Resilient Cryptographic Sharing: Defending Against CPA, CCA, and Framing Attacks

Anandarup ROY[1,3] Suprita TALNIKAR[1,2] Kouichi SAKURAI[1]

Abstract: This work explores recent advances in traceable secret sharing and threshold encryption, focusing on traceability, verifiability, and resistance to adversarial manipulation. These protocols are categorized by setup models and entities like tracers and verifiers. The cryptographic tools used rely on one-way functions, with security proofs in oracle-based tracing or silent setup models. These schemes often offer properties like anonymity and deniability, employing universal composability and algebraic tracing. It also highlights comparisons between secret sharing paradigms and threshold encryption, analyzing their resilience to CPA and CCA attacks, including framing attacks, revealing vulnerabilities to computational adversaries. computational power.

Keywords: Traceable Secret Sharing, Threshold Encryption, CPA, CCA

1. Introduction and Foundational Primitives

The increasing use of distributed systems—from global cloud infrastructures to decentralized blockchain networks—has rendered traditional, centrally-trusted security models inadequate. Secret sharing, first introduced by Shamir and Blakley in 1979, allows a secret to be split among n parties such that any t of them can reconstruct it, while fewer than t learn nothing [1], [2]. Threshold cryptography extends this idea to public-key operations: the private key is secret-shared, and decryption or signing occurs only via cooperation of t parties [3]. These primitives eliminate single points of failure and underpin secure multiparty computation (SMPC) and threshold encryption. This paper provides a systematic analysis of a critical set of cryptographic tools developed to address this challenge: secret sharing schemes and threshold cryptosystems endowed with mechanisms for traceability and accountability.

Our Contributions

(1) We analyse and compare existing TSS and threshold encryption schemes with black-box tracing [4], [5].

(2) We propose a unified framework for compartmented access structures and hashed trace keys, enhancing non-imputability and frameproofness.

(3) We outline open problems in collusion-deterrent constructions and post-quantum traceability [6], [7].

1.1 Principles of Secret Sharing Schemes (SSS)

Secret Sharing Schemes (SSS) are a fundamental cryptographic method for distributing a secret among a group of participants in a way that enforces shared control and provides fault tolerance. Initially conceived for securing high-value, static secrets like missile launch codes or numbered bank accounts, their principles have become a cornerstone for more dynamic and complex distributed protocols.

1.1.1 Definition and Core Components

A secret sharing scheme involves a *dealer* who holds a secret, s. The dealer processes this secret through a `Share` algorithm, which splits it into n pieces called *shares*, $(sh_1, sh_2, \ldots, sh_n)$. Each share is then distributed to a distinct participant. The scheme is defined by a threshold parameter, t, where $1 \leq t \leq n$, creating what is known as a (t, n)-threshold scheme. The functionality of such a scheme is governed by two primary properties:

(1) **Correctness (or Reconstructability):** Any group of t or more participants can pool their shares and use a public `Reconstruct` algorithm to perfectly recover the original secret s.

(2) **Security:** Any group of fewer than t participants, even if they collude and share their information, can gain no information whatsoever about the secret s.

This "all-or-nothing" property is what makes secret sharing powerful. It provides a mechanism to store sensitive information without a single point of failure; the loss or compromise of up to $t-1$ shares does not compromise the

[1] Department of Advanced Informatics, Kyushu University, Fukuoka, Japan
[2] Applied Statistics Unit, Indian Statistical Institute, Kolkata, India
[3] Dr. ROY is supported by the National Institute of Information and Communications Technology(NICT), Japan, under the NICT International Invitational Program.

secret's confidentiality, and the loss of up to $n - t$ shares does not compromise its availability.

1.1.2 Security Formulations

The intuitive notion that an insufficient number of shares provides "no information" has been formalized into a rigorous, information-theoretic guarantee known as *perfect security*. A secret sharing scheme is considered perfectly secure if for any two distinct secrets, s_0 and s_1, and any unauthorized set of $t-1$ shares, the probability distribution of those shares is identical regardless of whether the original secret was s_0 or s_1. Formally, for a scheme (`Share`, `Reconstruct`) over a message space M, it is perfectly secure if for all secrets $m, m' \in M$ and for all subsets of indices $S \subseteq \{1, \ldots, n\}$ with $|S| < t$, the following holds: This means that an adversary observing $t-1$ shares learns absolutely nothing that would help them distinguish which secret was shared; their knowledge remains the same as someone with zero shares. This strong, unconditional guarantee, which does not rely on any computational hardness assumptions, is what allows SSS to serve as a secure building block for more complex protocols like Secure Multi-Party Computation (SMPC), where the inputs of participants are shared in this manner to enable joint computation without revealing the private inputs.

1.1.3 Canonical Constructions

Two seminal and conceptually distinct constructions form the basis of most threshold secret sharing schemes.

- **Shamir's Secret Sharing (SSS):** Proposed by Adi Shamir in 1979, this scheme is based on polynomial interpolation. The core idea is that a unique polynomial of degree $t-1$ is defined by any t distinct points. To share a secret s, the dealer encodes it as the constant term of a randomly generated polynomial $f(x)$ of degree $t-1$ over a finite field $\mathbb{F}_p$, such that $f(0) = s$. The polynomial is defined as:

  $$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1} \pmod{p}$$

  where the coefficients $a_1, \ldots, a_{t-1}$ are chosen uniformly at random from $\mathbb{F}_p$. The shares are then generated by evaluating this polynomial at n distinct non-zero public points, $x_1, x_2, \ldots, x_n$. Each participant i receives the share $sh_i = (x_i, f(x_i))$. To reconstruct the secret, any t participants can use their t points $(x_i, f(x_i))$ to uniquely determine the polynomial $f(x)$ using a method like Lagrange interpolation. Once the polynomial is reconstructed, the secret is recovered by evaluating it at $x = 0$, yielding $s = f(0)$. The perfect security of the scheme stems from the fact that with only $t-1$ points, the value of $f(0)$ remains completely undetermined, as infinitely many polynomials of degree $t-1$ can pass through those $t-1$ points, each corresponding to a different possible secret.

- **Blakley's Geometric Scheme:** Proposed independently by George Blakley in 1979, this scheme uses a geometric approach. The secret is represented as a single point in a t-dimensional space. Each share is the equation of a hyperplane (a $(t-1)$-dimensional subspace) that contains this secret point. To reconstruct the secret, t participants intersect their t hyperplanes. The intersection of t distinct hyperplanes in a t-dimensional space uniquely defines a single point—the secret. With fewer than t hyperplanes, the intersection is at least a line, leaving the secret point undetermined among infinitely many possibilities.

1.2 The Architecture of Threshold Cryptosystems

Threshold cryptography extends the principles of secret sharing to the domain of public-key cryptosystems, enabling distributed cryptographic operations like decryption and digital signing. This paradigm is crucial for applications that require both the security of public-key cryptography and the resilience of distributed trust. Instead of protecting a static secret, threshold cryptography protects a dynamic *capability*—the ability to use a private key.

1.2.1 Core Concept

The fundamental innovation of a threshold cryptosystem is the distribution of a single private key sk among n parties. The corresponding public key pk remains a single, public value that can be used by anyone to encrypt messages or verify signatures. The private key sk is never stored in its entirety in any single location. Instead, each of the n parties holds a private key *share* sk_i. A cryptographic operation, such as decrypting a ciphertext or generating a signature, requires the active cooperation of a threshold t of these parties. This architecture inherently avoids single points of failure; an attacker cannot compromise the system by stealing a single key from one party, but must instead compromise at least t parties.

1.2.2 Threshold Public-Key Encryption (TPKE)

A Threshold Public-Key Encryption (TPKE) scheme formalizes the process of distributed decryption. It is defined by a set of five algorithms that manage the key lifecycle and decryption process:

(1) `Setup(n, k, Λ):` A key generation algorithm that takes the number of servers n, the threshold k (often denoted as t), and a security parameter Λ as input. It outputs a public key PK, a verification key VK, and a vector of private key shares $SK = (SK_1, \ldots, SK_n)$.

(2) `Encrypt(PK, M):` A standard encryption algorithm that uses the public key PK to encrypt a message M, producing a ciphertext C.

(3) `ShareDecrypt(PK, i, SK_i, C):` A distributed decryption protocol where each server i uses its private key share SK_i to compute a *partial decryption* of the ciphertext C.

(4) `ShareVerify(PK, VK, C, μ_i):` An algorithm that allows anyone (typically the entity requesting decryption)

to use the public verification key VK to check if a received partial decryption μ_i from server i is valid for the ciphertext C. This is crucial for robustness against faulty or malicious servers.

(5) `Combine(PK, VK, C, {`$\mu\_1, \ldots, \mu\_k$`})`**:** An algorithm that takes k valid partial decryptions and combines them to recover the original plaintext message M.

The security of a TPKE scheme is typically defined against powerful adversaries, most notably through a game-based definition of security against chosen-ciphertext attacks (CCA2). This ensures that the scheme is secure even if an adversary can obtain partial decryptions for ciphertexts of its choosing, with the exception of the target ciphertext itself.

1.2.3 Relationship to SSS

While threshold cryptography is built upon the conceptual foundation of secret sharing, it represents a significant step beyond it. A simple SSS could be used to split a private key, but this would require the t parties to first reconstruct the full private key in one place to use it, thereby creating a transient single point of failure and negating the core security benefit. The crucial distinction is that threshold cryptosystems employ interactive or non-interactive protocols that allow the parties to *use* their shares to collectively compute a result (e.g., a signature or a plaintext) *without* ever reconstructing the private key itself. For example, in threshold decryption, each server computes a partial result based on its share, and these partial results are then combined. The full private key is never assembled, preserving the distributed trust model throughout the entire operation. This makes threshold cryptography a far more complex but also more powerful primitive for active, distributed security applications.

2. Paradigms of Traceability in Cryptography

The concept of "traceability" in cryptography is not monolithic; it encompasses a range of models, goals, and techniques designed to provide accountability in systems with multiple participants. At its core, traceability is the ability to identify the source of leaked information or unauthorized actions. In the context of secret sharing and threshold cryptosystems, this specifically refers to identifying the colluding insiders who misuse their legitimate access to undermine the system's security. Understanding the paradigms of traceability requires examining its origins in traitor tracing, analyzing the inherent tension it creates with user anonymity, and distinguishing between different models of how accountability is enforced.

2.1 From Traitor Tracing to Traceable Secret Sharing

The intellectual roots of modern traceability lie in the field of *traitor tracing*, which was originally developed for broadcast encryption and Digital Rights Management (DRM) systems. In a typical broadcast scenario, a content provider distributes encrypted content (e.g., pay-per-view television) to n legitimate subscribers, each possessing a unique decryption key. The central problem arises when a coalition of malicious subscribers—the "traitors"—pool their keys to create a "pirate decoder." This decoder is a new device or piece of software that can decrypt the broadcasted content but whose internal workings are designed to be anonymous, hiding the identities of the traitors who created it. A traitor tracing scheme provides a mechanism for the content provider, upon capturing such a pirate decoder, to analyze its behavior and identify at least one of the traitors in the original coalition. This foundational model has been ingeniously adapted to the contexts of secret sharing and threshold encryption. The role of the "pirate decoder" is taken by a "reconstruction box" or a "decryption oracle":

- In **Traceable Secret Sharing (TSSS)**, a coalition of fewer than t malicious parties, who individually cannot reconstruct the secret, can create a *reconstruction box* R. This box is an algorithm that takes their $f < t$ shares as internal state and, when provided with an additional $t - f$ valid shares from honest parties, outputs the original secret. The goal of the tracing algorithm is to identify one of the malicious parties who contributed to building R.

- In **Traceable Threshold Decryption**, a coalition of t or more parties can create a decryption device D that functions as a decryption oracle. This device can decrypt ciphertexts without revealing the identities of the parties who contributed their key shares to its creation.

3. Adversary Models

We formalize three adversarial games that extend the frameworks introduced by Goyal et al. **?** and Boneh et al. **?**. These games model different adversarial goals in secret sharing schemes.

3.1 Secret Recovery Game

Algorithm 1 Secret Recovery Game

Input: Security parameter κ, number of parties n, threshold t
Output: 1 if adversary reconstructs the secret, else 0
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa, n, t)$
$s \leftarrow \mathbb{F}$
$(I, \mathsf{state}) \leftarrow \mathcal{A}^{\mathsf{Share},\mathsf{Recon}}(\mathsf{pp})$
$s' \leftarrow \mathcal{A}(\mathsf{state})$
if $s' = s$ **then**
    return 1
else
    return 0
end if

3.2 Traceability Game

Algorithm 2 Traceability Game

Input: Security parameter κ, number of parties n, threshold t
Output: 1 if tracing succeeds, else 0
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa, n, t)$
$(\mathsf{pk}, \mathsf{sk}, \mathsf{tkey}, \{\mathsf{sk}_i, \mathsf{vk}_i, \mathsf{tkey}_i\}_{i\in[n]}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
$s \leftarrow \mathbb{F}$
$I = \{i_1, \ldots, i_f\} \subseteq [n]$
$\mathsf{ShareSupply} \leftarrow \mathsf{Tracer}$
$\mathsf{Box} \leftarrow \mathsf{Tracer}$
$s' \leftarrow \mathsf{Box}$
$C \leftarrow \mathcal{A}$
if $s' = s$ and $\mathcal{A}$ produces valid NIZkPOK **then**
    return 1
else
    return 0
end if

3.3 Framing Game

Algorithm 3 Framing Game

Input: Security parameter κ, number of parties n, threshold t
Output: 1 if framing succeeds, else 0
$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa, n, t)$
$I \subseteq [n], i \notin I$
$(\mathsf{sk}_i, \mathsf{vk}_i, \mathsf{tkey}_i) \leftarrow \mathsf{Corrupt}(i)$
$(s', \mathsf{Proof}) \leftarrow \mathcal{A}(\mathsf{pp}, \{\mathsf{sk}_j, \mathsf{vk}_j, \mathsf{tkey}_j\}_{j\in I})$
if $\mathsf{Judge}(\mathsf{Proof}, I \cup \{i\}) = 1$ and $s' = s$ **then**
    return 1
else
    return 0
end if

3.4 Reconstruction Box Abstraction

Boneh, Partap, and Rotem **?** propose a **black-box model** for tracing, where the tracing algorithm has only oracle access to the reconstruction box $\mathcal{R}$. This abstraction is now standard in traceable VRFs and dealerless threshold encryption. A critical distinction in these models is between *white-box* and *black-box* tracing. In a white-box model, the tracer is assumed to have full access to the internal code and state of the pirate decoder. This is often an unrealistic assumption. The more challenging and practical model is *black-box tracing*, where the tracer can only interact with the decoder via its public interface—providing inputs (e.g., ciphertexts or additional shares) and observing the outputs. The challenge is to deduce the identity of a traitor purely from this input-output behavior. The majority of recent and impactful research focuses on achieving security in this stronger black-box model.

3.5 The Anonymity-Traceability Spectrum

There is a fundamental and inherent tension between the goals of user anonymity and actor traceability. Anonymity seeks to hide the link between an action and an identity, while traceability seeks to expose it. This conflict is a central theme in the design of any group-oriented cryptographic system, and the extensive research in group and ring signatures provides a valuable lens through which to understand this trade-off.

- **Group Signatures:** These schemes are designed to provide "anonymity by default." A member of a group can sign a message on behalf of the entire group, and a verifier can confirm that the signature is valid and originated from a legitimate group member, but cannot identify which specific member produced it. However, this anonymity is conditional. A designated *group manager* or *tracing authority* holds a special key that allows them to "open" any signature and reveal the identity of the signer. This provides accountability but places immense trust in the group manager, who has the unilateral power to de-anonymize any user, including innocent ones. This centralized power can be a significant vulnerability if the manager is compromised or acts maliciously.

- **Ring Signatures:** In contrast, ring signatures offer stronger, unconditional anonymity. A signer can form an ad-hoc "ring" of participants (including themselves) and produce a signature that is verifiable as coming from someone in that ring, without revealing who. There is no group manager and no mechanism for tracing. While this provides powerful privacy, it comes at the cost of accountability. Malicious users can abuse this anonymity with little fear of being identified.

The design of modern traceable systems for secret sharing and threshold cryptography is an attempt to navigate this spectrum and find a desirable middle ground. The objective is often described as achieving "privacy-preserved traceability" or "controlled anonymity". In such a system, the identities of honest participants engaged in legitimate protocol executions remain private. However, should a subset of participants deviate from the protocol to create a pirate box or leak information, the system guarantees that their actions can be traced back to them. This selective loss of anonymity serves as a powerful deterrent against misbehavior while preserving privacy as the default state.

3.6 Models of Accountability: Private vs. Public Tracing

The mechanism by which traceability is enforced gives rise to two distinct models of accountability, each with different trust assumptions and suitability for different system architectures.

- **Private (Centralized) Tracing:** This is the traditional and more straightforward model, directly analo-

gous to the group manager in group signatures. In this model, the `Setup` algorithm generates a secret *tracing key* in addition to the other system parameters. This tracing key is given to a trusted authority, and only this authority can execute the tracing algorithm. While simpler to design, this model introduces a centralized point of trust. The security of the entire accountability mechanism rests on the integrity and availability of this single authority. This model is well-suited for hierarchical organizations or systems where a trusted administrator is a natural part of the architecture.

- **Public (Decentralized) Tracing:** This is a more recent and powerful model that is far better aligned with the ethos of decentralized systems like blockchains. In a public tracing scheme, the ability to trace is not tied to any secret key. The tracing algorithm is a public procedure that anyone can run, taking as input the public system parameters and black-box access to a pirate decoder. This decentralizes accountability, removing the need for a trusted third party and making the tracing process itself transparent and verifiable. Achieving public traceability is cryptographically more demanding and often requires different and more advanced tools, such as the switch from binary to q-ary codes seen in recent work on threshold decryption.

The clear trend in the field, driven by the demands of blockchain and other decentralized applications, is a move away from centralized trust models. These applications are explicitly designed to operate without a single trusted party, making the private tracing model a philosophical and architectural mismatch. This has directly spurred research into public traceability, where accountability is an emergent property of the public protocol rules rather than a power vested in a privileged entity. This shift also reflects a deeper evolution in the threat model itself, from protecting against malicious actors to designing systems that are robust and incentive-compatible for rational actors. A rational participant in a decentralized system might be willing to collude if the potential reward is high and the risk of being caught by a single, potentially slow or fallible, authority is low. Public traceability dramatically increases this risk, as anyone in the system can act as a watchdog, fundamentally altering the game-theoretic calculus for potential cheaters.

4. State-of-the-Art in Traceable Secret Sharing Schemes (TSSS)

The field of Traceable Secret Sharing Schemes (TSSS) has undergone a remarkably rapid evolution, moving from a theoretical concept with significant overhead to a set of practical and efficient constructions in just a few years. This progression has been marked by refinements in security models, novel algorithmic techniques, and an expansion of scope from simple threshold policies to more complex access structures. This section charts this development by examining the key research contributions that define the state-of-the-art.

4.1 Foundational Work: The Goyal, Song, and Srinivasan Scheme (CRYPTO 2021)

The concept of TSSS was formally introduced and first constructed by Goyal, Song, and Srinivasan in a seminal paper presented at CRYPTO 2021. Their work was motivated by a scenario that highlights the vulnerability of standard secret sharing to rational actors. They imagine a collector, "Trudy," who publicly offers a financial reward for secret shares. Trudy provides a sophisticated cryptographic protocol that allows any server holding a share to submit it anonymously and receive payment, with a guarantee of "immunity" from being caught. A rational server, "Bob," analyzing this offer, would conclude that the potential reward outweighs the negligible risk, and would thus be incentivized to sell his share. To counter this, the authors introduced the first formal security definition for TSSS. The core guarantee is that any server that misbehaves by leaking its share (e.g., by participating in Trudy's collection scheme) runs a significant risk of being traced. The tracing procedure, upon identifying a cheating server, produces a piece of "evidence" that is computationally infeasible to forge and can be verified in a "court of law" to prove the server's dishonest behavior. The construction proposed to achieve this was highly non-trivial, underscoring the difficulty of the problem. It relied on the existence of a generic secure two-party computation (SMPC) protocol as a core building block. While this established the feasibility of TSSS, it came with significant practical drawbacks. The most notable limitations were that the size of each secret share was quadratic in the length of the secret itself, and the tracing algorithm, inheriting the complexity of the underlying SMPC, was quite involved. This foundational work successfully defined the problem and proved its solvability, but left the door open for the development of more efficient and practical schemes.

4.2 Advancements: The Boneh, Partap, and Rotem Schemes (CRYPTO 2024)

A significant leap towards practical TSSS was made by Boneh, Partap, and Rotem in a paper presented at CRYPTO 2024. They introduced a new security model that is both stronger and more aligned with practical attack scenarios. In their model, a coalition of $f < t$ corrupted servers uses their shares to create a *reconstruction box*, R. This box is treated as a black box; the tracer has no knowledge of its internal construction and can only provide it with additional shares and observe its output. The goal is to use this black-box access to trace R back to at least one of the f corrupted servers that created it. This model is stronger because it makes no assumptions about the methods used by the colluders, making it robust against unforeseen implementation strategies. Within this new framework, Boneh, Partap, and Rotem presented two highly efficient constructions based on classical secret sharing schemes.

4.2.1 Construction 1: Traceable Shamir's SSS

The first construction provides a novel and practical tracing algorithm for the widely-used Shamir's Secret Sharing scheme. By subtly modifying the share generation process, they enable a tracer to identify colluders. A key achievement of this scheme is its efficiency: the shares are only twice as large as the secret, a linear overhead that represents an exponential improvement over the quadratic size in the Goyal et al. construction. The tracing algorithm is also specific and efficient, avoiding the heavy machinery of generic SMPC.

4.2.2 Construction 2: Traceable Blakley's SSS

The second construction is based on an extension of Blakley's geometric secret sharing scheme. This scheme is particularly remarkable for the efficiency of its tracing mechanism. It is optimally efficient in the sense that it requires just **one** successful query to the reconstruction box R to identify a traitor. This "one-shot" tracing capability is extremely powerful in practical scenarios where interaction with a pirate box may be limited or risky. Like the Shamir-based scheme, the share size is also only twice that of the secret. This work, with its stronger security model and two concrete, highly efficient constructions, is widely seen as a critical step in moving TSSS from a theoretical curiosity to a deployable cryptographic tool. This rapid cycle of improvement—from a feasibility result with high overhead in 2021 to near-optimal, practical constructions in 2024—is indicative of both the perceived importance of the problem and the rapid pace of innovation in the field.

4.3 Extending Traceability Beyond Thresholds: General Access Structures

Standard (t, n)-threshold schemes are a special case of a broader concept known as *general access structures*. A general access structure is a formal way to specify any monotone set of authorized coalitions. For example, in a corporate setting, a policy might require approval from {any two executives} OR {one executive and three managers}. Such a policy cannot be represented by a simple (t, n)-threshold. The first work to extend the TSSS framework to these more expressive general access structures was presented by Farràs and Guiot in a 2025 ePrint paper. Their work broadens the applicability of traceability to scenarios with complex, hierarchical, or department-based authorization policies. Interestingly, the path to constructing a traceable scheme for general access structures led the authors to make significant contributions to a seemingly opposite concept: *anonymous secret sharing*. Anonymous SSS is a strengthened notion of secret sharing which requires that the shares themselves do not reveal the identities of the parties holding them. The work by Farràs and Guiot proposes new, stronger definitions for anonymity and presents an anonymous scheme for general access structures that satisfies them. This connection reveals a deep and non-obvious technical relationship between anonymity and traceability. It appears that in order to build a robust tracing mechanism that can handle the complexities of a general access structure, one must first build the scheme upon a foundation of strong anonymity. In a simple threshold scheme, all shares are structurally equivalent. In a general scheme, different parties may have shares with different "power" or structure. An anonymous scheme likely regularizes this structure, preventing colluders from using the unique characteristics of their shares to construct a reconstruction box that evades tracing. The traceability mechanism is then built atop this anonymous layer. This suggests that, far from being opposites, strong anonymity properties may be a prerequisite for achieving robust traceability in complex sharing scenarios.

4.4 Comparative Analysis of TSSS Schemes

The rapid development of TSSS can be best understood by comparing the key properties of the major schemes proposed to date. Table 1 provides a summary of their characteristics, highlighting the evolution in efficiency, security models, and supported access policies.

5. Accountability in Threshold Decryption and Signature Schemes

While TSSS addresses the leakage of static secret shares, a more dynamic and complex challenge arises in threshold cryptosystems, where shares are actively used to perform computations. Here, accountability is not just about tracing leaked data, but about attributing unauthorized cryptographic operations (decryptions or signatures) to the colluding parties responsible. Research in this area has been heavily influenced by the economic realities of decentralized systems, leading to the development of sophisticated schemes that balance accountability, privacy, and decentralization.

5.1 The Rational Actor Problem in Threshold Decryption

Standard threshold decryption schemes, while secure against classical cryptographic adversaries, exhibit a critical vulnerability when participants are modeled as rational economic actors. The core issue, as articulated by Boneh et al., is the existence of an "untraceable master key". A coalition of t or more malicious (but rational) parties can use their private key shares not to perform a single decryption, but to collaboratively reconstruct the master private key itself. They can then sell this master key to an adversary—for instance, a searcher in an encrypted mempool who wants to front-run transactions. This action is devastatingly effective because the reconstructed master key typically contains no information linking it back to the specific shares [8] used in its creation. From the perspective of the colluding parties, this is a "risk-free profit" opportunity. Naive penalty mechanisms, such as "slashing" (a financial penalty common in blockchain systems) for any party whose individual key share sk_i is publicly revealed, are completely ineffective. The colluders never reveal their individual shares; they only sell the derived, anonymous master key. This economic vulnerability demonstrates that cryptographic security alone is

Table 1: A comparative summary of state-of-the-art Traceable Secret Sharing Schemes.

Scheme	Underlying SSS	Access Structure	Share Size Overhead	Tracing Model	Tracing Efficiency	Key Assumptions
Goyal, Song, Srinivasan (CRYPTO '21)	Generic	Threshold	Quadratic ($O(\lambda \cdot \|s\|)$ where λ is security parameter)	Cryptographic Collection Protocol	Complex (relies on SMPC)	Existence of Secure Two-Party Computation
Boneh, Partap, Rotem (CRYPTO '24)	Shamir's SSS	Threshold	Linear ($2 \cdot \|s\|$)	Black-Box Reconstruction Box	Efficient (polynomial queries)	Standard (none)
Boneh, Partap, Rotem (CRYPTO '24)	Blakley's SSS	Threshold	Linear ($2 \cdot \|s\|$)	Black-Box Reconstruction Box	Optimally Efficient (1 query)	Standard (none)
Farràs, Guiot (ePrint '25)	Novel (based on anonymous SSS)	General Monotone	Polynomial	Black-Box Reconstruction Box	Not specified (focus on feasibility)	Standard (none)

insufficient. For a threshold decryption system to be viable in an incentivized environment, traceability must be an inescapable, inherent property of the scheme itself.

5.2 Publicly Traceable Threshold Decryption

The need for a scheme that could resist this rational attack led to the development of traceable threshold decryption. A key goal, especially for decentralized applications, is to achieve *public traceability*, where accountability is not reliant on a trusted central authority [9]. The work of Canard, Papon, and Phan [9] presents a concrete construction for publicly traceable threshold decryption. Their central technical innovation is to move from the binary collusion-secure codes used in classical traitor tracing [10] to more advanced **robust q-ary Identifiable Parent Property (IPP) codes** [11]. These codes, defined over a larger alphabet of size q, have properties that allow for public tracing algorithms. To integrate these codes, the underlying cryptographic primitive had to be generalized from a bipartite (2-party) Key Encapsulation Mechanism (KEM) to a **q-partite KEM**. The security model was also strengthened to a new notion of *adaptive one-sided security*, which is necessary to prove the soundness of the public tracing procedure. The tracing algorithm in this scheme works by interacting with the pirate decryption oracle D in a black-box manner. By feeding it specially crafted ciphertexts and observing its success or failure, the tracer reconstructs a "pirate codeword." The properties of the IPP code then guarantee that this codeword can be used to identify at least one of the "parent" codewords belonging to a traitor who contributed to the decoder D. This work shows a clear causal link: an economic problem (the rational actor vulnerability) in an existing cryptographic primitive (threshold decryption) directly spurred fundamental cryptographic innovation (the development of publicly traceable threshold KEMs based on q-ary IPP codes).

5.3 Hybrid Models: Accountable and Private Threshold Signatures (TAPS)

In many real-world applications, there is a desire for both internal accountability and external privacy. An organization might need to trace which of its executives authorized a fraudulent financial transaction, but it does not want to reveal its internal governance structure (e.g., the threshold t, or the identities of the signers on any particular valid transaction) to the general public. This requirement led to the development of a hybrid primitive. In their CRYPTO 2022 paper, Boneh and Komlo introduced **Threshold, Accountable, and Private Signatures (TAPS)**. A TAPS scheme produces signatures that are, from a public verification standpoint, completely private; they reveal nothing about the quorum of signers. However, a designated entity holding a secret *tracing key skt* can analyze any valid signature and trace it back to the exact set of t signers who produced it. The high-level construction involves having the signing parties first generate a standard **Accountable Threshold Signature (ATS)**, which by definition reveals the signers' identities. They then encrypt this ATS under a public key whose corresponding secret key is the tracing key *skt*. The final TAPS signature consists of this ciphertext along with a zero-knowledge proof that it is a valid encryption of a valid ATS for the given message. This elegant construction resolves the binary trade-off between fully private schemes (like Private Threshold Signatures, PTS) and fully accountable ones (ATS), creating a new point on the spectrum that is highly desirable for many organizational

use cases.

5.4 Decentralization and Dynamic Accountability (DeTAPS)

While TAPS provided a powerful new model, its initial construction was centralized. It relied on a single trusted *combiner* to assemble the signature shares and a single trusted *tracer* to hold the tracing key *skt*, reintroducing single points of failure and trust that are undesirable in distributed systems. To address these limitations, the **Decentralized TAPS (DeTAPS)** scheme was proposed. This work elevates the concept of accountability from a simple property to a complex, distributed protocol in its own right. DeTAPS introduces several key features:

- **Decentralized Combining and Tracing:** The roles of combiner and tracer are not fixed to single entities but can be performed by a dynamic set of nodes in a distributed network (e.g., elected via a blockchain consensus mechanism).
- **Enhanced Privacy:** The scheme is designed to be secure even against untrusted combiners and tracers, using techniques like Trusted Execution Environments (TEEs) to protect secret keys during computation.
- **Notarized and Dynamic Tracing:** The most significant innovation is that the act of tracing is itself a threshold-gated event. A tracing operation can only proceed if a quorum of t' designated "notaries" authorize it.

This notarization is achieved using sophisticated cryptographic tools. **Dynamic Threshold Public-Key Encryption (DTPKE)** is used to encrypt the accountable signature component such that only a dynamically chosen set of t' notaries can cooperate to decrypt it. **Key-Aggregate Searchable Encryption (KASE)** is used as a secure and efficient mechanism to "awaken" the correct notaries and provide them with the necessary information to perform their part of the tracing protocol. The evolution from ATS to TAPS and then to DeTAPS demonstrates a clear trend: accountability is becoming a first-class cryptographic protocol, not just a static property. The act of tracing in a high-stakes decentralized system must itself be a secure, robust, and accountable multi-party process. This represents a significant increase in both conceptual and technical complexity, signaling the emergence of a new sub-field focused on the "cryptography of accountability."

5.5 Comparative Analysis of Accountable Threshold Schemes

The various approaches to accountability in threshold cryptosystems occupy different points in the design space, balancing privacy, accountability, and decentralization. Table **??** summarizes the key characteristics of the landmark schemes in this domain.

6. Conclusion

Traceable secret sharing and threshold encryption have matured from theoretical constructs to practical tools. Efficient black-box tracing and advanced adversary models enable accountability in distributed trust environments. Future work includes post-quantum secure traceability and incentive-compatible designs [7], [12].

Acknowledgments We express our gratitude to the National Institute of Information and Communications Technology (NICT). This research was made possible through the support provided for Dr. Anandarup ROY's visit to Sakurai Lab at Kyushu University by the Foreign Researcher Invitation Program of NICT.

References

[1] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[2] G. R. Blakley. Safeguarding cryptographic keys. In *1979 International Workshop on Managing Requirements Knowledge (MARK)*, pages 313–318, 1979.

[3] Yvo Desmedt and Yair Frankel. Shared generation of authenticators and signatures. In J. Feigenbaum, editor, *Advances in Cryptology–CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 457–469. Springer, 1990.

[4] Jan Bormet, Jonas Hofmann, and Hussien Othman. Traceable threshold encryption without trusted dealer. Cryptology ePrint Archive, Paper 2025/342, 2025.

[5] Karim Baghery, Ehsan Ebrahimi, Omid Mirzamohammadi, and Mahdi Sedaghat. Traceable verifiable secret sharing and applications. Cryptology ePrint Archive, Paper 2025/318, 2025.

[6] Tiantian Gong, Aniket Kate, Hemanta K. Maji, and Hai H. Nguyen. Disincentivize collusion verifiable secret sharing. In *Advances in Cryptology – EUROCRYPT 2025: 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4–8, 2025, Proceedings, Part V*, page 34–64, Berlin, Heidelberg, 2025. Springer-Verlag.

[7] Rishiraj Bhattacharyya, Jan Bormet, Sebastian Faust, Pratyay Mukherjee, and Hussien Othman. Cca-secure traceable threshold (id-based) encryption and application. *IACR Cryptol. ePrint Arch.*, page 341, 2025.

[8] Ran Canetti, Ivan Damgård, Sebastian Kolby, Divya Ravi, and Sophia Yakoubov. Deniable secret sharing. Cryptology ePrint Archive, Paper 2025/525, 2025.

[9] Sébastien Canard, Nathan Papon, and Duong Hieu Phan. Public traceability in threshold decryption. *IACR Cryptology ePrint Archive*, 2025.

[10] Dan Boneh and Brent Waters. A fully collusion resistant broadcast, trace, and revoke system. In *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 211–220. ACM Press, 2006.

[11] Sanjam Garg, Adam D. Klivans, and Brent Waters. Traitor tracing with public tracing. In *Advances in Cryptology–EUROCRYPT 2018*, pages 277–306. Springer Cham, 2018.

[12] Dan Boneh, Aditi Partap, and Lior Rotem. Traceable verifiable random functions. Cryptology ePrint Archive, Paper 2025/312, 2025.