A Survey on Distributed Randomness

Kevin Choi

Spring 2021

Contents

1 Building Blocks

1.1 Commit-Reveal

In the case of a strawman solution, the classical commit-reveal provides an intuitive way to source a random number from a group of nodes by first allowing each node to commit (in the cryptographic sense) to its secret random number and then adding all shares (revealed later) from all nodes to compute the final random number in that round of DRB. The problem with this is that the last person to reveal his share can in fact check the round's output faster than other people and hence can decide not to reveal his number if he doesn't like the round's output, biasing the DRB. This is called the *last revealer attack*.

1.2 Verifiable Secret Sharing

The issue with the classical Shamir's secret sharing scheme is that either the dealer or other participants could in fact be acting maliciously, e.g. Alice (a participant) needs to trust that she has received her share correctly from the dealer while she also needs to trust that other participants' revealed shares are correct.

This issue can be fixed by the notion of verifiable secret sharing (VSS). The idea is that we add an additional verification process to the usual Shamir's secret sharing scheme. Here, we take note of two commonly used VSS schemes: Feldman-VSS and Pedersen-VSS.

1.2.1 Feldman-VSS

The following summarizes a simple VSS scheme (where t among n participants can reconstruct the group secret) proposed by Paul Feldman.

- $\mathbb{Z}_q$ and p are chosen, where primes p, q satisfy: p divides $q - 1$

- $f(x) = \sum_{i=0}^{t-1} a_i x^i$ is randomly selected by the dealer, where $f(0) = a_0$ is the secret

- The shares are $f(1), f(2), ..., f(n)$ and are distributed to n participants, respectively

- Also distributed from the dealer are commitments to coefficients of f modulo p, i.e. $c_j \equiv g^{a_j}$ for $j = 0, ..., t - 1$

- Given her share $f(k)$ and the polynomial coefficient commitments, Alice (a participant) can verify the validity of her share by checking:

$$g^{f(k)} = g^{\sum_{i=0}^{t-1} a_i k^i} = \prod_{j=0}^{t-1} c_j^{k^j} = c_0 c_1^k c_2^{k^2} \cdots c_{t-1}^{k^{t-1}}$$

- Any t number of participants (say) $i = 1, 2, ..., t$ can recover the secret a_0 by performing Lagrange interpolation involving Lagrange coefficients $\lambda_i = \prod_{j \neq i} \frac{j}{j-i}$:

$$a_0 = f(0) = \sum_{i=1}^{t} f(i)\lambda_i$$

What is new here (compared to Shamir's secret sharing scheme) is the inclusion of commitments to polynomial coefficients modulo p. These commitments enable participants to verify the validity of their corresponding shares.

1.2.2 Pedersen-VSS

Reminiscent of Pedersen commitment, Pedersen-VSS is a variation that involves two random polynomials generated by the dealer as opposed to one. Namely, the protocol runs as follows.

- $\mathbb{Z}_q$ and p are chosen, where primes p, q satisfy: p divides $q - 1$

- $f(x) = \sum_{i=0}^{t-1} a_i x^i$ and $f'(x) = \sum_{i=0}^{t-1} b_i x^i$ are randomly selected by the dealer, where $f(0) = a_0$ is the secret (as before)

- The shares are $(f(1), f'(1)), ..., (f(n), f'(n))$ and are distributed to n participants, respectively

- Also distributed from the dealer are commitments to coefficients of f and f' modulo p, i.e. $c_j \equiv g^{a_j} h^{b_j}$ for $j = 0, ..., t-1$

- Given her share $(f(k), f'(k))$ and the polynomial coefficient commitments, Alice (a participant) can verify the validity of her share by checking:

$$g^{f(k)} h^{f'(k)} = \prod_{j=0}^{t-1} c_j^{k^j}$$

- Any t number of participants (say) $i = 1, 2, ..., t$ can recover the secret a_0 by performing Lagrange interpolation involving Lagrange coefficients $\lambda_i = \prod_{j \neq i} \frac{j}{j-i}$:

$$a_0 = f(0) = \sum_{i=1}^{t} f(i)\lambda_i$$

Effectively, what Pedersen-VSS is able to achieve is the decoupling of g^{a_0} (as the public key corresponding to the secret key a_0) and $g^{a_0} h^{b_0}$ (as the published commitment for verification purposes). In other words, the verification process in which the participants verify their shares does not (even information-theoretically)

leak any information regarding the initial secret a_0, a fact that is not true with Feldman-VSS. This subtlety is explained with an example in the scope of distributed key generation in a later section.

1.3 Distributed Key Generation

The motivation for distributed key generation (DKG), which basically comprises n parallel instances of a VSS (run by each participant), is to achieve and utilize a group secret in a leaderless manner such that no participant has access to any of others' shares in the secret sharing process. Consequently, one round of DKG can pave the way for multiple rounds of (say) a threshold signature scheme (e.g. threshold BLS), which is an aspect of DKG that makes it different from PVSS (where each PVSS round involving a reconstruction of some unrevealed share needs to be followed by a new, separate PVSS round).

1.3.1 Pedersen-DKG

Parallel instances of Feldman-VSS, also called Joint-Feldman

1.3.2 Gennaro-DKG

Parallel instances of Pedersen-VSS

1.4 Publicly Verifiable Secret Sharing

While VSS allows verification of the involved shares, the fact of the matter is that such verification can only be done by the involved participants in the secret sharing process. In other words, VSS' verification process is not public. Publicly verifiable secret sharing (PVSS), on the other hand, involves a public verification process, which is useful and desirable in models such as the public bulletin model or blockchain. Here, we delineate Schoenmakers' PVSS scheme and Scrape's optimization of it.

1.4.1 Schoenmakers

A bit different from Shamir's → VSS

1.4.2 Scrape's Optimization

Involves Reed-Solomon codes

1.5 Verifiable Delay Function

1.5.1 Wesolowski's

1.5.2 Pietrzak's

1.5.3 Group Instantiation

RSA group vs class group of $\mathbb{Q}(\sqrt{p})$ (non-trusted setup)

2 Protocols

2.1 PVSS-based

2.1.1 RandHound

One-off protocol

2.1.2 Scrape

Commit-reveal every round, with reconstruction via PVSS

2.1.3 HydRand

Delayed commit-reveal with a leader per round, with reconstruction via PVSS

2.2 DVRF-based

Distributed verifiable random function (DVRF) is quite naturally a distributed version of VRF, where the VRF's secret key is distributed among a group of participants.

Hash of VUF $\rightarrow$ VRF

Threshold signature can be interpreted under this bucket

2.2.1 RandHerd

Involves threshold Schnorr + CoSi (collective signing = multisig aggregation + communication tree) in a fairly complicated way

2.2.2 Dfinity

League of Entropy's drand

Stalling the network? 100% eventual liveness assumption?

2.2.3 DDH-DVRF

Does not use pairings

2.3   VDF-based

2.3.1   Extending Commit-Reveal

Unicorn: pre-VDF
RANDAO + VDF

2.3.2   Extending Public Randomness

Taking some closing stock price or a block hash + VDF

2.3.3   RandRunner

A VDF chain involving a group of trapdoor VDFs

References

[1] Feldman, Paul. "A practical scheme for non-interactive verifiable secret sharing." 28th Annual Symposium on Foundations of Computer Science (sfcs 1987). IEEE, 1987.

[2] Dodis, Yevgeniy, and Aleksandr Yampolskiy. "A verifiable random function with short proofs and keys." International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2005.