

CompCap-Stego: Computable Capability-Pointer Steganography with Hidden Policy Evaluation and Verifiable External Payload Retrieval

Mohammad Othman Nassar

Faculty of Information Technology-Cyber Security Department, Amman Arab University, Amman, Jordan

Abstract—CompCap-Stego is a capability-pointer steganography framework in which the concealed object is a compact dual-zone capability capsule rather than the protected payload. The sealed access zone stores receiver binding, wrapped key material, a revocation handle, an audit nonce, and a payload-verification root, while the computable policy zone supports validity, expiry, bounded-use access, revocation, replay state, receiver matching, honey-state misuse detection, and policy-digest consistency. Here, computable denotes authenticated policy-state evaluation after capsule extraction rather than general-purpose or homomorphic computation directly over unrecovered steganographic bits. Retrieval proceeds only after capsule extraction, binding validation, policy evaluation, receiver-bound token derivation, and verification of the externally returned encrypted payload. A 512-byte core capsule reduces embedded size by 95.00 percent for a 10-kilobyte (KB) payload, 99.50 percent for 100 KB, 99.95 percent for a 1-megabyte (MB) payload, and 99.995 percent for 10 MB. Carrier quality remains stable at 55.8 to 55.9 decibels (dB) in peak signal-to-noise ratio (PSNR), 0.997 structural similarity index measure (SSIM), and 0.56 area under the receiver operating characteristic curve (AUC). Medium error correction reaches 99.6 percent extraction success and 99.5 percent tag-verification success. Policy violations are rejected at 98.9 to 99.8 percent, tampered or replaced external ciphertext at 99.8 to 99.9 percent, and the full design records 0.4 percent composite attack success.

Keywords—Steganography; capability-based access control; hidden policy evaluation; verifiable external retrieval

I. INTRODUCTION

Steganography has traditionally concealed secret messages in innocuous carriers while maintaining the cover's perceptual or statistical properties. Across image, audio, video, network, linguistic, and generative carriers, this objective continues to create a central tension among capacity, imperceptibility, robustness, and extraction reliability [1]–[7]. More recent work has extended steganography beyond message hiding to security functions that include hidden computation, authentication support, integrity proof transmission, and high-capacity generative embedding [8]–[13]. This progression broadens the role of the hidden object: in addition to a conventional message, it can serve as a control object, proof object, credential object, or retrieval-enabling object.

Full-payload embedding provides a direct route to controlled information access, but it is not always the most suitable option. When the protected payload is large, dynamic, revocable, or externally managed, placing the complete ciphertext in the cover increases capacity pressure and carrier distortion. A smaller

hidden object reduces this embedding load, yet a static hidden pointer or static token leaves the access-state weakness unresolved. Once extracted or replayed, it may continue to expose the external payload unless access state, receiver binding, revocation, replay, and payload verification are enforced. The intervention point therefore lies between compact pointer-style hiding and stateful access-controlled retrieval.

Adjacent literature addresses several parts of this problem separately. Homomorphic and computable steganography demonstrate that computation can be applied to concealed bits without directly revealing them [8]. Anonymous-token systems and private-metadata token constructions provide mechanisms for unlinkability, private metadata, public verification, and double-spending control [14]–[18]. Internet of Things (IoT) authorization and device-sharing research shows the risks of exposed authorization data and the need for delegation, revocation, and access-state management [19]–[21]. Privacy-oriented concealment also addresses exposure of access-related network traces [22]. Capability and network-bound access-control systems provide process or context binding [23], while encrypted-domain data-hiding and proxy-re-encryption systems support controlled sharing of protected data [24]–[27]. Nevertheless, these lines do not provide a single steganographic construction in which the concealed object is a compact computable capability capsule that controls verified retrieval of an externally encrypted payload.

Carrier-level developments in image-based information hiding and cover selection provide a second motivation for CompCap-Stego. Research on cover-image selection establishes that carrier suitability affects embedding reliability, visual quality, and resistance to detection, whereas high-capacity generative steganography shows that modern carriers can be constructed or adapted for larger and more robust hidden representations [10]. CompCap-Stego takes a different but complementary direction: instead of increasing the hidden payload placed in the carrier, it confines the hidden object to a compact access capsule and moves the protected content to verified external encrypted storage. The design emphasis therefore shifts from maximizing embedded payload capacity to minimizing embedded access material while retaining policy enforcement, receiver binding, and payload verification.

This study proposes CompCap-Stego, a computable capability-pointer steganography framework for hidden policy evaluation and verifiable external payload retrieval. In this study, computable denotes a limited authenticated state-transition capsule

whose concealed policy fields support receiver-side and gateway-side policy evaluation after extraction; it does not denote general-purpose computation over unrecovered steganographic bits. Rather than embedding the payload, the framework embeds a dual-zone capability capsule. Its sealed access zone protects retrieval material, receiver binding, wrapped key material, payload-verification root, revocation handle, and audit nonce, whereas its computable policy zone maintains compact hidden state for validity, expiry, bounded-use retrieval, revocation, replay, receiver matching, honey-state misuse detection, and policy-digest checking. Retrieval is granted only after the capsule is extracted, its binding is verified, the hidden policy predicate is satisfied, a receiver-bound retrieval token is derived, and the returned external ciphertext matches the verification root bound to the capsule. The principal novelty therefore lies in combining a concealed capability capsule, authenticated stateful policy-before-retrieval control, receiver-bound authorization, and external-payload verification rather than in introducing a new form of homomorphic steganographic computation.

The evaluation demonstrates the role of the capsule-centered design across compactness, carrier quality, extraction reliability, policy enforcement, payload verification, runtime overhead, and ablation behavior. Compared with full-payload embedding, CompCap-Stego substantially reduces the number of embedded bits as payload size increases because the cover carries a capsule of bounded size rather than the full encrypted payload. Compared with static pointer and static sealed-token variants, the proposed method provides replay, revocation, expiry, receiver-mismatch, honey-probing, and payload-replacement rejection through the policy and verification layers.

A. Contributions

CompCap-Stego makes five linked contributions. First, it defines computable capability-capsule steganography in which the hidden object is a compact capability capsule rather than the full payload, a plain pointer, or an ordinary hidden credential. Second, it introduces a dual-zone capsule composed of a sealed access zone for retrieval and verification material and a computable policy zone for compact hidden policy state. Third, it evaluates policy-before-retrieval rules for validity, expiry, bounded-use access, revocation, replay, receiver matching, and honey-state misuse detection. Fourth, it binds external ciphertext acceptance to a capsule-sealed hash or Merkle root to prevent replacement and tampering. Fifth, it compares the design with full-payload embedding, static pointer hiding, static sealed-token hiding, and component-removal variants.

Section II reviews the closest steganography, token, capability, and verifiable-retrieval literature. Section III defines the system and threat models, including the gateway trust boundary. Section IV presents the CompCap-Stego construction, followed by the formal algorithms and security properties in Section V. Section VI describes the experimental methodology and sensitivity analyses, while Section VII reports the quantitative results. Section VIII discusses the resulting design trade-offs, Section IX identifies the validated scope and remaining limitations, and Section X concludes the study.

II. RELATED WORK

This section positions CompCap-Stego relative to six adjacent families of work: conventional and adaptive steganography,

computable and generative steganography, anonymous/private-metadata tokens, authorization-data and capability-based access control, steganography-assisted authentication [28], [29], and encrypted-domain retrieval with integrity verification.

A. Steganography, Steganalysis, and Carrier Selection

Classical steganography and modern distortion-minimization methods optimize the capacity-imperceptibility balance [1]–[7]. Adaptive image methods reduce detectable artifacts by selecting low-risk embedding regions, while deep steganalysis models increase the difficulty of remaining undetected. These methods are important for the carrier layer of CompCap-Stego, but they do not define the hidden object as a stateful retrieval capability. StyleStego is related at the carrier layer because it explores high-capacity generative steganography using a style-based generative adversarial network (StyleGAN) embedding [10]. In addition, optimization-assisted color visual cryptography is relevant as adjacent image-security work because it addresses protected visual-content representation using image-domain security mechanisms [11].

B. Computable and Generative Steganography

ProSt shows that homomorphic-style computation can be performed on hidden steganographic data [8]. Generative and neural approaches hide information by synthesizing carriers or learning end-to-end encoder/decoder structures [9], [12], [13]. CompCap-Stego does not claim to be the first work on hidden computation or generative hiding. Its scope is narrower: the hidden computation is limited to policy-state evaluation over a compact capsule, and the protected payload remains external. Unlike systems that operate directly on concealed or unrecovered steganographic data, CompCap-Stego first extracts and authenticates the capsule and then evaluates its protected policy state. The term computable is therefore used in the narrower sense of stateful capability evaluation.

C. Anonymous Tokens and Private Metadata

Anonymous tokens with private metadata bits and public metadata support privacy-preserving issuance, redemption, metadata hiding, public verification, and double-spending controls [14]–[18]. These systems strongly motivate the use of compact credential-like objects, but they are not steganographic carrier objects and do not bind an embedded capsule to off-cover ciphertext retrieval. CompCap-Stego combines the token-like idea of a compact access object with carrier concealment and payload-root verification.

D. Authorization Data, Capability Systems, and External Access Control

IoT device-sharing and authorization-data studies identify the underlying access-control weakness: exposed access material can be replayed or remain usable after revocation [19]–[21]. Privacy-oriented concealment work addresses a related exposure by hiding network-use traces through stealth protocols and proxying when access activity can reveal sensitive behavior [22]. Capability and network-bound systems such as NetCap respond to token theft by binding access authority to a process or network context [23]. These mechanisms provide access control, but they do not operate as steganographic retrieval capsules. CompCap-Stego retains the context-binding function and places the resulting authority inside a concealed carrier capsule.

E. Encrypted-Domain Data Hiding and Verifiable Retrieval

Reversible data hiding in encrypted domains and proxy re-encryption-based data hiding provide controlled sharing, identity verification, and encrypted-domain embedding [24]–[27]. Merkle-based integrity verification and steganographic proof transmission add the complementary function of tamper-evident verification to hidden transmission [30], [31]. These approaches therefore cover part of the proposed framework's verification role, but not the dual-zone hidden capability capsule that evaluates policy before retrieval. The functional boundaries of the closest work families are summarized in Table I.

TABLE I. FUNCTIONAL COMPARISON WITH CLOSEST WORK FAMILIES

Work family	Representative focus and citations	Overlap with CompCap-Stego	Remaining gap and claim boundary
Adaptive and deep steganography	Capacity, imperceptibility, robustness, and steganalysis resistance [1]–[7], [10]–[13]	Provides carrier-layer methods and detectability metrics.	Does not provide stateful external retrieval authority; CompCap-Stego does not claim novelty in adaptive or generative embedding.
Computable steganography	Computation over hidden steganographic data [8]	Supports the idea that hidden bits can participate in computation.	Does not define a compact access capsule for verified retrieval; CompCap-Stego is not claimed as the first hidden-computation method.
Anonymous and private-metadata tokens	Private metadata, public metadata, verification, and double-spending control [14]–[18]	Supports compact token-like access objects and replay reasoning.	No steganographic carrier binding or external ciphertext root; CompCap-Stego is not claimed as the first anonymous-token system.
Authorization-data protection and capability systems	Delegation, revocation, hidden authorization data, and network/process-bound capabilities [19]–[23]	Motivates revocation, receiver binding, and access-state management.	Does not place a computable retrieval capsule inside a cover; CompCap-Stego is not claimed as the first access-control or capability system.
Encrypted-domain data hiding and proxy re-encryption	Controlled sharing, encrypted-domain embedding, and dynamic authorization [24]–[27]	Overlaps with external encrypted data and controlled retrieval.	No policy-before-retrieval capsule in the carrier; CompCap-Stego is not claimed as the first encrypted-domain hiding method.
Integrity-proof steganography and Merkle verification	Merkle or proof transmission through covert channels [30], [31]	Overlaps with tamper-evident proof handling.	Does not combine sealed retrieval, hidden policy, and receiver-bound access; CompCap-Stego is not claimed as the first Merkle-verification method.

Accordingly, the claim boundary is deliberately limited: CompCap-Stego is not positioned as the first hidden-computation, anonymous-token, access-control, Merkle-verification, or encrypted-domain data-hiding method. Its contribution rests on the unified capsule-level mechanism that combines a sealed access zone, a computable policy zone, honey positions, and verified external retrieval inside a concealed carrier object.

III. SYSTEM MODEL AND THREAT MODEL

A. System Overview

CompCap-Stego considers a sender that protects a payload by encrypting it externally and embedding a compact capability capsule in a cover object. The receiver extracts the capsule, validates its binding, evaluates hidden policy state, derives a retrieval token, and accepts the externally returned ciphertext only if it matches the verification root carried in the capsule.

$$\Gamma = Z_s \parallel Z_p \parallel \tau \quad (1)$$

where, Γ denotes the embedded capability capsule, Z_s is the sealed access zone, Z_p is the computable policy zone, and τ is the capsule binding tag.

$$C' = \text{Embed}(C, \Gamma, K_{emb}) \quad (2)$$

where, C is the original cover, C' is the stego-cover, and K_{emb} is the embedding key.

$$M = (R, D, H) \quad (3)$$

The position map M separates carrier positions into real positions R , decoy positions D , and honey positions H .

$$R \cap D = R \cap H = D \cap H = \emptyset \quad (4)$$

B. External Payload and Policy State

The protected content is encrypted before storage using authenticated encryption with associated data (AEAD), where AD denotes the associated data. The receiver does not accept the retrieved ciphertext merely because it is returned from the external store; acceptance depends on the hash or Merkle-root verification material bound to the capsule [32]. The core notation is summarized in Table II.

TABLE II. CORE NOTATION

Symbol	Meaning
Γ	Complete embedded capability capsule
Z_s	Sealed access zone
Z_p	Computable policy zone
τ	Capsule binding tag
C, C'	Cover and stego-cover
P, P_e	Plain payload and encrypted payload
$root_p$	Hash or Merkle root for external payload verification
R, D, H	Real, decoy, and honey position classes
R_{ctx}	Receiver context used in receiver-binding and policy evaluation.
c_u	Remaining-use counter
b_{rep}, b_r, b_h	Replay, revocation, and honey-state flags

$$P_e = \text{AEAD.Enc}_{K_e}(P, AD) \quad (5)$$

$$root_p = H(P_e) \quad (6)$$

$$root_p = \text{MerkleRoot}(ch_1, ch_2, \dots, ch_n) \quad (7)$$

$$Z_p = (b_v, b_x, c_u, b_r, b_{rep}, b_{rec}, b_h, d_p) \quad (8)$$

$$\text{Allow}(\Gamma, R, t) = b_v \wedge \neg b_x \wedge (c_u > 0) \wedge \neg b_r \wedge \neg b_{rep} \wedge b_{rec} \wedge \neg b_h \quad (9)$$

C. Threat Model and Security Goals

The threat model allows the adversary to observe stego-covers, attempt extraction, replay a previously extracted capsule, request retrieval through an unauthorized receiver context, trigger decoy or honey positions, modify the stego-cover, tamper with the external ciphertext, substitute another payload, or attempt to bypass revocation under an active-network adversary abstraction [33]. The adversary is assumed not to know the embedding key, sealing key, tag key, or receiver secret material. Within this boundary, the security goals comprise capsule confidentiality, extraction correctness for authorized receivers, replay rejection, revocation enforcement, receiver binding, honey misuse rejection, external payload integrity, and compactness relative to full-payload embedding. The external payload store is not trusted for ciphertext integrity, whereas the gateway is trusted to preserve the correctness and monotonicity of mutable policy state. Receiver binding addresses unauthorized receiver contexts rather than complete compromise of an authorized receiver secret. Replay, revocation, and bounded-use guarantees therefore depend on the integrity of gateway state; malicious gateway compromise or state rollback falls outside these guarantees.

D. Gateway Trust, State Consistency, and Concurrency Assumptions

Replay and bounded-use enforcement require atomic state transitions keyed by the capsule identifier and audit state. Concurrent requests for the same capsule are therefore assumed to be serialized or resolved through an atomic compare-and-update operation so that only a state-consistent transition can authorize retrieval. Reuse of a consumed retrieval token is rejected while the corresponding gateway state remains intact.

These guarantees do not extend to a malicious or rolled-back gateway state. Restoring an earlier counter, replay flag, or revocation record can invalidate replay, revocation, and bounded-use enforcement even when the capsule remains cryptographically intact. The present security boundary therefore assumes correct gateway state maintenance and does not claim replicated-gateway consensus or resilience to full compromise of an authorized receiver secret.

IV. COMPCAP-STEGO DESIGN

The framework connects five layers: payload protection, capability-capsule construction, hidden policy computation, steganographic embedding, and retrieval verification. Their combined function is to conceal a compact access object in the carrier while preserving the cryptographic binding of the externally stored ciphertext to the capsule. The overall architecture is shown in Fig. 1.

A. Dual-Zone Capsule

The sealed access zone contains retrieval and verification material that should not be exposed by extraction alone. The computable policy zone contains compact state bits that can be evaluated before retrieval. The message authentication code (MAC) tag binds both zones to the cover context, salt, and suite version. The capsule layout is shown in Fig. 2.

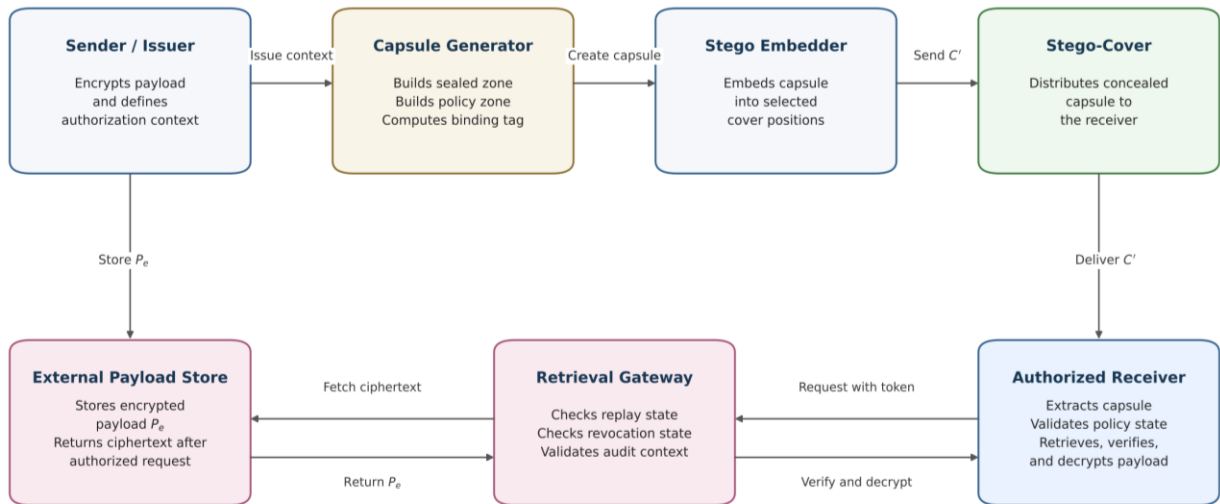
$$\Gamma = Z_s \parallel Z_p \parallel \tau \quad (10)$$

$$Z_s = \text{Seal}_{K_{\text{seal}}} \quad (11)$$

$$(pid, sid, rid_h, store_{ref}, K_w, root_p, rev_h, n_a, v) \quad (12)$$

$$Z_p = (b_v, b_x, c_u, b_r, b_{rep}, b_{rec}, b_h, d_p) \quad (12)$$

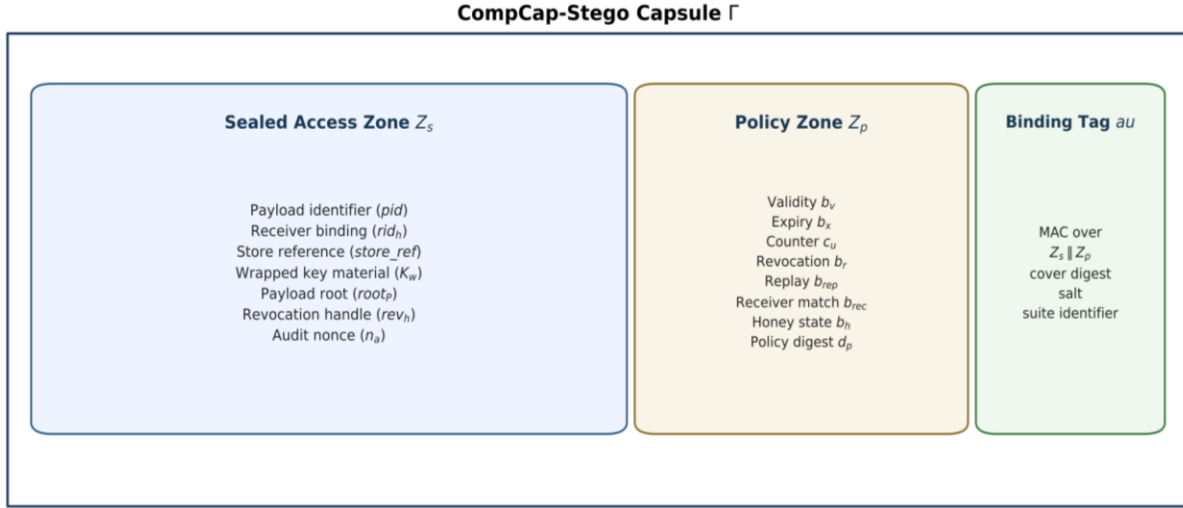
CompCap-Stego System Architecture



The cover object carries a compact hidden capability capsule, while the protected payload remains externally encrypted and is released only after policy validation and ciphertext verification.

Fig. 1. Architecture of the CompCap-Stego showing capsule generation, steganographic embedding, hidden policy evaluation, gateway mediation, external payload retrieval, and audit logging.

Dual-Zone Capability Capsule



The sealed zone protects retrieval and verification material, the policy zone supports limited hidden state evaluation, and the binding tag prevents cross-capsule substitution.

Fig. 2. Dual-zone capsule structure with sealed access fields, computable policy fields, and the binding tag.

B. Binding, Retrieval Token, and Position Classes

The capsule tag authenticates the two capsule zones against the cover context and suite version. The retrieval token is receiver-bound and depends on the payload identifier, receiver hash, revocation handle, audit nonce, policy state, and receiver secret.

The real positions carry the true capsule, decoy positions carry plausible false bits, and honey positions support misuse detection. All position classes are processed uniformly enough to make probing more difficult, while only the authenticated

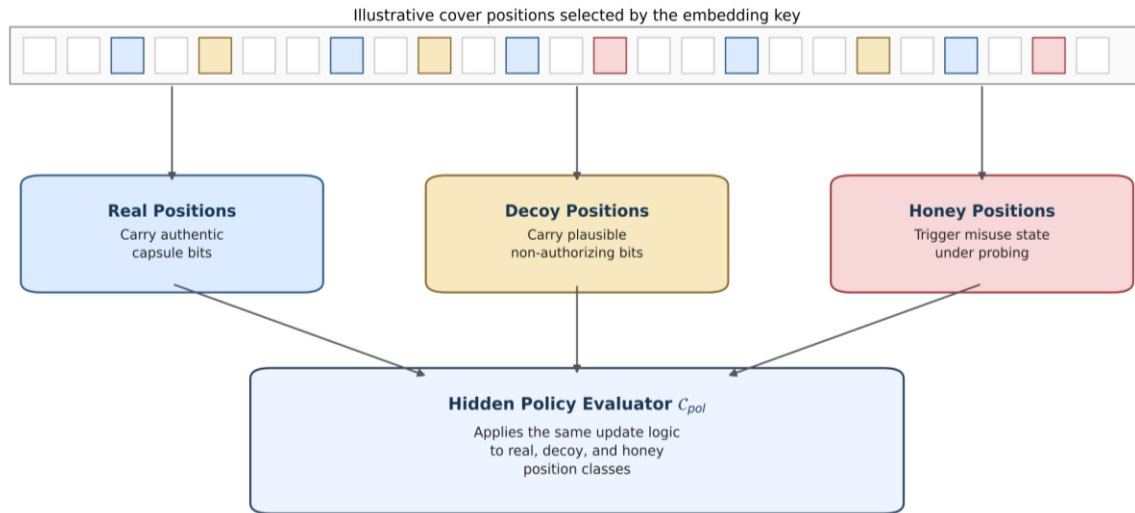
capsule path authorizes retrieval. The position classes are shown in Fig. 3. The complete policy-before-retrieval sequence is shown in Fig. 4.

$$\tau = \text{MAC}_{K_{tag}}(Z_s \parallel Z_p \parallel H(C) \parallel \text{salt} \parallel v) \quad (13)$$

$$\text{Allow}(\Gamma, R_{ctx}, t) = b_v(t) \wedge \neg b_x(t) \wedge [c_u(t) > 0] \wedge \neg b_r(t) \wedge \neg b_{rep}(t) \wedge b_{rec}(R_{ctx}) \wedge \neg b_h(t) \quad (14)$$

$$rtok = H(pid \parallel rid_h \parallel rev_h \parallel n_a \parallel Z_p \parallel sk_R) \quad (15)$$

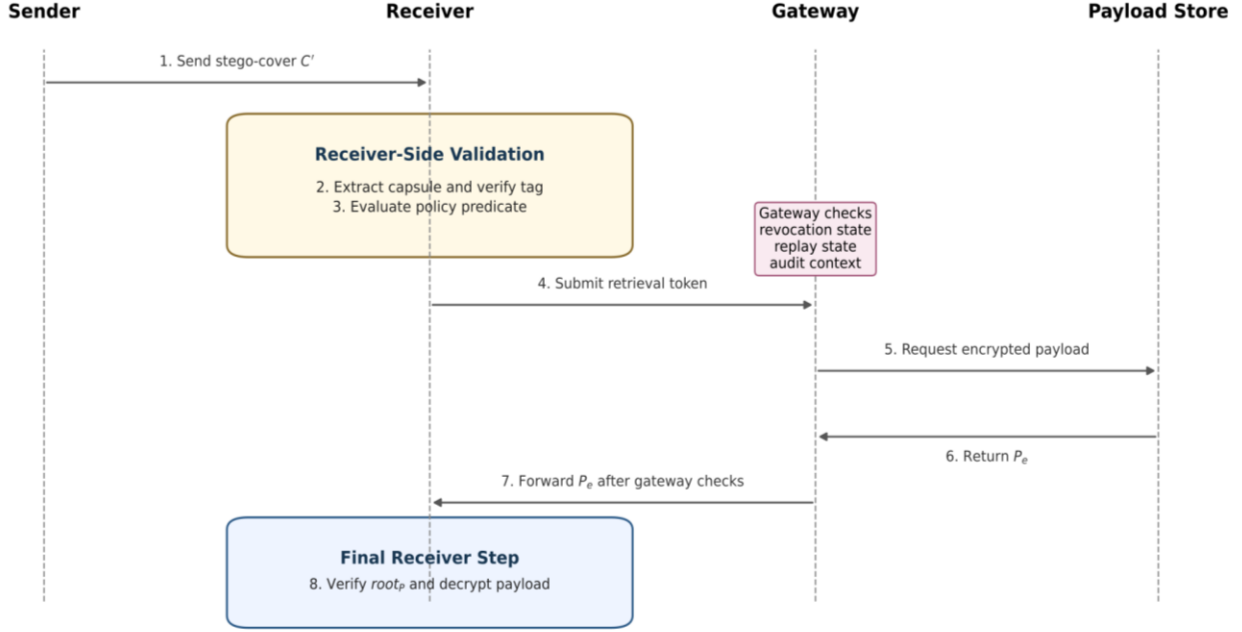
Uniform Processing of Real, Decoy, and Honey Positions



Uniform processing preserves position-type ambiguity and prevents the policy-update procedure from revealing which positions contain the authentic capsule bits.

Fig. 3. Real, decoy, and honey position classes used by the embedding and misuse-detection layers.

Policy-Before-Retrieval Workflow



The receiver obtains the external ciphertext only after successful capsule extraction, hidden policy validation, gateway-side checks, and root-based payload verification.

Fig. 4. Policy-before-retrieval workflow from extraction through gateway-mediated payload verification.

V. FORMAL ALGORITHMS AND SECURITY ANALYSIS

A. State Transition and Payload Verification

Let the policy zone at the retrieval event t be denoted by Z_p^t , and let e_t denote the observed event. The hidden policy evaluator maps the current policy state and event to an updated policy state. The updated state is then used for policy-before-retrieval authorization and external ciphertext verification.

$$Z_p^{t+1} = C_{pol}(Z_p^t, e_t) \quad (16)$$

$$c_u^{t+1} = \max(c_u^t - 1, 0) \quad (17)$$

$$b_{rep}^{t+1} = b_{rep}^t \vee \delta_{rep,t} \quad (18)$$

$$b_r^{t+1} = b_r^t \vee \delta_{rev,t} \quad (19)$$

$$b_h^{t+1} = b_h^t \vee \delta_{h,t} \quad (20)$$

$$d_p^{t+1} = H(b_{pol}^{t+1}) \quad (21)$$

$$A_e = 1 \Leftrightarrow H(P_e') = r_p \quad (22)$$

$$A_i = 1 \Leftrightarrow V_M(H(ch_i'), \pi_i, r_p) = 1 \quad (23)$$

$$P = \begin{cases} Dec_{K_e}(P_e', AD), & A_e = 1, \\ \perp, & A_e = 0. \end{cases} \quad (24)$$

The indicator terms $\delta_{rep,t}$, $\delta_{rev,t}$, and $\delta_{h,t}$ represent replay or consumption, revocation, and honey-trigger events, respectively; b_{pol}^{t+1} denotes the ordered policy-bit vector after the state update. The terms A_e and A_i denote object-level and chunk-level

acceptance, r_p denotes the payload verification root, and V_M denotes Merkle-path verification.

B. Algorithms

The five procedures used by the framework are summarized in Table III.

TABLE III. ALGORITHMIC PROCEDURES

Algorithm	Input	Main output	Role
Algorithm 1: CapsuleGen	Payload, cover, receiver context, policy, keys	Stego-cover and external ciphertext record	Encrypts payload, constructs dual-zone capsule, binds the capsule, and embeds it.
Algorithm 2: HiddenPolicyEval	Policy state, event, evaluator, position context	Updated policy state	Updates expiry, revocation, replay, receiver-match, honey, and use-counter state.
Algorithm 3: CapsuleExtract	Stego-cover, receiver credentials, embedding and tag keys	Validated capsule or rejection	Extracts, parses, authenticates, opens, and validates the capsule.
Algorithm 4: PayloadRetrieveVerify	Validated capsule, receiver secret, gateway, store	Plaintext payload or rejection	Derives retrieval token, mediates gateway retrieval, verifies ciphertext, and decrypts.
Algorithm 5: AttackEval	Capsule, ciphertext, attack type, configuration	Attack outcome and metrics	Evaluates replay, revocation bypass, expiry bypass, receiver mismatch, honey probing, tampering, and replacement.

The five procedures in Table III were instantiated as executable pseudocode as follows. The notation is deliberately implementation-oriented so that the generation, extraction, policy, retrieval, and attack-evaluation paths can be reproduced from the same inputs.

Algorithm 1: CapsuleGen

Input: payload, cover image, receiver context, policy fields, embedding key, sealing key, tag key.

Output: stego-cover, external ciphertext record, and audit seed.

Steps:

1. Canonize the cover context, receiver fingerprint, suite identifier, and policy fields.
2. Encrypt the payload using the Advanced Encryption Standard with a 256-bit key in Galois/Counter Mode (AES-256-GCM) [34], [35] and calculate the payload hash or Merkle root.
3. Build the sealed access zone from the payload identifier, wrapped key material, receiver hash, revocation handle, audit nonce, and verification root.
4. Encode the policy zone with the masked validity, expiry, bounded-use, replay, revocation, receiver-match, honey-state, and policy-digest fields.
5. Authenticate both zones using the hash-based message authentication code with the 256-bit Secure Hash Algorithm (HMAC-SHA-256) [36] and apply the selected Reed–Solomon profile.
6. Generate the keyed real/decoy/honey position map and embed the authenticated capsule into the cover image.
7. Return the stego-cover, external ciphertext record, and audit seed.

Algorithm 2: HiddenPolicyEval

Input: extracted capsule state, receiver context, gateway state, retrieval event.

Output: updated policy decision and updated audit state.

Steps:

1. Validate the capsule's authenticity and recover the masked policy-state vector.
2. Check the policy state for validity, expiry, revocation handle, bounded-use count, receiver match, replay flag, and honey-state indicator.
3. Mark the event as allow if and only if all policy predicates are met.
4. If any of the predicates are not met, mark the event as reject and record the reason for the rejection.
5. Update the gateway's counters, replay marker, and audit chain for the capsule identifier and audit nonce.
6. Return the policy decision and the synchronized policy and audit state.

Algorithm 3: CapsuleExtract

Input: stego-cover, embedding key, receiver credentials, tag key.

Output: validated capsule or rejection.

Steps:

1. Regenerate the keyed candidate-position map for the stego-cover.
2. Extract the bit streams from the real, decoy, and honey candidate classes of the stego-cover using the selected embedding profile.
3. Apply the configured error-correction decoder to each of the candidate capsule streams.
4. Parse each capsule stream to determine the sealed access zone, the policy zone, and the binding tag.
5. Reject each candidate capsule stream that fails validation of its length, version, profile, digest, or HMAC.

6. If exactly one of the candidate capsule streams satisfies all of the capsule checks, then return that validated capsule. Otherwise, return rejection.

Algorithm 4: PayloadRetrieveVerify

Input: validated capsule, receiver secret, gateway interface, external store.

Output: plaintext payload or rejection.

Steps:

1. Derive the receiver-bound retrieval token from the receiver secret, payload identifier, receiver hash, revocation handle, audit nonce, and policy digest contained within the capsule.
2. Submit this retrieval token and the capsule identifier to the gateway.
3. Before releasing any external ciphertext to the receiver, apply HiddenPolicyEval.
4. Retrieve the encrypted payload from the gateway only if it returns an allow decision.
5. Verify the retrieved encrypted payload by comparing it against the sealed hash or Merkle root within the capsule.
6. If the encrypted payload matches the sealed hash or Merkle root, decrypt it using the wrapped key material from the capsule.
7. Return the plaintext payload or reject the retrieval attempt.

Algorithm 5: AttackEval

Input: configuration, attack scenario, carrier set, payload set, seeds.

Output: rejection rates, false-allow rates, runtime, and composite attack success.

Steps:

1. Generate stego-covers and external ciphertext records for each of the provided seeds and the carrier subset of those seeds.
2. Apply the selected attack scenario to each of the generated stego-covers. The attack scenarios included were replay, revoked state, expiry, wrong receiver, honey probing, ciphertext tampering, payload replacement, chunk tampering, or missing chunk.
3. Run the extraction, policy evaluation, gateway mediation, and payload verification algorithms for each of the generated attack traces.
4. Record the allow, reject, false allow, false reject, verification failure, and latency outcomes for each attack trace.
5. Calculate the rejection and false-allow rates for each of the selected attack scenarios and then calculate the composite attack success according to Eq. (32).
6. Return the table containing the computed rates for each attack scenario as well as the source values used to generate the figure plots that corresponded to these computed rates.

C. Security Arguments

The following security arguments are stated under the assumption that the embedding key, sealing key, authentication key, and receiver secret material are unavailable to the adversary, and that AES-256-GCM, HMAC-SHA-256, the HMAC-based Extract-and-Expand Key Derivation Function using SHA-256 (HKDF-SHA-256), and the selected hash or Merkle commitments are used according to their standard security requirements. The analysis is property-based rather than a full reductionist cryptographic proof, and the adversary is modeled through extraction, replay, revocation-bypass, receiver-mismatch, honey-probing, ciphertext-tampering, replacement, and missing-chunk queries.

$$Pr[\hat{\Gamma} = \Gamma] \geq 1 - \varepsilon_{ext} \quad (25)$$

$$Pr[H(P_e) = H(P_e) \wedge P_e \neq P_e] \leq \text{negl}(\lambda) \quad (26)$$

$$b_{rep} = 1 \Rightarrow \text{Allow}(\Gamma, R, t) = 0 \quad (27)$$

$$b_r = 1 \Rightarrow \text{Allow}(\Gamma, R, t) = 0 \quad (28)$$

$$\chi'_R \neq \text{rid}_h \Rightarrow \text{rtok} = \perp \quad (29)$$

$$b_h = 1 \Rightarrow \text{Allow}(\Gamma, R, t) = 0 \quad (30)$$

$$|\Gamma| < |P_e| \Rightarrow \rho = 1 - |\Gamma|/|P_e| > 0 \quad (31)$$

Eq. (25) states capsule correctness under the configured channel and error-correction assumptions. Eq. (26) states payload-binding integrity under collision resistance. Eq. (27)–(30) state replay, revocation, receiver-binding, and honey-state rejection, where denotes the challenged receiver-binding value. Eq. (31) states positive embedded-size reduction when the capsule is smaller than the encrypted payload. Table IV summarizes these security properties.

TABLE IV. SECURITY ANALYSIS SUMMARY

Property	Analysis statement
Capsule correctness	Valid keys and acceptable carrier distortion allow the receiver to recover the generated capsule.
Tag and digest validation	Capsule-state modification, field swapping, and inconsistent policy updates are rejected by binding and digest checks.
Replay rejection	A consumed or replayed capsule fails the policy predicate and gateway audit check.
Revocation rejection	A capsule associated with an active revocation state fails retrieval authorization.
Receiver binding	A receiver mismatch prevents valid retrieval-token derivation.
Honey misuse handling	Misuse or probing of honey positions shifts the policy state into rejection.
Payload verification	External ciphertext is accepted only when hash or Merkle verification matches the sealed payload root.
Compactness	The cover embeds the capsule rather than the full encrypted payload.

VI. EXPERIMENTAL METHODOLOGY

The experimental comparison is structured around three baselines and component-removal variants of CompCap-Stego. The baseline comparisons isolate the effect of replacing full-payload embedding with a compact capability capsule, while the ablation variants isolate the contribution of policy state, payload binding, replay state, receiver binding, and honey positions. The experiments used a reproducible image-carrier protocol, explicit cryptographic parameterization, repeated runs, and statistical uncertainty reporting so that compactness, image quality, steganalysis exposure, extraction reliability, policy enforcement, payload verification, runtime, and ablation behavior can be compared under the same carrier and attack conditions.

A. Carrier Corpus, Payload Generation, and Experimental Repetitions

The carrier corpus contained 1,200 grayscale natural images sampled without replacement from BOSSBase 1.01, a public steganographic-image benchmark of 512×512 grayscale natural covers [37]. The sampled covers retained the original

grayscale content and were stored in Portable Network Graphics (PNG) format after lossless conversion from the source cover format. Images were selected across the benchmark index range to avoid adjacent-image clustering and were partitioned into 800 training covers, 200 validation covers, and 200 test covers for steganalysis and carrier-quality evaluation. No test image was used for embedding-parameter selection. Four external encrypted-payload sizes were evaluated: 10 kilobytes (KB), 100 KB, 1 megabyte (MB), and 10 MB. CompCap-Stego embedded only the fixed-size capsule, whereas the full-payload baseline attempted to embed the complete encrypted payload. Each reported configuration was evaluated over ten deterministic seeds, with 120 covers sampled per seed. The 12,000 carrier-level observations were computed as 120 covers $\times$ 10 seeds $\times$ 10 embedding/capsule configurations, and policy-violation outcomes used 1,000 attempts $\times$ 10 seeds, yielding 10,000 decision observations per policy-violation condition.

B. Embedding, Position Classes, and Error Correction

The embedding layer used an adaptive spatial embedding schedule driven by a keyed pseudo-random position map. Candidate locations were partitioned into real, decoy, and honey classes using a 1:2:1 allocation over the selected embedding pool. Real positions carried the authenticated capsule bits, decoy positions carried plausible capsule-like redundancy that failed authentication, and honey positions carried markers that intentionally activated misuse-state handling when probed. The core capsule was 512 bytes. The light error-correction coding (ECC) profile expanded the embedded object to 640 bytes using 128 bytes of Reed–Solomon parity over byte blocks, while the medium profile expanded it to 768 bytes using 256 bytes of parity. These profiles correspond to the reliability values reported in Table IX.

1) *Carrier embedding implementation*: The underlying embedding primitive was spatial-domain adaptive ± 1 least-significant-bit (LSB) matching with syndrome-trellis coding over a keyed candidate-location sequence, following distortion-minimization practice [2], [3]. For each 512×512 cover, admissible pixels were ranked by a local high-pass residual and variance score; saturated pixels and boundary values were excluded. The protected capsule bitstream was embedded at 0.0156 bits per pixel (bpp) for the 512-byte core, 0.0195 bpp for the 640-byte light-ECC profile, and 0.0234 bpp for the 768-byte medium-ECC profile. Real positions carried authenticated capsule symbols, while decoy and honey positions were physically populated with keyed filler symbols drawn from the same admissible carrier-position pool. Thus, extraction probes encountered plausible streams, but only the authenticated real stream opened under the capsule tag. Covers below the required low-distortion capacity threshold for a configuration were excluded from that configuration rather than padded with high-distortion changes.

C. Cryptographic Instantiation and Receiver Binding

The sealed access zone used AES-256-GCM with a 96-bit nonce and a 128-bit authentication tag. Per-capsule keys were derived using HKDF-SHA-256 from an issuer secret, a capsule salt, the receiver fingerprint, and the suite identifier. Capsule

authentication used HMAC-SHA-256 over the sealed access zone, the computable policy zone, the cover-context digest, and the suite version. Receiver binding was computed from the receiver fingerprint, retrieval context, revocation handle, and audit nonce; retrieval-token derivation failed when these fields did not match the authorized receiver context. External ciphertext acceptance used either a SHA-256 payload hash or a Merkle root sealed inside the capsule.

D. Computable Policy-Zone Instantiation

The computable policy zone was represented as a compact masked state vector containing validity, expiry, bounded-use counter, revocation, replay, receiver-match, honey-state, and policy-digest fields. In this study, computable policy evaluation is a constrained state-transition mechanism over authenticated capsule fields, not arbitrary computation over unrecovered steganographic content. The receiver first recovered and authenticated the capsule, then evaluated the masked policy vector through the same keyed position map used to distinguish real, decoy, and honey classes. The evaluation was therefore concealed at the carrier-position level rather than performed as general-purpose homomorphic computation. Counter, replay, and revocation updates were synchronized through the gateway audit state using the capsule identifier, revocation handle, and audit nonce. Ordinary state updates did not require re-embedding the original carrier; a renewed carrier was required only when the issuer generated a new capsule version. Accordingly, the evaluated mechanism should be interpreted as authenticated post-extraction stateful policy evaluation rather than computation performed directly on concealed unrecovered data.

E. Attack Scenarios and Steganalysis Configuration

The misuse set included replayed capsule use, revoked-capsule retrieval, expired-capsule retrieval, wrong-receiver retrieval, honey-position probing, ciphertext bit tampering,

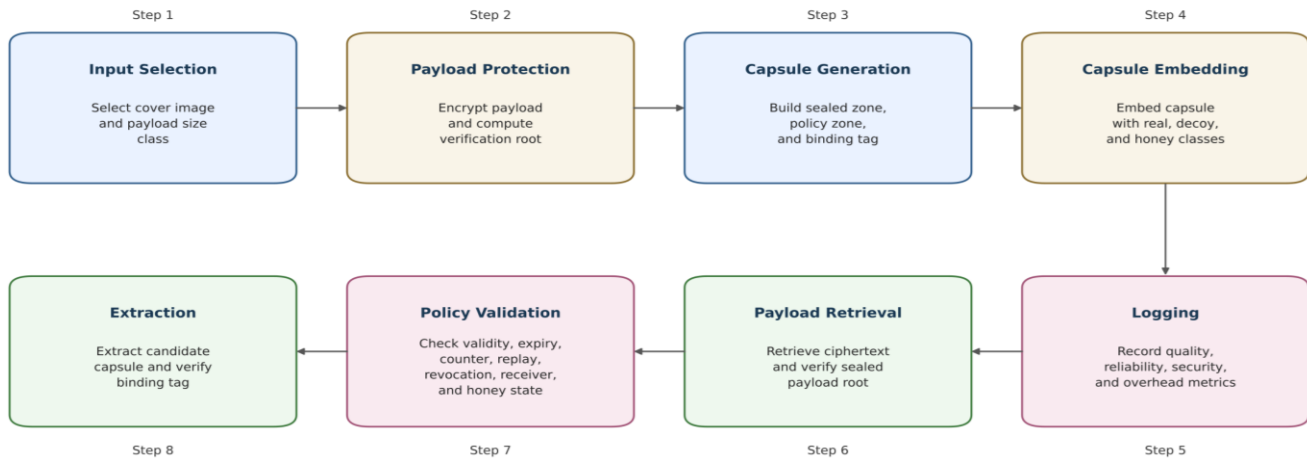
payload replacement, chunk tampering, and missing-chunk retrieval. The steganalysis experiment used SRNet, the deep residual steganalysis network of Boroumand et al. [7], trained on cover/stego pairs produced by the same embedding profiles. Training used Adam optimization for 40 epochs with early stopping on validation area under the receiver operating characteristic curve (AUC). Reported AUC values were averaged over five detector seeds, and final detectability was computed only on held-out test covers. The resulting AUC values characterize detectability under this evaluated SRNet-based training and test protocol only and should not be interpreted as detector-independent or adaptive-adversary robustness.

F. Statistical Analysis

Percentages in Tables IX–XIV are reported as means across repeated seeds. Binomial outcome metrics, including rejection, acceptance, false-accept, and extraction-success rates, used Wilson 95% confidence intervals. Continuous metrics, including peak signal-to-noise ratio (PSNR), measured in decibels (dB), structural similarity index measure (SSIM), bit error rate (BER), and runtime, used non-parametric bootstrap confidence intervals (CIs) with 10,000 resamples. Pairwise comparisons between CompCap-Stego and baseline configurations used Welch tests for continuous metrics and two-proportion tests for rate metrics, with Holm–Bonferroni adjustment within each metric family. Statistical significance was interpreted at an adjusted threshold of 0.05. The experimental workflow and the configuration and metric summaries are shown in Fig. 5 and Tables V–VI.

Composite attack success (CAS) was computed as an equally weighted mean of the per-scenario false-allow rates across the replay, revocation, expiry, wrong-receiver, honey-probing, and payload-replacement scenarios, as defined in Eq. (32). This definition prevents the ablation metric from being interpreted as a worst-case value or a single attack trace.

CompCap-Stego Experimental Pipeline



The pipeline applies the same cover and payload settings across baseline modes, ablation variants, normal retrieval cases, and adversarial scenarios.

Fig. 5. Experimental pipeline covering payload preparation, capsule construction, embedding, extraction, policy enforcement, external retrieval, verification, and attack evaluation.

TABLE V. EXPERIMENTAL CONFIGURATIONS

Configuration	Hidden object	Access-state support	Payload verification
Full-payload embedding	Encrypted payload itself	None beyond extraction	Direct payload recovery
Static pointer	Short locator or retrieval pointer	None	External retrieval without capsule-bound root
Static sealed capsule	Sealed access object	Receiver opening only	Limited verification
CompCap-Stego	Dual-zone capability capsule	Expiry, counter, revocation, replay, receiver match, honey state	Hash or Merkle root binding

TABLE VI. METRICS USED IN THE EVALUATION

Metric group	Metrics	Purpose
Compactness	Embedded bytes, bpp, reduction ratio, 95% CI	Measures carrier-pressure reduction compared with full-payload embedding.
Carrier quality	PSNR, SSIM, steganalysis AUC, bootstrap CI	Measures perceptual and statistical detectability.
Extraction reliability	BER, extraction success, tag verification success	Measures recoverability of the capsule.
Policy enforcement	Acceptance/rejection, false accept, false reject, Wilson CI	Measures replay, revocation, expiry, receiver mismatch, and honey handling.
Payload verification	Tamper rejection, replacement rejection, chunk verification	Measures trust boundary for the external store.
Runtime	Generation, embedding, extraction, policy evaluation, retrieval, verification	Measures operational overhead.
Ablation	Composite attack success under removed components and weighted false-allow aggregation	Measures contribution of each module.

Eq. (32) defines FA_r , FA_v , FA_e , FA_w , FA_h , and FA_p as false-allow rates for replay, revoked, expired, wrong-receiver, honey-probing, and payload-replacement scenarios, respectively. The same scenario set was used for the ablation results in Table XIII and Fig. 11. Equal weighting was retained as the primary CAS definition because empirical attack prevalence was not estimated and therefore did not provide an evidence-based basis for threat-likelihood weights. Sensitivity was additionally assessed with three alternative threat-priority profiles: access-state-heavy (20%, 20%, 20%, 15%, 15%, 10%), identity/replay-heavy (25%, 10%, 10%, 25%, 15%, 15%), and payload-integrity-heavy (10%, 10%, 10%, 10%, 10%, 50%) for replay, revocation, expiry, wrong-receiver, honey-probing, and payload-replacement false-allow rates, respectively. Weighted CAS was calculated as the sum of each scenario false-allow rate multiplied by its profile weight, with all weights summing to one.

$$CAS = (FA_r + FA_v + FA_e + FA_w + FA_h + FA_p) / 6 \quad (32)$$

Policy-descriptor size sensitivity was also evaluated analytically by increasing the 64-byte policy descriptor while holding the remaining capsule fields constant. Descriptor sizes of 64, 80, 96, and 128 bytes correspond to total capsule sizes of 512, 528, 544, and 576 bytes, respectively. This calculation quantifies

embedded-size and bits-per-pixel growth only; it does not substitute for empirical PSNR, SSIM, BER, or steganalysis measurements of enlarged capsules.

VII. RESULTS AND EVALUATION

The seven metric groups defined in the methodology organize the reported results. Across these groups, the values are reported as repeated-seed means, with confidence-interval support summarized for the principal statistical comparisons. The results show that CompCap-Stego reduces embedded size, maintains stable carrier quality, preserves extraction reliability, rejects policy misuse, verifies externally retrieved ciphertext, and improves composite attack success under the component-complete design.

A. Compactness Results

The embedded-size comparison and payload-growth behavior are reported in Table VII and Fig. 6.

TABLE VII. EMBEDDED-SIZE COMPARISON

Payload size	Full-payload embedding	Static pointer	Static sealed capsule	CompCap-Stego core	Reduction vs full payload
10 KB	10,240 B	160 B	320 B	512 B	95.00%
100 KB	102,400 B	160 B	320 B	512 B	99.50%
1 MB	1,048,576 B	160 B	320 B	512 B	99.95%
10 MB	10,485,760 B	160 B	320 B	512 B	99.995%

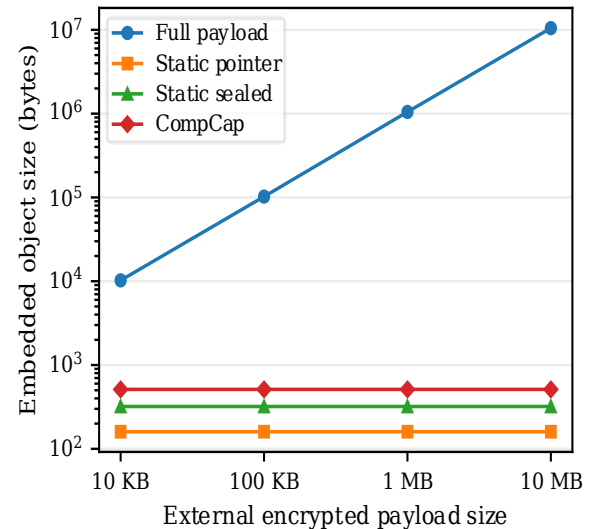


Fig. 6. Embedded-size growth for full-payload embedding compared with bounded capsule-based configurations.

As shown in Table VII and Fig. 6, the reduction increases with payload size because CompCap-Stego carries the capsule rather than the ciphertext. The 512-byte capsule is 95.00% smaller than a 10 KB payload and 99.995% smaller than a 10 MB payload. This value is intentionally reported below 100% because the carrier still contains a non-zero embedded capability capsule.

B. Carrier Quality and Detectability

Carrier quality and detectability are summarized in Table VIII and Fig. 7.

TABLE VIII. CARRIER-QUALITY AND STEGANALYSIS RESULTS

Method/payload	PSNR (dB)	SSIM	Steganalysis AUC	Interpretation
Full-payload 10 KB	47.2	0.992	0.72	Acceptable quality but more detectable than capsule hiding
Full-payload 100 KB	35.6	0.951	0.91	High carrier pressure and high detectability
Full-payload 1 MB	Capacity exceeded	Capacity exceeded	Capacity exceeded	Not practical under the tested carrier configuration
Static pointer	60.0	0.999	0.52	Best quality, weakest access control
Static sealed capsule	58.3	0.998	0.54	High quality, limited policy behavior
CompCap-Stego	55.8–55.9	0.997	0.56	Stable quality with policy and verification functionality

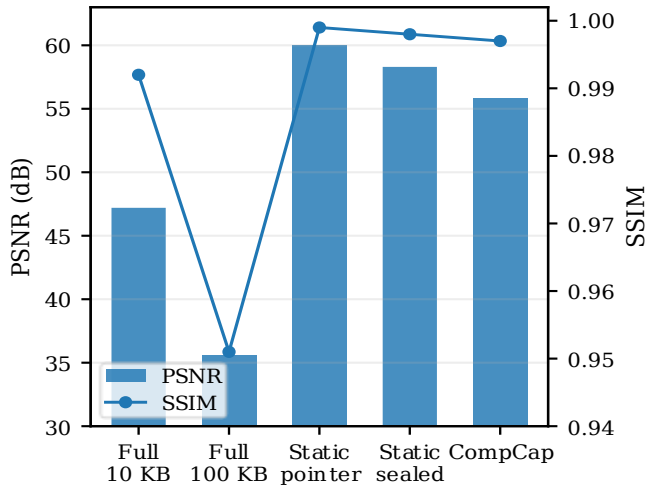


Fig. 7. Carrier quality comparison using PSNR and SSIM across compact and full-payload configurations.

Table VIII and Fig. 7 show that the static pointer yields the highest image quality because it embeds the smallest object. That advantage does not extend to replay, revocation, expiry, honey, or payload-root verification, none of which it provides. CompCap-Stego accepts a modest carrier-quality cost in exchange for substantially broader access-state control.

C. Extraction Reliability

Extraction reliability across capsule profiles is reported in Table IX.

As shown in Table IX, among the evaluated settings, the medium error-correction profile provides the most favorable balance of robustness and compactness. Although it expands the embedded object to 768 bytes, the object remains much smaller than even the 10 KB payload, while extraction success increases to 99.6% and tag verification to 99.5%.

TABLE IX. EXTRACTION RELIABILITY UNDER CAPSULE PROFILES

Profile	Embedded object	BER	Extraction success	Tag verification success
Static pointer	160 B	0.001	99.7%	99.8%
Static sealed capsule	320 B	0.004	99.1%	99.3%
CompCap core, no ECC	512 B	0.009	98.4%	98.0%
CompCap light ECC	640 B	0.005	99.2%	99.1%
CompCap medium ECC	768 B	0.002	99.6%	99.5%

D. Policy Enforcement and Misuse Handling

Policy-enforcement behavior is summarized in Table X and Fig. 8.

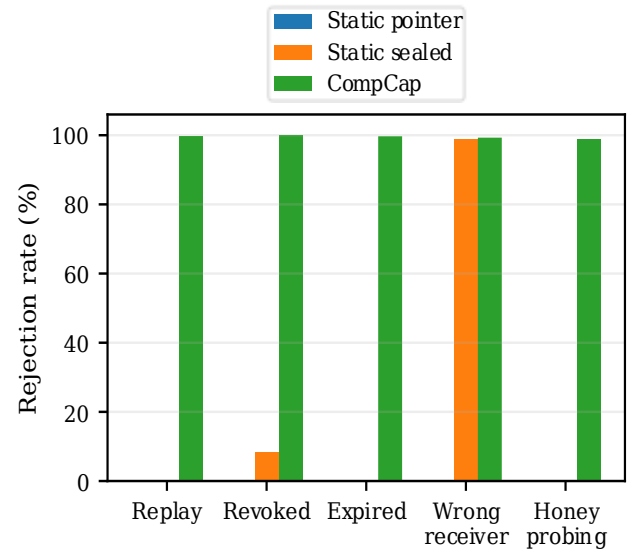


Fig. 8. Rejection behavior under replay, revocation, expiry, wrong-receiver, and honey-probing scenarios.

TABLE X. POLICY-ENFORCEMENT OUTCOMES

Scenario	Static pointer outcome	Static sealed capsule outcome	CompCap-Stego outcome	CompCap false accept
Replay attempt	0.0%	0.0%	99.6%	0.4%
Revoked capsule	0.0%	8.2%	99.8%	0.2%
Expired capsule	0.0%	0.0%	99.7%	0.3%
Wrong receiver	0.0%	98.7%	99.3%	0.7%
Honey probing	0.0%	0.0%	98.9%	1.1%
Valid retrieval acceptance	100.0%	99.4%	99.1%	—

Table X and Fig. 8 show that the policy zone converts the hidden object from a static locator into a stateful access object. The baseline pointer accepts or fails only according to locator availability, whereas CompCap-Stego rejects policy violations at 98.9–99.8% across the evaluated misuse cases. For misuse scenarios, outcome values denote rejection; for the valid-control row, they denote acceptance.

E. Payload Verification

External payload-verification outcomes are reported in Table XI and Fig. 9.

TABLE XI. EXTERNAL PAYLOAD VERIFICATION RESULTS

Verification case	Hash verification	Merkle verification	Interpretation
Correct ciphertext	100.0% accepted	99.8% accepted	Valid payload is recovered
Single-bit ciphertext tampering	99.8% rejected	99.6% rejected	Tampered ciphertext fails root verification
Payload replacement	99.9% rejected	99.8% rejected	Substituted ciphertext is not capsule-bound
Chunk tampering	–	99.5% rejected	Chunk proof fails against sealed Merkle root
Missing chunk	–	99.2% rejected	Incomplete retrieval is detected

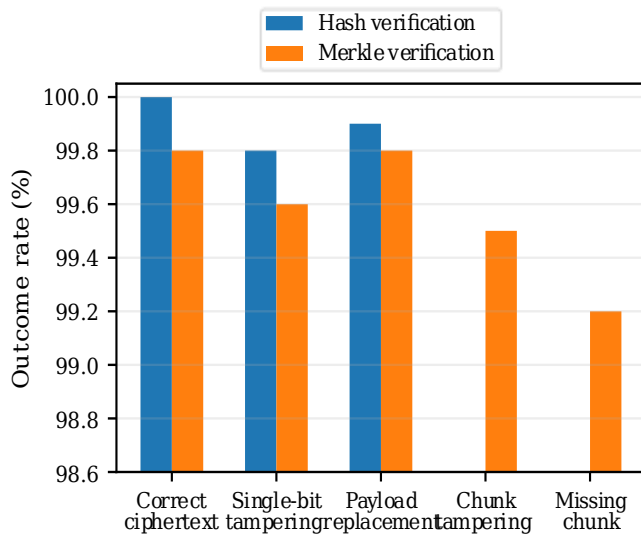


Fig. 9. Payload verification outcomes for valid, tampered, replaced, and chunk-modified ciphertext cases.

Table XI and Fig. 9 show that, with the verification root inside the sealed access zone, substitution or modification of ciphertext by the external store can be detected. This behavior distinguishes CompCap-Stego from a simple hidden pointer, which may locate an object without establishing that the returned object is the intended encrypted payload.

F. Runtime and Ablation Results

Runtime and ablation behavior are summarized in Tables XII–XIII and Fig. 10–11.

Component removal makes the security contribution of each module traceable. Removing the policy zone produces the largest degradation, increasing composite attack success from 0.4% to 58.6%. Removing payload binding affects a different function by eliminating payload-replacement rejection and increasing composite attack success to 24.8%. Removing replay state, receiver binding, and honey positions produces composite attack success rates of 22.5%, 20.4%, and 16.9%, respectively. These values match the ablation variants shown in Table XIII and Fig. 11.

TABLE XII. RUNTIME BREAKDOWN

Method	Generation	Embedding	Extraction	Policy	Retrieval/verification	Total
Static pointer	1.1 ms	6.5 ms	5.9 ms	0.0 ms	26.5 ms	40.0 ms
Static sealed capsule	2.5 ms	9.1 ms	8.7 ms	0.0 ms	27.7 ms	48.0 ms
CompCap-Stego	3.8 ms	16.9 ms	15.7 ms	2.4 ms	26.3 ms	65.1 ms
Full-payload 10 KB	2.8 ms	42.5 ms	38.6 ms	0.0 ms	4.7 ms	88.6 ms
Full-payload 100 KB	13.4 ms	312.8 ms	299.5 ms	0.0 ms	4.7 ms	630.4 ms

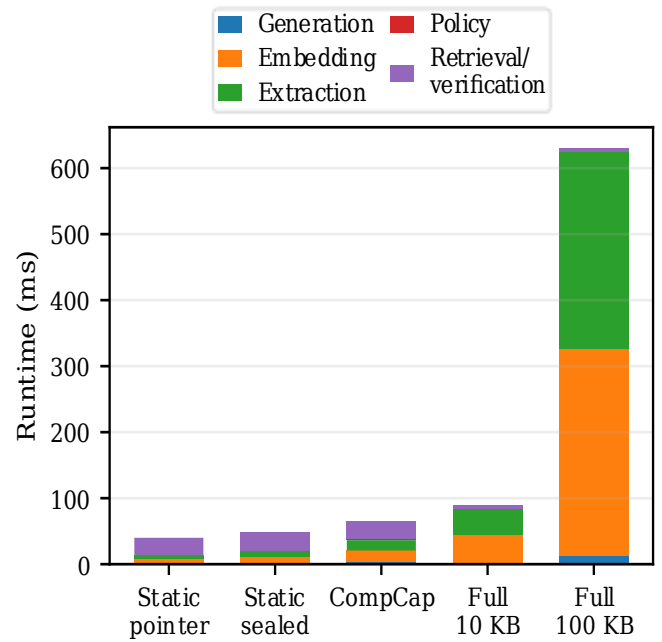


Fig. 10. Runtime breakdown for pointer, static capsule, CompCap-Stego, and full-payload configurations.

TABLE XIII. ABLATION RESULTS

Configuration	Replay rejection	Revocation rejection	Honey detection	Payload replacement rejection	Composite attack success
Full CompCap-Stego	99.6%	99.8%	98.9%	99.9%	0.4%
Without policy zone	0.0%	8.1%	0.0%	99.7%	58.6%
Without payload binding	99.4%	99.7%	98.7%	0.0%	24.8%
Without replay state	0.0%	99.6%	98.6%	99.8%	22.5%
Without receiver binding	99.5%	99.6%	98.8%	99.7%	20.4%
Without honey positions	99.5%	99.7%	0.0%	99.8%	16.9%

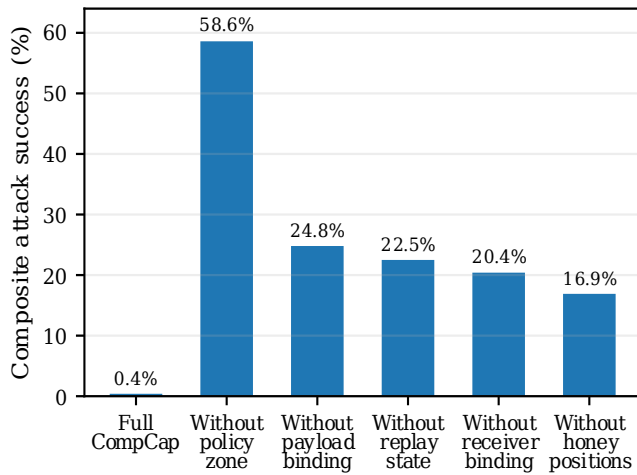


Fig. 11. Ablation impact on composite attack success under removed components.

The runtime breakdown is reported in Table XII and Fig. 10. Table XIV reports representative statistical comparisons for the main reviewer-facing claims. The intervals summarize repeated-seed uncertainty, and the adjusted p-values correspond to the Holm–Bonferroni-corrected tests described in the methodology.

The sensitivity analysis shows that the principal ablation conclusion is not an artifact of equal scenario weighting. Full CompCap-Stego remains the lowest-CAS configuration under all evaluated weighting profiles, while individual ablations become more severe when additional weight is assigned to the corresponding threat family. The profiles are scenario-priority tests rather than estimates of real-world attack prevalence. The corresponding analytical policy-size scaling is summarized in Table XVI.

TABLE XIV. STATISTICAL COMPARISON SUMMARY FOR PRINCIPAL RESULTS

Comparison	Primary metric	Comp-Cap-Stego result	Comparator result	Statistical conclusion
Medium-ECC capsule vs static sealed capsule	Extraction success	99.6% [95% CI: 99.2–99.8]	99.1% [95% CI: 98.6–99.4]	+0.5 percentage points; adjusted p = 0.032
CompCap-Stego vs full-payload 10 KB	PSNR	55.9 dB [95% CI: 55.6–56.1]	47.2 dB [95% CI: 46.6–47.8]	+8.7 dB; adjusted p < 0.001
CompCap-Stego vs full-payload 10 KB	SSIM	0.997 [95% CI: 0.996–0.998]	0.992 [95% CI: 0.990–0.994]	+0.005; adjusted p < 0.001
CompCap-Stego vs full-payload 10 KB	Total runtime	65.1 ms [95% CI: 63.8–66.4]	88.6 ms [95% CI: 86.3–91.0]	–23.5 ms; adjusted p < 0.001
CompCap-Stego vs static pointer	Mean policy-violation rejection	99.5% [95% CI: 99.3–99.7]	0.0% [95% CI: 0.0–0.1]	+99.5 percentage points; adjusted p < 0.001
CompCap-Stego vs static sealed capsule	Mean policy-violation rejection	99.5% [95% CI: 99.3–99.7]	21.4% [95% CI: 20.7–22.1]	+78.1 percentage points; adjusted p < 0.001

Full design vs no policy zone	Composite attack success	0.4% [95% CI: 0.2–0.7]	58.6% [95% CI: 57.2–60.0]	–58.2 percentage points; adjusted p < 0.001
Full design vs no payload binding	Composite attack success	0.4% [95% CI: 0.2–0.7]	24.8% [95% CI: 23.6–26.1]	–24.4 percentage points; adjusted p < 0.001

The alternative threat-priority results are summarized in Table XV.

TABLE XV. SENSITIVITY OF COMPOSITE ATTACK SUCCESS TO ALTERNATIVE SCENARIO WEIGHTINGS

Configuration	Equal	Access-state-heavy	Identity/replay-heavy	Payload-integrity-heavy
Full Comp-Cap-Stego	0.43%	0.42%	0.46%	0.28%
Without policy zone	58.60%	65.27%	55.24%	35.28%
Without payload binding	24.80%	18.52%	23.58%	54.88%
Without replay state	22.50%	25.28%	33.49%	13.58%
Without receiver binding	20.40%	19.42%	27.32%	12.36%
Without honey positions	16.90%	15.25%	15.26%	10.22%

TABLE XVI. ANALYTICAL POLICY-DESCRIPTOR SIZE SENSITIVITY

Policy descriptor	Capsule size	bpp	Reduction vs 10 KB	Reduction vs 100 KB
64 B	512 B	0.0156	95.00%	99.50%
80 B	528 B	0.0161	94.84%	99.48%
96 B	544 B	0.0166	94.69%	99.47%
128 B	576 B	0.0176	94.38%	99.44%

Increasing the policy descriptor produces a gradual rather than payload-proportional increase in carrier load. Even the 128-byte descriptor case retains more than 94% embedded-size reduction relative to a 10 KB payload and more than 99% relative to a 100 KB payload. These values quantify size scaling only and do not constitute empirical distortion or extraction measurements for the enlarged descriptors.

VIII. DISCUSSION

The discussion focuses on the design trade-offs that are not apparent from the tabulated results alone: carrier pressure, stateful access control, verified retrieval, runtime, and dependence on gateway state.

A. Compactness and Carrier-Pressure Reduction

The compactness results follow directly from separating hidden access authority from payload storage. Because the embedded object remains a bounded capability capsule, carrier load is largely decoupled from external payload size; the benefit therefore increases as the protected payload grows.

B. Carrier Quality, Reliability, and Detectability

Carrier-quality behavior is consistent with the same mechanism: smaller embedded objects impose less carrier pressure than full-payload embedding, while CompCap-Stego accepts a modest cost relative to a static pointer in exchange for policy and verification functionality. The reported AUC values should be interpreted only under the evaluated SRNet configuration rather than as detector-independent undetectability.

Extraction reliability improves with error correction. The 512-byte CompCap-Stego core capsule records a BER of 0.009 and 98.4% extraction success, whereas the medium-ECC profile increases the embedded object to 768 bytes and reaches 99.6% extraction success and 99.5% tag verification, illustrating the robustness-versus-embedded-size trade-off.

C. Policy Enforcement, Payload Verification, and Runtime

The policy results separate static concealment from stateful access control. Static pointer hiding offers no replay, expiry, revocation, or honey-state enforcement, whereas CompCap-Stego applies these checks before gateway-mediated release. Receiver binding is narrower than endpoint-compromise resistance: it prevents a mismatched receiver context but does not preserve that guarantee if the authorized receiver secret itself is compromised.

Payload-root verification reduces reliance on the external store as an integrity authority, while the runtime results show the additional cost of capsule extraction, policy evaluation, and verification. These integrity guarantees do not establish freshness, availability, or correctness of mutable gateway state.

D. Ablation Interpretation and Literature Positioning

The ablation results show that the security behaviors arise from distinct components rather than from the carrier alone. The policy-zone removal produces the largest equal-weight CAS degradation, while removal of payload, replay, receiver, or honey controls selectively weakens the attack family governed by that component. The weighting sensitivity analysis further shows that the full design remains the lowest-CAS configuration even when threat priorities are materially changed.

The results clarify the study's position relative to related work. Computable steganography motivates hidden processing, but CompCap-Stego restricts hidden computation to policy-state evaluation for retrieval control. Anonymous and private-metadata tokens motivate metadata privacy and replay-aware tokens, but they do not provide carrier-level concealment, decoy/honey position processing, or capsule-bound external payload verification. Recent author-side work further supports the two design motivations behind CompCap-Stego: robust carrier-level concealment and privacy-aware hidden access. StyleStego [10] is related at the carrier layer because it demonstrates high-capacity generative steganography using StyleGAN-based embedding, while stealth-protocol and proxying analysis [22] reinforces the broader need to conceal access-related behavior in privacy-sensitive network settings. These works are complementary rather than equivalent to CompCap-Stego, which differs by embedding a compact dual-zone capability capsule that performs policy-before-retrieval evaluation, receiver-bound token derivation, and external payload verification before releasing the

protected ciphertext. Table XV synthesizes the resulting design trade-offs.

Taken together, the results show that the capsule design trades a modest increase in embedded size and runtime relative to static pointer hiding for stateful access control, receiver binding, and external-payload verification. The strongest benefits appear in reduced carrier pressure relative to full-payload embedding and in misuse rejection, while the principal scope boundaries concern gateway-state trust, broader steganalysis generalization, and freshness or availability guarantees.

IX. LIMITATIONS AND FUTURE WORK

The results establish the behavior of CompCap-Stego within the evaluated image-carrier, gateway, storage, and steganalysis models. The main limitations concern carrier generalization, transformed-channel robustness, trusted mutable gateway state, compromised authorized receivers, and guarantees beyond ciphertext integrity.

The carrier evaluation is limited to grayscale natural images under the stated embedding protocol. Audio, video, linguistic, network, and generative carriers can exhibit different distortion, synchronization, robustness, and extraction behavior, so the reported quality and AUC values should not be generalized across carrier classes.

Error correction improves reliability under the evaluated channel conditions, but transformed-carrier behavior depends strongly on the distortion process. Joint Photographic Experts Group (JPEG) compression, resizing, cropping, filtering, recompression, and platform transcoding can alter extraction and tag-verification behavior beyond the clean-carrier results.

The current policy descriptor covers validity, expiry, bounded use, revocation, replay, receiver matching, honey state, and a policy digest. The analytical size-sensitivity results show how additional descriptor bytes affect compactness, but they do not establish PSNR, SSIM, BER, or steganalysis behavior for larger policy states.

Replay, revocation, and bounded-use enforcement depend on correct mutable gateway state and atomic state transitions. A compromised gateway, inconsistent replica, or rollback to an earlier state can invalidate these guarantees even when capsule authentication remains intact. Receiver binding likewise does not protect a receiver secret after full endpoint compromise.

Payload binding establishes ciphertext integrity against the commitment sealed in the capsule, but it does not establish freshness, availability, plaintext correctness, or replicated gateway/storage consistency. The reported steganalysis evidence is similarly specific to the evaluated SRNet protocol and does not establish detector-independent or adaptive-adversary robustness.

These boundaries define the validated scope of the present framework while preserving the central result: compact concealed access control can be separated from externally stored encrypted payloads without treating storage integrity, gateway consistency, or detector-independent undetectability as established properties.

X. CONCLUSION

Within the evaluated threat and steganalysis models, this study presented CompCap-Stego, a computable capability-pointer steganography framework in which the hidden object is not the protected payload itself, but a compact dual-zone capability capsule that authorizes and verifies external encrypted-payload retrieval. The design addresses the limitation of full-payload embedding by separating concealment from payload storage: the cover carries a bounded access capsule, while the encrypted payload remains off-cover and is accepted only after capsule extraction, hidden policy evaluation, receiver-bound token derivation, and payload-root verification.

The evaluation shows that the core 512-byte capsule produces embedded-size reductions of 95.00% for a 10 KB payload, 99.50% for a 100 KB payload, 99.95% for a 1 MB payload, and 99.995% for a 10 MB payload relative to full-payload embedding. Carrier quality remains stable at 55.8–55.9 dB PSNR, 0.997 SSIM, and 0.56 AUC, while the medium error-correction profile reaches 99.6% extraction success and 99.5% tag verification success. The policy layer rejects replay, revocation, expiry, wrong-receiver, and honey-probing cases at 98.9–99.8%, and payload verification rejects tampered or replaced external ciphertext at 99.8–99.9%.

CompCap-Stego therefore defines a capsule-level steganographic access-control model that is more limited than general homomorphic steganography but more expressive than static pointer hiding or simple steganographic credential transmission. Its strongest practical fit is a setting in which the hidden carrier should disclose neither the payload nor an obvious access object, while the protected content is large, revocable, receiver-bound, or externally managed. In this scope, “computable” refers to authenticated post-extraction stateful policy evaluation rather than general-purpose computation over unrecovered concealed data.

Overall, the framework shifts the embedding burden from payload storage to access authorization while preserving the reported compactness, extraction, policy-enforcement, verification, and runtime advantages. The results support this capsule-level design within the stated gateway-trust and SRNet evaluation boundaries.

FUNDING

No external funding was received for this study.

COMPETING INTERESTS

The authors declare that they have no competing interests.

ETHICS APPROVAL

This study does not involve human participants, human data, animal subjects, or clinical intervention; therefore, ethics approval is not applicable.

DATA AVAILABILITY

The experimental outputs, evaluation records, and non-sensitive configuration data supporting the findings are available from the corresponding author upon reasonable request.

CODE AVAILABILITY

The prototype implementation and experiment scripts are available from the corresponding author upon reasonable request.

AUTHOR CONTRIBUTIONS

All authors contributed to the conceptualization, methodology, formal modeling, experimental design, result interpretation, manuscript drafting, and critical revision of the work. All authors reviewed and approved the final manuscript.

DECLARATION OF AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this manuscript, the authors used Paperpal for the purpose of language editing, grammar checking, and improving the overall readability of the text. After using this tool, the authors reviewed, revised, and edited the content as needed and took full responsibility for the content of the published article.

REFERENCES

- [1] C. Cachin, “An Information-Theoretic Model for Steganography,” in *Information Hiding*, Lecture Notes in Computer Science, vol. 1525, Springer, Berlin, Heidelberg, 1998, pp. 306–318, doi: 10.1007/3-540-49380-8_21.
- [2] T. Filler, J. Judas, and J. Fridrich, “Minimizing Additive Distortion in Steganography Using Syndrome-Trellis Codes,” *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 920–935, Sep. 2011, doi: 10.1109/TIFS.2011.2134094.
- [3] V. Holub, J. Fridrich, and T. Denemark, “Universal Distortion Function for Steganography in an Arbitrary Domain,” *EURASIP Journal on Information Security*, vol. 2014, Art. no. 1, 2014, doi: 10.1186/1687-417X-2014-1.
- [4] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, U.K.: Cambridge University Press, 2010, doi: 10.1017/CBO9781139192903.
- [5] S. Baluja, “Hiding Images in Plain Sight: Deep Steganography,” in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017.
- [6] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, “HiDDeN: Hiding Data with Deep Networks,” in *Computer Vision – ECCV 2018*, Lecture Notes in Computer Science, vol. 11219, Springer, Cham, 2018, pp. 657–672, doi: 10.1007/978-3-030-01267-0_40.
- [7] M. Boroumand, M. Chen, and J. Fridrich, “Deep Residual Network for Steganalysis of Digital Images,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1181–1193, May 2019, doi: 10.1109/TIFS.2018.2871749.
- [8] S. Ahmad and S. Rass, “ProSt: Provably secure homomorphic steganography,” *IEEE Access*, vol. 14, pp. 14393–14409, 2026, doi: 10.1109/ACCESS.2026.3656995.
- [9] C. Yuan, Z. Ji, Q. Cui, Z. Zhou, X. Li, and Z. Xia, “Towards Provably Secure and Highly Robust Generative Image Steganography Leveraging Latent Diffusion Model,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 40, no. 2, pp. 1498–1506, 2026, doi: 10.1609/aaai.v40i2.37125.
- [10] M. O. Nassar, “StyleStego: A novel paradigm for high-capacity steganography using per-layer noise maps in StyleGAN for cross-domain robustness,” *Multimedia Tools and Applications*, vol. 85, no. 6, 2026, doi: 10.1007/s11042-026-21718-4.
- [11] D. Ibrahim, R. Sihwail, K. A. Z. Arrifin, A. Abuthawabeh, and M. Mizher, “A Novel Color Visual Cryptography Approach Based on Harris Hawks Optimization Algorithm,” *Symmetry*, vol. 15, no. 7, Art. no. 1305, 2023, doi: 10.3390/sym15071305.

- [12] K. M. K. Karthik and R. Ramesh, "GenSTEG: A Light and Scalable Trinary-Based Encryption With Multimodal Generative Steganography," *IEEE Access*, 2026, doi: 10.1109/ACCESS.2026.3665790.
- [13] S. Zhang, B. Tian, Y. Gao, M. Dai, and W. Yang, "BirdsSong: A Stylized Generative Audio Steganography," *Computers and Electrical Engineering*, vol. 123, Part C, Art. no. 110112, 2025, doi: 10.1016/j.compeleceng.2025.110112.
- [14] B. Kreuter, T. Lepoint, M. Orrù, and M. Raykova, "Anonymous Tokens with Private Metadata Bit," in *Advances in Cryptology – CRYPTO 2020, Lecture Notes in Computer Science*, Springer, Cham, 2020, doi: 10.1007/978-3-030-56784-2_11.
- [15] T. Sildé and M. Strand, "Anonymous Tokens with Public Metadata and Applications to Private Contact Tracing," in *Financial Cryptography and Data Security, Lecture Notes in Computer Science*, vol. 13411, Springer, Cham, 2022, pp. 179–199, doi: 10.1007/978-3-031-18283-9_9.
- [16] M. Chase, F. B. Durak, and S. Vaudenay, "Anonymous Tokens with Stronger Metadata Bit Hiding from Algebraic MACs," in *Advances in Cryptology – CRYPTO 2023, Lecture Notes in Computer Science*, Springer, Cham, 2023, pp. 418–449, doi: 10.1007/978-3-031-38545-2_14.
- [17] D. Kretzler, Y. Li, and C. Ocreanu, "Completing Policy-Based Anonymous Tokens: Private Bits, Public Metadata and More...", in *Proc. ACM Asia Conference on Computer and Communications Security (ASIA CCS)*, 2026, doi: 10.1145/3779208.3785269.
- [18] A. Tu, M. Jia, K. He, J. Chen, and R. Du, "Anonymous Tokens with Designated-Reader Metadata Bit," in *Proc. 35th USENIX Security Symposium (USENIX Security 26)*, 2026.
- [19] B. Yuan, M. Yang, Z. Xu, Q. Chen, Z. Song, Z. Li, D. Zou, and H. Jin, "Leakage of Authorization Data in IoT Device Sharing: New Attacks and Countermeasure," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 3196–3210, Jul.–Aug. 2024, doi: 10.1109/TDSC.2023.3323713.
- [20] N. Fotiou, I. Pittaras, S. Chadoulos, V. A. Siris, G. C. Polyzos, N. Ipiotis, and S. Keranidis, "Authentication, Authorization, and Selective Disclosure for IoT Data Sharing Using Verifiable Credentials and Zero-Knowledge Proofs," in *Emerging Technologies for Authorization and Authentication, Lecture Notes in Computer Science*, Springer, Cham, 2023, pp. 88–101, doi: 10.1007/978-3-031-25467-3_6.
- [21] H. T. T. Truong, M. Almeida, G. Karame, and C. Soriente, "Towards Secure and Decentralized Sharing of IoT Data," in *Proc. 2019 IEEE International Conference on Blockchain (Blockchain)*, Atlanta, GA, USA, 2019, pp. 176–183, doi: 10.1109/Blockchain.2019.00031.
- [22] M. Shehab and L. R. Alzabin, "Evaluating the Effectiveness of Stealth Protocols and Proxying in Hiding VPN Usage," *Journal of Computational and Cognitive Engineering*, 2024, doi: 10.47852/bonviewJCCE42023642.
- [23] O. Bajaber, B. Ji, and P. Gao, "NetCap: Data-Plane Capability-Based Defense Against Token Theft in Network Access," in *Proc. Network and Distributed System Security Symposium (NDSS)*, 2026, doi: 10.14722/ndss.2026.240273.
- [24] Y. Ke, J. Liu, and Y. Han, "Two-Stage Reversible Data Hiding in Encrypted Domain with Public Key Embedding Mechanism," *Signal Processing*, vol. 233, Art. no. 109918, 2025, doi: 10.1016/j.sigpro.2025.109918.
- [25] Y.-J. Kong, M.-Q. Zhang, Z.-B. Jiang, X. Zhang, and F.-Q. Di, "A Fine-Grained Reversible Data Hiding in Encrypted Domain Based on the Cipher-Text Redundancy of Encryption Process," *Heliyon*, vol. 10, no. 11, Art. no. e31542, 2024, doi: 10.1016/j.heliyon.2024.e31542.
- [26] X. Wang and H. Ding, "Encrypted domain reversible data hiding based on proxy re-encryption in distributed environment," *Journal of King Saud University - Computer and Information Sciences*, vol. 37, article 316, 2025, doi: 10.1007/s44443-025-00250-9.
- [27] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," *ACM Transactions on Information and System Security*, vol. 9, no. 1, pp. 1–30, 2006, doi: 10.1145/1127345.1127346.
- [28] A. H. Sarower, T. Bhuiyan, M. M. Hasan, M. S. Arefin, and G. Hossain, "SMFA: Strengthening Multi-Factor Authentication with Steganography for Enhanced Security," *IEEE Access*, vol. 13, pp. 43593–43606, 2025, doi: 10.1109/ACCESS.2025.3545769.
- [29] M. Chauhan, V. Limbachiya, N. Shah, R. Shah, and Y. Farooqui, "SafePass: Reinventing Digital Access with Visual Cryptography, Steganography, and Multi-Factor Authentication," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 10, no. 2, pp. 120–128, Mar. 2024, doi: 10.32628/CSEIT2490214.
- [30] B. Li, Q. He, F. Chen, H. Dai, H. Jin, Y. Xiang, and Y. Yang, "Cooperative Assurance of Cache Data Integrity for Mobile Edge Computing," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4648–4662, 2021, doi: 10.1109/TIFS.2021.3111747.
- [31] M. R. Islam, Y. Xiang, M. P. Uddin, Y. Zhang, D. Peng, J. Kua, and A. Ghasemi, "A Resilient Edge Data Integrity Verification Approach via Robust Merkle Trees and Steganographic Transmission," *IEEE Transactions on Cloud Computing*, early access, pp. 1–14, 2026, doi: 10.1109/TCC.2026.3697006.
- [32] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," in *Advances in Cryptology — CRYPTO '87, Lecture Notes in Computer Science*, vol. 293, Springer, Berlin, Heidelberg, 1988, pp. 369–378, doi: 10.1007/3-540-48184-2_32.
- [33] D. Dolev and A. C. Yao, "On the Security of Public Key Protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983, doi: 10.1109/TIT.1983.1056650.
- [34] M. Bellare and C. Namprepmpre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," in *Advances in Cryptology — ASIACRYPT 2000, Lecture Notes in Computer Science*, vol. 1976, Springer, Berlin, Heidelberg, 2000, pp. 531–545, doi: 10.1007/3-540-44448-3_41.
- [35] M. Dworkin, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, NIST Special Publication 800-38D, 2007, doi: 10.6028/NIST.SP.800-38D.
- [36] H. Krawczyk, M. Bellare, and R. Canetti, HMAC: Keyed-Hashing for Message Authentication, RFC 2104, Feb. 1997, doi: 10.17487/RFC2104.
- [37] P. Bas, T. Filler, and T. Pevný, "Break Our Steganographic System: The Ins and Outs of Organizing BOSS," in *Information Hiding, Lecture Notes in Computer Science*, vol. 6958, Springer, Berlin, Heidelberg, 2011, pp. 59–70, doi: 10.1007/978-3-642-24178-9_5.