sciendo

Proceedings on Privacy Enhancing Technologies ; 2020 (2):397–415

Bailey Kacsmar*, Chelsea H. Komlo*, Florian Kerschbaum, and Ian Goldberg

Mind the Gap: Ceremonies for Applied Secret Sharing

Abstract: Secret sharing schemes are desirable across a variety of real-world settings due to the security and privacy properties they can provide, such as availability and separation of privilege. However, transitioning secret sharing schemes from theoretical research to practical use must account for gaps in achieving these properties that arise due to the realities of concrete implementations, threat models, and use cases. We present a formalization and analysis, using Ellison's notion of ceremonies, that demonstrates how simple variations in use cases of secret sharing schemes result in the potential loss of some security properties, a result that cannot be derived from the analysis of the underlying cryptographic protocol alone. Our framework accounts for such variations in the design and analysis of secret sharing implementations by presenting a more detailed user-focused process and defining previously overlooked assumptions about user roles and actions within the scheme to support analysis when designing such ceremonies. We identify existing mechanisms that, when applied to an appropriate implementation, close the security gaps we identified. We present our implementation including these mechanisms and a corresponding security assessment using our framework.

Keywords: secret sharing, applied cryptography, protocol analysis, ceremony analysis

DOI 10.2478/popets-2020-0033
Received 2019-08-31; revised 2019-12-15; accepted 2019-12-16.

1 Introduction

The security properties that theoretical secret sharing purports to provide are particularly meaningful for high-risk users such as journalists, as demonstrated by the security-critical effort required for the investigation and reporting of the Panama Papers [29]. However, while the security of theoretical secret sharing is well documented in academic research, in practice, the security guarantees are more complicated.

The descriptions in the literature of secret sharing schemes, which we additionally refer to as threshold schemes, often lack sufficient evidence of the security of real-world deployments of the schemes. This shortcoming is due to the descriptions leaving a large number of assumptions and decisions to the participants, as these are considered to be outside of the protocol. Just as the design of the highly successful TLS protocol accounts for more real-world practicalities than the underlying Diffie-Hellman key exchange protocol, the practical use of threshold schemes, as we demonstrate, is no different. For example, although Shamir secret sharing (see Section 2) is information theoretically secure, ultimately the shares must be communicated to participants through a channel, which in most cases will rely on symmetric or asymmetric encryption, and therefore rely on computational assumptions. Furthermore, unlike cryptographic protocols such as Diffie-Hellman, threshold schemes require significant user involvement and decisions at nearly every stage of the protocol. Consequently, analyzing the security of threshold schemes requires assessing both the protocol and the actions and decisions required of users.

Ellison [14] introduced the concept of a *ceremony* in security analysis, which requires the inclusion of both the cryptographic *protocol* as well as any possible *user actions or decisions* in the security analysis. Surprisingly, the state of research literature for threshold schemes does not include a complete, end-to-end, formal definition and assessment of the security of the ceremony of threshold schemes. Without such definitions, deployments of threshold schemes lack the necessary structure required for formal analysis as their flexibility in terms of applications is broad.

Without strict boundaries for a specific threat model and use case, it is impossible to provide both a generalized framework and a formal analysis for secret sharing ceremonies. This work provides a structure, in

***Corresponding Author: Bailey Kacsmar:** University of Waterloo, E-mail: bkacsmar@uwaterloo.ca
***Corresponding Author: Chelsea H. Komlo:** University of Waterloo, E-mail: ckomlo@uwaterloo.ca
Florian Kerschbaum: University of Waterloo, E-mail: florian.kerschbaum@uwaterloo.ca
Ian Goldberg: University of Waterloo, E-mail: iang@uwaterloo.ca

the form of a framework, that can be used for a given threshold scheme to define and analyze its particular ceremony, by structuring the ceremony as a series of stages and steps as is necessary to assess the ceremony's end-to-end security. Although an unbounded number of possible user interactions exist, our framework can be used to guide the definition and formalization of the ceremony. We identify ceremony-related issues, such as requiring the dealer to delete sensitive material and the requirement for users to authenticate one another. The ceremony for an application, defined in terms of our framework, accounts for the specific goals and adversaries of the ceremony and therefore provides the needed structure that must precede efforts to formally analyze a specific ceremony.

Contributions. We provide a framework to facilitate the process of defining an accurate ceremony for a given threshold scheme; we then use this framework to assess the security of several threshold schemes, and define a lightweight set of improvements that are useful to threshold schemes based on Shamir secret sharing. Our framework is useful for comparing different existing secret sharing schemes; however, what it primarily provides is a structure for defining threshold scheme ceremonies with the necessary details to perform a more accurate security analysis that accounts for the setting in which the threshold scheme is used. Overall, our contributions include:

- a demonstration of the variability in the ceremony of threshold schemes and how this variability can lead to gaps in the security properties achieved by the threshold scheme;
- formalizations of the adversaries and of several use cases of threshold schemes used in practice;
- a framework to facilitate security analyses of threshold schemes used in real-world settings;
- exemplar applications of our ceremony framework via security analyses of three threshold scheme case studies; and
- techniques to close security gaps uncovered in our above analysis and an implementation of these improvements in Rust.

Organization. This paper is organized as follows. Background, motivation, and related work are Sections 2, 3, and 4 respectively. Section 5 is our framework for our analysis. Formalized ceremonies for two modes of operation for threshold schemes are in Sections 6 and 7. Section 8 summarizes our analysis for several threshold schemes, Section 9 is our improved ceremony and implementation and our conclusion is Section 10.

Table 1. Parameters and additional notation used within our analysis

n	number of participants
t	threshold
s	secret recovered by a (t, n)-threshold scheme
$\mathcal{S}$	secret space of a (t, n)-threshold scheme
D	dealer
D_P	dealer that later becomes a participant
$\mathcal{F}$	sensitive information requiring protection
Base	$s = \mathcal{F}$
Ext	$s \neq \mathcal{F}$
P_r	participant performing a recovery of s
$\mathcal{U}$	participant performing an update
$\vec{C}$	commitment used to validate any share

2 Threshold Schemes

We summarize the notation used throughout our analysis in Table 1, including both notation standard to the literature as well as new notation we introduce in later sections for the purpose of our analysis. Notably, we denote the secret information that is protected by the threshold scheme as $\mathcal{F}$, while the secret input into the threshold scheme is s. This differentiation will become important when we define modes of operation where $\mathcal{F}$ can either be equal to s, or distinct, as defined in Section 5.2.

In general, cryptographic secret sharing schemes enable a group of n participants, possessing a secret s, to divide s into n shares. Before creating the n shares, a threshold value t is chosen such that a collection of t shares must be used to learn the value of s. A (t, n)-*threshold scheme* is a secret sharing scheme where n and t are positive integers such that $t \leq n$, n representing the number of participants, and t the desired threshold.

In a (t, n)-*threshold scheme* we designate a *dealer* D as the entity that selects the *secret* s and generates the n shares such that each of the n participants in the scheme receives a share that preserves the following properties:

Reconstruction: any size-t subset of the n participants can compute the secret given their t shares, and

Secrecy: no subset of the n participants consisting of $t-1$ or fewer participants is able to gain any knowledge of the secret given their combined shares.

In a conventional (t, n)-*threshold scheme*, the set of n participants does not contain the dealer D. However, in our analysis we work in the setting where the dealer, now labeled a *participant dealer* D_P, may continue to be involved in the scheme as a participant, as typically occurs in real-world practical settings.

While some variants of threshold schemes, such as threshold signature schemes [21], allow participants to use their shares of s individually and to perform reconstruction only on the results, we focus our attention on threshold schemes in which s is reconstructed directly. A well-known such construction, due to Shamir [41], distributes points lying on a polynomial (of degree $t-1$) as shares. We refer to this construction (summarized in Appendix A) as *classic Shamir secret sharing*. Combining t of these shares using polynomial interpolation would recover the secret; combining any smaller number of shares does not leak any information about the secret. The construction is information theoretically secure; that is, a Shamir threshold scheme can withstand adversaries with unlimited computational power.

3 Motivation

Threshold schemes allow for a high degree of variability in user goals and potential adversaries, where even slight variations significantly influence the security of the scheme overall. We describe two practical examples, where both cases utilize an identical underlying threshold scheme protocol, however, the threat model, context, goals, and thus ceremony vary dramatically between the two examples. For both examples, Alice, Bob, and Carol are journalists at the same organization.

Case One. Alice received highly sensitive files from a source. She fears external parties will act against her to prevent the distribution of the files, and wants to ensure that even if an adversary succeeds at targeting her, the information can still be accessed by either herself or trusted colleagues. She enlists the help of Bob and Carol in her efforts to preserve the availability of the files.

Alice acquires a laptop to encrypt and store the secret information. She inputs a key k into a tool that implements classic Shamir secret sharing using k as the secret ($s = k$) and inputs her desired parameters $t = 2$ and $n = 3$. The tool outputs the corresponding shares and Alice messages Bob and Carol over an established communication channel. Alice sends one share to Bob and one share to Carol. Bob and Carol confirm they received the share and each store their respective shares on a USB, which is then stored in a chosen safe place. Alice stores the laptop containing the encrypted secret information in a safety deposit box at a bank.

Alice leaves the news organization and Bob, who has lost his share, is assigned the story. Fortunately, Alice left the organization on good terms, so Bob contacts Alice and Carol, requesting their shares. Bob retrieves the laptop and decrypts the ciphertext using the key recovered from Alice and Carol's shares.

Case Two. Alice has received a decryption key that corresponds to a publicly released ciphertext [3, 4]. She fears external parties will attempt to distribute the key in a way that could endanger individuals. Alice wants to ensure the information remains confidential to those she has not authorized (herself or her trusted colleagues Bob and Carol) to distribute it.

Alice meets Bob and Carol at a previously agreed upon location. Using an airgapped[1] laptop, Alice inputs the key from her source into a tool that implements classic Shamir secret sharing, choosing $t = 2$, and $n = 3$. After the tool has output the corresponding shares, Alice, Bob, and Carol each save one share to their respective USB. Finally, Alice deletes all information off of the airgapped laptop. Everyone keeps their respective USB devices on their person at all times.

Alice's USB is taken from her while crossing a border. Fortunately the USB is insufficient to learn the secret data, however, Alice can meet Bob and Carol in person to request their shares in order to recover the key.

Observations. In both cases, Alice made a number of choices, including selection of participants, selection of communication methods, and selection of storage mechanisms. The choices Alice made affect the security and privacy properties of each case. For instance, only Case Two requires the physical presence of each participant. Such a requirement may limit the availability of the information if an adversary had the power to prevent participants from meeting up; for example, if the participants are initially separated by a geographical border. Furthermore, storing the encrypted data on the laptop creates a single point of failure for an adversary targeting the availability of the ciphertext.

The above examples demonstrate how the range of choices and prioritization impact the threshold scheme ceremony. For instance, Case One preserves availability and prevents the information from being released

1 As a security mechanism, an airgapped laptop is protected against connecting to the Internet.

preemptively, but still requires confidentiality, otherwise multiple copies of the information could simply be stored. More significantly, these examples highlight the need to consider the ceremony of the threshold scheme when performing a security analysis, factoring in both the threat model that users operate within, along with how users perform the actions required of the threshold scheme in context of their use case, such as whether users operate online or entirely offline. This crucial observation motivates our next sections, where we formalize several ceremonies of threshold schemes as both protocol *and* explicit user actions and decisions in the effort to more accurately assess the security of the threshold scheme under consideration.

4 Related Work

Ceremony Analyses of Security Tools. Our analysis follows in the style of prior work analyzing cryptographic tools used in practice [13, 19], verifying security properties, and documenting decision paths taken by users when participating in cryptographic protocols. Our security analysis specifically includes the ceremony performed by end users [14], encompassing everything out-of-band to a cryptographic protocol but required of users and thus subject to security consequences [12, 23]. While prior ceremony analyses have been performed for a number of cryptographic protocols, the research literature currently lacks a similar analysis assessing the security of ceremonies for threshold schemes specifically.

Previous work on ceremony analysis includes specifying how to model users' devices [26] and formal analysis of Public-Key Infrastructures (PKIs) [27]. While frameworks have been proposed to assess the security of existing ceremonies such as that of Carlos et al. [9], our framework specifically defines possible threshold scheme ceremonies and facilitates their analysis.

In 2013, Carlos et al. [10] focused on threat modeling within ceremonies, highlighting that threat models for ceremonies must be adaptive. Threat models must evolve to match varying user goals and contexts even when these ceremonies utilize the same underlying protocol. Radke et al. [34] highlight adaptive threat models as a weaknesses of ceremony analysis, as the context of the ceremony must be as well defined in order to accurately model potential adversaries and threats against the goals of end users to claim the ceremony as secure. As ceremony analysis of threshold schemes is highly dependent on users' threat models, and as the threat models can differ depending on the user, context, or use case, instead of providing a narrow ceremony analysis for a specific threat model, we present a generalized *framework* for performing a ceremony analyses of threshold schemes in both theory and practice, across several common real-world threat models.

Applications of Threshold Schemes. Sunder is an existing applied secret sharing tool created by Freedom of the Press Foundation [18] to support journalists protecting long-term secrets such as the Snowden archives. Another tool building on secret sharing is Callisto, which provides a safety-in-numbers approach to exposing names of sexual abusers [35]. While these use cases give insight into the setting and application of threshold schemes used in practice, many other use cases of threshold schemes exist [1, 2, 43].

Shatter [2] is a framework for desktop and mobile platforms that performs key sharing across a user's devices. Shatter uses secret sharing to leverage users' increasing numbers of devices by requiring a threshold number of devices to provide consensus for actions such as performing a login. Although Shatter uses secret sharing it is actually an example of a threshold signature scheme. Nonetheless, it still shares a number of properties with secret sharing schemes that make our analysis applicable.

Shatter Secrets [1] is an advance on Shatter that provides protection to users' data when crossing borders. With Shatter Secrets, a user could encrypt their primary device and then distribute shares to their friends at their destination with the encryption key serving as the secret s. Once over the border, the user with the encrypted device visits t of their friends, physically NFC-taps their devices to retrieve the shares, reconstructs the secret, and decrypts their device.

Pico [43] stores shares on hardware tokens instead of utilizing users' existing devices. One explicit use case of Pico is as a replacement for password managers as it uses public key cryptography challenge-response instead of typical passwords. Pico exists as a mobile application and is intended to block a thief who has stolen fewer than t tokens from violating confidentiality, while preserving availability as long as t tokens remain.

Secret Sharing Variants. Verifiable Secret Sharing (VSS) is a variant on threshold schemes in which any participant can verify the integrity of their share using a public commitment. Well-known VSS schemes include Feldman's [15] and Pedersen's [32] schemes.

Proactive secret sharing, introduced by Ostrovsky and Young [31] and used in a secret-sharing scheme by Herzberg et al. [22], protects against a *mobile adver-*

sary. A mobile adversary can control a subset of players over time, but the members belonging to this subset can change between epochs. To defend against such an adversary, proactive secret sharing relies on proactively updating shares to enable a form of forward security.

5 A Framework for Ceremony Analysis

In this section we present the components we need for performing a ceremony analysis. We define threshold scheme adversaries, distinguish two modes of operation, identify security goals, and provide additional terminology that we use throughout our analysis. We conclude this section with an outline of how to use our framework to produce a complete ceremony for a specific secret sharing protocol used in a specific setting and purpose.

5.1 Formalizing Threshold Scheme Adversaries

First, we formalize a range of possible adversaries against threshold schemes used in practice, and describe the possible capabilities and powers these adversaries can hold. We outline several conventional adversarial models and identify variations within each model.

Adversary Power. We define three levels of power an adversary may possess. Although we utilize the terms 'high', 'middle', and 'low', these terms are simply points of reference for comprehension and are not intended as prescriptive classifications.

A *high-powered adversary* has the power and resources of a government actor. High-powered adversaries can access state-of-the-art computing resources and have significant quantities of time and money at their disposal. Such an adversary has the power to take legal action, bounded only by the political environment of that jurisdiction. For example, the NSA is known to masquerade as well-known sites, installing malware capable of exfiltrating data from a victim's device [20], governments are known to use informants to infiltrate activist groups (Martin Luther King Jr.'s friend and photographer was an FBI informant [28]), and some countries have proposed laws allowing legal orders requiring technology companies to work on behalf of the government to provide access to encrypted devices [11].

A *low-powered adversary* has similar computational, temporal, and monetary resources as the participants of the threshold scheme. A *middle-powered adversary* exists somewhere between the powers of a government actor and the powers of the participants. Such an adversary has the same legal powers as a low-powered adversary, but may have the same money and time available to them as a government actor.

Adversary Capabilities. We limit our analysis to the capabilities of the below-mentioned adversarial models. The adversaries may be participants in the ceremony or outsiders. A previously trusted participant may become an adversary at a later time in the protocol. That is, we do not assume participant roles are static.

An *honest-but-curious* (HBC) adversary will not deviate from the ceremony, but will try to learn as much information as they can within the bounds of the ceremony. An HBC adversary will view any information that is exposed to them, and may collude with other participants in an effort to learn additional information.

A *malicious* adversary is not bound to any expectation of behaviour, and she can participate both honestly and dishonestly in the ceremony at will. A malicious adversary can impersonate other actors, elect to not participate in the ceremony, or participate disruptively by, for example, providing false shares and attempting to deceive other parties into providing the adversary with their shares.

Adversaries who compromise operating systems or hardware infrastructure are also a real threat to users defending against a high-powered adversary. However, this class of threats are out of scope for our analysis as details of physical infrastructure vary widely between implementations. In practice, secret-sharing implementations should employ well-known device protection techniques such as single-use strategies[2] and using secure operating systems such as Qubes [39].

Adversary Goals. Here we identify goals for an active adversary of threshold schemes.

Learning secret information. An adversary motivated to learn the sensitive information $\mathcal{F}$ can work to gain knowledge of the secret or the shares, subsequently allowing the adversary to recover $\mathcal{F}$.

Modifying secret information. An adversary may wish to modify $\mathcal{F}$ without detection, resulting in participants recovering information that is different than the original input into the threshold scheme.

2 One example of a single-use strategy is using "burner" phones. A burner phone is one that is newly purchased and used for a short period, after which it is discarded. [40]

Preventing secret recovery. Adversaries may also seek to prevent others from accessing or disseminating $\mathcal{F}$. For example, an adversary seeking to hide information—such as a government seeking to prevent public distribution of evidence of war crimes—can work to disrupt communication, destroy shares, or even to destroy the sensitive data $\mathcal{F}$.

Causing harm to participants. In some countries, working with material that is prohibited can be a crime, putting all parties at risk [5]. An adversary may be motivated to harm the participants of the threshold scheme, and can seek to perform actions such as attributing ownership of $\mathcal{F}$ to those participants.

5.2 Modes of Operation

We define two manners of use, termed 'modes of operation', to manage the sensitive information $\mathcal{F}$. The Base Mode is defined as a ceremony for classic Shamir secret sharing [41]. The Extended Mode is an extension to Base Mode, and is documented to be used in practice in high-risk settings [1, 18].

Base. In the first mode of operation, the confidential information, $\mathcal{F}$, is small in size, such that each of the shares distributed to the participants can reasonably be about the size of $\mathcal{F}$. The secret s can then be the information itself, $s = \mathcal{F}$.

Extended. The second mode of operation, addresses when the sensitive information is too large to be used directly as the secret s. The **Ext** mode of operation is modeled after a common real-world use case described in the documentation of the secret sharing tool Sunder [18]. In this case, the confidential data $\mathcal{F}$ is first encrypted and the encryption key is then used as input as s into **Base**. After the secret s is reconstructed, additional steps must be taken to retrieve the data $\mathcal{F}$ using s as a key.

5.3 Identifying Security Goals

We next define several security goals that are commonly cited for implementations of threshold schemes used in practice [2, 18]. The below identified goals are specific to the context and use case of threshold schemes in general. However, the context of the threshold scheme under consideration can impact the security goals for the scheme. We use this set of security goals for our analysis, but other analysts using our framework should identify the security goals appropriate for their specific scheme. Such goals are not limited to the ones listed here and may include some of these goals, and others as well.

1. **t-Separation of Privilege:** We define *t-Separation of Privilege* as a specific case of the well-known *Separation of Privilege* security principle first introduced by Saltzer and Schroeder [38]. Threshold schemes require t participants' shares to perform a recovery of $\mathcal{F}$, where t is the chosen threshold.
2. **Availability:** The secret information $\mathcal{F}$ is accessible to honest participants so long as at least t valid shares remain accessible. For Extended Mode the availability of the encrypted version of $\mathcal{F}$ will be enforced by the choice of safe storage mechanism (see Section 5.5).
3. **Information Theoretic Security:** Even given unlimited computational power, adversaries inside or outside the ceremony cannot access $\mathcal{F}$ while possessing fewer than t shares.
4. **Confidentiality:** Adversaries outside of the protocol cannot gain knowledge of $\mathcal{F}$. Note that in a real-world setting this goal requires revocation of participants to achieve confidentiality across epochs where participants move from a trusted to an untrusted state.
5. **Integrity/Corruption Detection:** Corruption of an individual share or the sensitive information is detected by honest participants before completing the Reconstruction stage.

5.4 Threshold Ceremony Analysis Outline

We next describe our framework to structure our assessment of the security of threshold schemes in practice, including both the protocol and ceremony of the threshold scheme under consideration.

Identify stages of the ceremony of the threshold scheme. Security ceremonies can be broken down into components called *stages*. Fully specifying the complete ceremony and its component stages is the first step towards evaluating the security of the threshold scheme under consideration. We provide two formalizations (Sections 6 and 7) as a skeletal frame of reference for future analyses of threshold schemes derived from Shamir secret sharing.

Define the threat model. First, define possible *adversaries* of the threshold scheme or of the users participating in the scheme, including the adversaries' goals. In Section 5.1, we demonstrate a range of possible adversaries against threshold schemes. Second, determine the desired *security goals*. We present several possible secu-

1. Ceremony identification and formalization (stages)
2. Threat Model (selection of adversaries and security goals)
3. Mode of operation (identification of use cases)
4. Evaluation of Security (assessment of security goals relative to threat model)

Fig. 1. Framework for security analysis for threshold schemes derived from Shamir secret sharing

rity goals of threshold schemes in Section 5.3. At times, certain security goals may prove to be in conflict. For example, a system operating in Extended Mode that prioritizes availability over confidentiality might distribute an encrypted ciphertext publicly in order to decrease the possibility of destruction.

Define the mode of operation. Threshold schemes can potentially allow for many modes of operation. For example, classic Shamir secret sharing can support both the Base Mode and Extended Mode of operation. To evaluate the security of a threshold scheme, a single mode of operation must first be specified. If a scheme supports more than one mode of operation, the security evaluation should be performed once for each. Note that transitioning between modes of operation for the same, or updated, secret is not supported by such evaluations as it introduces new potential attack vectors (including issues related to using shares at most once; see Section 5.5).

Evaluate security goals against adversaries. Using knowledge of adversary goals and capabilities along with the ceremony formalization, the security goals for the system can be evaluated in the context of the given mode of operation and threat model. For each stage in the threshold scheme, and for each step within a stage, evaluate if the adversary's capabilities can defeat the system goal. If the adversary can defeat the system goal, this goal is considered unmet. See Figure 1 for an overview of our framework.

5.5 Assumptions and Limitations

We maintain several assumptions for the purpose of providing a structured analysis and designing our framework. However, we acknowledge these assumptions may not always hold in real-world settings.

Secure Communication and Storage. We emphasize the existence and availability of a secure communication channel as well as a mechanism for safe storage. Communicating and storing data securely are both critical to the security of a practical threshold scheme.

Safe storage is a storage mechanism such that data is guaranteed to be recoverable in the future. Such mechanisms must avoid single points of failure such as due to server crashes; preventing such failures requires storing copies of the data on multiple servers, for example. We assume a safe storage mechanism provides the properties of availability to participants. We also assume that if participants require authentication before accessing the stored data, the safe storage mechanism can provide this authentication mechanism.

Using Shares At Most Once. We maintain that secrets, and consequently shares, should be single-use as otherwise new security risks are introduced. For example, a multi-use setting requires the assumption that the participant performing the recovery securely deletes both the secret and the collected shares from their local device *after* completing the recovery. If the recovering participant breaks this trust and stores shares or the secret information locally after the first recovery, the participant can bypass the step of gathering $t-1$ shares from other participants in future recoveries. Furthermore, a device containing t shares is a single point of failure—an appealing target for an adversary trying to learn the secret.

Honesty of the Dealer. We assume that the dealer is honest both in the case of D and D_P, as classic Shamir secret sharing is trivially broken when the dealer is dishonest.

In the Extended Mode the dealer is also responsible for determining sufficient protection for the encrypted $\mathcal{F}$ in terms of the secure storage selected.

Erasure Assumption. For the purposes of our analysis, we work within the erasure assumption [8], which assumes that participants are able to securely erase data when required. We recognize that if the erasure assumption does not hold, then many of the security properties we define are broken, as an adversary could perform analysis post-hoc on stolen machines and recover sensitive material.

Non-Collusion. Honest participants will not collude with external parties. For example, honest participants will not attempt a recovery initiated by an unauthorized person.

6 Base Mode Stages

We now more formally identify the possible choices and actions for users participating in a threshold scheme, and introduce a formalization for a general ceremony of threshold schemes based on Shamir secret sharing. Starting with the Base Mode of operation, we present three stages consisting of *share generation*, *share distribution*, and *reconstruction*. The ceremony framework for the Base Mode of operation is outlined in Figure 2.

Classification of Steps. We annotate each step in a stage to classify how participants are involved. We annotate steps as *Device* for expected implementation actions, as *Choice* for user decisions, and as *Action* for expected user actions.

6.1 Share Generation

The generation stage allows minimal variation and choice from the user. In this stage, we assume a secret s has previously been selected. A dealer D possesses s and selects the values for t and n. The dealer may or may not be a participant in the scheme. Regardless, the dealer provides t, n, and s to a tool that follows the steps for Share Generation defined in Appendix A, resulting in the generation of n shares. After these shares have been generated, the device should securely delete all r_i's (which it created) while the dealer deletes all copies of s.

The choices and actions required of the dealer at this stage consist of selecting appropriate values for t and n.

Choice: Determine Parameters. When generating shares, the dealer chooses the appropriate threshold and number of participants. As these choices are highly context dependent, the dealer is trusted to make these choices taking into account their respective threat model.

An adversary in this setting can leverage poor or uninformed choices of n and t to gain unauthorized access to or prevent participants from accessing s. For example, an adversary hoping to prevent a group of journalists from accessing the sensitive information need only destroy $x > n - t$ shares to prevent journalists from accessing the sensitive information in the future.

Choosing t and n requires identifying the trade-off in prioritization for availability and t-separation of privilege or risk of collusion. A larger value for t (for fixed n) increases the number of participants that can collude without learning the secret, while lower t increases the number of participants that can be unavailable while still keeping the secret in a recoverable state. The value of n, on the other hand, is likely to be determined by

Share Generation

1. Choice: The dealer chooses values for n and t.
2. Device: Let the secret space be $\mathcal{S} = GF(q)^\ell$, where q is a prime or a prime power, $q \geq n+1$, and $\ell \geq 1$. Let $s \in \mathcal{S}$ be the secret.
3. Device: Selects $t-1$ values independently and uniformly at random from $\mathcal{S}$ as $r_1, \ldots, r_{t-1}$ and sets $f : GF(q) \to \mathcal{S}$ as $f(x) = r_{t-1}\, x^{t-1} + r_{t-2}\, x^{t-2} + \cdots + r_1\, x + s$.
4. Device: Generates shares $s_i = (a_i, f(a_i))$ for $1 \leq i \leq n$, where the a_i are arbitrary distinct non-zero elements of $GF(q)$.
5. Device: Delete r_i's.
6. Action: Delete all copies of s.

Share Distribution

1. Choice: Select n participants (possibly including the dealer).
2. Choice: Select a secure communication channel (in person, Signal, etc.).
3. Action: The dealer distributes $s_i = (a_i, f(a_i))$ to participant P_i for $1 \leq i \leq n$.
4. Action: Delete each s_i from the dealer's device. Exception is if the dealer is a participant and keeps one share.
5. Choice: Each participant selects an appropriate storage mechanism for their share.
6. Action: Each participant stores their share in the selected storage mechanism.

Reconstruction

1. Choice: Select a communication channel to bring t or more shares together.
2. Action: P_r and the contacted participants authenticate one another.
3. Choice: Contacted participants elect whether to proceed and participate in a reconstruction.
4. Action: If proceeding, a contacted participant sends their share to P_r.
5. Device: Combine the t or more shares using polynomial interpolation to recover the secret $s = f(0)$.

Fig. 2. Ceremony Framework for Base Mode of Operation

Table 2. Network Model: Properties of Communication Channels

●=achieved; ◐=potential loss; ○=not achieved

	Confidentiality	Integrity	Info. Theor. Sec.
In-person	●	○	●
Signal	●	●	○
TLS 1.3	●	●	○
PGP	◐	●	○
SMS	○	○	○

context—specifically by how many trusted participants are available, as opposed to n being easily chosen.

Action: Perform Secure Deletion. Secure deletion is always required when performing share generation. Verification that secret material has been securely deleted[3] is difficult, thus for our analysis we work under the assumption of the erasure mode defined in Section 5.5. Achieving the desired security properties of the threshold ceremony requires the dealer to delete s and all r_i's after generating shares. This fact demonstrates the higher level of trust required in the dealer beyond that of the other participants.

If the dealer fails to delete s and the r_i's off their machine it becomes an easy and highly profitable target for an adversary. This single point of failure allows the adversary to bypass reconstructing t shares and instead target the dealer's machine. Thus, the presence of s on the machine of the dealer presents a formidable risk and underscores the necessity of secure deletion.

6.2 Share Distribution

Share distribution determines who receives shares and how the shares are transmitted to participants. The responsibility of the dealer includes selection of participants, selection of a secure communication channel, and transmission of shares over this channel.

After receiving their share, a participant is responsible for selecting a safe storage mechanism for the share until required for the Recovery stage. After all shares are distributed, the dealer securely deletes all shares from their device, with the exception of their own share, if applicable.

Choice: Select Secure Channel. Shamir secret sharing assumes the existence of a secure communication channel. However, the dealer holds responsibility to assess and choose an appropriate channel where all aforementioned security properties hold. Unsurprisingly, users often struggle to make safe choices when using security-critical tools in similar contexts [44]. Communication channels that could be used in practice which are not information-theoretically secure include TLS [37], Signal [33],and PGP [7], while in-person communication achieves information theoretic security. Notably, each of these transmission methods achieve divergent security properties when used in a secret sharing ceremony. TLS and Signal support confidentiality and integrity assuming that participants authenticate one another before sending any messages. In-person communication achieves confidentiality but does not support integrity, due to the lack of a defined integrity mechanism. While PGP encrypts data in transit, thereby achieving confidentiality at the moment data is transmitted, PGP is not forward-secure. Consequently, PGP does not strictly preserve confidentiality of future transmitted data in the case that a user's private key is compromised. Cellular networks' Short Message Service (SMS), while commonly used for security protocols such as two-factor authentication [24], does not achieve any of our desired properties. We provide a summarized analysis of the security properties of various channels in Table 2.

Choice: Select Participants. In a (t, n)-threshold scheme, the dealer is responsible for selecting which participants are entrusted with shares. The dealer in some cases is also free to decide if they will become a participant dealer D_p and retain a share for themselves. Choosing appropriate participants is heavily context dependent and influenced by the users' threat model.

Action: Perform Secure Deletion. As with Share Generation, there is a need for secure deletion. After distributing the shares the dealer must delete all shares that are not their own from their device. Failure to do so provides an adversary with a single target to gain the secret s just as leaving the secret itself would.

6.3 Reconstruction

This stage occurs when a valid participant chooses to initiate a recovery. The participant P_r performing the recovery contacts and authenticates other participants, who authenticate P_r as a valid participant. These participants then decide for themselves whether reconstruction is appropriate and whether to participate at that

3 For further details on existing secure deletion solutions see the analysis from Reardon [36].

time. If so, they transmit their share over a secure channel. Once P_r possesses t or more shares, P_r can perform reconstruction using a tool for polynomial interpolation to extract the secret s.

Choice: Select Secure Channel. Performing a recovery of s assumes a secure communication channel to transmit shares from other participants to P_r.

Action: Perform Authentication. During recovery it is left to the participants to authenticate each other even assuming a secure channel. For instance, the participants in the scheme need to know whether or not it is permissible for the initiating participant to perform a recovery. In one example, from Section 3, Alice left the organization and Bob determined it was okay to include Alice in the recovery and contacted her. However, in another setting we can imagine Alice left the organization and initiated a recovery by requesting a share from Carol. Without a revocation mechanism, there is nothing preventing Alice from recovering the secret if Carol provides her with a share. Therefore, even after losing authorization, Alice can learn the secret and break confidentiality. Thus, in this latter setting, the ceremony as stated is insecure. Such an insecurity is an example of how a ceremony secure in one case may be insecure in another, and so it is important to specify the ceremony as part of the security analysis, as opposed to just analyzing the underlying protocol. Additionally, we will address this particular insecurity in Section 9.

7 Extended Mode Stages

The Extended Mode is one possible extension of Shamir secret sharing, and is a practical use case for users seeking to protect sensitive information that is large in size. For example, Sunder requires operating in the Extended Mode when the secret is larger than 1 MB [18].

We now formalize the choices and actions users must make in the Extended Mode. We introduce the stages *Secret Preparation* which is performed before the Base Mode Share Generation stage, and *Extended Reconstruction* which is performed after the Base Mode Reconstruction stage; see Figure 3 for an outline.

7.1 Secret Preparation

Secret Preparation begins with an existing plaintext and a secret key. Using this key, the plaintext is encrypted via a symmetric encryption algorithm, and the output of the ciphertext is stored using safe storage (see Section 5.5) for later use in the Reconstruction stage. This secret key is subsequently used as the input s into the Share Generation phase.

Secret Preparation
1. Device: Generate a secret key to be used as s.
2. Device: Encrypt $\mathcal{F}$ using s and an appropriate authenticated encryption algorithm.
3. Choice: Select a safe storage mechanism for the ciphertext.
4. Action: Safely store the ciphertext.

Extended Reconstruction
1. Action: Acquire ciphertext from selected safe storage.
2. Device: Use recovered s to decrypt the ciphertext.

Fig. 3. Ceremony Framework Additions for Extended Mode of Operation

Action: Generate Secret Key. The sensitive information $\mathcal{F}$ should be encrypted using the chosen authenticated encryption algorithm and a secret key generated by the dealer. Note that authenticated encryption does *not* provide end-to-end integrity against an attacker that is able to acquire s. In that event, an adversary could modify the stored ciphertext without detection. We present a way to block this attack in Section 9.

Choice: Select Safe Storage. Even if s has been successfully reconstructed, the user must have chosen a reliable storage mechanism (see Section 5.5 for requirements of safe storage) to recover the ciphertext of $\mathcal{F}$ after performing the Reconstruction stage.

7.2 Secret Recovery

After s is recovered in the Reconstruction stage, the participant initiating the recovery can retrieve the ciphertext of $\mathcal{F}$ from the chosen storage location, use the secret s as a key for the symmetric encryption algorithm for decryption, and produce the original sensitive information $\mathcal{F}$.

For this stage, the probability of successfully recovering $\mathcal{F}$ is dependent on choices made by the user in the Secret Preparation stage; for example, if the user did not choose an adequate storage mechanism, the user may fail to recover $\mathcal{F}$ in the Secret Recovery stage.

8 Application of Ceremony Framework Analysis

We now apply our ceremony framework to aid the security analysis of threshold schemes, as specified in Section 5. We present three case studies to highlight how seemingly straightforward implementations can achieve or miss assumed security goals.

8.1 Defined Threat Model

We maintain a specific threat model for our security analysis of each case study.

Adversaries. We assume a high-powered adversary (defined in Section 5.1) which has access to exceptional computational power, time, and money, along with significant legal resources. We do not assume fixed roles for participants; a once-trusted participant can become an adversary at a later time.

Security Goals. We evaluate each case study against the sample security goals defined in Section 5.3. Specifically, we evaluate the security goals of t-separation of privilege, availability, information theoretic security, confidentiality, and integrity against the above-defined adversary.

8.2 Case Study One: Classic Shamir Threshold Scheme

Classic Shamir secret sharing is not a complete protocol and by extension not a complete ceremony. Unsurprisingly, classic Shamir secret sharing in isolation cannot achieve all desired security properties. A summary of the analysis detailed below for classic Shamir secret sharing (and indeed all of the ceremonies we analyze) can be found in Table 3.

t-Separation of Privilege is achieved by classic Shamir secret sharing. Within the Extended Mode, an adversary with access to the ciphertext still requires at minimum t shares to decrypt the ciphertext.

The loss of *Availability* in Extended Mode demonstrates how security properties can be lost when moving from Base Mode to a seemingly innocuous extension of Shamir secret sharing. In the Base Mode, availability is preserved as long as $t < n$. In the Extended Mode, the loss of the ciphertext renders the secret unavailable, regardless of the number of shares that remain available. As the protocol does not define a safe storage mechanism to protect against loss of the ciphertext, this becomes a single point of failure.

Information theoretic security is achieved in theory by the mathematics of classic Shamir secret sharing. When evaluating the scheme in practice, we must consider the channel used to transmit shares to participants. We grant a half-circle in the table for information theoretic security in Base Mode as the protocol can remain entirely offline if desired, requiring shares to be transmitted in person or via a trusted physical channel. However, when operating in online mode, shares are transmitted over an online channel. As online communication channels rely on encryption protocols that are not information theoretically secure, classic Shamir secret sharing loses information theoretic security when used with an online channel. Furthermore, as Extended Mode requires a symmetric encryption algorithm, working within the Extended Mode similarly is not information theoretically secure.

Confidentiality is not achieved in either the Base or Extended Mode of operation due to the lack of a revocation mechanism. Once shares have been distributed, classic Shamir secret sharing does not consider the case where a once-trusted participant moves to an untrusted state, such as by voluntarily or involuntarily leaving an organization. For example, nothing prevents a participant who was fired, but possesses a valid share, from participating in a future recovery protocol by colluding with other participants who similarly may or may not be currently within a trusted state in the organization. We therefore only grant half-circles in the table for all of these cases, as they only achieve confidentiality as long as there is no need for revocation.

Integrity of shares is not a goal that classic Shamir secret sharing guarantees. In some settings, for example, $t = 2$ but four shares are available during the reconstruction phase, the correct secret can be determined if a limited number (in this case, one) of shares is corrupted or maliciously changed. However, using these techniques for integrity requires raising the required number of shares during reconstruction, as well as assumptions about the number of corrupted shares. Ideally, a separate integrity check for the shares would enable the detection of corrupted shares directly. Once detected and identified, corrupted shares would be excluded from the recovery and, if necessary, additional validated shares could be included; we provide this functionality in our extensions in Section 9. Note that in the Extended Mode the ciphertext carries its own integrity check, but that check is not entirely sufficient, as discussed in Section 7.1.

Table 3. Ceremony Analysis Summary

●=achieved; ◐=ceremony dependent; ○=not achieved

IT-Sec=Information Theoretic Security

	Classic Shamir				Sunder Ceremony				Shatter Secrets		Our Proactive VSS			
	Base		*Ext*		*Base*		*Ext*		*Ext*		*Base*		*Ext*	
	HBC	MAL	HBC	MAL	HBC	MAL	HBC	MAL	HBC	MAL	HBC	MAL	HBC	MAL
t-Sep. Priv.	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Availability	●	●	○	○	●	●	◐	◐	○	○	●	●	●	●
IT Sec.	◐	◐	○	○	○	○	○	○	○	○	○	○	○	○
Conf.	◐	◐	◐	◐	◐	◐	◐	◐	●	●	●	●	●	●
Integrity	○	○	○	○	●	●	◐	◐	◐	◐	●	●	●	●

8.3 Case Study Two: Sunder

Freedom of the Press Foundation's tool Sunder [17, 18] is a desktop application for journalists to generate a configurable number of key shares for encrypted documents. While this tool supports both Base and Extended Modes of operation and can accommodate a wide range of threat models and ceremonies, we bound our analysis to the online setting with a high-risk threat model and high adversary capabilities. We define an explicit ceremony for Sunder in Appendix B.

Sunder is a straightforward implementation of classic Shamir secret sharing, and many of the security properties for classic Shamir secret sharing apply to Sunder. In the secret-sharing implementation used by Sunder [42], every character in the ℓ-character secret provided to Sunder becomes an input into a Shamir protocol acting over the Galois Field $GF(256)$. (Equivalently, the secret as a whole is treated as an element of the vector space $GF(256)^{\ell}$.) Each of the n shares will then be ℓ bytes long. One deviation worth highlighting is Sunder's support for share integrity. The underlying cryptographic library [42] that Sunder utilizes provides share integrity by generating a public-private ephemeral key pair to sign shares during the Generation stage. Share signatures are validated during the Recovery stage to ensure both the validity of shares and also that all shares are signed by the same public key. However, Sunder does not include document encryption within the tool and thus does not support integrity validation for the encrypted documents; consequently, Sunder only partially attains integrity for Extended Mode.

Sunder can achieve confidentiality if no more than t participants holding valid shares leave the organization, however, Sunder is limited to confidentiality without revocation and thus only has a half-circle in the table.

Finally, while Sunder provides availability in Base Mode, it leaves to the user the decision of how to store the encrypted file in Extended Mode. If the file is not safely stored, availability could be compromised.

8.4 Case Study Three: Shatter Secrets

Shatter Secrets [1] is an open-source protocol that uses Shamir secret sharing of a key that encrypts a user's device. Shatter secrets can be used to distribute shares (each encrypted with a key held by the user) to other devices and friends of the device owner such that a threshold number is required to decrypt the device. A device owner crossing an international border can encrypt their device using Shatter Secrets such that they are unable to decrypt their device without the physical presence of a threshold number of participants holding shares. We define an explicit ceremony for Shatter Secrets in Appendix C.

The shares in Shatter Secrets are themselves encrypted by a key held by the primary data owner (on a secondary device). Violating confidentiality thus requires compromising the encrypted primary device, the secondary device storing the share decryption key, and a sufficient number of the friends' devices storing the encrypted shares. Specifically, unlike in Sunder, shareholders alone cannot recover the secret. Thus confidentiality of the device's data is preserved. Availability of the device's data, however, can be easily compromised as authorities can seize the encrypted device (assuming the device owner does not have a backup). Shatter Secrets' design does not adequately provide the property

of availability as it prioritizes confidentiality such that the device can be a single point of attack. The authenticated encryption of the shares themselves provide integrity for the shares, but the integrity of the encrypted device depends on whether that encryption mechanism can detect modifications to the ciphertext.

9 Lightweight Integratable Improvements to Shamir Secret Sharing

As can be seen from our example case studies, implementations based on classic Shamir secret sharing have gaps limiting the security properties, such as confidentiality and integrity in Base Mode and availability in Extended Mode. To address these gaps, we introduce a lightweight set of improvements which are fully compatible with classic Shamir secret sharing. These improvements are extensions to Shamir secret sharing and can be applied to implementations of Shamir secret sharing. Using our framework for ceremony security analysis, we assess the security properties provided by these improvements within the Base and Extended Modes.

9.1 Overview

We define a lightweight Proactive Verifiable Secret Sharing (Proactive VSS) scheme and specify three new stages: Share Update, Share Validate, and Generate Commitment. These stages provide participants with the capability to update shares and revoke access to individuals who are no longer trusted. Users can verify the integrity of shares and verify the integrity of $\mathcal{F}$.

We use the protocol and model of Herzberg et al. [22], in which adversarially controlled players during an update stage count against both adjacent epochs. Alternatively, one could use the protocol and somewhat stronger adversarial model of Nikov and Nikova [30], at the cost of requiring the dealer to select t knowing that $t-1$ corrupted players could compromise the secret, but t are needed to reconstruct it.

Assumptions. Our commitment scheme assumes a sufficiently random s, such as a key. If s is not sufficiently random, several of our improvements can still be used; see Section 9.4.

Share Generation
1. Steps 1–4 as before in Figure 2
2. Device: Generates a commitment $\vec{C} = \langle \phi_0, \ldots, \phi_{t-1} \rangle$, where $\phi_0 = g^s$, and $\phi_j = g^{r_j}$ for $1 \leq j \leq t-1$.
3. Choice/Action: The dealer publishes $\vec{C}$ to a trusted public location all participants can access.
4. Steps 5–6 from Figure 2

Share Distribution
1. Steps 1–6 unchanged from Figure 2

Share Validation
1. Action: The participant fetches the commitment $\vec{C}$ from its trusted public location.
2. Device: Using $\phi_0, \ldots, \phi_{t-1}$ which constitute $\vec{C}$, the participant will then calculate ψ by evaluating $\prod_{j=0}^{t-1} \phi_j^{a_i{}^j}$.
3. Device: The participant can then validate the correctness of her share by validating ψ is equal to $g^{f(a_i)}$.

Share Updates
1. Action: $\mathcal{U}$ executes the Share Generation stage, with unchanged values for t and n, to generate a new polynomial $h(x)$ where s=0. For each authorized participant holding a share $a_i, f(a_i)$, use $h(x)$ to generate a share as the update $u_i = (a_i, h(a_i))$.
2. Action: $\mathcal{U}$ publishes the updated commitments to the trusted public location.
3. Action: $\mathcal{U}$ distributes the update $u_i = (a_i, h(a_i))$ to the (authorized) participant with share $(a_i, f(a_i)$ for $1 \leq i \leq m$, where $m \leq n$.
4. Action/Device: Each participant will apply the share update u_i, to their share s_i to produce $s_i = (a_i, f(a_i) + h(a_i))$.

Reconstruction
1. Step 1 as before in Figure 2
2. Action/Device: P_r ensures the validity of each share using the public commitment $\vec{C}$.
3. Step 2 from Figure 2

Fig. 4. An Improved Base Mode Ceremony via Verifiable Secret Sharing (VSS) and proactive share updates

9.2 Base Protocol Description

Our modifications can be used in conjunction with an existing secret sharing implementation as demonstrated by the modifications to the stages indicated in Figure 4.

Modified Share Generation. In addition to the original steps, the dealer $\mathcal{D}$ generates a commitment $\vec{C}$ to the polynomial. The j^{th} index of $\vec{C}$ is equal to g^{r_j}, where r_j is the randomly selected coefficient, while the zeroth element in $\vec{C}$ is equal to g^s. (Here, g is the generator of a group in which the Decisional Diffie-Hellman problem is hard.) The dealer publishes $\vec{C}$ to a trusted location that every participant can access. Examples of such a location include a commitment verification party, a public blockchain, or some other location all participants can reliably view in the same state. Note that the choice of trusted location is influenced by the powers of an adversary—a trusted Twitter account could be effective against a low-powered adversary, but not a high-powered adversary.

Share Validation. Once $\vec{C}$ has been published, any participant P_i can validate the integrity of her share $s_i = (a_i, f(a_i))$, where $f(a_i)$ is the value generated for participant i using the assigned value a_i and the dealer-selected function f. Validation requires first fetching the public commitment and computing $\psi = \prod_{j=0}^{t-1} \phi_j^{a_i{}^j}$ where $\phi_0 = g^s$, and $\phi_j = g^{r_j}$ for $1 \leq j \leq t-1$. Next, each participant validates that $g^{f(a_i)}$ is equal to ψ, where $f(a_i)$ is taken from their respective share.

Share Updates. Share updates use commitments to zero to preserve the original secret value upon secret reconstruction. To perform an update on m shares, where $t \leq m \leq n$, the participant performing the update assumes the temporary role of the updater $\mathcal{U}$, where $\mathcal{U}$ can be any valid participant. If $m < n$, then there will be shares that do not receive an update and therefore are effectively revoked.

The updater $\mathcal{U}$ first generates m share updates and one commitment update. The set of share updates is generated by running the Share Generation stage with $s = 0$. (Let the polynomial used in this stage be $h(x)$.) This step ensures that shares from a prior epoch cannot be used in conjunction with updated shares in the next epoch to reconstruct s. Thus, shares can be proactively "rotated" forward while protecting the original secret.

After Generation, $\mathcal{U}$ distributes the m share updates to the selected participants. Additionally, $\mathcal{U}$ must apply the commitment update to the original commitment by performing pointwise multiplication between the original commitment to $f(x)$ and a new commitment to the new polynomial $h(x)$. Consequently, $\mathcal{U}$ must be able to safely access the commitment such that the commitment update can be securely applied.

Upon receiving a share update, each participant updates their share by computing $(a_i, f(a_i)+h(a_i))$, where $s_i = (a_i, f(a_i))$ is their original share and $u_i = (a_i, h(a_i))$ is the received update. After computing the updated share each participant performs the Share Validation stage with their updated share and the updated commitment to ensure the integrity of their updated share. If share validation fails, the participant should delete the updated share and continue with their old share. If the new share is valid, the participant should delete the old share and store the updated share for future use.

Reconstruction. Reconstruction of s is as described in a classic Shamir threshold scheme. However, before recovering the secret, the participant performing the recovery first executes the Share Validate stage for each share. If a share is invalid, the Reconstruction stage requires the acquisition of a replacement share such that a set of t valid shares is produced.

Secret Preparation

1. Steps 1–2 unchanged from Figure 3
2. Action: Generate integrity value for the ciphertext.
3. Choice: Select a safe storage mechanism for the ciphertext.
4. Action: Store the ciphertext.

Share Distribution$^+$

1. Steps 1–3 unchanged from Figure 2
2. Action: The dealer distributes ciphertext integrity value, along with a copy of the ciphertext, to P_i for $1 \leq i \leq n$.
3. Steps 4–6 from Figure 2
4. Action: Each participant stores their copy of the ciphertext integrity value with their share.

Extended Reconstruction

1. Action: Acquire ciphertext from selected location.
2. Action: Verify integrity value.
3. Device: Use recovered s to decrypt the ciphertext.

Fig. 5. Improved Ceremony Framework for an Extended Mode of Operation

9.3 Extended Protocol Description

Figure 5 summarizes the additional steps of our improved extended mode ceremony for Secret Preparation, Share Distribution$^+$, and Extended Reconstruction.

Secret Preparation. Secret Preparation now includes the generation of a separate integrity value for the ciphertext to be distributed in Share Distribution$^+$. This can be as simple as a collision-resistant hash of the ciphertext.

Share Distribution$^+$. Additional steps are added to the share distribution stage to enable ciphertext integrity in the Reconstruction stage. In addition to the shares, the above integrity value is distributed to each participant. Using a separate integrity primitive checked by the reconstructor during the Reconstruction stage protects integrity even against adversaries who know s. Each participant stores the integrity value along with their share in the selected safe storage mechanism.

Extended Reconstruction. Before decrypting the ciphertext as required, the participant performing the reconstruction checks their copy of the integrity value against the ciphertext.

9.4 Security and Limitations

We now apply our framework from Section 5 to assess the use of our defined improvements. We maintain the identical threat model and security goals as in our case studies from Section 8. We assume a high-powered adversary and desire the security goals of t-Separation of Privilege, Availability, Information Theoretic Security, Confidentiality, and Integrity.

Our improvements (summarized in Figures 4 and 5) guarantee Availability, Confidentiality, and Integrity of shares and the secret information for both the Base and Extended Modes of operation. Integrity is achieved due to the use of a proactive VSS scheme for share verification. We will now discuss each security goal in more depth and how it is achieved.

In Extended Mode, availability of the ciphertext is achieved via redundancy (defined in Step 2 of Share Distribution$^+$ in Figure 5). By distributing the ciphertext to each participant, n independent copies are made while only one is required to recover the secret $\mathcal{F}$. This approach falls within our assumed trust model, as giving a copy of the ciphertext to participants who are trusted with a share does not result in additional powers or capabilities for these participants. In this case, the safe storage mechanism becomes the set of participant devices which store the copy of the ciphertext, achieving availability via redundancy.

The 'Share Updates' stage allows for removing participants from participating in a future recovery. Removing participants enables the preservation of confidentiality with revocation.

For the Extended Mode, we require the dealer to distribute the ciphertext integrity value to each participant, as defined in Figure 5 (Step 2 of Share Distribution$^+$). Distributing the ciphertext integrity value to each participant allows for any participant performing the Share Recovery stage to verify the integrity of the ciphertext while the integrity vector $\vec{C}$ is used to verify individual shares.

Requiring a Sufficiently Random Secret. Note that the commitment scheme for Base Mode requires a sufficiently random s. In the case that s is not sufficiently random, the entropy of s can be increased by moving to the Extended Mode by encrypting the low-entropy secret with a high-entropy key. Alternatively, the dealer can pad s with a sufficiently large number of random bits (e.g., 256 bits). As a final requirement on secrets and shares having sufficient entropy, we highlight the importance of generating secrets and shares on a machine with sufficient entropy sources to prevent amplifying an adversary's guessing attack [16].

9.5 Implementation

The techniques we employ in our implementation, specifically for proactive verified secret sharing, are derived from those of Herzberg et al. [22]. Our implementation differs slightly from their Share Update function, which requires *every* server in the threshold scheme to generate a new update value and distribute it to each other server in the system. In our protocol, any participant may generate an update and send it (noninteractively) to the other participants; the correctness of the update can be verified from the commitments. These other participants may be online or offline. If they are offline they will perform the update when they come back online. If a participant does not trust an update they can initiate another update. Finally, we do not require all participants to perform the update generation, or to be online. Therefore, our derived implementation (as initiated by any one participant) is noninteractive.

Our implementation is in Rust and uses curve25519-dalek [25] for group operations, which we make publicly available (https://crysp.uwaterloo.ca/software/vss/). Group operations are performed in Edwards form

for speed and safety properties. The benefit of using Rust for our implementation is multi-language interoperability and memory safety. Furthermore, our changes can be integrated with implementations of threshold schemes in Rust such as RustySecrets [42] and Sunder [17].

10 Conclusion

Although the theoretical study of secret sharing protocols began decades ago, the use of secret sharing schemes in practice remains poorly defined. Interest in standardization of practical implementations of threshold cryptography is growing, including by NIST [6]. However, to enable practical use, researchers and practitioners must account for gaps in security that arise when moving from a theoretical setting to a real-world application. As a step towards practical secret sharing, we present a framework to facilitate the security analysis of threshold schemes based on Shamir secret sharing. We distinguish between operating in a base or an extended mode of operation, and through case studies, we demonstrate that variations in the ceremony of secret sharing schemes can lead to changes in the fundamental security properties provided to end users. Our framework can aid the design and analysis of future implementations of secret sharing by providing a more detailed ceremony definition and accounting for previously undefined assumptions about adversaries, user roles, and user actions or decisions within the scheme. Finally, we introduce and implement a secret-sharing protocol with improved security properties that can be directly integrated with existing Shamir secret sharing implementations.

Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments and feedback. We gratefully acknowledge the support of NSERC for grants RGPIN-03858, RGPIN-05849, RGPAS-507908, CRDPJ-531191, CRDPJ-534381, and DGDND-00085, the NSERC Alexander Graham Bell Canada Graduate Scholarship program, and the Royal Bank of Canada for funding this research. This research was undertaken, in part, thanks to funding from the Canada Research Chairs program.

References

[1] Erinn Atwater and Ian Goldberg. Shatter Secrets: Using Secret Sharing to Cross Borders with Encrypted Devices. In *Cambridge International Workshop on Security Protocols*, pages 289–294. Springer, 2018.

[2] Erinn Atwater and Urs Hengartner. Shatter: Using Threshold Cryptography to Protect Single Users with Multiple Devices. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 91–102. ACM, 2016.

[3] James Ball. Unredacted US embassy cables available online after WikiLeaks breach. https://www.theguardian.com/world/2011/sep/01/unredacted-us-embassy-cables-online. Accessed 2019-05-29.

[4] James Ball. WikiLeaks publishes full cache of unredacted cables. https://www.theguardian.com/media/2011/sep/02/wikileaks-publishes-cache-unredacted-cables. Accessed 2019-05-29.

[5] Elana Beiser. Record Number of Journalists Jailed as Turkey, China, Egypt Pay Scant Price for Repression. *Committee to Protect Journalists*, 2017.

[6] Luís T.A.N. Brandão, Nicky Mouha, and Apostol Vassilev. NISTIR 8214 Threshold Schemes for Cryptographic Primitives. https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8214.pdf, 2019. Accessed 2019-05-24.

[7] Jon Callas, Lutz Donnerhacke, Hal Finney, David Shaw, and Rodney Thayer. OpenPGP Message Format. https://tools.ietf.org/html/rfc4880, November 2007.

[8] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 136–145. IEEE, 2001.

[9] Marcelo Carlomagno Carlos, Jean Everson Martina, Geraint Price, and Ricardo Felipe Custódio. A Proposed Framework for Analysing Security Ceremonies. In *SECRYPT*, pages 440–445, 2012.

[10] Marcelo Carlomagno Carlos, Jean Everson Martina, Geraint Price, and Ricardo Felipe Custódio. An updated threat model for security ceremonies. In *Proceedings of the 28th annual ACM symposium on applied computing*, pages 1836–1843. ACM, 2013.

[11] Department of Homeland Affairs, Australian Government. The Assistance and Access Act 2018. https://www.homeaffairs.gov.au/about-us/our-portfolios/national-security/lawful-access-telecommunications/data-encryption. Accessed 2019-05-20.

[12] Rachna Dhamija, J Doug Tygar, and Marti Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 581–590. ACM, 2006.

[13] Benjamin Dowling and Kenneth G Paterson. A Cryptographic Analysis of the WireGuard Protocol. In *International Conference on Applied Cryptography and Network Security*, pages 3–21. Springer, 2018.

[14] Carl M. Ellison. Ceremony Design and Analysis. *IACR Cryptology ePrint Archive*, 2007:399, 2007.

[15] Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. *Annual Symposium on Foundations of*

Computer Science (Proceedings), pages 427–438, 11 1987.

[16] Diogo AB Fernandes, Liliana FB Soares, João V Gomes, Mário M Freire, and Pedro RM Inácio. Security Issues in Cloud Environments: A Survey. *International Journal of Information Security*, 13(2):113–170, 2014.

[17] Freedom of the Press Foundation. Sunder is a user-friendly graphical interface for Shamir's Secret Sharing. https://github.com/freedomofpress/sunder, 2018. Accessed 2019-05-28.

[18] Freedom of the Press Foundation. Welcome to Sunder. https://sunder.readthedocs.io/en/latest/, 2018. Accessed 2019-05-28.

[19] Tilman Frosch, Christian Mainka, Christoph Bader, Florian Bergsma, Jörg Schwenk, and Thorsten Holz. How secure is TextSecure? In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 457–472. IEEE, 2016.

[20] Ryan Gallagher and Glenn Greenwald. How the NSA Plans to Infect 'Millions' of Computers with Malware. *The Intercept*, 2014.

[21] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust Threshold DSS Signatures. In *EUROCRYPT*, pages 354–371, 1996.

[22] Amir Herzberg, Stanisław Jarecki, Hugo Krawczyk, and Moti Yung. Proactive Secret Sharing Or: How to Cope With Perpetual Leakage. In Don Coppersmith, editor, *Advances in Cryptology — CRYPT0' 95*, pages 339–352, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[23] Markus Jakobsson. The human factor in phishing. *Privacy & Security of Consumer Information*, 7(1):1–19, 2007.

[24] Javelin Strategy & Research. 2017 State of Authentication Report. https://fidoalliance.org/wp-content/uploads/The-State-of-Authentication-Report.pdf, 2017.

[25] Isis Agora Lovecruft and Henry De Valence. https://doc.dalek.rs/curve25519_dalek/, 2018.

[26] Taciane Martimiano, Jean Everson Martina, M Maina Olembo, and Marcelo Carlomagno Carlos. Modelling user devices in security ceremonies. In *2014 Workshop on Socio-Technical Aspects in Security and Trust*, pages 16–23. IEEE, 2014.

[27] Jean Everson Martina, Túlio Cícero Salavaro de Souza, and Ricardo Felipe Custodio. Ceremonies Formal Analysis in PKI's Context. In *2009 International Conference on Computational Science and Engineering*, volume 3, pages 392–398. IEEE, 2009.

[28] Chris McGreal. Martin Luther King friend and photographer was FBI informant. *The Guardian*, 2010.

[29] Susan E. McGregor, Elizabeth Anne Watkins, Mahdi Nasrullah Al-Ameen, Kelly Caine, and Franziska Roesner. When the Weakest Link is Strong: Secure Collaboration in the Case of the Panama Papers. In *26th USENIX Security Symposium (USENIX Security 2017)*, pages 505–522, Vancouver, BC, 2017. USENIX Association.

[30] Ventzislav Nikov and Svetla Nikova. On Proactive Secret Sharing Schemes. In *International Workshop on Selected Areas in Cryptography*, pages 308–325. Springer, 2004.

[31] Rafail Ostrovsky and Moti Yung. How to Withstand Mobile Virus Attacks (Extended Abstract). In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '91, pages 51–59, New York, NY, USA, 1991. ACM.

[32] Torben P. Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '91, pages 129–140, London, UK, UK, 1992. Springer-Verlag.

[33] Trevor Perrin and Moxie Marlinspike. The Double Ratchet Algorithm. https://signal.org/docs/specifications/doubleratchet/, 2016.

[34] Kenneth Radke, Colin Boyd, Juan Gonzalez Nieto, and Margot Brereton. Ceremony analysis: Strengths and weaknesses. In *IFIP International Information Security Conference*, pages 104–115. Springer, 2011.

[35] Anjana Rajan, Lucy Qin, David W Archer, Dan Boneh, Tancrede Lepoint, and Mayank Varia. Callisto: A cryptographic approach to detecting serial perpetrators of sexual misconduct. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, page 49. ACM, 2018.

[36] Joel Reardon. *Secure Data Deletion*. Springer International Publishing, Cham, 2016.

[37] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. https://tools.ietf.org/html/rfc8446, August 2018.

[38] Jerome H. Saltzer and Michael D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, Sep. 1975.

[39] Bruce Schneier. The Operating System That Can Protect You Even if You Get Hacked. https://freedom.press/news/the-operating-system-that-can-protect-you-even-if-you-get-hacked/, 2014. Accessed 2019-05-14.

[40] Bruce Schneier. Cell Phone Opsec. https://www.schneier.com/blog/archives/2015/04/cell_phone_opse.html, 2019. Accessed 2019-05-24.

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

[42] Spin Research. https://github.com/spinresearch/rustysecrets, 2018.

[43] Frank Stajano. Pico: No More Passwords! In *Proceedings of the 19th International Conference on Security Protocols*, SP'11, pages 49–81, Berlin, Heidelberg, 2011. Springer-Verlag.

[44] Alma Whitten and J Doug Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *USENIX Security Symposium*, volume 348, 1999.

A Shamir Secret Sharing Construction

Share Generation and Distribution

1. Let the secret space be $\mathcal{S} = GF(q)^\ell$, where q is a prime or a prime power, $q \geq n+1$, and $\ell \geq 1$. Let $s \in \mathcal{S}$ be the secret.
2. The dealer D selects $t-1$ values independently and uniformly at random from $\mathcal{S}$ as $r_1, \ldots, r_{t-1}$.
3. The dealer computes $f : GF(q) \to \mathcal{S}$ as $f(x) = r_{t-1}\, x^{t-1} + r_{t-2}\, x^{t-2} + \cdots + r_1\, x + s$.

4. The dealer generates shares $s_i = (a_i, f(a_i))$ for $1 \leq i \leq n$, where the a_i are arbitrary distinct non-zero elements of $GF(q)$.
5. The dealer distributes $s_i = (a_i, f(a_i))$ to participant P_i for $1 \leq i \leq n$.
6. Delete r_i's from the device.

Reconstruction

1. A coalition of t participants combines their shares $\langle s_i = (x_i, y_i)\rangle_{i=1}^{t}$ and performs polynomial interpolation to recover the secret $s = f(0) = \sum_{j=1}^{t} y_j \prod_{1 \leq h \leq t, h \neq j} \frac{x_h}{x_h - x_j}$.

B A Ceremony for Sunder

In the following we define a ceremony based on the information contained within the documentation for Sunder [18]. We use our framework to define the ceremony included in our analysis of Sunder. Note that we have re-categorized steps from our framework, for example device to action, depending on Sunder's implementation of each step.

B.1 Base Mode Stages

Share Generation

1. Choice: The dealer chooses values for n and t.
2. Device: Generates a signature keypair.
3. Device: Generates n shares (of the secret $s \in GF(256)^{\ell}$) $s_i = (a_i, f(a_i))$ for $1 \leq i \leq n$, where the a_i are arbitrary distinct non-zero elements of $GF(256)$. The shares are signed with the signature key.
4. Action: Delete all copies of s and the signature key. (The device retains the public verification key.)

Share Distribution

1. Choice: Select n participants (possibly including the dealer).
2. Choice: Select a secure communication channel (in person, Signal, etc.).
3. Action: The dealer distributes $s_i = (a_i, f(a_i))$ along with the public verification key to participant P_i for $1 \leq i \leq n$.
4. Action: Delete each s_i from the dealer's device. Exception is if the dealer is a participant and keeps one share.
5. Choice: Each participant selects an appropriate storage mechanism for their share.
6. Action: Each participant stores their share in the selected storage mechanism.

Reconstruction

1. Choice: Select a communication channel to bring t or more shares together.
2. Not Defined: P_r and the contacted participants authenticate one another.
3. Choice: Contacted participants elect whether to proceed and participate in a reconstruction.
4. Action: If proceeding, a contacted participant sends their share to P_r.
5. Device: Checks the signature on the received shares.
6. Device: Combines the t or more valid shares using polynomial interpolation to recover the secret $s = f(0)$.

B.2 Extended Mode Stages

Secret Preparation

1. Action: Generate a secret key to be used as s.
2. Action: Encrypt $\mathcal{F}$ using s and an appropriate authenticated[4] encryption algorithm.
3. Action: Store the ciphertext.

Extended Reconstruction

1. Action: Acquire ciphertext from storage.
2. Action: Use recovered s to decrypt the ciphertext.

C A Ceremony for Shatter Secrets

Shatter Secrets only has an Extended mode of operation, which we define below. Shatter Secrets uses devices with Near Field Communication (NFC) capability in order to secret share a key s that is used to encrypt a sensitive drive such as a laptop.

Secret Preparation

1. Action: Generate a secret key to be used as s.
2. Action: Encrypt the sensitive drive using s.

Share Generation

1. Choice: The user chooses values for n and t.

4 Although we specify an authenticated encryption algorithm, this is not specified by Sunder.

2. Device: Generates a symmetric key.
3. Device: Generates n shares (of the secret $s \in GF(256)^{\ell}$) $s_i = (a_i, f(a_i))$ for $1 \leq i \leq n$, where the a_i are arbitrary distinct non-zero elements of $GF(256)$. The shares are encrypted using authenticated encryption using the symmetric key.
4. Action: Delete all copies of s. (The device retains the symmetric key.)

Share Distribution

1. Choice: Select n participants.
2. Choice: Select a secure communication channel (in person, Signal, etc.).
3. Action: The user distributes the encrypted share $E(s_i)$ to participant P_i for $1 \leq i \leq n$.
4. Action: Delete each s_i from the user's device.
5. Device: Each participant's device stores their share.

Reconstruction

1. Action: Meet participant with shares. NFC tap user's device to participant's device to transfer encrypted share. Repeat until t or more shares are retrieved.
2. Device: Decrypt each share with stored symmetric key, discarding unsuccessful decryptions.
3. Device: Combine the t or more valid shares using polynomial interpolation to recover the secret $s = f(0)$.
4. Device: Use recovered s to decrypt the sensitive drive.