Applications of Threshold Signature Schemes for Physical Access Control

John Sahhar
Entropy
United States
john@entropy.xyz

Abstract
Threshold signature schemes (TSS) have received increased attention as demand for applications enabling distributed control increases. Most ongoing discussion, and development, around TSS focuses on protecting digital assets for clients of major financial institutions. Entropy is building a trustless, decentralized asset custodian that utilizes TSS to do just that for individuals, organizations, and institutions alike.

However, few are discussing the application of TSS to physical systems. This non-technical document aims to highlight real world applications for threshold based access control (**TBAC**).

1. Introduction

A threshold signature scheme (TSS) enables a some *t* parties to collectively compute a **digital signature** without learning information about the private key [JPA'20]. Digital signatures produced by traditional signature schemes are used widely in modern applications for authentication. In cryptocurrencies like bitcoin; a digital signature is required to authenticate ownership of a digital wallet for performing privileged actions, such as spending funds for that wallet.

Unlike regular signature schemes that require only one secret key to produce a digital signature. TSS splits this ability across *n* parties and requires some *t* parties where $t <= n$ to each produce a partial signature with their secret share which, in combination with $t - 1$ other partial signature, produces a valid digital signature.

This enables constructing systems that distribute authority for privileged actions of a given key to *t-of-n* parties while retaining the cryptographic security properties of the underlying signature scheme.

A real world example is the usage by financial institutions who offer custodial services. They use TSS to increase the security and distribute access controls for assets held in custody across multiple parties. This is analogous to traditional contractual agreements that require multiple parties to sign a physical document that allocates assets in a particular manner. The difference being TSS applications provide cryptographic security in conjunction with the programmatic control of funds.

2. Key concepts

There are two key concepts of TSS relevant to this document; **Key Generation** and **Threshold Operation**.

Key Generation is the model used to generate and distribute threshold keys to our *n* parties. **Threshold Operation** is the model used for *t* parties to perform any threshold action.

The two models for Key Generation are **Distributed Key Generation** (**DKG**) and **Centralized Key Generation** (**CKG**). In **DKG**, *n* parties work together to generate threshold keys *without revealing their individual key to any other party*. In CKG, one party is responsible for generating and distributing threshold keys to the other *n* parties.

The important distinction between the two models is that in CKG the party who generated and distributed threshold keys can choose to **retain the ability to reproduce any other parties' threshold operation on their own**. Inversely, DKG **disallows any party the ability to reproduce another parties portion of a signature**. Both models have their use-case and a full paper discussing the tradeoffs of each model is well warranted but currently non-existent.

The two models for **Threshold Operations** are **Physical Based Operations** (**PBO**) and **Digitally Based Operations** (**DBO**). **PBO** require that *t* parties be present in a physical location and often involve use of physical hardware authentication devices such as a YubiKey.

DBO has no requirements on the physicality of the *t* parties nor the devices used to generate signature shares.

A financial institution with global clients likely utilizes DBO for custodial applications, whereas a secure facility likely utilizes **PBO** for physical access control.

Before moving forward, let us recognize that all TSS have tunable parameters to suit the needs of each real-world application of TSS. Additionally, it should be noted that any successful implementation of the applications discussed would require significant effort and collaboration of cryptographers, software engineers, hardware vendors, and security researchers.

3. Threshold Based Access Control

Now that we have a basic high level understanding of TSS, the remainder of this paper will outline the optimistic and curious applications of Threshold Based Access Control (TBAC) systems.

The first application of TBAC is the secure-shared locker. Originally, a shared locker used by multiple parties would distribute physical, identical keys to each party and every party retains the ability to access the physical locker for adding or removing items as needed.

Two problems exist with this model. Firstly, any party can access the locker without

informing any other party. Second, each party relies on every other party to maintain the security of their key, as one party having a key compromised results in all parties having access compromised.

TBAC solves both problems with a *t-of-n* **TSS** using **PBO** where $t = n$.

In this system, all parties must perform a PBO, therefore removing the ability for any party to act without knowledge of all other parties. Additionally, ***t-1* keys can be compromised without any locker access compromise**, therefore eliminating the need to trust all parties ability to retain their key securely.

In the event of a key compromise, TSS define key regeneration and key rebroadcasting protocols for both DKG and CKG.

A concrete implementation of this system would involve an analog lock controlled digitally that has interfaces to accept input from hardware based authentication products such as a YubiKey. Each party plugs in their YubiKey and produces their unique signature share, once *t* shares have been produced the digital control system opens the analog lock.

Replacing the locker in this example, we can expand this concept to include physical access control for any location. A research facility may require multiple researchers to be present for sensitive data to be accessed. A construction worksite may require a buddy system for access to a dangerous part of the site for employee safety. Any organization that has the requirement of physical access control can benefit from TBAC.

4. Blocks

While these systems sound interesting in practice, multiple blocks exist before such systems will exist in the real world.

First, lack of hardware support. The most common hardware authentication product is YubiKey, which supports the most commonly used signature schemes. TSS utilize the same underlying signature schemes, but require specific implementations of the underlying scheme to work for TSS. Yubikey supports zero TSS.

Second, Lack of software support. There are few open source libraries available for building TSS applications, all of which are newly developed and some of which are actively maintained. Additionally, the influx of quality academic publications on TSS within the last six months even is quite high, resulting in little time for implementations to catch up.

Third, lack of educated developers. While TSS is certainly on the simpler side in terms of theoretical cryptography, a lack of developers educated enough to securely implement these protocols is apparent.

5. Optimistic views going forward

A large increase in venture capital funding for companies building applications utilizing

TSS has occurred over the past two years. The development of open source libraries is slowly but steadily under way. A NIST call for feedback on TSS recently occurred. Multiple qualified cryptographers actively research the area. The proliferation of hardware authentication devices is ever-increasing, specifically with Apple's announcement of Passkeys, which pushes for increased interoperability for hardware based authentication. The concept of everyone with a smartphone being able to easily take part in applications using TSS by means of their phone would make the potential real world application landscape broaden drastically.

Resources

A Rust library for *t-of-n* TSS with GG'20: https://github.com/axelarnetwork/tofn

A Go library for *t-of-n* TSS with GG'18: https://github.com/bnb-chain/tss-lib

References

[NIST 8214B] Notes on Threshold EdDSA/Schnorr Signatures **https://csrc.nist.gov/publications/detail/nistir/8214B/draft**

[JPA'20] A Survey of ECDSA Threshold Signing https://eprint.iacr.org/2020/1390.pdf