'_{r,\tau}) \stackrel{?}{=} e(g, A_1) e(\rho_{r,\tau}, A_2),$$

$$e(Y'_{r,\tau}, \widetilde{g}) \stackrel{?}{=} e(g, \widetilde{Y}'_{r,\tau}),$$

$$e(g, T'_{r,\tau}) \stackrel{?}{=} e(Y'_{r,\tau}, H_2(\eta_r)).$$

If the check succeeds, accept only the first valid posting with replay tag $\rho_{r,\tau}$ in ledger order.

Phase IV: Threshold Opening

`RepDec`$(sk_i^{\mathsf{th}}, \mathsf{ct}, \mathsf{Tok}_{r,\tau})$**.**

D1. After public opening, set $\mathsf{ad} = \mathsf{Tok}_{r,\tau}$ and run TGS.`CtVrfy`$(pk_{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$. Parse $\mathsf{ct} = (v, \widetilde{v}, c_M, \bar{w})$, set $\bar{u} = H_1(v, \mathsf{ad})$, and require $e(v, \widetilde{g}) = e(g, \widetilde{v})$ and $e(\bar{w}, \widetilde{g}) = e(\bar{u}, \widetilde{v})$.

D2. Run TGS.`ShareDec`$(sk_i^{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$. For $sk_i^{\mathsf{th}} = (i, \alpha_i, \Theta_i, \widetilde{\Theta}_i)$, compute $w_i = v^{\alpha_i}$ and $d_i = (i, w_i)$.

D3. Append $\widehat{d}_i \leftarrow \mathsf{Enc}_{k_{\mathcal{BB}}}(d_i, (\mathsf{ct}, \mathsf{Tok}_{r,\tau}))$ over an authenticated channel. $\mathcal{BB}$ records $(i, \widehat{d}_i)$ with $\mathcal{CM}_i$'s identity.

`RepCom`$(pk_c, k_{\mathcal{BB}}, \mathsf{ct}, \mathsf{Tok}_{r,\tau}, \{\widehat{d}_j\})$**.**

C1. Set $\mathsf{ad} = \mathsf{Tok}_{r,\tau}$, run TGS.`CtVrfy`$(pk_{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$, parse $\mathsf{ct} = (v, \widetilde{v}, c_M, \bar{w})$, compute $\bar{u} = H_1(v, \mathsf{ad})$, and require $e(v, \widetilde{g}) = e(g, \widetilde{v})$ and $e(\bar{w}, \widetilde{g}) = e(\bar{u}, \widetilde{v})$. Initialize $\mathcal{J} \leftarrow \emptyset$.

C2. For each attributed entry $(j, \widehat{d}_j)$, decrypt $d' \leftarrow \mathsf{Dec}_{k_{\mathcal{BB}}}(\widehat{d}_j, (\mathsf{ct}, \mathsf{Tok}_{r,\tau}))$. If decryption fails or d' does not parse as (j, w_j), ignore the entry. Otherwise set $d_j = (j, w_j)$ and run TGS.`ShareVrfy`$(pk_c, j, \mathsf{ct}, \mathsf{ad}, d_j)$: accept iff $e(w_j, \widetilde{g}) = e(v, \widetilde{\Theta}_j)$. If it holds, add j to $\mathcal{J}$; otherwise, add $\mathcal{CM}_j$ to blame_i.

C3. Run TGS.`Combine`$(pk_c, \mathsf{ct}, \mathsf{ad}, \{d_j\}_{j\in\mathcal{J}})$. If $|\mathcal{J}| < t$, fail; otherwise compute:

$$\lambda_j = \prod_{\ell\in\mathcal{J}_t\setminus\{j\}} \frac{\ell}{\ell - j} \in \mathbb{Z}_p,$$

$$w = \prod_{j\in\mathcal{J}_t} w_j^{\lambda_j},$$

for any t-subset $\mathcal{J}_t \subseteq \mathcal{J}$. Then derive $k = H_K(v, w)$ and compute $M \leftarrow D_s(k, c_M)$. If $M = \perp$, reject; otherwise, output M and publish blame_i.

Fig. 2. Main AWARE protocol. Registration and scheduled pre-token delivery involve $\mathcal{U}$ and $\mathcal{AS}$. During report submission, $\mathcal{U}$ acts anonymously as $\mathcal{W}$. Values not posted to $\mathcal{BB}$ remain local to the entity executing the corresponding algorithm.

large eligibility classes, conservative token expiration, uniform response formatting, and padding. A user who exhausts its token pool should wait for the next refresh rather than request tokens on the reporting path.

Multiple reports. Each report consumes one unused token. Under Definition 3, distinct tokens from the same user's

pool are unlinkable at the token layer: posted tokens do not reveal whether two accepted reports came from the same registered user, when report content and posting metadata are fixed. This does not hide correlation through writing style, attachments, content similarity, or posting schedule. Those sources of correlation should be addressed by operational guidance, batching, or dummy reporting traffic.

Cross-epoch token use. A user may maintain a bounded pool of unused tokens across epochs. Each token carries a public TNIBS tag η_r that identifies its validity class, not the report time. Committee members accept a token only if η_r is within the public acceptance window. This supports cross-epoch use without contacting $\mathcal{AS}$ at report time: the epoch label is visible to $\mathcal{AS}$ at issuance, but no fresh authentication record is created when the report is posted. Expiration policy is expressed through tag-acceptance rules, which should be coarse enough that the accepted tag set does not collapse the anonymity set.

Bulletin-board model. $\mathcal{BB}$ is modeled as an append-only public record after entries are accepted. The model does not cover mempool-like pending states, relay ordering, or adversarial front-running that copies a pending submission onto $\mathcal{BB}$ first. Deployments can reduce this risk by accepting reports through an anonymous confidential transport, such as an onion-service endpoint, before $\mathcal{BB}$ publication, preventing network observers from copying a pending bearer token. $\mathcal{BB}$ can be implemented by a blockchain [43] or a centralized append-only service; the security analysis uses only append-only order, public readability, and authenticated attribution for committee-side entries.

Committee-internal opening. Decryption shares are encrypted under $k_{\mathcal{BB}}$ and posted to $\mathcal{BB}$ with authenticated committee metadata. This creates an append-only opening record, but reconstruction is committee-internal: only members holding $k_{\mathcal{BB}}$ can recover and verify the shares. Opening is therefore not publicly verifiable; outsiders see only encrypted share messages. Share-payload confidentiality is only against outsiders that do not collude with committee members. Inside the committee, invalid shares are detected and attributed through blame lists.

VI. Security Analysis

We prove four goals under the threat model of Section IV-B: whistleblower anonymity, token-based authentication, report confidentiality before threshold opening, and committee-internal opening integrity. The reductions use the concrete primitives in AWARE: TNIBS blindness and one-more unforgeability, Boneh–Shoup AD-only CCA threshold decryption, authenticated share encryption, and the append-only bulletin board. Authentication is counting-based: accepted postings with distinct replay tags cannot outnumber issued pre-tokens. It does not imply report truthfulness or protection for stolen bearer tokens. We first record the DKG bridge lemma. Detailed proofs are in Appendix A; the sketches below give the reduction route.

Lemma 1 (Non-aborting DKG realization of $\mathsf{TGS}.\mathtt{DKeyGen}$)**.** *Assume authenticated private channels, authenticated broadcast, and at most $t-1$ corrupted committee members. Conditioned on non-abort, the dealerless DKG of Section III-C realizes the key-generation distribution expected by $\mathsf{TGS}.\mathtt{DKeyGen}(1^\lambda, n, t)$: it outputs $(pk_{\mathsf{th}}, pk_c, (sk_i^{\mathsf{th}})_{i\in[n]})$ such that*

$$pk_{\mathsf{th}} = g^{f(0)}, \qquad sk_i^{\mathsf{th}} = (i, f(i), g^{f(i)}, \widetilde{g}^{f(i)})$$

for an implicit degree-$(t-1)$ polynomial f. Except for the intended public value $g^{f(0)}$, no PPT adversary controlling at most $t-1$ members learns the scalar $f(0)$, under discrete-logarithm hardness in the source groups. The lemma does not claim DKG liveness against disruptive dealers.

Proof sketch. For a non-aborting execution, view each accepted dealer as defining one degree-$(t-1)$ polynomial. The receiver checks force the delivered share $s_{j\to i}$ to match the committed value $f_j(i)$, and the pairing checks force the commitments in $\mathbb{G}_1$ and $\mathbb{G}_2$ to encode the same coefficients. Summing the accepted dealer polynomials gives $f = \sum_j f_j$, hence $pk_{\mathsf{th}} = g^{f(0)}$, $\Theta_i = g^{f(i)}$, and $\widetilde{\Theta}_i = \widetilde{g}^{f(i)}$. A corrupted set of size at most $t-1$ obtains fewer than t evaluations of the honest aggregate polynomial; the remaining public data are only discrete-log-hard encodings of its coefficients. Thus a distinguisher for the hidden scalar $f(0)$, beyond the intended public value $g^{f(0)}$, would give a discrete-log distinguisher in the source groups. □

A. Whistleblower anonymity

Users register to $\mathcal{AS}$ under real identities; AWARE does not hide registration-time identity from $\mathcal{AS}$. The anonymity claim is token-layer only: a posted token and ciphertext should not reveal the registered public key or issuance-side pre-token record that produced them, except through the leakage fixed below. Content, writing style, external facts, and posting schedules are not hidden by this game.

Views and leakage.

Let

$$\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}} := \begin{pmatrix} \mathsf{ord}_{\mathcal{AS}}, \mathsf{ask}, \mathsf{apk}, \mathsf{TagPol}, \{(\mathsf{id}_i, \mathsf{upk}_i)\}_{i\in\mathcal{U}}, \\ \{(\mathsf{upk}, \eta_r, \tau, \mathsf{nonce}_{r,\tau}, y_{r,\tau}, \mathsf{ptok}_{r,\tau}, \tau^{\mathsf{iss}})\}, \end{pmatrix}$$

denote the authentication server's private issuance-side view. It contains the TNIBS signing key, the tag policy, the registered identities and public keys, the concrete tags, nonces, issuer randomness values, pre-tokens, and issuance order labels generated by `RegIss`, but not the user's local `RegObt` step.

Let

$$\mathsf{View}_{\mathcal{BB}} := \Big(\{(\mathsf{Tok}_{r,\tau}, \mathsf{ct}, \tau^{\mathsf{post}})\}, \{(i, \widehat{d}_i, \tau_i^{\mathsf{open}})\}, \mathsf{ord}_{\mathcal{BB}}\Big)$$

denote the public bulletin-board view: report postings, encrypted opening-share entries with authenticated committee identities, and ledger order.

The overall deanonymization view is $\mathsf{View}^{\mathsf{anon}} := (\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}, \mathsf{View}_{\mathcal{BB}})$.

Deanonymization means linking an issuance-side entry in $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$ to a posted token in $\mathsf{View}_{\mathcal{BB}}$. Since `RegObt` is local, it is visible only through the final posted token. Honest users also recompute $H_1(\mathsf{nonce}_{r,\tau})$ and reject mismatched $h_{r,\tau}$, so $\mathcal{AS}$ cannot use malformed $h_{r,\tau}$ values as fingerprints.

We write $\mathsf{Leak}^{\mathsf{anon}}$ for the non-token metadata revealed by the scheduled-refresh model: registration-set size, eligibility classes, refresh epochs, public tags, issuance timing/order labels, tag policy, posting times, ledger order, report lengths, and report-content leakage. For non-challenge reports it also contains the public bulletin-board record. For challenge reports it contains only this shape and metadata, not $\mathsf{Tok}^\star$, $\mathsf{ct}^\star$, or the full challenge record. The private issuance-side state is modeled separately by $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$. The game only claims anonymity for histories with identical $\mathsf{Leak}^{\mathsf{anon}}$.

Scheduled-refresh assumption. Eligible users in the same class receive the same number m of padded pre-tokens per epoch, and delivery is independent of later reporting. Delivery delays, small classes, fresh-token use, and network paths are either in $\mathsf{Leak}^{\mathsf{anon}}$ or handled by batching, padding, and anonymous transport.

Definition 2 (Single-report anonymity)**.** *AWARE satisfies* single-report whistleblower anonymity *if, for every PPT adversary $\mathcal{A}$, the following advantage is negligible:*

$$\mathsf{Adv}^{\mathsf{Anon}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Anon}_{\mathcal{A}}(\lambda) = 1\right] - \tfrac{1}{2}\right|.$$

In $\mathsf{Anon}_{\mathcal{A}}(\lambda)$:

1) *The challenger runs $\mathtt{Setup}$ and creates two honest challenge users $\mathcal{U}_0 = (\mathsf{usk}_0, \mathsf{upk}_0)$ and $\mathcal{U}_1 = (\mathsf{usk}_1, \mathsf{upk}_1)$.*
2) *The adversary receives the public parameters, $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$ for non-challenge executions (including* ask, TagPol*, tags, and auxiliary issuance records independent of the challenge bit), and the non-challenge public bulletin-board record.*
3) *The adversary outputs two admissible histories $\mathcal{H}_0, \mathcal{H}_1$ for the challenge users. The challenger aborts unless their $\mathsf{Leak}^{\mathsf{anon}}$ values are equal; the histories may differ only in which challenge user supplies the finalized token for the report.*
4) *The challenger samples $b \xleftarrow{\$} \{0,1\}$. It executes the common refresh schedule for both challenge users, includes the resulting issuance-side records in $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$, finalizes the selected challenge token for $\mathcal{U}_b$, and runs $\mathtt{RepGen}(M^\star, \mathsf{Tok}^\star_{b,r,\tau^\star})$. It returns the issuance-side view, the non-challenge bulletin-board record, the challenge posting $(\mathsf{Tok}^\star_{b,r,\tau^\star}, \mathsf{ct}^\star, \tau^{\mathsf{post},\star})$, and $\mathsf{Leak}^{\mathsf{anon}}$.*
5) *$\mathcal{A}$ outputs $\hat{b}$; the experiment returns 1 iff $\hat{b} = b$.*

Definition 3 (Multi-report token unlinkability)**.** *AWARE satisfies* multi-report token unlinkability *if, for every polynomial $q = q(\lambda)$ and every PPT adversary $\mathcal{A}$, the advantage*

$$\mathsf{Adv}^{\mathsf{MAnon}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{MAnon}_{\mathcal{A}}(\lambda) = 1\right] - \tfrac{1}{2}\right|$$

is negligible.

The experiment $\mathsf{MAnon}_{\mathcal{A}}(\lambda)$ is the same as Definition 2, except that the histories contain q challenge report slots. They must have identical $\mathsf{Leak}^{\mathsf{anon}}$: the same ordered payloads $(M^\star_1, \ldots, M^\star_q)$, report-content leakage, posting schedule, and accepted tag sequence, but not the same concrete tokens or ciphertexts. They may differ only in the assignment from report slots to issuance-side pre-token records. Same-user and different-user assignments are both allowed, but every spent token must be unused and valid, and no issuance record may be assigned twice in the same history. The challenger samples $b \xleftarrow{\$} \{0,1\}$, realizes the assignments prescribed by $\mathcal{H}_b$, posts the resulting q challenge records $\{(\mathsf{Tok}^\star_j, \mathsf{ct}^\star_j, \tau^{\mathsf{post}}_j)\}_{j=1}^q$, and gives the resulting view to $\mathcal{A}$. The adversary wins if it outputs b.

Definition 4 (Multi-session issuer-side TNIBS blindness)**.** *A TNIBS scheme satisfies* multi-session issuer-side blindness *if this polynomial-session extension of Hanzlik's tagged recipient-blindness and nonce-blindness games gives every PPT adversary only negligible advantage. The game allows polynomially many challenge sessions, auxiliary finalized tokens, public tags, and repeated recipient public keys.*

The challenger runs $(\mathsf{ask}, \mathsf{apk}) \leftarrow \mathtt{TNIBS.KeyGen}(1^\lambda)$ and gives $(\mathsf{ask}, \mathsf{apk})$ to $\mathcal{B}$. It maintains recipient keys and an issuance ledger. The adversary may request recipient public keys, issuance records, and non-challenge finalizations. For an issuance request $(i, \eta, \mathsf{nonce})$, the challenger runs $\mathsf{psig} \leftarrow \mathtt{TNIBS.Issue}(\mathsf{ask}, \mathsf{upk}_i, \eta, H_1(\mathsf{nonce}))$ using fresh issuer randomness y, sets $\mathsf{ptok} = (\eta, \mathsf{nonce}, H_1(\mathsf{nonce}), \mathsf{psig})$, records $I = (i, \mathsf{upk}_i, \eta, \mathsf{nonce}, y, \mathsf{ptok})$, and returns I to $\mathcal{B}$. A non-challenge finalization returns the token obtained with the hidden recipient secret key.

At challenge time, $\mathcal{B}$ outputs two equal-length vectors $\mathbf{I}_0 = (I_{0,1}, \ldots, I_{0,q})$ and $\mathbf{I}_1 = (I_{1,1}, \ldots, I_{1,q})$ of previously issued but unfinalized records. They must have the same public tag sequence and metadata, but may use different recipient keys or nonce/pre-token transcripts, and may repeat a recipient key. No record may appear twice in the same vector, and no record in $\mathbf{I}_0 \cup \mathbf{I}_1$ may be finalized before or after the challenge. The challenger samples $b \xleftarrow{\$} \{0,1\}$, finalizes the records in $\mathbf{I}_b$, and returns the vector of finalized tokens $(\mathsf{Tok}^\star_1, \ldots, \mathsf{Tok}^\star_q)$ together with all allowed auxiliary non-challenge tokens and public metadata. The adversary outputs $\hat{b}$, and the experiment returns 1 iff $\hat{b} = b$. We write

$$\mathsf{Adv}^{\mathsf{MSIBnd}}_{\mathcal{B},\mathsf{TNIBS}}(\lambda) := \left|\Pr\left[\mathsf{MSIBnd}_{\mathcal{B},\mathsf{TNIBS}}(\lambda) = 1\right] - \tfrac{1}{2}\right|.$$

Lemma 2 (Hybrid amplification of TNIBS blindness)**.** *Assume that the tagged recipient-blindness and nonce-blindness games of the TNIBS scheme are* auxiliary-token closed*: in the single-challenge game, the adversary may additionally receive any polynomial-size set of honestly generated non-challenge finalized tokens, issuer-side records, public tags, and metadata whose distribution is independent of the challenge bit. These auxiliary tokens may include tokens generated under recipient public keys that also occur in the challenge. Then, for $q = q(\lambda)$ challenge records,*

$$\mathsf{Adv}^{\mathsf{MSIBnd}}_{\mathcal{B}}(\lambda) \le 2q\Big(\mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{B}'}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{B}''}(\lambda)\Big) + \mathrm{negl}(\lambda),$$

for suitable PPT adversaries $\mathcal{B}'$ and $\mathcal{B}''$ with comparable running time. Hence auxiliary-token-closed tagged recipient blindness and nonce blindness imply multi-session issuer-side TNIBS blindness.

Proof sketch. Let G_j be the hybrid in which the first j challenge slots are finalized according to $\mathbf{I}_1$ and the remaining slots according to $\mathbf{I}_0$. To distinguish G_{j-1} from G_j, the

reduction gives all other challenge slots, all non-challenge tokens, and all metadata as auxiliary input to the corresponding single-session blindness game. If the changed slot differs only in the recipient key, the reduction uses tagged recipient blindness; if it differs only in the nonce or pre-token transcript, it uses tagged nonce blindness. If both parts differ, insert one intermediate slot distribution and apply the two games in sequence. The reduction outputs the multi-session adversary's bit as its own guess, and the triangle inequality over the q slots gives the stated loss. □

Lemma 3 (Auxiliary-token closure condition for the Hanzlik TNIBS construction)**.** *For the Hanzlik tagged NIBS instantiation used by AWARE, assume the single-session tagged recipient-blindness and tagged nonce-blindness games of [42] remain secure when the issuer-side adversary also receives polynomially many honestly generated non-challenge tokens, issuance records, public tags, and metadata independent of the challenge bit. Then the instantiation satisfies Definition 4; for q challenge records and q_{aux} auxiliary finalized tokens, the advantage is bounded by*

$$\mathsf{Adv}^{\mathsf{MSIBnd}}_{\mathcal{B}}(\lambda) \le 2(q+q_{\mathsf{aux}})\mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{D}'}(\lambda) + 2(q+q_{\mathsf{aux}})\mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{D}''}(\lambda) + \mathrm{negl}(\lambda).$$

Proof sketch. Handle auxiliary finalized tokens before embedding the main blindness challenge. For one auxiliary token, the reduction replaces its concrete recipient key, nonce, and pre-token transcript by an honestly generated dummy token with the same public tag and a valid final signature. The recipient-key replacement is reduced to tagged recipient blindness, and the nonce/transcript replacement is reduced to tagged nonce blindness. The auxiliary distribution is independent of the challenge bit, so these replacements can be made one token at a time without changing the hidden bit in the main experiment. A union bound over q_{aux} tokens, followed by Lemma 2, gives the stated bound. □

Theorem 4 (Single-report anonymity)**.** *Assume the Hanzlik TNIBS instantiation used by* `RegIss/RegObt` *satisfies Lemma 3 under the accepted public tags, and the scheduled-refresh assumption above holds. Then AWARE satisfies single-report whistleblower anonymity as defined in Definition 2.*

Proof sketch. The reduction receives the TNIBS blindness challenge interface for the two candidate issuance histories and uses it only for the challenge token-finalization step. It generates the remaining AWARE setup itself, answers all non-challenge issuance and finalization queries honestly, and keeps the bulletin-board record consistent with the leakage fixed by the anonymity experiment. For the challenge report, it uses the finalized token returned by the TNIBS challenger as $\mathsf{Tok}^\star$, runs report encryption with fresh randomness and associated data $\mathsf{Tok}^\star$, and returns the resulting posting. This posting is public post-processing of the hidden TNIBS token and values independent of the challenge bit. Therefore the AWARE adversary's output bit is a valid guess in the TNIBS blindness game, with only the auxiliary-token and scheduling losses already stated. □

Theorem 5 (Multi-report token unlinkability)**.** *Assume that the Hanzlik TNIBS instantiation used by* `RegIss/RegObt` *satisfies the auxiliary-token-closed tagged recipient-blindness and tagged nonce-blindness condition in Lemma 3 under the accepted public tags, and that the scheduled-refresh assumption above holds. Then AWARE satisfies multi-report token unlinkability as defined in Definition 3.*

Proof sketch. The reduction maps the two admissible AWARE histories to two TNIBS challenge vectors $\mathbf{I}_0$ and $\mathbf{I}_1$, one record for each challenge report slot. It asks the TNIBS challenger to finalize the hidden vector, then encrypts the fixed report payloads under the returned tokens, using each token as the associated data for its ciphertext. All non-challenge issuance records, bulletin-board entries, payloads, and timing leakage are simulated identically in the two worlds. Thus the only hidden choice in the AWARE view is which TNIBS vector was finalized. A distinguisher for the AWARE histories therefore gives a multi-session TNIBS blindness distinguisher, and Lemmas 2 and 3 give the final loss. □

B. Token-Based Whistleblower Authentication

We next analyze token-based authentication. The committee checks token validity and replay order, not the submitter's real identity. The property says nothing about report truthfulness and does not prevent theft, transfer, or front-running of a finalized bearer token. Against malicious users and external submitters, the claim is quantitative: q issued pre-tokens can lead to at most q accepted postings with distinct replay tags.

We first define when a posted report is counted as accepted in this game. Let $\mathsf{Tok} = (\eta, \rho, \sigma)$ be a posted authentication token, and let ct be the accompanying report ciphertext. Given a set $\mathcal{S}$ of already consumed replay tags, the predicate $\mathsf{PostAccept}(\mathsf{Tok}, \mathsf{ct}; \mathcal{S})$ returns 1 if and only if all of the following conditions hold:

1) η lies inside the public acceptance window;
2) $\rho \notin \mathcal{S}$;
3) $\mathsf{TNIBS.Verify}(\mathsf{apk}, \eta, (\rho, \sigma)) = 1$;
4) $\mathsf{TGS.}\texttt{CtVrfy}(\mathsf{pk}_{\mathsf{th}}, \mathsf{ct}, \mathsf{Tok}) = 1$.

If the predicate returns 1, then ρ is added to $\mathcal{S}$. Therefore, only the first valid posting with a given replay tag can be accepted. Malformed ciphertexts do not consume replay tags.

Definition 5 (Authentication-token unforgeability)**.** *AWARE satisfies authentication-token unforgeability if, for every PPT adversary $\mathcal{A}$, the following advantage is negligible:*

$$\mathsf{Adv}^{\mathsf{TokUnf}}_{\mathcal{A}}(\lambda) := \Pr[\mathsf{TokUnf}_{\mathcal{A}}(\lambda) = 1].$$

The experiment $\mathsf{TokUnf}_{\mathcal{A}}(\lambda)$ is defined as follows.

1) The challenger runs

$$(\mathsf{pp}; \mathsf{ask}, \mathsf{apk}; \mathsf{pk}_{\mathsf{th}}, \mathsf{pk}_{\mathsf{c}}, \{\mathsf{sk}^{\mathsf{th}}_i\}_{i\in[n]}) \leftarrow \mathsf{Setup}(1^\lambda, n, t),$$

and gives all public parameters to $\mathcal{A}$.

2) The challenger maintains admitted eligible public keys and an issuance counter q_{iss}, initially 0. A key is admitted only through the registration interface; for adversarial-key admission, the one-more assumption in Theorem 6 is

interpreted for that interface. Invalid keys or tags return $\perp$ and are not counted.

3) *The adversary has oracle access to $\mathcal{O}_{\mathsf{Issue}}(\mathsf{upk}, \eta)$. On a valid query, the challenger samples* nonce*, sets $h = H_1(\mathsf{nonce})$, runs TNIBS issuance on (upk, η, h), and returns one pre-token $\mathsf{ptok} = (\eta, \mathsf{nonce}, h, \mathsf{psig})$ distributed as in* RegIss*. It increments $q_{\mathsf{iss}} \leftarrow q_{\mathsf{iss}} + 1$. A returned pre-token is counted as one issued authentication token, even if the adversary later finalizes it locally by running* RegObt*.*
4) *Eventually, $\mathcal{A}$ outputs an ordered list of purported bulletin-board postings $\mathcal{P} = \{(\widetilde{\mathsf{Tok}}_j, \widetilde{\mathsf{ct}}_j)\}_{j=1}^{\ell}$.*
5) *The challenger processes $\mathcal{P}$ in ledger order. It initializes $\mathcal{S} \leftarrow \emptyset$ and $N_{\mathsf{acc}} \leftarrow 0$. For each posting $(\widetilde{\mathsf{Tok}}_j, \widetilde{\mathsf{ct}}_j)$, if $\mathsf{PostAccept}(\widetilde{\mathsf{Tok}}_j, \widetilde{\mathsf{ct}}_j; \mathcal{S}) = 1$, then the challenger adds the corresponding replay tag to $\mathcal{S}$ and sets $N_{\mathsf{acc}} \leftarrow N_{\mathsf{acc}} + 1$. Otherwise, the posting is ignored.*
6) *The experiment outputs 1 if and only if $N_{\mathsf{acc}} > q_{\mathsf{iss}}$.*

Definition 6 (TNIBS one-more unforgeability)**.** *A TNIBS scheme satisfies one-more unforgeability if the following experiment gives every PPT adversary only negligible success probability. The challenger runs $(\mathsf{ask}, \mathsf{apk}) \leftarrow \mathsf{TNIBS.KeyGen}(1^\lambda)$, gives* apk *to $\mathcal{B}$, and answers adaptive presignature-issuance queries $\mathcal{O}_{\mathsf{TNIBS.Issue}}(\mathsf{upk}, \eta, \mu)$ by returning $\mathsf{psig} \leftarrow \mathsf{TNIBS.Issue}(\mathsf{ask}, \mathsf{upk}, \eta, \mu)$. Let q_η be the number of answered issuance queries under tag η. The adversary wins if it outputs a public tag $\eta^\star$ and more than $q_{\eta^\star}$ triples $\{(\eta^\star, \rho_j, \sigma_j)\}_{j=1}^{q_{\eta^\star}+1}$ such that $\mathsf{TNIBS.Verify}(\mathsf{apk}, \eta^\star, (\rho_j, \sigma_j)) = 1$ for every j, and the messages $\{\rho_j\}_{j=1}^{q_{\eta^\star}+1}$ are pairwise distinct. Multiple randomized signatures on the same pair $(\eta^\star, \rho)$ are counted only once. We denote the corresponding advantage by $\mathsf{Adv}^{\mathsf{OM\text{-}Unf}}_{\mathcal{B},\mathsf{TNIBS}}(\lambda)$.*

Theorem 6 (Authentication-token unforgeability)**.** *If the underlying TNIBS scheme satisfies one-more unforgeability for the recipient-key admission interface used by registration, then AWARE satisfies authentication-token unforgeability.*

Proof sketch. The reduction runs the AWARE experiment with the public key supplied by the TNIBS one-more unforgeability challenger. For every issuance query under a tag η, it samples the nonce and recipient message as in AWARE, obtains the corresponding TNIBS presignature from its oracle, and returns the pre-token to the adversary while recording one issued slot for that tag. At the end, it applies the same replay filter and WAuth predicate as AWARE to the adversary's bulletin-board record. Each accepted posting contains a valid TNIBS signature (η, ρ, σ) on a distinct randomized message ρ, because repeated replay tags are counted once. If accepted postings outnumber issued slots, then for some tag the reduction has more valid signatures than oracle queries and outputs them as a one-more forgery. □

C. Report confidentiality

We analyze confidentiality before opening. The adversary may corrupt fewer than t committee members and query shares for other associated-data values, but it receives no honest share for the challenge token before the public opening condition holds. The report-encryption component is the Boneh–Shoup threshold decryption construction with $\mathsf{ad} = \mathsf{Tok}$; we use its threshold AD-only CCA security as a black-box assumption.

Definition 7 (Report confidentiality)**.** *AWARE satisfies* report confidentiality before threshold opening *if, for every PPT adversary $\mathcal{A}$, the following advantage is negligible:*

$$\mathsf{Adv}^{\mathsf{Conf}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{Conf}_{\mathcal{A}}(\lambda) = 1\right] - \tfrac{1}{2}\right|.$$

In $\mathsf{Conf}_{\mathcal{A}}(\lambda)$, the adversary chooses $\mathcal{L} \subseteq \{1, \ldots, n\}$ with $|\mathcal{L}| \leq t-1$. The challenger runs setup, gives public parameters, and reveals only the corrupted committee states. If $\mathcal{L} \neq \emptyset$, this view also includes $k_{\mathcal{BB}}$. This key reveals only opening-share payloads that are actually posted.

The adversary may obtain honest tokens and may query decryption shares for arbitrary postings $(\mathsf{Tok}, \mathsf{ct})$, before and after the challenge, subject to the underlying AD-only CCA admissibility rules. For each query, $\mathsf{ad} = \mathsf{Tok}$. At challenge time, $\mathcal{A}$ submits equal-length messages (M_0, M_1) and a token $\mathsf{Tok}^\star$. The token is adversarially chosen associated data; for accepted AWARE reports, we require $\mathsf{WAuth}(\mathsf{Tok}^\star) = 1$. The challenge associated data must be fresh: no answered share query or simulated opening event may have used $\mathsf{Tok}^\star$. Otherwise the experiment aborts. The challenger samples $b \xleftarrow{\$} \{0, 1\}$, sets $\mathsf{ad}^\star = \mathsf{Tok}^\star$, computes the challenge report $(\mathsf{Tok}^\star, \mathsf{ct}^\star) \leftarrow \mathtt{RepGen}(M_b, \mathsf{Tok}^\star)$, and returns the challenge posting $(\mathsf{Tok}^\star, \mathsf{ct}^\star)$ to $\mathcal{A}$.

The experiment models the pre-opening interval. For $\mathsf{ad}^\star = \mathsf{Tok}^\star$, the challenger rejects every share query and simulated opening event with associated data $\mathsf{ad}^\star$, including modified ciphertexts under the same token. The adversary may compute corrupted members' shares, but it gets no honest share for $\mathsf{ad}^\star$ before opening. Finally, $\mathcal{A}$ outputs $\hat{b}$, and the experiment returns 1 iff $\hat{b} = b$.

Theorem 7 (Report confidentiality)**.** *Assume the DKG initialization satisfies Lemma 1, and assume the Boneh–Shoup threshold decryption component used inside $\mathtt{RepGen}$ satisfies threshold AD-only CCA security against adversaries corrupting fewer than t committee members, with $\mathsf{ad} = \mathsf{Tok}$. Then AWARE satisfies Definition 7. More precisely, for a suitable PPT reduction $\mathcal{R}$,*

$$\mathsf{Adv}^{\mathsf{Conf}}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{ADCCA}}_{\mathcal{R},\mathsf{TGS}}(\lambda) + \mathrm{negl}(\lambda).$$

Proof sketch. The reduction forwards the corrupted committee set to the Boneh–Shoup AD-only CCA challenger and uses the returned public key, corrupted shares, and verification material as the AWARE threshold state. It simulates TNIBS issuance, token verification, and the bulletin board honestly. For every admissible non-challenge opening query, it sets $\mathsf{ad} = \mathsf{Tok}$ and forwards the corresponding share request to the threshold challenger. For the challenge, it submits $(M_0, M_1, \mathsf{Tok}^\star)$ as the AD-only CCA challenge, receives the challenge ciphertext, and returns the AWARE posting $(\mathsf{Tok}^\star, \mathsf{ct}^\star)$. The AWARE experiment rejects all pre-opening share queries under $\mathsf{Tok}^\star$, exactly matching the challenge-associated-data restriction. Hence the adversary's final bit is the reduction's bit in the AD-only CCA game, with the DKG realization loss added separately. □

D. Opening integrity

Opening integrity is committee-internal. Because shares are encrypted under $k_{\mathcal{BB}}$, public observers cannot verify shares or recompute the opening. The goal is only that an honest `RepCom` execution accepts a plaintext consistent with the accepted ciphertext, associated data, and t verified shares. The definition does not assert public verifiability, liveness, or protection against t colluding committee members.

Definition 8 (Opening integrity)**.** *AWARE satisfies* opening integrity *if, for every PPT adversary corrupting at most $t-1$ committee members, the following event occurs with negligible probability: the adversary produces an accepted bulletin-board record and an honest committee member running `RepCom` on an accepted posting $(\mathsf{Tok}, \mathsf{ct})$, with $\mathsf{ad} = \mathsf{Tok}$, outputs $M \neq \perp$, although the share entries used in this accepting execution do not contain a t-element set J_t satisfying all of the following conditions:* $\mathsf{TGS}.\mathtt{CtVrfy}(pk_{\mathsf{th}}, \mathsf{ct}, \mathsf{ad}) = 1$*; each attributed entry $(j, \widehat{d}_j)$ for $j \in J_t$ decrypts under $k_{\mathcal{BB}}$ with associated data $(\mathsf{ct}, \mathsf{Tok})$ to a share payload $d_j = (j, w_j)$, so its decrypted inner index matches its authenticated bulletin-board identity;* $\mathsf{TGS}.\mathtt{ShareVrfy}(pk_c, j, \mathsf{ct}, \mathsf{ad}, d_j) = 1$ *for every $j \in J_t$; and* $M = \mathsf{TGS}.\mathtt{Combine}(pk_c, \mathsf{ct}, \mathsf{ad}, \{d_j\}_{j \in J_t})$.

Theorem 8 (Opening integrity)**.** *Assume the public ciphertext and share verification algorithms of the Boneh–Shoup threshold decryption component are sound, reconstruction has a unique output for any t verified shares under fixed $(\mathsf{ct}, \mathsf{ad})$, share encryption binds its associated data, and $\mathcal{BB}$ is append-only with authenticated committee entries. Then AWARE satisfies Definition 8.*

Proof sketch. This proof is a bad-event argument rather than a secrecy reduction. Fix an honest accepting `RepCom` execution. The append-only bulletin board fixes the first accepted posting, so the execution is bound to one pair $(\mathsf{Tok}, \mathsf{ct})$ and associated data $\mathsf{ad} = \mathsf{Tok}$. The algorithm decrypts committee-share payloads only with associated data $(\mathsf{ct}, \mathsf{Tok})$, checks that the decrypted inner index matches the authenticated committee identity, and then applies `ShareVrfy` before reconstruction. If `RepCom` outputs a plaintext without a valid threshold witness as required by Definition 8, then at least one bad event occurred: ciphertext-verification soundness failed, share-verification soundness failed, the share-encryption layer accepted rebinding or tampering, or the bulletin board misattributed a committee entry. The theorem bounds the violation by the union of these negligible events; it does not assert liveness or public verifiability. □

VII. CCA Strengthening for Report Encryption

The main AWARE construction uses the AD-only CCA Boneh–Shoup threshold-decryption component from Section III. This section gives an optional wrapper for deployments that require a threshold CCA interface for report encryption. The wrapper prevents a modified challenge ciphertext from becoming an admissible decryption-share query. It does not change token issuance, report acceptance, or committee setup, and it does not protect a finalized bearer token after theft, transfer, or front-running before acceptance on $\mathcal{BB}$. Since the wrapper is optional, the main security theorems in Section VI and the prototype evaluation do not rely on it.

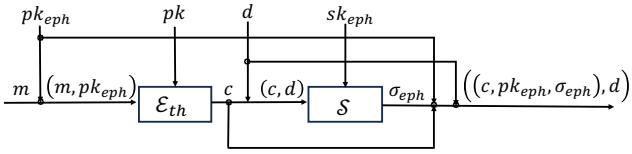


Fig. 3. Optional CCA wrapper for report encryption. A fresh one-time verification key binds the threshold ciphertext to the external AWARE associated data.

A. Optional Threshold CCA Wrapper

The wrapper follows the usual one-time-signature transformation from AD-only CCA security to CCA security. For each report, the encryptor samples a fresh one-time verification key pk_{eph}, uses pk_{eph} as the associated data inside the threshold encryption, and signs the threshold ciphertext together with the external AWARE associated data. A modified accepted query under the same pk_{eph} must therefore contain a new valid one-time signature, which is a strong one-time forgery. Fig. 3 shows the message format.

1) Syntax and Correctness: Let $\mathcal{S} = (\mathtt{G}, \mathtt{S}, \mathtt{V})$ be a strongly one-time secure signature scheme. Let $\mathcal{E} = (\mathtt{G}_{\mathsf{th}}, \mathtt{Enc}, \mathtt{Dec}, \mathtt{Combine})$ denote the underlying AD-only CCA threshold-opening component, where `Dec` is the share-decryption algorithm. The wrapped interface is $\mathcal{E}' = (\mathtt{G}', \mathtt{E}', \mathtt{D}', \mathtt{C}')$.

Definition 9 (Wrapped report encryption)**.** *The wrapped report-encryption interface consists of the following algorithms.*

- $(pk_{\mathsf{th}}, pk_c, sk_1^{\mathsf{th}}, \ldots, sk_n^{\mathsf{th}}) \leftarrow \mathtt{G}'(n, t)$*: output the result of the distributed* $\mathsf{TGS}.\mathtt{DKeyGen}$ *interface.*
- $(pk_{eph}, \mathsf{ct}, \sigma_{eph}) \leftarrow \mathtt{E}'(pk_{\mathsf{th}}, M, d)$*: sample* $(pk_{eph}, sk_{eph}) \leftarrow \mathtt{G}(\lambda)$*, compute*

  $$\mathsf{ct} \leftarrow \mathtt{Enc}(pk_{\mathsf{th}}, M, pk_{eph}), \qquad \sigma_{eph} \leftarrow \mathtt{S}(sk_{eph}, (\mathsf{ct}, d)),$$

  and output $(pk_{eph}, \mathsf{ct}, \sigma_{eph})$. Here d is the external AWARE associated data, e.g., $d = \mathsf{Tok}$; the underlying threshold ciphertext uses pk_{eph} as its associated data.
- *To compute a wrapped share, run*

  $$d_i \leftarrow \mathtt{D}'(sk_i^{\mathsf{th}}, (pk_{eph}, \mathsf{ct}, \sigma_{eph}), d).$$

  If $\mathtt{V}(pk_{eph}, (\mathsf{ct}, d), \sigma_{eph}) = 0$, output $\perp$. Otherwise output $d_i \leftarrow \mathtt{Dec}(sk_i^{\mathsf{th}}, \mathsf{ct}, pk_{eph})$.
- $(M/\perp, \mathsf{blame}) \leftarrow \mathtt{C}'(pk_c, \mathsf{ct}, pk_{eph}, \{d_j\}_{j \in \mathcal{J}})$*: output the result of* $\mathtt{Combine}(pk_c, \mathsf{ct}, pk_{eph}, \{d_j\}_{j \in \mathcal{J}})$.

Definition 10 (Correctness)**.** *For any $n, t \in \mathbb{N}$ with $t \leq n$, any $(pk_{\mathsf{th}}, pk_c, sk_1^{\mathsf{th}}, \ldots, sk_n^{\mathsf{th}})$ generated by $\mathtt{G}'(n, t)$, any $(pk_{eph}, \mathsf{ct}, \sigma_{eph})$ generated by $\mathtt{E}'(pk_{\mathsf{th}}, M, d)$, and any set $\{d_j\}_{j \in \mathcal{J}}$ of honestly generated non-$\perp$ decryption shares with $|\mathcal{J}| \geq t$, the combiner output $(M, \mathsf{blame}) \leftarrow \mathtt{C}'(pk_c, \mathsf{ct}, pk_{eph}, \{d_j\}_{j \in \mathcal{J}})$ recovers the encrypted message and does not blame an honest share generator.*

2) Security of the Optional Wrapper: The following experiment defines threshold CCA security for the wrapped interface. The adversary may corrupt fewer than t decryption servers and make adaptive decryption-share queries, but exact replays of challenge ciphertexts are rejected. Security requires that the challenge bit remains hidden for equal-length messages.

Definition 11 (Threshold CCA security)**.** *A threshold scheme $\mathcal{E}' = (G', E', D', C')$ satisfies* threshold CCA security *if, for every PPT adversary $\mathcal{A}$, the advantage*

$$\mathsf{Adv}^{\mathsf{thCCA}}_{\mathcal{A}}(\lambda) := \left|\Pr\left[\mathsf{thCCA}_{\mathcal{A},\mathcal{E}'}(\lambda) = 1\right] - \tfrac{1}{2}\right|$$

is negligible.

The experiment $\mathsf{thCCA}_{\mathcal{A},\mathcal{E}'}(\lambda)$ proceeds as follows.

1) *The adversary outputs parameters $(n, t, \mathcal{L})$, where $0 < t \le n$, $\mathcal{L} \subseteq \{1, \dots, n\}$, and $|\mathcal{L}| \le t - 1$.*
2) *The challenger runs*

$$(pk_{\mathsf{th}}, pk_c, sk_1^{\mathsf{th}}, \dots, sk_n^{\mathsf{th}}) \leftarrow G'(n, t),$$

samples $b \xleftarrow{\$} \{0, 1\}$, gives $(pk_{\mathsf{th}}, pk_c, \{sk_\ell^{\mathsf{th}}\}_{\ell \in \mathcal{L}})$ to $\mathcal{A}$, and initializes an empty challenge list List.
3) *On an encryption query (m_{i0}, m_{i1}, d_i) with $|m_{i0}| = |m_{i1}|$, the challenger computes*

$$(pk_{eph,i}, \mathsf{ct}_i, \sigma_{eph,i}) \leftarrow E'(pk_{\mathsf{th}}, m_{ib}, d_i),$$

returns this triple to $\mathcal{A}$, and records $(pk_{eph,i}, \mathsf{ct}_i, \sigma_{eph,i}, d_i)$ in List.
4) *On a decryption-share query*

$$((\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}), \widehat{d}),$$

the challenger rejects exact challenge-query replays, namely queries for which

$$(\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}, \widehat{d}) \in \mathsf{List}.$$

Otherwise, for each $i \in \{1, \dots, n\}$ it computes

$$\widehat{d}_i \leftarrow D'(sk_i^{\mathsf{th}}, (\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}), \widehat{d})$$

and returns $\{\widehat{d}_i\}_{i=1}^n$, where entries may be $\perp$.
5) *Finally, $\mathcal{A}$ outputs a bit $\widehat{b}$, and the experiment returns 1 iff $\widehat{b} = b$.*

Definition 12 (Multiple one-time signature security)**.** *Let $\mathcal{S} = (G, S, V)$ be a signature scheme. In the experiment $\mathsf{mstSIG}_{\mathcal{A},\mathcal{S}}(\lambda)$, the challenger maintains polynomially many one-time signing sessions. On a fresh-key query, it samples $(pk_i, sk_i) \leftarrow G(\lambda)$, stores (i, pk_i, sk_i), and returns pk_i. On the first signing query (i, m_i) for an existing unsigned session i, it returns $\sigma_i \leftarrow S(sk_i, m_i)$ and marks session i as signed; additional signing queries for the same session are rejected. The adversary wins if it outputs $(pk^\star, m^\star, \sigma^\star)$ such that $pk^\star = pk_i$ for some signed session i, $(m^\star, \sigma^\star) \neq (m_i, \sigma_i)$ for that session, and $V(pk^\star, m^\star, \sigma^\star) = 1$. The advantage $\mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{A}}(\lambda)$ is the probability that $\mathcal{A}$ wins this experiment.*

Lemma 9. *If $\mathcal{S}$ is strongly one-time secure, then $\mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{A}}(\lambda)$ is negligible for every PPT adversary $\mathcal{A}$.*

Proof sketch. The reduction guesses the fresh-key session used in the final forgery. It places the strong one-time verification key from its challenger in that session and generates all other one-time keys itself. The first signing query for the guessed session is forwarded to the one-time signing oracle; signing queries for other sessions are answered locally, and repeated signing queries are rejected as in the mstSIG game. If the adversary forges under the guessed signed session, the forged pair is different from the oracle-signed pair and is valid under the challenge verification key. The reduction outputs this pair as a strong one-time forgery, losing only the probability of guessing the right session. The full proof is given in Appendix A-I. □

Proposition 10 (Wrapper security)**.** *If the underlying threshold decryption scheme is AD-only CCA-secure and $\mathcal{S}$ is strongly one-time secure, then the wrapped interface $\mathcal{E}'$ satisfies threshold CCA security. More precisely, for every PPT adversary $\mathcal{A}$ making at most q_{enc} encryption queries, there are suitable PPT adversaries $\mathcal{R}$ and $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{thCCA}}_{\mathcal{A}}(\lambda) \le q_{\mathsf{enc}}\, \mathsf{Adv}^{\mathsf{ADCCA}}_{\mathcal{R}}(\lambda) + \mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{B}}(\lambda).$$

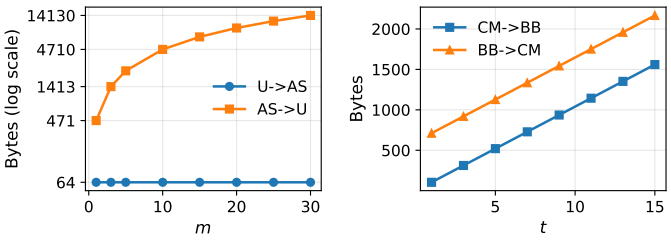
Proof sketch. Use a game sequence over the encryption queries. Let Forge be the event that the adversary submits a valid decryption-share query reusing an ephemeral verification key from an encryption query but changing the signed pair (ct, d, σ). This event gives an mstSIG forgery by the preceding lemma. Conditioned on $\neg$Forge, a query under an already used ephemeral key is either an exact replay, which the threshold CCA experiment rejects, or invalid. Every other accepted query uses a fresh ephemeral key and therefore a fresh associated-data value for the underlying AD-only CCA scheme. The reduction embeds one encryption query at a time into the AD-only CCA challenge, signs the returned ciphertext with the wrapper's one-time key, and forwards admissible share queries with the corresponding associated data. Summing over q_{enc} possible challenge positions and adding $\Pr[\mathsf{Forge}]$ gives the stated bound. The full proof is given in Appendix A-J. □

B. Integration with AWARE

Adopting the optional wrapper changes only the report-time algorithms. Setup is unchanged: $\mathcal{AS}$ runs TNIBS key generation, the committee runs $\mathsf{TGS}.\mathtt{DKeyGen}$ via DKG, and $\mathcal{BB}$ remains the same append-only public record. Strongly one-time signatures, such as Lamport signatures [44], are used only as a black-box primitive.

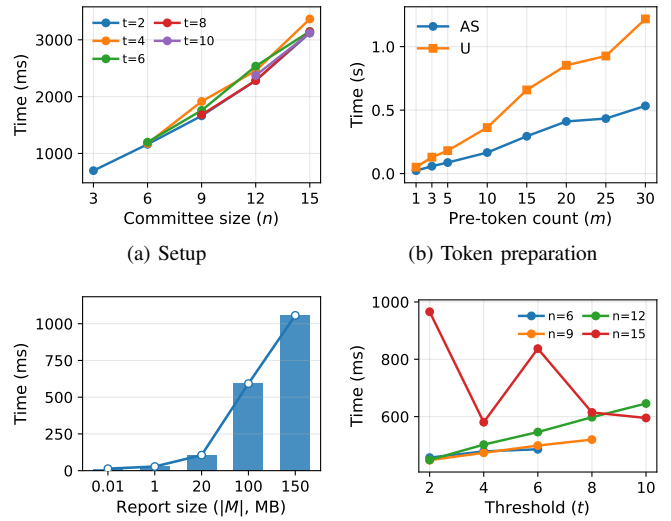
RepGen. The reporting user chooses an unused $\mathsf{Tok}_{r,\tau}$, sets $d_{\mathsf{ad}} = \mathsf{Tok}_{r,\tau}$, and computes $(pk_{eph}, \mathsf{ct}, \sigma_{eph}) \leftarrow E'(pk_{\mathsf{th}}, M, d_{\mathsf{ad}})$. The posted report becomes $(pk_{eph}, \mathsf{ct}, \sigma_{eph}, \mathsf{Tok}_{r,\tau})$; token validity and replay are still interpreted from the ledger order on $\mathcal{BB}$.

RepDec. For an accepted posting, committee member $\mathcal{CM}_i$ sets $d_{\mathsf{ad}} = \mathsf{Tok}_{r,\tau}$ and runs D' on its local share sk_i^{th}, the wrapped ciphertext, and d_{ad}. If the output is $\perp$, it rejects the posting. Otherwise, it encrypts the share payload under $k_{\mathcal{BB}}$ with associated data $(pk_{eph}, \mathsf{ct}, \sigma_{eph}, \mathsf{Tok}_{r,\tau})$ and appends the encrypted share to $\mathcal{BB}$ under its authenticated committee identity.



(a) Scheduled pre-token delivery (b) Threshold-opening traffic

Fig. 4. Variable communication overhead in AWARE. Scheduled delivery grows with the number of pre-tokens m, while threshold opening grows with the number of released shares t.



(a) Setup (b) Token preparation

(c) Report encryption (d) Threshold opening

Fig. 5. Computation overhead by AWARE protocol component. Token preparation runs during scheduled refresh, whereas report encryption and threshold opening are on the report-time path.

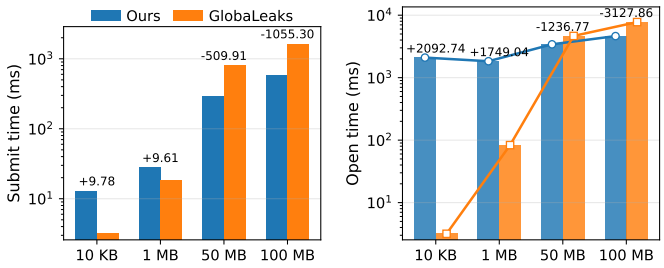
RepCom. A committee member first verifies the one-time signature on $(\mathsf{ct}, d_{\mathsf{ad}})$ under pk_{eph}. It then checks the bulletin-board attribution of each encrypted share, decrypts it with associated data $(pk_{eph}, \mathsf{ct}, \sigma_{eph}, \mathsf{Tok}_{r,\tau})$, and keeps only payloads that parse as $d_j = (j, w_j)$ with matching identity j. From the retained set $\{d_j\}_{j\in\mathcal{J}}$, it runs $\mathtt{C}'(pk_c, \mathsf{ct}, pk_{eph}, \{d_j\}_{j\in\mathcal{J}})$ and outputs the recovered report together with any blame information.

VIII. Performance Evaluation

A. Evaluation Setup

We measure the protocol operations that appear in AWARE's security model and report the main costs in a Java prototype. The evaluation is limited to protocol computation and communication; it is not an end-to-end deployment measurement. The prototype implements the main AWARE construction in Fig. 2, and the source code is available at https://github.com/Zora-G/AWARE. Hashing uses SHA-256 from `java.security`. Report encryption uses the Boneh–Shoup threshold-decryption component from Section III, and opening shares are encrypted as in `RepDec`/`RepCom`. Payloads are encrypted with AES/GCM/NoPadding from `javax.crypto`. Bilinear groups use JPBC Type-F parameters [45], which give an asymmetric Type-3 pairing interface for the TNIBS token layer and the threshold-decryption checks. These parameters are used only for prototype timing, not as deployment parameters.

The implementation uses fixed-base exponentiation preprocessing for public generators and batch verification for threshold shares, with individual verification used only for blame. The optional CCA wrapper in Section VII-A is not included in the measurements. All experiments were run on a MacBook Air with macOS 15.6.1, an Apple M2 chip, and 24 GB unified memory. We report wall-clock averages over three measured runs after one warm-up run. The number of runs and the prototype implementation are limited; we use the results as protocol-operation measurements, not as scalability claims.



(a) Submission path (b) Opening path

Fig. 6. Report-time cost relative to GlobaLeaks. AWARE posts token and ciphertext metadata with the encrypted report; during opening, it verifies the token and reconstructs once from threshold shares after the opening condition holds, while the baseline decrypts at all configured recipients.

B. Protocol Overhead

We separate communication from computation. Communication consists of scheduled pre-token delivery, report upload, and opening-share exchange. Computation follows the protocol phases in Fig. 2: `RegIss` creates pre-tokens, `RegObt` finalizes them locally, `RepGen` encrypts the report, `WAuth` verifies the token, and `RepDec`/`RepCom` perform threshold opening.

Fig. 4, Fig. 5, and Table II give the measured costs. The variable communication terms are linear in the number of pre-tokens m or in the opening threshold t; the report upload is dominated by the payload size. Token issuance and token finalization are linear in m, but they run during scheduled refresh and are not on the report-time path. Report-time work consists of report encryption, TNIBS token verification, ciphertext and share verification, and threshold reconstruction.

C. Comparison

We compare AWARE with GlobaLeaks (GL) [2] on the report-time path. The comparison covers the operations performed when a report is submitted and later opened: report upload, authentication-token verification, payload decryption, and threshold reconstruction. The two systems do not provide the same security semantics. To make the baseline closer to AWARE's threshold-opening setting, we configure GlobaLeaks with $r = 5$ recipients and aggregate retrieval and decryption across all recipients, instead of measuring a single recipient that can decrypt alone.

Fig. 6 and Table III show the resulting cost profile. AWARE has a fixed pairing cost on the submission path, which gives

TABLE II
COMMUNICATION AND COMPUTATION COSTS FOR AWARE. COMMUNICATION TERMS ARE PER PROTOCOL EXECUTION; TIMINGS ARE PROTOTYPE MEASUREMENTS UNDER THE EVALUATION SETUP.

Phase	**Operation**	**Traffic**	**Comm.**	**Comp.**	**Remark**
Setup	Committee setup	–	–	depends on n, t	One-time; see (b).
Registration	Public-key upload	$U \to AS$	64 B	–	One-time.
Refresh	Pre-token issuance	$AS \to U$	$471m$ B	$\approx 0.018m$ s	Scheduled refresh; linear in m.
Refresh	Token finalization	local	–	$\approx 0.040m$ s	Local; linear in m.
Reporting	Report upload	$W \to BB$	$\lvert M\rvert + 743$ B	depends on $\lvert M\rvert$	Token, ciphertext metadata, and encryption framing.
Opening	Share upload	$CM \to BB$	$104t$ B	depends on n, t	Local opening work; see (c).
Opening	Share download	$BB \to CM$	$\approx 711 + 104(t-1)$ B	–	Metadata plus other shares.

(a) Protocol summary

$n\backslash t$	2	4	6	8	10
3	697.060	–	–	–	–
6	1160.935	1174.493	1198.550	–	–
9	1661.589	1914.622	1757.309	1679.919	–
12	2283.009	2456.322	2536.406	2283.112	2373.466
15	3119.472	3366.467	3148.193	3140.419	3120.274

(b) Setup time (ms)

$n\backslash t$	2	4	6	8	10
3	740.038	–	–	–	–
6	456.951	478.778	486.067	–	–
9	448.148	473.761	499.034	519.924	–
12	448.916	502.339	546.148	597.838	645.765
15	965.829	580.473	837.174	614.566	595.304

(c) Opening time (ms)

$\lvert M\rvert$ (MB)	0.01	0.1	0.5	1	20	50	100	150
Time (ms)	13.45	11.71	19.80	28.04	105.79	297.60	592.51	1056.77

(d) Encryption time by payload size (ms)

Note: Setup in (b) measures local setup computation; DKG communication and bulletin-board latency are excluded. Report opening in (c) measures local ciphertext validation, share generation, share verification, and reconstruction from t shares. Network, bulletin-board latency, and separate share-encryption time for posting are excluded.

TABLE III
MEASURED REPORT-TIME COST RELATIVE TO GLOBALEAKS [2]. POSITIVE Δ VALUES MEAN HIGHER MEASURED RUNTIME FOR AWARE; NEGATIVE VALUES MEAN LOWER MEASURED RUNTIME IN THIS CONFIGURED PROTOTYPE EXPERIMENT.

Size	**Submit (ms)**			**Open (ms)**		
	Ours	**GL**	Δ	**Ours**	**GL**	Δ
10 KB	13.01	3.23	+9.78	2095.91	3.17	+2092.74
1 MB	28.04	18.43	+9.61	1832.30	83.26	+1749.04
50 MB	297.60	807.51	**-509.91**	3417.96	4654.73	**-1236.77**
100 MB	592.51	1647.81	**-1055.30**	4628.05	7755.91	**-3127.86**

Note: AWARE uses $(n, t) = (5, 3)$. To compare against a multi-recipient baseline, GlobaLeaks uses $r = 5$ recipients. Submit excludes offline pre-token issuance and local token finalization. AWARE Open includes one `WAuth` token verification and threshold opening; GlobaLeaks Open aggregates retrieval and decryption by all recipients.

higher measured runtime for small reports. For larger reports, symmetric payload encryption dominates, and the measured AWARE implementation is lower in runtime than the configured multi-recipient GlobaLeaks baseline in this prototype setting. This comparison is limited to protocol operations; it does not measure end-to-end deployment performance. In the Type-3 token instantiation, AWARE adds 743 B of fixed upload overhead in Table II: 711 B for token and ciphertext metadata, and the remaining bytes for symmetric-encryption framing. The configured GlobaLeaks baseline adds 440 B, excluding the encrypted payload. For opening, the AWARE column includes one measured `WAuth` call, encrypted share generation, share decryption, batch share verification, threshold reconstruction, and payload decryption, following Fig. 2. The fixed opening cost is visible for small reports; for larger reports, payload handling dominates the measured report-time path.

D. Evaluation Summary

Overall, AWARE adds constant protocol metadata beyond the report payload. The variable communication terms depend on m or t, not on the number of registered users. Scheduled refresh moves token issuance and token finalization off the report-time path; report submission later requires no contact with the authentication server. On the measured report-time path, small submissions are affected by fixed pairing costs, while large submissions are dominated by symmetric payload encryption. Batch share verification reduces opening cost, but token verification and ciphertext validation remain the main fixed costs in the Type-3 pairing implementation. The measurements cover protocol computation and communication only. Network transport, bulletin-board latency, operational response time, and large-scale deployment behavior are outside the experiment.

IX. CONCLUSION

This paper presented *AWARE*, a non-interactive whistleblowing protocol for anonymous report submission by eligible insiders. AWARE separates identity authentication from report

submission through scheduled-refresh anonymous tokens, so a later report does not require a new contact with the authentication server. It also replaces unilateral recipient access with threshold opening, where reconstruction uses committee decryption shares that pass verification after the public opening condition holds. Under the stated leakage and corruption model, we proved token-use unlinkability, token unforgeability, confidentiality of reports before opening, and opening integrity. The Java implementation and measurements show practical costs for the tested parameter ranges. Future work will study deployment integration and user-facing reporting workflows with journalists, auditors, and organizational compliance teams.

APPENDIX A
SECURITY PROOFS

This appendix contains the detailed proofs for Section VI. The main text keeps only proof sketches.

A. Proof of Lemma 1

Proof. We prove the realization statement in the non-aborting case.

Step 1: every accepted dealer defines one committed polynomial. For an accepted dealer $\mathcal{CM}_j$, there exist exponents $(a_{j,0}, \ldots, a_{j,t-1})$ such that $C_{j,k} = g^{a_{j,k}}$. The receiver checks imply that every accepted private value $s_{j\to i}$ equals $f_j(i)$ for the polynomial $f_j(X) = a_{j,0} + a_{j,1}X + \cdots + a_{j,t-1}X^{t-1}$. The pairing checks $e(C_{j,k}, \widetilde{g}) = e(g, \widetilde{C}_{j,k})$ for $0 \le k < t$ ensure that the $\mathbb{G}_1$ and $\mathbb{G}_2$ commitments represent the same exponents. Thus each accepted dealer is bound to one degree-$(t-1)$ polynomial.

Step 2: the aggregate shares are Shamir shares. Define the aggregate polynomial $f(X) = \sum_{j=1}^{n} f_j(X)$. This polynomial has degree at most $t-1$. Each member computes

$$\alpha_i = \sum_{j=1}^{n} s_{j\to i} = \sum_{j=1}^{n} f_j(i) = f(i),$$

so α_i is the Shamir share of $f(0)$ at point i. The published threshold key satisfies

$$\Theta = \prod_{j=1}^{n} C_{j,0} = g^{\sum_j a_{j,0}} = g^{f(0)},$$

and the verification keys satisfy

$$\Theta_i = \prod_{j=1}^{n}\prod_{k=0}^{t-1} C_{j,k}^{i^k} = g^{f(i)}, \qquad \widetilde{\Theta}_i = \prod_{j=1}^{n}\prod_{k=0}^{t-1} \widetilde{C}_{j,k}^{i^k} = \widetilde{g}^{f(i)}.$$

Thus the public key, verification key, and local states match the threshold-decryption key-generation output.

Step 3: fewer than t corrupted shares do not reveal the scalar secret. Let $\mathcal{C}$ be the corrupted committee set with $|\mathcal{C}| \le t-1$. The corrupted members know their own dealer polynomials, all shares delivered to indices in $\mathcal{C}$, their aggregate shares $\{f(i)\}_{i\in\mathcal{C}}$, and the public commitments. Private channels hide honest-to-honest shares. Ignoring public commitments, Shamir secrecy gives that the honest contribution to $f(0)$ is statistically undetermined by fewer than t evaluations, because at least one honest dealer contributes a uniformly random degree-$(t-1)$ polynomial. The commitments add only group encodings of the honest coefficients, including the intended public key $g^{f(0)}$. Under discrete-logarithm hardness, these encodings do not reveal the scalar coefficients to a PPT adversary. Therefore the corrupted view does not reveal the scalar $f(0)$, while the same commitments still bind the verification material used by $\mathsf{TGS}.\texttt{ShareVrfy}$.

Step 4: conclude. The non-aborting DKG execution produces the state expected by the Shamir-based threshold encryption scheme. The trusted dealer is replaced by an interactive computation of the same implicit polynomial f. Since the lemma is conditioned on non-abort, malicious attempts to stop the DKG are outside this realization statement. □

B. Proof of Lemma 2

Proof. Fix a PPT adversary $\mathcal{B}$ for Definition 4. We use hybrids.

Step 1: define the hybrids. Let $\mathbf{I}_0 = (I_{0,1}, \ldots, I_{0,q})$ and $\mathbf{I}_1 = (I_{1,1}, \ldots, I_{1,q})$ be the two challenge vectors. For $j \in \{0, \ldots, q\}$, game G_j finalizes the first j records according to $\mathbf{I}_1$ and the remaining records according to $\mathbf{I}_0$. It gives $\mathcal{B}$ the token vector, public tags, issuer-side records, and allowed auxiliary tokens. Thus G_0 is bit 0, and G_q is bit 1.

Step 2: reduce a distinguishing gap to one slot. Let Δ_j be $\mathcal{B}$'s output probability difference between G_{j-1} and G_j. By the triangle inequality,

$$\mathsf{Adv}^{\mathsf{MSIBnd}}_{\mathcal{B}}(\lambda) \le \sum_{j=1}^{q} |\Delta_j|.$$

It is enough to bound each adjacent gap. Games G_{j-1} and G_j differ only in slot j. All other tokens, records, tags, metadata, and auxiliary tokens are unchanged, and the auxiliary-token-closed single-session games allow the reduction to provide them.

Step 3: handle one adjacent pair. Write the two candidate records for slot j as

$$I_{0,j} = (\mathsf{upk}_0, \eta, \mathsf{nonce}_0, y_0, \mathsf{ptok}_0, \ldots),$$
$$I_{1,j} = (\mathsf{upk}_1, \eta, \mathsf{nonce}_1, y_1, \mathsf{ptok}_1, \ldots),$$

where η is the same by admissibility. If only the recipient key differs, a distinguisher for G_{j-1} and G_j gives a tagged recipient-blindness adversary: it uses the two keys as challenge keys, places the challenge token in slot j, and supplies the other slots as auxiliary tokens. If only the nonce/pre-token transcript differs under the same key, the same construction gives a tagged nonce-blindness adversary.

If both the recipient key and the nonce/pre-token transcript differ, insert the intermediate slot distribution

$$(\mathsf{upk}_0, \mathsf{nonce}_0) \to (\mathsf{upk}_1, \mathsf{nonce}_0) \to (\mathsf{upk}_1, \mathsf{nonce}_1).$$

The first transition is bounded by recipient blindness, and the second by nonce blindness. The single-session challenger performs the challenge finalization, so the reduction never uses an unknown recipient secret key.

Step 4: sum the losses. For each slot j,

$$|\Delta_j| \le \mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{B}'_j}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{B}''_j}(\lambda) + \mathrm{negl}(\lambda),$$

where the negligible term covers malformed or non-admissible records. Summing the q adjacent gaps and absorbing the overhead into $\mathcal{B}'$ and $\mathcal{B}''$ gives

$$\mathsf{Adv}^{\mathsf{MSIBnd}}_{\mathcal{B}}(\lambda) \le 2q\Big(\mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{B}'}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{B}''}(\lambda)\Big) + \mathrm{negl}(\lambda).$$

Since q is polynomial, the bound is negligible when the two single-session blindness advantages are negligible. □

C. Proof of Lemma 3

Proof. We use the Hanzlik formulas from Fig. 2. The issuer-side adversary knows the signing key $\mathsf{ask} = (a_1, a_2)$, the public key $\mathsf{apk} = (A_1, A_2) = (\widetilde{g}^{a_1}, \widetilde{g}^{a_2})$, the public tags, and the issuance records. Only recipient secret keys and the challenge bit are hidden.

Step 1: isolate one auxiliary finalized token. Fix an auxiliary issuance record $I = (\mathsf{upk}, \eta, \mathsf{nonce}, y, \mathsf{ptok})$ that is not the blindness challenge. Its honest final token is obtained by `TNIBS.Obtain`. We compare it to a dummy token under the same tag η. Choose a dummy recipient key (x_0, g^{x_0}), a fresh dummy nonce, dummy issuer randomness y_0, and recipient randomization ω_0, and run the same finalization equations. Equivalently, for

$$h_0 = H_1(\mathsf{nonce}_0), \quad \rho_0 = h_0^{x_0^{-1}}, \quad r_0 = (y_0\omega_0)^{-1},$$

the dummy finalized token is

$$\mathsf{Tok}_0 = (\eta, \rho_0, \sigma_0), \qquad \sigma_0 = (Z_0', Y_0', \widetilde{Y}_0', T_0')$$

with

$$Y_0' = g^{r_0}, \quad \widetilde{Y}_0' = \widetilde{g}^{r_0}, \quad T_0' = H_2(\eta)^{r_0}, \quad Z_0' = (g^{a_1}\rho_0^{a_2})^{r_0^{-1}}.$$

This token verifies because

$$\begin{aligned} e(Z_0', \widetilde{Y}_0') &= e(g, A_1)e(\rho_0, A_2), \\ e(Y_0', \widetilde{g}) &= e(g, \widetilde{Y}_0'), \\ e(g, T_0') &= e(Y_0', H_2(\eta)). \end{aligned}$$

Step 2: replace the concrete auxiliary record by the dummy record. Use two single-session transitions in the auxiliary-input form of the Hanzlik games. Recipient blindness changes the recipient key from upk to g^{x_0} under the same tag. Nonce blindness changes the nonce and presignature transcript from $(\mathsf{nonce}, y, \mathsf{ptok})$ to $(\mathsf{nonce}_0, y_0, \mathsf{ptok}_0)$ under the dummy key and the same tag. After these changes, the simulator knows x_0 and finalizes the dummy token exactly. Thus any distinguisher gives either a recipient-blindness or nonce-blindness adversary, up to negligible public-check rejection probability.

Step 3: handle all auxiliary tokens. Apply Step 2 one token at a time. Keep all public tags, issuer-side records, and metadata in the view. The replacement is identical in both challenge-bit branches because the auxiliary distribution is independent of the challenge bit. By the triangle inequality, replacing all q_{aux} auxiliary finalized tokens by dummy finalized tokens loses at most

$$2q_{\mathsf{aux}}\Big(\mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{C}'}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{C}''}(\lambda)\Big) + \mathrm{negl}(\lambda).$$

After the replacements, the single-session reduction can generate every auxiliary token without any challenge recipient secret key. Thus the challenge-independent auxiliary tokens required by AWARE are absorbed by the same Hanzlik blindness assumptions, in their auxiliary-input form. Lemma 2 then gives the stated multi-session blindness. □

D. Proof of Theorem 4

Proof. Let $\mathcal{A}$ be a PPT adversary for Definition 2. We construct a TNIBS blindness adversary from $\mathcal{A}$.

Step 1: set up the simulated AWARE instance. The reduction $\mathcal{R}$ participates in the one-session TNIBS blindness game. It receives the signer-side values exposed to $\mathcal{AS}$, including $(\mathsf{ask}, \mathsf{apk})$, and the challenge recipient public keys $\mathsf{upk}_0, \mathsf{upk}_1$; the corresponding secret keys stay with the TNIBS challenger. $\mathcal{R}$ registers these keys as the challenge users and honestly samples all other AWARE state. This matches `Setup` and `RegU`.

Step 2: answer non-challenge activity. For each non-challenge issuance, $\mathcal{R}$ runs TNIBS issuance with ask, samples nonce and issuer randomness as in `RegIss`, and records the issuer-side tuple. It computes non-challenge finalized tokens using the generated user secret keys, or receives them as allowed auxiliary tokens from Lemma 3. It simulates non-challenge postings by running `RepGen` honestly and simulates committee entries according to the prescribed leakage.

Step 3: embed the challenge. Given admissible histories, $\mathcal{R}$ checks identical $\mathsf{Leak}^{\mathsf{anon}}$. The histories fix the challenge users, refresh metadata, and public tag. The TNIBS blindness interface generates the concrete challenge nonces, issuer randomness values, and pre-token records as honest `RegIss` would. $\mathcal{R}$ inserts the two unfinalized records into $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$ and submits them as candidate records. The TNIBS challenger returns a finalized token $\mathsf{Tok}^\star$ for one hidden candidate.

Step 4: post-process the TNIBS challenge into an AWARE report. The histories fix the same report payload $M^\star$, the same report-content leakage, and the same posting metadata. The reduction runs the public report generation algorithm with fresh threshold-encryption randomness: $\mathsf{ct}^\star \leftarrow \mathtt{RepGen}(M^\star, \mathsf{Tok}^\star)$, where the Boneh–Shoup threshold-decryption associated data is $\mathsf{Tok}^\star$. This randomized post-processing adds no issuer-side identifier: $\mathsf{ct}^\star$ depends only on $\mathsf{Tok}^\star$, $M^\star$, the public threshold key, and fresh encryption randomness. The reduction gives $\mathcal{A}$ the challenge posting $(\mathsf{Tok}^\star, \mathsf{ct}^\star, \tau^{\mathsf{post},\star})$ together with the common leakage and the simulated non-challenge record.

Step 5: compare views. If the TNIBS challenge bit is b, the simulated token is distributed as the token of $\mathcal{U}_b$. All other values are honestly generated or fixed by the common leakage. The scheduled-refresh assumption excludes a submission-time issuance event outside the token layer. Thus the view is $\mathsf{View}^{\mathsf{anon}}_b$, up to negligible rejection probability.

Step 6: convert the guess. $\mathcal{R}$ outputs $\mathcal{A}$'s bit. Any non-negligible advantage for $\mathcal{A}$ gives a TNIBS blindness distinguisher. Let q_{aux} be the number of auxiliary finalized tokens in the simulated view. Lemma 3 handles them, and Lemma 2 handles candidate records that differ in both recipient key and nonce/pre-token transcript. Therefore, for suitable PPT

adversaries $\mathcal{B}'$ and $\mathcal{B}''$,

$$\mathsf{Adv}^{\mathsf{Anon}}_{\mathcal{A}}(\lambda) \le 2(1+q_{\mathsf{aux}})\Gamma_{\mathsf{B}}(\lambda) + \mathrm{negl}(\lambda),$$
$$\Gamma_{\mathsf{B}}(\lambda) = \mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{B}'}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{B}''}(\lambda).$$

The two TNIBS advantages are negligible by assumption, so AWARE satisfies single-report whistleblower anonymity. □

E. Proof of Theorem 5

Proof. Let $\mathcal{A}$ be a PPT adversary for Definition 3, and let $q = q(\lambda)$ be the number of challenge report slots.

Step 1: reduce to TNIBS token-vector blindness. $\mathcal{R}$ receives a multi-session TNIBS blindness interface and simulates AWARE as in the single-report proof. It checks that $\mathcal{H}_0, \mathcal{H}_1$ have identical $\mathsf{Leak}^{\mathsf{anon}}$. It then uses the TNIBS issuance interface to generate the unfinalized challenge records named by the histories, inserts the issuer-side records into the simulated $\mathsf{View}^{\mathsf{priv}}_{\mathcal{AS}}$, and forms two vectors $\mathbf{I}_0, \mathbf{I}_1$. Position j is the record assigned to report slot j in $\mathcal{H}_0$ or $\mathcal{H}_1$. By admissibility, no record is reused in one vector, and both vectors have the same tag sequence and metadata. Hence they are valid TNIBS challenge vectors.

Step 2: embed the TNIBS challenge vector. The TNIBS challenger samples a hidden bit b, finalizes all records in $\mathbf{I}_b$, and returns the token vector $(\mathsf{Tok}^\star_1, \ldots, \mathsf{Tok}^\star_q)$. For each challenge slot j, the reduction runs report generation on fixed payload $M^\star_j$ and token $\mathsf{Tok}^\star_j$, using fresh encryption randomness and associated data $\mathsf{Tok}^\star_j$. It obtains $\{(\mathsf{Tok}^\star_j, \mathsf{ct}^\star_j, \tau^{\mathsf{post}}_j)\}_{j=1}^q$ and gives them to $\mathcal{A}$ with the common leakage and non-challenge record.

Step 3: prove distributional equivalence. If the TNIBS bit is 0, the token vector follows $\mathcal{H}_0$; if it is 1, it follows $\mathcal{H}_1$. Ciphertexts are randomized post-processing of the tokens, fixed payloads, and fresh encryption randomness. They contain no issuer-side record value except the posted token as associated data. All metadata and non-challenge entries are equal by admissibility, and scheduled refresh rules out submission-time issuance. Thus the simulated view is the real $\mathsf{MAnon}_{\mathcal{A}}$ view for bit b, up to negligible rejection probability.

Step 4: apply the TNIBS hybrid lemma. If $\mathcal{A}$ distinguishes the two histories, $\mathcal{R}$ uses the same output bit to distinguish $\mathbf{I}_0$ from $\mathbf{I}_1$. Let q_{aux} be the number of auxiliary finalized tokens exposed to the adversary. By Lemmas 3 and 2, for suitable PPT adversaries $\mathcal{B}'$ and $\mathcal{B}''$,

$$\mathsf{Adv}^{\mathsf{MAnon}}_{\mathcal{A}}(\lambda) \le 2(q+q_{\mathsf{aux}})\Gamma_{\mathsf{B}}(\lambda) + \mathrm{negl}(\lambda),$$
$$\Gamma_{\mathsf{B}}(\lambda) = \mathsf{Adv}^{\mathsf{RBnd}}_{\mathcal{B}'}(\lambda) + \mathsf{Adv}^{\mathsf{NBnd}}_{\mathcal{B}''}(\lambda).$$

Since q is polynomial and the two TNIBS blindness advantages are negligible, AWARE satisfies multi-report token unlinkability. □

F. Proof of Theorem 6

Proof. Let $\mathcal{A}$ be any PPT adversary in Definition 5. We construct a PPT adversary $\mathcal{B}$ for the TNIBS one-more unforgeability experiment.

Step 1: initialize the simulation. The TNIBS one-more challenger gives $\mathcal{B}$ apk and the presignature-issuance oracle $\mathcal{O}_{\mathsf{TNIBS.Issue}}$. The reduction uses this apk as the AWARE authentication public key and honestly generates all other public state. It handles registration as in Definition 5: accepted keys enter Reg, and rejected keys cannot be used in counted issuance queries.

Step 2: answer issuance queries without knowing ask. On a valid query (upk, η), the reduction samples nonce, sets $h = H_1(\mathsf{nonce})$, and forwards (upk, η, h) to its TNIBS issuance oracle. The oracle returns a presignature psig. The reduction returns $\mathsf{ptok} = (\eta, \mathsf{nonce}, h, \mathsf{psig})$ to $\mathcal{A}$ and increments q_η. This matches `RegIss`: the nonce is fresh and the presignature comes from the same TNIBS issuance algorithm. Thus q_η is exactly the number of counted AWARE pre-tokens issued under tag η.

Step 3: process the adversary's final posting record. When $\mathcal{A}$ outputs an ordered posting list $\mathcal{P} = \{(\widetilde{\mathsf{Tok}}_j, \widetilde{\mathsf{ct}}_j)\}_{j=1}^{\ell}$, the reduction scans it in ledger order with the same consumed-tag set $\mathcal{S}$. A posting is counted only if $\mathsf{PostAccept}(\widetilde{\mathsf{Tok}}_j, \widetilde{\mathsf{ct}}_j; \mathcal{S}) = 1$. The reduction can evaluate this predicate from public parameters. Let J be the accepted indices. For each $j \in J$, parse $\widetilde{\mathsf{Tok}}_j = (\widetilde{\eta}_j, \widetilde{\rho}_j, \widetilde{\sigma}_j)$. By the definition of $\mathsf{PostAccept}$, $\mathsf{TNIBS.Verify}(\mathsf{apk}, \widetilde{\eta}_j, (\widetilde{\rho}_j, \widetilde{\sigma}_j)) = 1$. The replay rule makes accepted replay tags pairwise distinct. Thus, for each tag η, the set $J_\eta = \{j \in J : \widetilde{\eta}_j = \eta\}$ gives $|J_\eta|$ valid TNIBS signatures on pairwise distinct messages $\{\widetilde{\rho}_j\}_{j \in J_\eta}$ under tag η.

Step 4: extract a one-more forgery. If $\mathcal{A}$ wins the AWARE experiment, then

$$N_{\mathsf{acc}} = \sum_{\eta} |J_\eta| > q_{\mathsf{iss}} = \sum_{\eta} q_\eta.$$

By the pigeonhole principle, there exists a public tag $\eta^\star$ such that $|J_{\eta^\star}| > q_{\eta^\star}$. The reduction outputs $\eta^\star$ and the valid triples $\{(\eta^\star, \widetilde{\rho}_j, \widetilde{\sigma}_j)\}_{j \in J_{\eta^\star}}$ to the TNIBS challenger. They verify under the same tag, have pairwise distinct messages, and outnumber $\mathcal{B}$'s presignature queries under $\eta^\star$. Thus $\mathcal{B}$ wins whenever $\mathcal{A}$ wins.

Step 5: conclude. The simulation is perfect until extraction, and extraction succeeds on every winning transcript. Therefore

$$\mathsf{Adv}^{\mathsf{TokUnf}}_{\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{OM\text{-}Unf}}_{\mathcal{B},\mathsf{TNIBS}}(\lambda).$$

The right-hand side is negligible by the assumed one-more unforgeability of the TNIBS scheme for the admitted-key interface. Thus AWARE satisfies authentication-token unforgeability. □

G. Proof of Theorem 7

Proof. Let $\mathcal{A}$ be a PPT adversary for Definition 7. We build a PPT reduction $\mathcal{R}$ against the Boneh–Shoup threshold-decryption AD-only CCA game.

Step 1: forward the corruption set. The adversary chooses $\mathcal{L} \subseteq [n]$ with $|\mathcal{L}| \le t-1$. $\mathcal{R}$ forwards $\mathcal{L}$ to the Boneh–Shoup threshold-decryption challenger and receives the public keys and corrupted local states allowed in the AD-only CCA game. It uses them as the AWARE threshold state.

Step 2: simulate the non-encryption parts. $\mathcal{R}$ honestly generates TNIBS keys and token-issuance records. It samples $k_{\mathcal{BB}}$ and gives it to $\mathcal{A}$ when $\mathcal{L} \neq \emptyset$. This matches the threat

model and does not reveal the challenge plaintext, because no honest challenge share is released.

Step 3: answer decryption-share queries. For each adversarial query $(\mathsf{Tok}, \mathsf{ct})$, the reduction computes $\mathsf{ad} = \mathsf{Tok}$. If the query is admissible, $\mathcal{R}$ forwards $(\mathsf{ct}, \mathsf{ad})$ to the Boneh–Shoup threshold-decryption share-decryption oracle and returns the response. Otherwise it rejects, exactly as the AWARE experiment does. After the challenge associated data is fixed, no query with that associated data is answered.

Step 4: embed the Boneh–Shoup threshold-decryption challenge. When $\mathcal{A}$ submits $(M_0, M_1, \mathsf{Tok}^\star)$, the reduction sets $\mathsf{ad}^\star = \mathsf{Tok}^\star$. Freshness makes $\mathsf{ad}^\star$ a valid challenge associated-data value in the AD-only CCA game. The reduction submits $(M_0, M_1, \mathsf{ad}^\star)$, receives challenge ciphertext $\mathsf{ct}^\star$, and gives $(\mathsf{Tok}^\star, \mathsf{ct}^\star)$ to $\mathcal{A}$.

Step 5: enforce the pre-opening rule. After the challenge, the reduction forwards all admissible share queries except those with associated data $\mathsf{Tok}^\star$. It rejects those queries, including modified ciphertexts under the same token. This matches the pre-opening experiment: honest members release no challenge share, while corrupted local states are already exposed.

Step 6: transfer the advantage. The view is distributed as in Definition 7: TNIBS values are honest, the corruption view is the Boneh–Shoup threshold-decryption corruption view, and share responses come from the same interface. Thus $\mathcal{R}$ outputs $\mathcal{A}$'s bit and wins with the same advantage, except for negligible invalid-DKG or malformed-query events. Hence

$$\mathsf{Adv}^{\mathsf{Conf}}_{\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{ADCCA}}_{\mathcal{R},\mathsf{TGS}}(\lambda) + \mathrm{negl}(\lambda).$$

The right-hand side is negligible by the assumed AD-only CCA security of the Boneh–Shoup threshold decryption component and Lemma 1. □

H. Proof of Theorem 8

Proof. We show that any accepting honest `RepCom` execution contains the witness required by Definition 8, except with negligible probability.

Step 1: bind the execution to one accepted posting. Fix the accepted posting processed by the honest committee member: $(\mathsf{Tok}_{r,\tau}, \mathsf{ct})$, with $\mathsf{ad} = \mathsf{Tok}_{r,\tau}$. The append-only log fixes one ledger order and prevents deletion, reordering, or reattribution. The replay rule fixes the first valid posting for $\rho_{r,\tau}$; honest members do not open later postings with the same tag.

Step 2: check ciphertext validity. The first operation in `RepCom` is $\mathsf{TGS}.\mathtt{CtVrfy}(pk_{\mathsf{th}}, \mathsf{ct}, \mathsf{ad})$. If it fails, the honest member rejects. If it succeeds, soundness of the threshold-decryption ciphertext-verification relation implies that, except with negligible probability, the ciphertext is valid for $(\mathsf{ct}, \mathsf{ad})$. Let $\mathsf{Bad}_{\mathsf{ct}}$ be the event that an invalid ciphertext passes.

Step 3: bind each encrypted share to the same posting. Consider a share entry attributed by $\mathcal{BB}$ to $\mathcal{CM}_j$. The honest combiner decrypts it only with associated data $(\mathsf{ct}, \mathsf{Tok}_{r,\tau})$. If decryption fails, the entry is ignored. If it succeeds, the payload must parse as $d_j = (j, w_j)$, so the inner index matches the authenticated bulletin-board identity. The committee-only share encryption prevents outsiders without $k_{\mathcal{BB}}$ from modifying, rebinding, or replaying a share under different associated data, except with negligible probability. Corrupted committee members may form payloads, but those payloads remain attributed to their authenticated identities.

Step 4: accept only algebraically valid shares. A parsed payload $d_j = (j, w_j)$ is inserted into the candidate set only if $\mathsf{TGS}.\mathtt{ShareVrfy}(pk_c, j, \mathsf{ct}, \mathsf{ad}, d_j) = 1$. By share-verification soundness, except with negligible probability every accepted d_j is a valid share for the same $(\mathsf{ct}, \mathsf{ad})$ and index j. Let $\mathsf{Bad}_{\mathsf{sh}}$ be the event that an invalid share passes. A union bound covers polynomially many entries.

Step 5: reconstruct only from a verified threshold set. If fewer than t shares pass, `RepCom` rejects. Thus any accepting execution has at least t posting-bound, attributed, verified shares. Let $\mathcal{J}_t$ be the t-subset used by `Combine`. Uniqueness of the reconstruction relation gives $M = \mathsf{TGS}.\mathtt{Combine}(pk_c, \mathsf{ct}, \mathsf{ad}, \{d_j\}_{j\in\mathcal{J}_t})$.

Step 6: conclude the bad-event bound. If Definition 8 is violated, then an honest combiner outputs $M \neq \perp$ without such a t-element witness. The previous steps show that this requires $\mathsf{Bad}_{\mathsf{ct}}$, share-encryption forgery or rebinding, log attribution failure, or $\mathsf{Bad}_{\mathsf{sh}}$. Each event is negligible, and the number of processed entries is polynomial. Their union is negligible. This proves committee-internal opening integrity. The proof does not assert public verifiability or liveness. □

I. Proof of Lemma 9

Proof. Let $\mathcal{A}$ win $\mathsf{mstSIG}_{\mathcal{A},\mathcal{S}}(\lambda)$ with probability ϵ, and let $q_1(\lambda)$ bound its fresh-key queries. We build $\mathcal{R}$ for strong one-time security. It receives a challenge key $pk^\star$, may ask for one signature under it, and must output a different valid message-signature pair.

Step 1: guess the challenged one-time key. The reduction samples $k \xleftarrow{\$} \{1, \ldots, q_1(\lambda)\}$. It honestly answers every fresh-key query except the kth. For the kth query, it returns $pk^\star$ and stores no signing key.

Step 2: answer signing queries. For honest sessions, $\mathcal{R}$ signs with the stored key. For the guessed session, it forwards the message to its one-time signing oracle. Each session is signed at most once, so the view is distributed as in the real mstSIG game.

Step 3: use the forgery. If $\mathcal{A}$'s forgery uses the guessed key $pk^\star$, then it is a different valid pair $(m^\star, \sigma^\star)$. $\mathcal{R}$ forwards this pair. The guess is correct with probability at least $1/q_1(\lambda)$, so

$$\Pr[\mathcal{R} \text{ wins the strong one-time signature game}] \ge \epsilon/q_1(\lambda).$$

A non-negligible ϵ would contradict strong one-time security. Hence $\mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{A}}(\lambda)$ is negligible. □

J. Proof of Proposition 10

Proof. Let $\mathcal{A}$ be a PPT adversary for Definition 11. We use two games.

Step 1: define the signature-forgery bad event. For an encryption query (m_{i0}, m_{i1}, d_i), the wrapped encryption algorithm returns

$$(pk_{eph,i}, \mathsf{ct}_i, \sigma_{eph,i}), \qquad \sigma_{eph,i} \leftarrow \mathrm{S}(sk_{eph,i}, (\mathsf{ct}_i, d_i)).$$

Call a later decryption-share query

$$((\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}), \widehat{d})$$

a *valid key-reuse query* if $\widehat{pk}_{eph} = pk_{eph,i}$ for some previous encryption query,

$$(\widehat{\mathsf{ct}}, \widehat{d}, \widehat{\sigma}_{eph}) \neq (\mathsf{ct}_i, d_i, \sigma_{eph,i}),$$

and

$$\mathrm{V}(\widehat{pk}_{eph}, (\widehat{\mathsf{ct}}, \widehat{d}), \widehat{\sigma}_{eph}) = 1.$$

Let Forge be the event that such a query occurs.

Step 2: bound $\Pr[\mathsf{Forge}]$. Build $\mathcal{B}$ for $\mathsf{mstSIG}_{\mathcal{B},\mathcal{S}}(\lambda)$. It simulates the threshold CCA experiment. On encryption query (m_{i0}, m_{i1}, d_i), $\mathcal{B}$ obtains a fresh one-time key $pk_{eph,i}$ and computes

$$\mathsf{ct}_i \leftarrow \mathrm{Enc}(pk_{\mathsf{th}}, m_{ib}, pk_{eph,i})$$

using honest threshold-encryption state. It asks the mst-SIG oracle to sign (ct_i, d_i) under $pk_{eph,i}$ and returns $(pk_{eph,i}, \mathsf{ct}_i, \sigma_{eph,i})$. It answers decryption-share queries as the wrapped interface does. If Forge occurs, then

$$(\widehat{pk}_{eph}, (\widehat{\mathsf{ct}}, \widehat{d}), \widehat{\sigma}_{eph})$$

is a valid signature under a used one-time key, on a different message-signature pair. Thus $\mathcal{B}$ outputs an mstSIG forgery, and

$$\Pr[\mathsf{Forge}] \leq \mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{B}}(\lambda).$$

By Lemma 9, this probability is negligible if $\mathcal{S}$ is strongly one-time secure.

Step 3: condition on no signature forgery. Condition on $\overline{\mathsf{Forge}}$. Build $\mathcal{R}$ for the AD-only CCA security of the underlying threshold scheme. If the cited AD-only CCA game is single-challenge, use the standard hybrid over encryption-query indices; equivalently, run $\mathcal{R}$ against the multi-challenge variant.

For each encryption query (m_{i0}, m_{i1}, d_i), $\mathcal{R}$ samples $(pk_{eph,i}, sk_{eph,i}) \leftarrow \mathrm{G}(\lambda)$ and submits

$$(m_{i0}, m_{i1}, pk_{eph,i})$$

to its AD-only CCA challenger, using $pk_{eph,i}$ as associated data. It receives ct_i, signs (ct_i, d_i), and returns the wrapped ciphertext. This is exactly E''s distribution.

Step 4: answer decryption-share queries. On a query

$$((\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}), \widehat{d}),$$

the reduction first verifies the one-time signature on $(\widehat{\mathsf{ct}}, \widehat{d})$. If verification fails, it returns $\{\perp\}_{i=1}^{n}$. If verification succeeds, then under $\overline{\mathsf{Forge}}$ there are two cases. If $\widehat{pk}_{eph}$ was never used in an encryption query, the associated data is fresh. If $\widehat{pk}_{eph} = pk_{eph,i}$ for a previous encryption query, then the only valid query with that key is the exact replay

$$(\widehat{pk}_{eph}, \widehat{\mathsf{ct}}, \widehat{\sigma}_{eph}, \widehat{d}) = (pk_{eph,i}, \mathsf{ct}_i, \sigma_{eph,i}, d_i),$$

which Definition 11 rejects. Thus every answered query is admissible for the AD-only CCA oracle. The reduction forwards $(\widehat{\mathsf{ct}}, \widehat{pk}_{eph})$ and relays the shares.

Step 5: conclude the bound. Under $\overline{\mathsf{Forge}}$, the simulation is exactly Definition 11. Any distinguishing advantage becomes a multi-challenge AD-only CCA advantage. If only the single-challenge theorem is cited, the hybrid over q_{enc} queries loses a factor q_{enc}. Therefore

$$\mathsf{Adv}^{\mathsf{thCCA}}_{\mathcal{A}}(\lambda) \leq q_{\mathsf{enc}}\, \mathsf{Adv}^{\mathsf{ADCCA}}_{\mathcal{R}}(\lambda) + \mathsf{Adv}^{\mathsf{mstSIG}}_{\mathcal{B}}(\lambda).$$

The first term is negligible because q_{enc} is polynomial and the threshold component is AD-only CCA-secure; the second is negligible by strong one-time signature security. Hence the wrapped interface is threshold CCA-secure. □

REFERENCES

[1] A. Swartz. (2024) Securedrop. [Online]. Available: https://securedrop.org/

[2] S. Ferro, G. Pellerano, F. Pietrosanti, A. Rodolfi, and R. Sghaier. (2024) Globaleaks. [Online]. Available: https://www.globaleaks.org/

[3] J. Assange. (2024) Wikileaks. [Online]. Available: https://www.wikileaks.org/

[4] Freedom of the Press Foundation. (2024) Introducing securedrop protocol. [Online]. Available: https://freedom.press/tech/news/introducing-securedrop-protocol/

[5] H. Jayakrishnan and R. Murali, "A simple and robust end-to-end encryption architecture for anonymous and secure whistleblowing," in *Proc. IC3*, 2019, pp. 1–6.

[6] S. Reijenga, K. Aslam, and E. Guzmán, "Whistleblowing in the software industry: A survey," in *Proc. ESEM*, 2023, pp. 1–12.

[7] M. Terracol. (2022) Policy brief: Internal whistleblowing systems. [Online]. Available: https://files.transparencycdn.org/images/2022_PolicyBrief_InternalWhistleblowingSystems_English-1.pdf

[8] T. Agrikola, G. Couteau, and S. Maier, "Anonymous whistleblowing over authenticated channels," in *Proc. TCC*, 2022, pp. 685–714.

[9] W. Quach, L. Tyner, and D. Wichs, "Lower bounds on anonymous whistleblowing," in *Proc. TCC*, 2023, pp. 3–32.

[10] M. Kelkar, G. Ganesh, J. Bonneau, and S. M. Weinberg, "Breaking omertà: On threshold cryptography, smart collusion, and whistleblowing," in *Proc. CCS*, 2025.

[11] R. Dingledine, N. Mathewson, and P. F. Syverson, "Tor: The second-generation onion router," in *Proc. USENIX Security*, 2004, pp. 303–320.

[12] A. Arora and C. Garman, "Improving the performance and security of tor's onion services," *Proc. Priv. Enhancing Technol.*, 2025.

[13] S. Angel and S. Setty, "Unobservable communication over fully untrusted infrastructure," in *Proc. OSDI*, 2016, pp. 551–569.

[14] S. Eskandarian, H. Corrigan-Gibbs, M. Zaharia, and D. Boneh, "Express: Lowering the cost of metadata-hiding communication with cryptographic privacy," in *Proc. USENIX Security*, 2021, pp. 1775–1792.

[15] A. Kwon, D. Lu, and S. Devadas, "XRD: Scalable messaging system with cryptographic privacy," in *Proc. NSDI*, 2020, pp. 759–776.

[16] D. Schadt, C. Coijanovic, and T. Strufe, "Aimless onions: Mixing without topology information," *Proc. Priv. Enhancing Technol.*, 2025.

[17] A. Davidson, J. Iyengar, and C. A. Wood, "The Privacy Pass Architecture," RFC 9576, Jun. 2024. [Online]. Available: https://www.rfc-editor.org/rfc/rfc9576

[18] S. Celi, A. Davidson, S. Valdez, and C. A. Wood, "Privacy Pass Issuance Protocols," RFC 9578, Jun. 2024. [Online]. Available: https://www.rfc-editor.org/rfc/rfc9578

[19] H. Wang, D. He, Z. Liu, and R. Guo, "Blockchain-based anonymous reporting scheme with anonymous rewarding," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1514–1524, 2019.

[20] Y. Wang, X. Wang, H.-N. Dai, X. Zhang, and M. Imran, "A data reporting protocol with revocable anonymous authentication for edge-assisted intelligent transport systems," *IEEE Trans. Ind. Informat.*, 2022.

[21] M. Jia, J. Chen, Y. Wang, and K. He, "Multi-authority anonymous credentials with efficient and decentralized supervision," *IEEE Trans. Inf. Forensics Security*, 2025.

[22] W. Xue, Y. Yang, M. Huang, Y. Li, H. H. Pang, and R. H. Deng, "Dkvsso: Delegatable keyed-verification credentials for efficient anonymous single sign-on," *IEEE Trans. Inf. Forensics Security*, 2025.

[23] R. Shi, Y. Yang, H. Feng, F. Yuan, H. Xie, and J. Zhang, "PriRPT: Practical blockchain-based privacy-preserving reporting system with rewards," *J. Syst. Archit.*, vol. 143, p. 102985, 2023.

[24] J. Müller, B. Pejó, and I. Pryvalov, "DeVoS: Deniable yet verifiable vote updating," *Proc. Priv. Enhancing Technol.*, 2024.

[25] R. Küsters, J. Liedtke, J. Müller, D. Rausch, and A. Vogt, "Ordinos: A verifiable tally-hiding e-voting system," in *Proc. EuroS&P*, 2020, pp. 216–235.

[26] P. Chaidos, V. Cortier, G. Fuchsbauer, and D. Galindo, "BeleniosRF: A non-interactive receipt-free electronic voting scheme," in *Proc. CCS*, 2016, pp. 1614–1625.

[27] T. Haines, J. Müller, and I. Querejeta-Azurmendi, "Scalable coercion-resistant e-voting under weaker trust assumptions," in *Proc. SAC*, 2023, pp. 1576–1584.

[28] N. Huber, R. Küsters, T. Krips, J. Liedtke, J. Müller, D. Rausch, P. Reisert, and A. Vogt, "Kryvos: Publicly tally-hiding verifiable e-voting," in *Proc. CCS*, 2022, pp. 1443–1457.

[29] M. Bougon, H. Chabanne, V. Cortier, A. Debant, E. Dottax, J. Dreier, P. Gaudry, and M. Turuani, "Themis: An on-site voting system with systematic cast-as-intended verification and partial accountability," in *Proc. CCS*, 2022, pp. 397–410.

[30] E. Zhai, D. I. Wolinsky, R. Chen, E. Syta, C. Teng, and B. Ford, "AnonRep: Towards tracking-resistant anonymous reputation," in *Proc. NSDI*, 2016, pp. 583–596.

[31] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing atop PoS blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, 2019.

[32] M. A. Azad, S. Bag, and F. Hao, "PrivBox: Verifiable decentralized reputation system for online marketplaces," *Future Gener. Comput. Syst.*, vol. 89, pp. 44–57, 2018.

[33] J. Blömer, F. Eidens, and J. Juhnke, "Practical, anonymous, and publicly linkable universally-composable reputation systems," in *Proc. CT-RSA*, 2018, pp. 470–490.

[34] W. Zhao, X. Yang, S. Qi, J. Wei, X. Dong, and Y. Qi, "Secure blockchain-based reputation system for iiot-enabled retail industry with resistance to sybil attack," *Future Gener. Comput. Syst.*, vol. 166, p. 107705, 2025.

[35] R. Xiong, J. Cheng, J. Pu, X. Dong, C. Chen, and Z. Xu, "Enhancing trust and collaboration: A reputation-driven mechanism for cross-chain iot data sharing," *Comput. Netw.*, vol. 264, p. 111266, 2025.

[36] H. Zou, X. Liu, W. Ren, and T. Zhu, "A decentralized electronic reporting scheme with privacy ion based on proxy signature and blockchain," *Secur. Commun. Netw.*, vol. 2022, pp. 1–18, 2022.

[37] M. Chase, T. Perrin, and G. Zaverucha, "The signal private group system and anonymous credentials supporting efficient verifiable encryption," in *Proc. CCS*, 2020, pp. 1445–1459.

[38] J. Blömer, J. Bobolz, D. Diemert, and F. Eidens, "Updatable anonymous credentials and applications to incentive systems," in *Proc. CCS*, 2019, pp. 1671–1685.

[39] J. Blömer and J. Bobolz, "Delegatable attribute-based anonymous credentials from dynamically malleable signatures," in *Proc. ACNS*, 2018, pp. 221–239.

[40] Y. Zhang, C. Xu, H. Li, K. Yang, N. Cheng, and X. Shen, "PROTECT: Efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage," *IEEE Trans. Mobile Comput.*, vol. 20, no. 6, pp. 2297–2312, 2020.

[41] D. Boneh and V. Shoup, *A Graduate Course in Applied Cryptography*. Self-published, 2023, version 0.6. [Online]. Available: https://toc.cryptobook.us/

[42] L. Hanzlik, "Non-interactive blind signatures for random messages," in *Proc. Eurocrypt*, 2023, pp. 722–752.

[43] C. Garman, M. Green, and I. Miers, "Decentralized anonymous credentials," *Cryptology ePrint Archive*, 2013.

[44] L. Lamport, "Constructing digital signatures from a one way function," SRI International Computer Science Laboratory, Tech. Rep. CSL-98, 1979.

[45] A. D. Caro. (2018) The java pairing based cryptography library (JPBC). [Online]. Available: http://gas.dia.unisa.it/projects/jpbc/