KU LEUVEN

FACULTY OF
ENGINEERING SCIENCE

The best master's thesis ever

First Author
Second Author

Thesis submitted for the degree of
Master of Science in Cybersecurity

Supervisor
Prof. dr. ir. Knows Better

Assessors
Ir. Kn. Owsmuch
K. Nowsrest

Assistant-supervisors
Ir. An Assistent
A. Friend

Academic year 2024 – 2025

Published by First Author and Second Author,
Faculty of Engineering Science, Kasteelpark Arenberg 1 bus 2200, B-3001 Leuven

Preface

I would like to thank everybody who kept me busy the last year, especially my promoter and my assistants. I would also like to thank the jury for reading the text. My sincere gratitude also goes to my wive and the rest of my family.

First Author
Second Author

Contents

Preface i

Abstract iv

List of Figures and Tables v

List of Abbreviations and Symbols vi

1 Literature Review 1

2 Preliminaries 3
- 2.1 Notation . . . 3
- 2.2 Coding Theory . . . 3
  - 2.2.1 Reed Solomon Codes . . . 5
- 2.3 Packed Shamir Secret Sharing . . . 6
- 2.4 Sigma Protocols . . . 8
  - 2.4.1 Chaum-Pedersen Protocol for DL Equality . . . 9
  - 2.4.2 NIZK PoK for Polynomial DL . . . 9
- 2.5 Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS) . . . 10
- 2.6 Conclusion . . . 11

3 Packed Pre-Constructed Publicly Verifiable Secret Sharing 13
- 3.1 Definitions . . . 13

4 Revisiting a Randomness Beacon Protocol 15
- 4.1 Computational Complexity . . . 15
  - 4.1.1 Computational Cost analysis . . . 17
- 4.2 Communication Complexity . . . 18
  - 4.2.1 Communication Cost analysis . . . 18

5 Conclusion 21

A The First Appendix 25
- A.1 More Lorem . . . 25
  - A.1.1 Lorem 15–17 . . . 25
  - A.1.2 Lorem 18–19 . . . 26
- A.2 Lorem 51 . . . 26

B The Last Appendix 27
- B.1 Lorem 20-24 . . . 27
- B.2 Lorem 25-27 . . . 28

Bibliography **29**

Abstract

The `abstract` environment contains a more extensive overview of the work. But it should be limited to one page.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

List of Figures and Tables

List of Figures

2.1 Packed Shamir Secret Sharing 7
2.2 Chaum-Pedersen NIZK PoK for DLEQ 9
2.3 A NIZK PoK for Polynomial DL based on Schoenmakers' PVSS 10
2.4 PPVSS Scheme 12

4.1 Commit and Reveal phase of the Randomness Beacon using PPPVSS . 16
4.2 Recovery and Output phase of the Randomness Beacon using PPPVSS 19

List of Tables

4.1 Computational cost of dealer and shareholders, $\mathbb{E}_x$ =group exponentiation and $\mathbb{P}_e$ =polynomial evaluation in group G with order q, where q is a large prime 17
4.2 Communication cost of dealer and (each) shareholder, R_o being the random oracle, G =group of order q and $\mathbb{Z}_q$ = modular group of order q, where q is a large prime 18

List of Abbreviations and Symbols

Abbreviations

DL	Discrete Logarithm
PPT	Probabilistic Polynomial Time
NIZK	Non-Interactive Zero Knowledge
PoK	Proof of Knowledge
AoK	Arguement of Knowledge
PSSS	Packed Shamir Secret Sharing
PVSS	Publicly Verifiable Secret Sharing
PPVSS	Pre-Constructed Publicly Verifiable Secret Sharing
PPPVSS	Packed Pre-Constructed Publicly Verifiable Secret Sharing

Symbols

q	prime number
$\mathbb{G}$	Cyclic group of order q
$\mathbb{Z}_q$	Modular ring with q elements
$\mathbb{Z}_q[X]$	Univariate polynomial ring in the variable X with coefficients in $\mathbb{Z}_q$
$mathbbZ_q$\$[X]$_t$	Set of polynomials in $\mathbb{Z}_q[X]$ of degree t
$\mathbb{Z}_q[X]_{\leq t}$	Set of polynomials in $\mathbb{Z}_q[X]$ of degree at most t
λ	Security Parameter
$negl$	Negligible function
$\mathcal{O}$	Big-O notation

Chapter 1

Literature Review

In 1979, Shamir introduced a threshold secret sharing scheme called Shamir Secret Sharing scheme [18], which is now a well-known and widely used secret sharing scheme to this day because of its numerous applications in cryptography. It was first of its kind to have Information Theoretic (IT) security under certain assumptions against passive adversaries who can only see the secret shares of the parties they have corrupted. In reality, however, the adversaries are usually stronger than just being passive, moreover, they possess the power to manipulate the share values of the corrupted parties itself. Shamir's scheme is not tailored to defend against active adversaries as one cannot verify the correctness of the shares. This led to numerous inventions of Verifiable Secret Sharing (VSS) schemes, which not only does allow the parties to verify the correctness of the shares shared by the dealer but also allows the parties to verify the correctness of the shares when opened by the parties during the reconstruction phase. Because of the feature of verifiability, VSS schemes can defend the applications against active adversaries.

There are many VSS schemes ([9], [10]) in the literature which are based on Shamir Secret Sharing scheme. Throughout the years, many advancements have been made in the field of VSS schemes, and as of writing this report the efficient VSS schemes are Π_F, Π_P and Π_{LA} [3], each of which have distinct security features. In VSS, only shareholders can actually verify the correctness of the shares. Certain applications demand to have verifiability feature available to anyone, which is solved by Publicly Verifiable Secret Sharing (PVSS) schemes. PVSS is an extension of VSS, where the correctness of the shares can be verified by anyone. Many cool applications exist today which use PVSS schemes, such as, e-voting [17], randomness beacons [6], etc. In [4], authors have noticed that the Schoenmakers' PVSS scheme used for the e-voting application in [17] is actually more than a PVSS scheme, and they coined the term Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS) scheme. PPVSS is a special type of PVSS where the dealer additionally publishes a commitment to the secret itself. The authors have also shown that any PVSS scheme can be transformed into a PPVSS scheme with minimal changes, and constructed a PPVSS Λ_{RO} from the PVSS Π_S [3] as an example, where they used Λ_{RO} to build an efficient e-voting application.

With PPVSS, one can build versatile applications and also can improve the efficiency of existing applications. In ALBATROSS [7], authors built a randomness beacon application using a PVSS. We have an intuition that an efficient randomness beacon application can be built using a scheme based on PPVSS on certain conditions. In this report, we will introduce Packed PPVSS (PPPVSS) along with its security proofs and give an example based on Λ_{RO}, which will be used to improve ALBATROSS in many cases.

Chapter 2

Preliminaries

2.1 Notation

Let $(\mathbb{G}, \times)$ be a cyclic group of prime order q with hard Discrete Log (DL) and its generator being g. Also, we write $\mathbb{Z}_q[X]_d$ to denote the set of all d degree polynomials univariate in X with coefficients in the finite field $\mathbb{Z}_q$. For remainder of the chapter we let $n > t$ for some positive integers n and t.

2.2 Coding Theory

This subsection is a brief recall of linear codes and their properties.

Definition 2.2.1 (Codeword)**.** *A* ***codeword*** *of length n is a vector $c \in \mathbb{Z}_q^n$.*

Definition 2.2.2 (Linear Code)**.** *[13] If $\mathcal{C}$ be a vector subspace of $\mathbb{Z}_q^n$ with dimension k, then $\mathcal{C}$ is said to be a* ***linear code****(/ linear $q-$ary code) of length n and dimension k.*

In the remainder of the subsection, we let $\mathcal{C}$ be a linear $q-$ary code of length n and dimension k.

Definition 2.2.3 (Hamming distance)**.** *The hamming distance d of two codewords of equal length is the number of positions at which the codewords differ. Also, the hamming distance of $\mathcal{C}$, $d(\mathcal{C})$ is defined to be the minimum hamming distance of any two codewords in $\mathcal{C}$.*

Definition 2.2.4 (Hamming weight)**.** *The hamming weight wt of a codeword c is the number of non-zero positions in c. Also, the hamming weight of $\mathcal{C}$, $wt(\mathcal{C})$ is defined to be the minimum hamming weight of any codeword in $\mathcal{C}$.*

Lemma 2.2.1. *[13] Given a tuple of codewords of equal length n, (u, v, w), let $d(u, v)$ and $wt(w)$ denote the hamming distance of u, v and the hamming weight of w respectively. Then $d(u, v) = wt(u - v)$ and $d(u, v) \leq d(u, w) + d(w, v)$.*

Definition 2.2.5 (error)**.** *A vector r is said to be an **error** of a codeword $c \in \mathcal{C}$ if $r = c + e$ for some $e \neq 0$ and e is called error term of r.*

It is trivial to observe that the hamming distance of error r of $c \in \mathcal{C}$ is the minimum of the hamming distances of r with each codeword in $\mathcal{C}$.

Definition 2.2.6 (Detectable Error)**.** *An error r of $c \in \mathcal{C}$ is said to be **detectable** in $\mathcal{C}$ if $r \notin \mathcal{C}$, otherwise it is said to be an **undetectable**.*

Theorem 2.2.1. *[13] An error r of $c \in \mathcal{C}$ is detectable if the hamming distance of c and r is less than the hamming distance of $\mathcal{C}$, more precisely $d(r, c) < d(\mathcal{C})$.*

Proof. Consider the negation of the statement, i.e., the hamming distance of c and r is less than the hamming distance of $\mathcal{C}$ and r is an undetectable error in $\mathcal{C}$, mathematically we have $d(r, c) < d(\mathcal{C})$ and $r \in \mathcal{C}$. The distance of any two codewords in $\mathcal{C}$ should be at least $d(\mathcal{C})$ implying $d(r, c) \geq d(\mathcal{C})$, which is a contradiction to the negation of our statement. □

The theorem 2.2.1 says that any error of a codeword in $\mathcal{C}$ is detectable as long as their hamming distance is strictly less than the hamming distance of $\mathcal{C}$ itself.

Definition 2.2.7 (Correctable Error)**.** *A detectable error r of $c \in \mathcal{C}$ is said to be **correctable** if one can obtain its error term e such that $c + e = r$.*

Theorem 2.2.2. *[13] One can find the error term e of the detectable error r of $c \in \mathcal{C}$ if $wt(e) < \frac{d(\mathcal{C})}{2}$.*

Proof. We have the following triangular inequality from lemma 2.2.1 for any $w \in \mathcal{C}$ with $w \neq c$:

$$d(\mathcal{C}) \leq d(w, c) \leq d(w, r) + d(r, c). \tag{2.1}$$

From the equation (2.1), we get

$$d(w, r) \geq d(\mathcal{C}) - d(r, c). \tag{2.2}$$

Since, $w \neq c$ we always will have

$$d(w, r) > d(r, c). \tag{2.3}$$

. From the equations (2.2) and (2.3), we have the following result:

$$d(\mathcal{C}) - d(r, c) > d(r, c) \implies d(\mathcal{C}) > 2d(r, c) \iff d(\mathcal{C}) > 2wt(e).$$

□

If one wants to correct a detectable error of a codeword in $\mathcal{C}$ then from theorem 2.2.2, its hamming distance with the codeword should be strictly less than half the hamming distance of $\mathcal{C}$ itself.

Definition 2.2.8 (Dual Code)**.** *The vector subspace $\mathcal{C}^{\perp}$ is called a Dual (Code) of $\mathcal{C}$ if it is orthogonal to $\mathcal{C}$.*

Definition 2.2.9 (Generating Matrix)**.** *The $k \times n-$matrix $\mathcal{G}$ is said to be a generating matrix of $\mathcal{C}$ if it generates $\mathcal{C}$, more precisely, the rows of G form a basis for $\mathcal{C}$. Also, $\mathcal{G}$ is said to be in its **standard form** if it is of the form*

$$\mathcal{G} = \begin{bmatrix} I_k & P \end{bmatrix},$$

where I_k is the $k \times k$ identity matrix and P is some $k \times (n-k)$ matrix.

Definition 2.2.10 (Parity Check Matrix)**.** *Consider the linear transformation ϕ as follows:*

$$\phi : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n-k},$$

*where kernel of ϕ is $\mathcal{C}$. Then the matrix associated to ϕ, $\mathcal{H}$, is called the **parity check matrix** of $\mathcal{C}$.*

Lemma 2.2.2. *[13] If $\mathcal{G}$ is a generating matrix of $\mathcal{C}$ in its standard form, i.e., $\mathcal{G} = \begin{bmatrix} I_k & P \end{bmatrix}$, then $\mathcal{H}$ being a parity check matrix of $\mathcal{C}$ is given by*

$$\mathcal{H} = \begin{bmatrix} -P^T & I_{n-k} \end{bmatrix},$$

where I_{n-k} is the $(n-k) \times (n-k)$ identity matrix where P^T is the transpose of P.

One can easily check if a codeword is in $\mathcal{C}$ by multiplying it with its corresponding parity check matrix $\mathcal{H}$.

2.2.1 Reed Solomon Codes

Consider the set of all univariate polynomials in X of degree at most t over $\mathbb{Z}_q$, denoted by $\mathbb{Z}_q[X]_{\leq t}$. It is trivial to observe that $\mathbb{Z}_q[X]_{\leq t}$ is isomorphic to the $t+1$ dimensional vector space $\mathbb{Z}_q^{t+1}$, where each vector consists the coefficients of a unique polynomial in $\mathbb{Z}_q[X]_{\leq t}$. Now, consider the following set of codewords determined by the evaluation of the polynomials in $\mathbb{Z}_q[X]_{\leq t}$ at n **distinct** points $x_1, \ldots, x_n \in \mathbb{Z}_q$:

$$RS = \{(f(x_1), \ldots, f(x_n)) : f \in \mathbb{Z}_q[X]_{\leq t}, x_i \neq x_j \text{ for } i \neq j\}. \tag{2.4}$$

Lemma 2.2.3. *Assume $n > t$. The hamming distance of any two codewords in RS is at least $n-t$. Furthermore, RS is a linear code of length n and dimension $t+1$.*

Proof. For $n > t$, saying that the hamming distance of any two codewords is at least $n-t$ is same as saying that the two codewords in RS are equal in at most t positions. Assume by contradiction that there exists two **distinct** t degree polynomials f, g in $\mathbb{Z}_q[X]$ with corresponding codewords in RS are equal in $t+1$ positions, i.e., their hamming distance is $n-t-1$. As $\mathbb{Z}_q$ is an integral domain, any $t+1$ distinct points in the set $\mathbb{Z}_q \times \mathbb{Z}_q$ will determine a unique t degree polynomial in $\mathbb{Z}_q[X]_t$. As a consequence, we should have $f = g$ in $\mathbb{Z}_q[X]$ which is a contradiction.

The remainder of the proof is by a consequence of the first part. More precisely, each codeword in RS determined by a polynomial in $\mathbb{Z}_q[X]_{\leq t}$ is actually a unique representation of the polynomial itself as it consists at least $t+1$ distinct evaluations of that polynomial where $n > t$. That is, RS is isomorphic to $\mathbb{Z}_q[X]_{\leq t}$ as a vector space which has dimension $t+1$. □

Definition 2.2.11 (Reed Solomon Code)**.** *The $q-$ary linear code RS of length n and dimension $t+1$ defined in (2.4) with minimum hamming distance $n-t$ is called a $[n, t+1, n-t]-$**Reed Solomon Code** [15] in $\mathbb{Z}_q$.*

Corollary 2.2.1. *All errors in a $[n, t+1, n-t]-$Reed Solomon (RS) code are detectable if their hamming distances with RS are at most t and $2t < n$. Moreover, all errors of hamming distance with RS being at most t can be corrected if $3t < n$.*

Proof. We have that any error of RS has hamming distance at most t with RS. From theorem 2.2.1, any error of a codeword in RS is detectable if its hamming distance is strictly less than the hamming distance of RS itself, i.e., $t < n-t$, which is equivalent to $2t < n$.

To be able to correct the errors of hamming distance at most t with RS, we need $t < \frac{n-t}{2}$ from theorem 2.2.2 which is equivalent to $3t < n$. □

In this report, we will use $[n, t+1, n-t]-$Reed Solomon code of the following form:

$$RS = \{(f(1), \ldots, f(n)) : f \in \mathbb{Z}_q[X]_{\leq t}\}. \tag{2.5}$$

2.3 Packed Shamir Secret Sharing

(n, t, ℓ)-Packed Shamir secret sharing ([12],[5]) scheme is a threshold secret sharing scheme which is a variant of (n, t)-Shamir's secret sharing scheme [18] where $n > 2t + \ell - 1$. In a nutshell, the $t + \ell - 1$ degree secret polynomial with coefficients in $\mathbb{Z}_q$ which evaluates to ℓ secrets is secret shared amongst n parties such that any $t+\ell$ parties can reconstruct back the secret polynomial. Recall that Shamir's secret sharing scheme requires at least $t+1$ parties to reconstruct the secret polynomial in contrast to the $t+\ell$ parties in the Packed Shamir secret sharing scheme. The scheme is summarized in the Figure 2.1.

One can observe that all the secret shares of a secret polynomial chosen by the dealer form a codeword in $[n, t+l, n-t-l+1]-$RS code. If the adversary is malicious and can corrupt at most t parties, then from corollary 2.2.1, the honest shareholders can detect the errors if $2t \leq n-l$ and moreover all such errors can be corrected if $3t \leq n-l$. Also, one can use the Berlekamp-Welch algorithm [19] to correct the errors.

But Shamir secret sharing scheme is designed particularly to defend against passive adversaries and not against malicious adversaries. A class of threshold secret

Packed Shamir Secret Sharing

Given ℓ secrets to share amongst n parties, where at most t of them can be *(passively)* corrupt, the (n, t, ℓ)-Packed Shamir secret sharing scheme description is as follows:

Sharing Algorithm:

- Dealer constructs the secret polynomial $f \in \mathbb{Z}_q[X]_{t+l-1}$ via the lagrange interpolation by choosing $t + \ell$ elements in $\mathbb{Z}_q$ where ℓ of them are secrets, $\{s_i\}_{i=0}^{\ell-1}$, with $f(-i) = s_i$ for all i and remaining t are chosen uniformly at random in $\mathbb{Z}_q$.
- Each party P_i receives their share $f(i)$ from the Dealer for each $i \in \{1, \ldots, n\}$

Reconstruction Algorithm:

- Any Q set containing at least $t + \ell$ parties can use the lagrange interpolation to compute $\{s_i\}_{i=0}^{\ell-1}$ as follows:

$$s_m = \sum_{i \in Q} f(i) \left[\prod_{j \in Q, j \neq i} \frac{-m - j}{i - j} \right] \quad , m \in \{0, \ldots, \ell - 1\}$$

- The secrets $\{s_i\}_{i=0}^{\ell-1}$ are outputted as the result.

FIGURE 2.1: Packed Shamir Secret Sharing

sharing schemes which are designed to defend against malicious adversaries is Verifiable Secret Sharing (VSS). There are many VSS schemes in the literature, and as of writing this report the efficient VSS schemes based on Shamir secret sharing are Π_F, Π_P and Π_{LA} [3] which are alternatives to the original VSS schemes from Feldman [10], Pedersen [14] and the more recent ABCP [2]. VSS schemes based on Shamir secret sharing allow shareholders to verify the correctness of the shares obtained during both the sharing and reconstruction phases. This enables these VSS schemes to defend against malicious adversaries who can actively corrupt t parties as long as $t \leq \frac{n-1}{2}$ (In contrast to $t < \frac{n}{3}$ in Shamir secret sharing). Publicly Verifiable Secret Sharing (PVSS) is an extension of VSS where anyone can verify the validity of the secret shares during the sharing phase. More recently, Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS)[4] was proposed which is an extension of PVSS. The main tools used in VSS, PVSS and PPVSS schemes are the **Sigma Protocols** which we overview in the next section 2.4.

2.4 Sigma Protocols

The agenda of this subsection is to give a brief formal background about some important primitives used in the PVSS ,Π_S [3], and the PPVSS ,Λ_{RO} [4], schemes. Let X and W be two sets with R being a relation on $X \times W$, and $L = \{x \in X : \exists w \in W, xRw\}$ be the language defined by R where xRw says that w is a witness for a given $x \in L$. Also, let $\mathcal{R}$ be a PPT algorithm such that $\mathcal{R}(1^\lambda)$ outputs pairs (x, w) with $x \in L$ and xRw where λ is a security parameter.

Given a relation R and its corresponding language L, a **Sigma ($\sum$) Protocol** is a 3-round *interactive* protocol between two Probabilistic Polynomial Time (PPT) algorithms, a prover P and a verifier V. For some $x \in L$ with xRw, in the first round P sends a commitment a to V. To which V sends a challenge d to P in the second round and finally P responds back with the response z to V in the third round. V outputs **true** or **false** upon the proof verification on transcript $trans := (a, d, z)$. Informally, with a $\sum -$protocol a prover P tries to convince a verifier V that they know a witness w for a given statement $x \in L$ without revealing any information about w. To state it formally, a $\sum -$protocol is supposed to satisfy *completeness*, *Honest Verifier Zero Knowledge* (HVZK) and *Special Soundness* which are defined as follows.

Definition 2.4.1 (Completeness)**.** *A $\sum -$protocol is said to be **complete** for $\mathcal{R}$ if the verifier V always accepts the honest prover P for any $x \in L$.*

Definition 2.4.2 (HVZK)**.** *A $\sum -$protocol is said to be Honest Verifier Zero Knowledge(**HVZK**) for $\mathcal{R}$ if there exist a PPT algorithm S that simulates trans of the scheme corresponding to a given $x \in L$ with any witness w of x. That is, given $x \in L$,*

$$trans(P(x, w) \leftrightarrow V(x)) \approx trans(S(x) \leftrightarrow V(x)) \qquad \textit{, for any witness } w \textit{ of } x.$$

Where $trans(P(\cdot) \leftrightarrow V(\cdot))$ is the transcript of the $\sum -$protocol amongst P and V and $\approx$ denotes the indistinguishability of the two transcripts.

Definition 2.4.3 (Special Soundness)**.** *A $\sum -$protocol is said to satisfy **Special Soundness** for $\mathcal{R}$, if there exists a PPT extractor $\mathcal{E}$ for any two valid transcripts, (a, d, z) and (a, d', z'), corresponding to a given $x \in L$ with only a unique witness w and $d \neq d'$ such that $\mathcal{E}(a, d, z, d', z')$ outputs the witness w.*

It is shown that a public-coin, complete, HVZK, special soundness $\sum -$protocol can be made into a Non Interactive Zero Knowledge (NIZK) Proof of Knowledge (PoK) or Argument of Knowledge (AoK) in the Random Oracle (RO) model using Fiat-Shamir transform [11]. In the following subsections, we recall two important NIZK PoK schemes which are used in Π_S and Λ_{RO} schemes.

2.4.1 Chaum-Pedersen Protocol for DL Equality

Recall $\mathbb{G}$ being the cyclic group of prime order q with hard Discrete Logarithm (DL). For some $g, h \in \mathbb{G}$ consider the following relation:

$$R_{DLEQ} = \{(g, h, a, b), x : a = g^x, b = h^x\}.$$

In [8], Chaum and Pedersen proposed a NIZK PoK scheme for the DL Equality relation, R_{DLEQ}. Informally, a prover P can convince a verifier V that they know x such that it can be used with both g and h to obtain a and b respectively. This protocol is widely used in many cryptographic applications like threshold decryption, e-voting and Randomness Beacons. We summarize the protocol in Figure 2.2.

Chaum-Pedersen Protocol for DLEQ

Let $(g, h, a, b) \in L_{DLEQ}$ be a statement with its corresponding witness being x where L_{DLEQ} is the language defined by the relation R_{DLEQ}.

Prover

- Samples $r \in_R \mathbb{Z}_q$ uniformly at random and sets $c_1 = g^r$ and $c_2 = h^r$.
- Sets $d \leftarrow \mathcal{H}(a, b, c_1, c_2)$, where $\mathcal{H}$ is an agreed upon Random Oracle (RO).
- Sets $z \equiv r + dx \pmod q$ and returns the proof(/transcript) $\pi := (d, z)$.

Verifier

- Checks if $d \leftarrow \mathcal{H}(a, b, \frac{g^z}{a^d}, \frac{h^z}{b^d})$ and outputs **true** or **false** accordingly.

FIGURE 2.2: Chaum-Pedersen NIZK PoK for DLEQ

2.4.2 NIZK PoK for Polynomial DL

Recall $\mathbb{G}$ being the cyclic group of prime order q with hard Discrete Logarithm (DL) and g being its generator. Consider the following relation for some polynomial $f \in \mathbb{Z}_q[X]_t$ with degree $t < n$:

$$R_{PDL} = \{(g, x_1, \ldots, x_n, F(x_1), \ldots, F(x_n)), f(X) : F(x_i) = g^{f(x_i)}, 1 \leq i \leq n\}.$$

In [3], Baghery formally introduced a NIZK PoK scheme for the Polynomial DL relation, R_{PDL}, which is a generalization of Schnorr's ID protocol [16]. Informally, a prover P can convince a verifier V that they know a t degree polynomial f such that it can be used with g to obtain $F(x_i)$ for $1 \leq i \leq n$. This protocol is used to construct the PPVSS Λ_{RO} [4], which was essential in building an efficient e-voting protocol. We summarize the protocol in Figure 2.3.

A NIZK PoK for Polynomial DL

Let $(g, x_1, \ldots, x_n, F(x_1), F(x_n)) \in L_{PDL}$ be a statement with its corresponding witness being $f \in \mathbb{Z}_q[X]_t$ where L_{PDL} is the language defined by the relation R_{PDL}.

Prover

- Samples $r \in_R \mathbb{Z}_q[X]_t$ uniformly at random and sets $\Gamma_i = g^{r(x_i)}$ for $1 \leq i \leq n$.
- Sets $d \leftarrow \mathcal{H}(F_1, \ldots, F_n, \Gamma_1, \ldots, \Gamma_n)$, where $\mathcal{H}$ is an agreed upon Random Oracle (RO).
- Sets $z(X) \equiv r(X) + df(X) \pmod q$ and returns the proof(/transcript) $\pi := (d, z(X))$.

Verifier

- First checks if z is a t degree polynomial in $\mathbb{Z}_q[X]_t$. If so, they proceed with the next step.
- Checks if $d \leftarrow \mathcal{H}(F_1, \ldots, F_n, \frac{g^{z(x_1)}}{F_1^d}, \ldots, \frac{g^{z(n)}}{F_n^d})$.
- If first two steps are correct then they output **true**, otherwise **false**.

Figure 2.3: A NIZK PoK for Polynomial DL based on Schoenmakers' PVSS

In the next section we will give a brief overview of the Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS) and its security guarantees.

2.5 Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS)

PPVSS was first introduced in [4], which is used as a building block to construct an efficient e-voting protocol based on Schoenmakers' PVSS [17]. Interestingly, the authors in [4] observed that the original e-voting protocol published in 1999 by Schoenmakers is an unusual application of PVSS, which led them to discover that Schoenmakers PVSS is actually a PPVSS. What sets PPVSS apart from standard PVSS schemes is that it can be used to build versatile applications, such as e-voting, randomness beacons etc., and can also improve efficiency of some existing protocols. The subtle difference between PPVSS and PVSS is that the secret itself is committed by the prover along with all its corresponding secret shares. We now will recall Λ_{RO}, the PPVSS scheme introduced in [4] which is actually based on Π_S[3] in figure 2.4.

The security guarantees of Λ_{RO} are explained in [4] which follows from the security guarantees of Π_S, also the authors replaced Schoenmakers' (P)PVSS with Λ_{RO} in the original e-voting protocol[17] and showed that they achieved improvement in the verification phase (7 to 30 times faster in implementation for some parameters).

2.6 Conclusion

In summary, we recalled some basic concepts of linear codes and Reed Solomon codes as an example. We then recalled the Packed Shamir secret sharing scheme which is a generalization of the Shamir Secret Sharing scheme [18] that was originally designed to defend against passive adversaries and showed its correspondence with Reed Solomon codes. As one of the goals of this thesis is to explore threshold secret sharing schemes that can defend against malicious adversaries, we then briefly recalled PPVSS [4] and an example based on PVSS introduced in [3]. Moving forward, in the next chapter we will introduce a new scheme called Packed Pre-Constructed Publicly Verifiable Secret Sharing (PPPVSS) scheme and give an example whose inspiration is drawn from Λ_{RO}.

Λ_{RO} [4]

Initialization: All parties $\{P_i\}_{i=1}^n$ and dealer D agree on the prime field $\mathbb{Z}_q$, a group $(\mathbb{G}, \times)$ of order q with a generator g, random oracle $\mathcal{H}$. Also, each party P_i registers their public key PK_i in the public ledger and all agree on $PK_0 \neq g$, a commitment key or a public key whose secret key is known to a target person.

Share:

- Dealer D samples a t-degree polynomial $f \in \mathbb{Z}_q[X]$ uniformly at random and sets g^{f_0} as the secret to be shared, where $f_0 = f(0)$.
- For each $1 \leq i \leq n$, D computes $f(i) = f_i$ and uses PK_i to encrypt the secret share $PK_i^{f_i} = y_i$. D also encrypts(/commits) to the secret g^{f_0} using the encryption(/commitment) key PK_0 to obtain $y_0 = PK_0^{f_0}$.
- D uses the PDL proof scheme 2.4.2 to generate π_{PDL} as follows:
  - Samples a $t-$degree polynomial $r \in \mathbb{Z}_q[X]$ uniformly at random and computes $c_i = PK_i^{r_i}$ where $r_i = r(i)$ for $0 \leq i \leq n$.
  - Using $\mathcal{H}$, $d = \mathcal{H}(y_0, \ldots, y_n, c_0, \ldots, c_n))$ is computed.
  - Sets $z(X) = r(X) + df(X)$, hence $\pi_{PDL} = (d, z(X))$ is obtained.
- D broadcasts the encryptions of the shares along with the encryption(/commitment) of the secret with π_{PDL} which proves the validity of the encrypted shares and encrypted(/committed) secret, i.e., broadcasts $\{y_i\}_{i=0}^n$ and $(d, z(X))$.

Verification: Given public keys $\{PK_i\}_{i=1}^n$ and public(commitment) key PK_0, any entity can check π_{PDL} to verify the correctness of the encrypted shares $\{y_i\}_{i=1}^n$ and the secret y_0. They will output **true** or **false** based on the verification of the proof. The procedure is outlined as follows:

- The entity checks if $z(X)$ is a $t-$degree polynomial or not.
- Checks if $d = \mathcal{H}(y_0, y_1, \ldots, y_n, \frac{PK_0^{z(0)}}{y_0^d}, \frac{PK_1^{z(1)}}{y_1^d}, \ldots, \frac{PK_n^{z(n)}}{y_n^d})$.
- Outputs **true** if both of the above checks are satisfied, otherwise **false**.

Reconstruction: There are two approaches to reconstruct the secret g^{f_0} based on the cooperation of the dealer D which are as follows:

- **Optimistic Reconstruction:** D publishes f_0, then any verifier (not necessarily shareholders) when giveen g, PK_0, y_0 can check if $y_0 = PK_0^{f_0}$ and returns g^{f_0} if the check passes, if not they output **false**.
- **Pessimistic Reconstruction:** If D refuses to reveal f_0, then any set $\mathcal{Q}$ consisting at least $t+1$ honest shareholders will do the following (which is same as the reconstruction step in the PVSS Π_S protocol):
  - Each party $P_i \in \mathcal{Q}$ decrypts their share y_i using their private key SK_i corresponding to PK_i to obtain g^{f_i}. Then they publish g^{f_i} along with a DLEQ proof 2.4.1, π_{DLEQ} which proves that the g^{f_i} is the correct decryption of y_i.
  - If $\mathcal{Q}$ consists at least $t+1$ honest parties, they can use the lagrange interpolation to compute the secret g^{f_0} as follows:

$$g^{f_0} = \prod_{i=1}^{t+1} g^{f_i^{\lambda_i}} = g^{\sum_{i=1}^{t+1} f_i \cdot \lambda_i},$$

where $\lambda_i = \prod_{j \neq i} \frac{j}{j-i}$ are lagrange coefficients.

FIGURE 2.4: a Pre-Constructed Publicly Verifiable Secret Sharing (PPVSS) scheme

Chapter 3

Packed Pre-Constructed Publicly Verifiable Secret Sharing

In most of the real time distributed applications, each participant runs as a dealer once to share their secrets with other participants. To retrieve back the secret, remaining participants need to open their shares and perform a bunch of operations to reconstruct the secret, which requires a lot of communication amongst the participants. During the whole time of secret reconstruction one could have asked to the dealer itself to reveal the secret which will save a lot of time and communication. This is the main motivation behind the work of *Pre-Constructed Publicly Verifiable Secret Sharing* (PPVSS) [4], where the authors have given the complete description of PPVSS and an example Λ_{RO} to improve the original e-voting protocol proposed by Schoenmakers [17].

In this chapter we will introduce Packed Pre-Constructed Publicly Verifiable Secret Sharing (PPPVSS) which merely is an extension to the notion of PPVSS. As Packed Shamir Secret Sharing 2.3 allows a dealer to share multiple secrets encoded in a single polynomial, we want to add this feature to PPVSS. In the subsequent sections we will give our definitions for PPPVSS, give a construction of our packed secret sharing scheme and finally show some security guarantees our scheme can achieve.

3.1 Definitions

The following definition directly follows the definition of PPVSS in [4].

Definition 3.1.1 (Packed Pre-Constructed Publicly Verifiable Secret Sharing (PPPVSS))**.** *A Packed PPVSS scheme should have four algorithms, namely, Initial, Share, Verify and Reconstruction whose descriptions are as follows:*

- ***Initial*** $(1^\lambda) \rightarrow \{PK_i\}_{i=1}^{n} \sqcup \{PK_j\}_{j=-(\ell-1)}^{0}$*: When given* 1^λ*, each party* P_i *for* $1 \leq i \leq n$ *registers their public key* PK_i *in a public ledger and withholds the* *corresponding secret key sk_i. Also, all parties and the dealer D agree on ℓ commitment keys or public keys whose secret keys are known to some target people.*

- ***Share*** $(n, t, f_0, \{PK_i\}_{i=1}^{n} \sqcup \{PK_j\}_{j=-(l-1)}^{0}) \rightarrow (\{y_i\}_{i=0}^{n}, \pi_{PPPVSS})$*: It secret shares*

Chapter 4

Revisiting a Randomness Beacon Protocol

4.1 Computational Complexity

See table 4.1 for an overview.

- In ALBATROSS, a dealer(as a part of **commit**) should compute $n(\mathbb{E}_x + \mathbb{P}_e)$ commitments and to give a proof he should do an additional $n(\mathbb{P}_e + \mathbb{E}_x)$. Also, on dealer should do $l(\mathbb{P}_e + \mathbb{E}_x)$ for computing secrets and keeping it to himself. In total dealer needs to do $(2n + l)[\mathbb{E}_x + \mathbb{P}_e]$.
  - In **Reveal**, a verifier should compute $2n\mathbb{E}_x$ which internally requires additional $n\mathbb{P}_e$, i.e., in total it requires $(n-1)n(2\mathbb{E}_x + \mathbb{P}_e)$ computations for each verifier.
    * In **Robust case** where t dealers do not open their polynomials, a verifier should verify $n - t$ polynomials of honest dealers, i.e., for each honest dealer, a verifier has to do $n\mathbb{P}_e$ to evaluate secret share exponents and does $n\mathbb{E}_x$ to get secret shares and cross checks them in the public ledger. Also, finally the verifier computes $l\mathbb{P}_e$ to get secret exponents and get l secrets by doing $l\mathbb{E}_x$. As there are $n - t$ honest dealers, the verifier has to compute $(n - t)(n + l)(\mathbb{E}_x + \mathbb{P}_e)$.
    * In **Honest case**, everyone would have been honest and so each verifier has to do $(n - 1)(n + l)(\mathbb{E}_x + \mathbb{P}_e)$.
  - **Recovery** phase only exists if some party does not open the polynomial leading to PVSS reconstruction phase, in the worst case there should be reconstruction for the secrets of t malicious parties. Given a malicious shareholder who has not opened the secret polynomial, each shareholder/reconstructor has to decrypt their share, which requires $1\mathbb{E}_x$ and should give a DLEQ proof that they have decrypted correctly, which additionally requires $2\mathbb{E}_x$; Also the re-constructor should verify DLEQ proofs of correct share decryption from $n - t$ honest shareholders requiring them to do $4(n - t)\mathbb{E}_x$. In total, each re-constructor requires $[3 + 4(n - t)]t\mathbb{E}_x$.

Randomness Beacon using PPPVSS

Our protocol with PPPVSS is run between a set $\mathcal{P}$ of n parties $P_1, \ldots, P_n$ who have access to a public ledger where they can post information for later verification. It is assumed that the Setup phase of Π_{PPPVSS} is already done and the public keys pk_i of each party P_i along with $\{\mathbb{P}_i\}_{i=1}^{l}$ being Commitment keys (or public keys of target people) to encrypt the l secrets are already registered in the ledger. In addition, the parties have agreed on a Vandermonde $(n-2t)\times(n-t)$-matrix $M = M(\omega, n-2t, n-t)$ with $\omega \in \mathbb{Z}_q^*$.

1. **Commit:** For $1 \leq j \leq n$:
   - Shareholder P_j executes the Distribution phase of the PPPVSS as Dealer for $\ell = n-2t$ secrets, publishing commitments (/encryptions) of secrets, $y^j_{-(l-1)}, \ldots, y^j_{-1}, y^j_0$, and encryptions of shares $\{y^j_i\}_{i=1}^n$ along with π^j_{proof}, which is a NIZK PoK for proving the correctness of committed(/encrypted) secrets and encrypted secret shares on the public ledger, also learning the secrets $h^{s^j_0}, \ldots, h^{s^j_{-(l-1)}}$ and their corresponding exponents $s^j_0, \ldots, s^j_{-(l-1)}$.
2. **Reveal:**
   - Each shareholder checks the validity of the proof π^j_{proof}, i.e., the **verification phase of PPPVSS protocol**.
   - After a set $\mathcal{C}$ containing at least $n-t$ shareholders publish their shares in the public ledger, $P_j \in \mathcal{C}$ reveals l secrets.
   - Every shareholder verifies the validity of secrets by reproducing the commitments using the commitment keys (/public keys of target people).
   - At this point, if every party in $\mathcal{C}$ has opened their secrets correctly, go to step 4' in Figure **??**. Otherwise, proceed to step 3 in Figure **??**.

Figure 4.1: Commit and Reveal phase of the Randomness Beacon using PPPVSS

- Using PPPVSS in randomness beacon protocol, a dealer(as a part of **commit**) requires to do $(n+l)[\mathbb{E}_x + \mathbb{P}_e]$ and $(l-1)\mathbb{M}_G$ to compute $\{y_i\}_{i=0}^n$. For generating the proof that y_i's are valid encryptions of the secret shares and also y_0 is a commitment of the l secrets, the dealer should do $(n+l)[\mathbb{E}_x + \mathbb{P}_e]$ which internally requires additional $(l-1)\mathbb{M}_G$. In total, a dealer has to do

Protocol	**Output size**	**Commit** *(by Dealer)*	**Reveal** *(by shareholder)*	**Recovery** *(by shareholder)*
ALBATROSS, *Honest case*	l^2	$(2n+l)[\mathbb{E}_x+\mathbb{P}_e]$	**Share Verification -** $(n-1)n[2\mathbb{E}_x+\mathbb{P}_e]$ **Secret Verification -** $(n-1)(n+l)[\mathbb{E}_x+\mathbb{P}_e]$	-
with PPPVSS, *Honest case*	l^2	$2(n+l)[\mathbb{E}_x+\mathbb{P}_e]$	**Share Verification -** $(n-1)(n+l)[2\mathbb{E}_x+\mathbb{P}_e]$ **Secret Verification -** $(n-1)l\mathbb{E}_x$	-
ALBATROSS, *Robust case*	l^2	$(2n+l)[\mathbb{E}_x+\mathbb{P}_e]$	**Share Verification -** $(n-1)n[2\mathbb{E}_x+\mathbb{P}_e]$ **Secret Verification -** $(n-t-1)(n+l)[\mathbb{E}_x+\mathbb{P}_e]$	$[3+4(n-t)]t\mathbb{E}_x$
with PPPVSS, *Robust case*	l^2	$2(n+l)[\mathbb{E}_x+\mathbb{P}_e]$	**Share Verification -** $(n-1)(n+l)[2\mathbb{E}_x+\mathbb{P}_e]$ **Secret Verification -** $(n-t-1)l\mathbb{E}_x$	$[3+4(n-t)]t\mathbb{E}_x$

TABLE 4.1: Computational cost of dealer and shareholders, $\mathbb{E}_x$ =group exponentiation and $\mathbb{P}_e$ =polynomial evaluation in group G with order q, where q is a large prime

$2\left[(n+l)[\mathbb{E}_x+\mathbb{P}_e]+(l-1)\mathbb{M}_G\right]$.

- In **Reveal**, a verifier should do $(n+l)(2\mathbb{E}_x+\mathbb{P}_e)$ and $(l-1)\mathbb{M}_G$ for each proof. In total, a verifier has to do $(n-1)(n+l)[2\mathbb{E}_x+\mathbb{P}_e]+(n-1)(l-1)\mathbb{M}_{\mathbb{G}}$.
  - In **Robust case** with t malicious parties not opening the secret polynomials, a verifier should do $l\mathbb{E}_x+(l-1)\mathbb{M}_G$ to verify each proof, so in total each verifier should do $(n-t-1)[l\mathbb{E}_x+(l-1)\mathbb{M}_G]$.
  - In **Honest case** where everyone is honest, a verifier will do $(n-1)l(\mathbb{E}_x+\mathbb{M}_G)$.
- The computational complexity of each re-constructor in **Recovery** phase is exactly same as in the case of ALBATROSS.

4.1.1 Computational Cost analysis

The dealer has to do a bit more work in the case of our protocol in contrast to ALBATROSS

4.2 Communication Complexity

Protocol	**Commit** *(by Dealer)*	**Reveal** *(by Dealer)*	**Recovery** *(by shareholder)*
ALBATROSS	$nG + (t + l)\mathbb{Z}_q$	$(t + l)\mathbb{Z}_q$	$1G + 1\mathbb{Z}_q + 1R_o$
with PPPVSS	$(n + 1)G + (t + l)\mathbb{Z}_q$	$l\mathbb{Z}_q$	$1G + 1\mathbb{Z}_q + 1R_o$

Table 4.2: Communication cost of dealer and (each) shareholder, R_o being the random oracle, G =group of order q and $\mathbb{Z}_q$ = modular group of order q, where q is a large prime

See table 4.2 for an overview.

- In ALBATROSS, a dealer (as a part of **commit**) should send n group elements as commitments, $t + l$ elements in $\mathbb{Z}/q\mathbb{Z}$ that defines the polynomial used in the ZKP and 1 extra element in $\mathbb{Z}/q\mathbb{Z}$ from RO.
  - In **Reveal**, an honest dealer would broadcast $t + l$ coefficients in $\mathbb{Z}/q\mathbb{Z}$ concerning the secret polynomial.
  - If some party has not revealed their polynomial, then in **Recovery** phase a re-constructor using PVSS reconstruction protocol should broadcast 1 element in group which is being the decrypted secret, for the proof of correct decryption, they have to broadcast 3 more group elements along with a polynomial which requires $t + l$ coefficients in $\mathbb{Z}/q\mathbb{Z}$ and 1 group element from RO.
- Using PPPVSS in randomness beacon protocol, a dealer (as a part of **commit**) should send $n+1$ group elements as commitments, $t+l$ elements in $\mathbb{Z}/q\mathbb{Z}$ that defines the polynomial used in the ZKP and 1 extra element in $\mathbb{Z}/q\mathbb{Z}$ from RO.
  - In **Reveal**, an honest dealer would broadcast l elements in $\mathbb{Z}_q$ concerning the exponents to construct the secret.
  - If some part has not revealed their secrets, then the communication cost of each re-constructor is exactly same as in the case of ALBATROSS.

4.2.1 Communication Cost analysis

Randomness Beacon using PPPVSS (cont.)

3. **Recovery:** Let $\mathcal{C}_a$ be the set containing at most t malicious shareholders(as Dealers) who did not open the exponents corresponding to their l secrets, $\{h^{s_i^k}\}_{i=0}^{-(l-1)}$ for each $P_k \in \mathcal{C}_a$, in *Reveal* phase.

    - Every shareholder P_j should decrypt the secret share of each malicious shareholder(Dealer) in $\mathcal{C}_a$, and give a DLEQ NIZK PoK which asserts that the decryption is performed correctly, $h^{s_j^k}$ and NIZK PoK for $(g, h^{s_j^k}, pk_j, y_j^k) \in L$ for each $P_k \in \mathcal{C}_a$, i.e., each shareholder should perform the *pessimistic* reconstruction phase of PPPVSS for every shareholder(Dealer) who has not revealed the exponents corresponding to their secrets.

4 **Output:** Let T be the $(n-t) \times l$ matrix with rows indexed by the shareholders in $\mathcal{C}$ and where the row corresponding to $P_a \in \mathcal{C}$ is $(h^{s_0^a}, ..., h^{s^a_{-(l-1)}})$.

- Each computes the $l \times l$-matrix $R = M \circ T$ by applying FFTE to each column $T^{(j)}$ of T, resulting in column $R^{(j)}$ of R (since $R^{(j)} = M \circ T^{(j)}$ and M is Vandermonde) for $j \in [0, l-1]$.
- Shareholders output the l^2 elements of R as final randomness.

4' **Alternative Output:** if every party in $\mathcal{C}$ has opened her secrets correctly in step *Reveal*, then:

- Shareholders compute $R = M \circ T$ in the following way: Let S be the $(n-t) \times l$ matrix with rows indexed by the shareholders in $\mathcal{C}$ and where the row corresponding to $P_a \in \mathcal{C}$ is $(s_0^a, ..., s^a_{-(l-1)})$. Then each party computes $U = M \circ S \in \mathbb{Z}_q^{l \times l}$ (using the standard FFT in $\mathbb{Z}_q$ to compute each column) and $R = h^U$.
- Shareholders output the l^2 elements of R as final randomness.

FIGURE 4.2: Recovery and Output phase of the Randomness Beacon using PPPVSS

Chapter 5

Conclusion

The final chapter contains the overall conclusion. It also contains suggestions for future work and industrial applications.

Appendices

Appendix A

The First Appendix

Appendices hold useful data which is not essential to understand the work done in the master's thesis. An example is a (program) source. An appendix can also have sections as well as figures and references[1].

A.1 More Lorem

Quisque facilisis auctor sapien. Pellentesque gravida hendrerit lectus. Mauris rutrum sodales sapien. Fusce hendrerit sem vel lorem. Integer pellentesque massa vel augue. Integer elit tortor, feugiat quis, sagittis et, ornare non, lacus. Vestibulum posuere pellentesque eros. Quisque venenatis ipsum dictum nulla. Aliquam quis quam non metus eleifend interdum. Nam eget sapien ac mauris malesuada adipiscing. Etiam eleifend neque sed quam. Nulla facilisi. Proin a ligula. Sed id dui eu nibh egestas tincidunt. Suspendisse arcu.

A.1.1 Lorem 15–17

Nulla in ipsum. Praesent eros nulla, congue vitae, euismod ut, commodo a, wisi. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Aenean nonummy magna non leo. Sed felis erat, ullamcorper in, dictum non, ultricies ut, lectus. Proin vel arcu a odio lobortis euismod. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Proin ut est. Aliquam odio. Pellentesque massa turpis, cursus eu, euismod nec, tempor congue, nulla. Duis viverra gravida mauris. Cras tincidunt. Curabitur eros ligula, varius ut, pulvinar in, cursus faucibus, augue.

Nulla mattis luctus nulla. Duis commodo velit at leo. Aliquam vulputate magna et leo. Nam vestibulum ullamcorper leo. Vestibulum condimentum rutrum mauris. Donec id mauris. Morbi molestie justo et pede. Vivamus eget turpis sed nisl cursus tempor. Curabitur mollis sapien condimentum nunc. In wisi nisl, malesuada at, dignissim sit amet, lobortis in, odio. Aenean consequat arcu a ante. Pellentesque porta elit sit amet orci. Etiam at turpis nec elit ultricies imperdiet. Nulla facilisi.

In hac habitasse platea dictumst. Suspendisse viverra aliquam risus. Nullam pede justo, molestie nonummy, scelerisque eu, facilisis vel, arcu.

Curabitur tellus magna, porttitor a, commodo a, commodo in, tortor. Donec interdum. Praesent scelerisque. Maecenas posuere sodales odio. Vivamus metus lacus, varius quis, imperdiet quis, rhoncus a, turpis. Etiam ligula arcu, elementum a, venenatis quis, sollicitudin sed, metus. Donec nunc pede, tincidunt in, venenatis vitae, faucibus vel, nibh. Pellentesque wisi. Nullam malesuada. Morbi ut tellus ut pede tincidunt porta. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam congue neque id dolor.

A.1.2 Lorem 18–19

Donec et nisl at wisi luctus bibendum. Nam interdum tellus ac libero. Sed sem justo, laoreet vitae, fringilla at, adipiscing ut, nibh. Maecenas non sem quis tortor eleifend fermentum. Etiam id tortor ac mauris porta vulputate. Integer porta neque vitae massa. Maecenas tempus libero a libero posuere dictum. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean quis mauris sed elit commodo placerat. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Vivamus rhoncus tincidunt libero. Etiam elementum pretium justo. Vivamus est. Morbi a tellus eget pede tristique commodo. Nulla nisl. Vestibulum sed nisl eu sapien cursus rutrum.

Nulla non mauris vitae wisi posuere convallis. Sed eu nulla nec eros scelerisque pharetra. Nullam varius. Etiam dignissim elementum metus. Vestibulum faucibus, metus sit amet mattis rhoncus, sapien dui laoreet odio, nec ultricies nibh augue a enim. Fusce in ligula. Quisque at magna et nulla commodo consequat. Proin accumsan imperdiet sem. Nunc porta. Donec feugiat mi at justo. Phasellus facilisis ipsum quis ante. In ac elit eget ipsum pharetra faucibus. Maecenas viverra nulla in massa.

A.2 Lorem 51

Maecenas dui. Aliquam volutpat auctor lorem. Cras placerat est vitae lectus. Curabitur massa lectus, rutrum euismod, dignissim ut, dapibus a, odio. Ut eros erat, vulputate ut, interdum non, porta eu, erat. Cras fermentum, felis in porta congue, velit leo facilisis odio, vitae consectetuer lorem quam vitae orci. Sed ultrices, pede eu placerat auctor, ante ligula rutrum tellus, vel posuere nibh lacus nec nibh. Maecenas laoreet dolor at enim. Donec molestie dolor nec metus. Vestibulum libero. Sed quis erat. Sed tristique. Duis pede leo, fermentum quis, consectetuer eget, vulputate sit amet, erat.

Appendix B

The Last Appendix

Appendices are numbered with letters, but the sections and subsections use arabic numerals, as can be seen below.

B.1 Lorem 20-24

Nulla ac nisl. Nullam urna nulla, ullamcorper in, interdum sit amet, gravida ut, risus. Aenean ac enim. In luctus. Phasellus eu quam vitae turpis viverra pellentesque. Duis feugiat felis ut enim. Phasellus pharetra, sem id porttitor sodales, magna nunc aliquet nibh, nec blandit nisl mauris at pede. Suspendisse risus risus, lobortis eget, semper at, imperdiet sit amet, quam. Quisque scelerisque dapibus nibh. Nam enim. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nunc ut metus. Ut metus justo, auctor at, ultrices eu, sagittis ut, purus. Aliquam aliquam.

Etiam pede massa, dapibus vitae, rhoncus in, placerat posuere, odio. Vestibulum luctus commodo lacus. Morbi lacus dui, tempor sed, euismod eget, condimentum at, tortor. Phasellus aliquet odio ac lacus tempor faucibus. Praesent sed sem. Praesent iaculis. Cras rhoncus tellus sed justo ullamcorper sagittis. Donec quis orci. Sed ut tortor quis tellus euismod tincidunt. Suspendisse congue nisl eu elit. Aliquam tortor diam, tempus id, tristique eget, sodales vel, nulla. Praesent tellus mi, condimentum sed, viverra at, consectetuer quis, lectus. In auctor vehicula orci. Sed pede sapien, euismod in, suscipit in, pharetra placerat, metus. Vivamus commodo dui non odio. Donec et felis.

Etiam suscipit aliquam arcu. Aliquam sit amet est ac purus bibendum congue. Sed in eros. Morbi non orci. Pellentesque mattis lacinia elit. Fusce molestie velit in ligula. Nullam et orci vitae nibh vulputate auctor. Aliquam eget purus. Nulla auctor wisi sed ipsum. Morbi porttitor tellus ac enim. Fusce ornare. Proin ipsum enim, tincidunt in, ornare venenatis, molestie a, augue. Donec vel pede in lacus sagittis porta. Sed hendrerit ipsum quis nisl. Suspendisse quis massa ac nibh pretium cursus. Sed sodales. Nam eu neque quis pede dignissim ornare. Maecenas eu purus ac urna tincidunt congue.

Donec et nisl id sapien blandit mattis. Aenean dictum odio sit amet risus. Morbi purus. Nulla a est sit amet purus venenatis iaculis. Vivamus viverra purus vel magna. Donec in justo sed odio malesuada dapibus. Nunc ultrices aliquam nunc. Vivamus facilisis pellentesque velit. Nulla nunc velit, vulputate dapibus, vulputate id, mattis ac, justo. Nam mattis elit dapibus purus. Quisque enim risus, congue non, elementum ut, mattis quis, sem. Quisque elit.

Maecenas non massa. Vestibulum pharetra nulla at lorem. Duis quis quam id lacus dapibus interdum. Nulla lorem. Donec ut ante quis dolor bibendum condimentum. Etiam egestas tortor vitae lacus. Praesent cursus. Mauris bibendum pede at elit. Morbi et felis a lectus interdum facilisis. Sed suscipit gravida turpis. Nulla at lectus. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Praesent nonummy luctus nibh. Proin turpis nunc, congue eu, egestas ut, fringilla at, tellus. In hac habitasse platea dictumst.

B.2 Lorem 25-27

Vivamus eu tellus sed tellus consequat suscipit. Nam orci orci, malesuada id, gravida nec, ultricies vitae, erat. Donec risus turpis, luctus sit amet, interdum quis, porta sed, ipsum. Suspendisse condimentum, tortor at egestas posuere, neque metus tempor orci, et tincidunt urna nunc a purus. Sed facilisis blandit tellus. Nunc risus sem, suscipit nec, eleifend quis, cursus quis, libero. Curabitur et dolor. Sed vitae sem. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Maecenas ante. Duis ullamcorper enim. Donec tristique enim eu leo. Nullam molestie elit eu dolor. Nullam bibendum, turpis vitae tristique gravida, quam sapien tempor lectus, quis pretium tellus purus ac quam. Nulla facilisi.

Duis aliquet dui in est. Donec eget est. Nunc lectus odio, varius at, fermentum in, accumsan non, enim. Aliquam erat volutpat. Proin sit amet nulla ut eros consectetuer cursus. Phasellus dapibus aliquam justo. Nunc laoreet. Donec consequat placerat magna. Duis pretium tincidunt justo. Sed sollicitudin vestibulum quam. Nam quis ligula. Vivamus at metus. Etiam imperdiet imperdiet pede. Aenean turpis. Fusce augue velit, scelerisque sollicitudin, dictum vitae, tempor et, pede. Donec wisi sapien, feugiat in, fermentum ut, sollicitudin adipiscing, metus.

Donec vel nibh ut felis consectetuer laoreet. Donec pede. Sed id quam id wisi laoreet suscipit. Nulla lectus dolor, aliquam ac, fringilla eget, mollis ut, orci. In pellentesque justo in ligula. Maecenas turpis. Donec eleifend leo at felis tincidunt consequat. Aenean turpis metus, malesuada sed, condimentum sit amet, auctor a, wisi. Pellentesque sapien elit, bibendum ac, posuere et, congue eu, felis. Vestibulum mattis libero quis metus scelerisque ultrices. Sed purus.

Bibliography

[1] D. Adams. *The Hitchhiker's Guide to the Galaxy.* Del Rey (reprint), 1995. ISBN-13: 978-0345391803.

[2] S. Atapoor, K. Baghery, D. Cozzo, and R. Pedersen. VSS from distributed ZK proofs and applications. Cryptology ePrint Archive, Paper 2023/992, 2023.

[3] K. Baghery. π: A unified framework for computational verifiable secret sharing. Cryptology ePrint Archive, Paper 2023/1669, 2023.

[4] K. Baghery, N. Knapen, G. Nicolas, and M. Rahimi. Pre-constructed publicly verifiable secret sharing and applications. Cryptology ePrint Archive, Paper 2025/576, 2025.

[5] G. R. Blakley and C. Meadows. Security of ramp schemes. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 242–268. Springer, 1984.

[6] I. Cascudo and B. David. SCRAPE: Scalable randomness attested by public entities. Cryptology ePrint Archive, Paper 2017/216, 2017.

[7] I. Cascudo and B. David. ALBATROSS: publicly AttestabLe BATched randomness based on secret sharing. Cryptology ePrint Archive, Paper 2020/644, 2020.

[8] D. Chaum and T. P. Pedersen. Wallet databases with observers. In E. F. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, pages 89–105, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[9] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults (extended abstract). In *SFCS '85*, pages 383–395. IEEE, 1985. DBLP's bibliographic metadata records provided through http://dblp.org/search/publ/api are distributed under a Creative Commons CC0 1.0 Universal Public Domain Dedication. Although the bibliographic metadata records are provided consistent with CC0 1.0 Dedication, the content described by the metadata records is not. Content may be subject to copyright, rights of privacy, rights of publicity and other

restrictions.; 26th Annual Symposium on Foundations of Computer Science ; Conference date: 21-10-1985 Through 23-10-1985.

[10] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 427–437. IEEE Computer Society, 1987.

[11] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

[12] M. Franklin and M. Yung. Communication complexity of secure computation (extended abstract). In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '92, page 699–710, New York, NY, USA, 1992. Association for Computing Machinery.

[13] J. Gallian. *Contemporary Abstract Algebra*. CRC Press, 2024.

[14] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[15] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[16] C.-P. Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1989.

[17] L. Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In M. Wiener, editor, *Advances in Cryptology - CRYPTO'99 (Proceedings 19th Annual International Cryptology Conference, Santa Barbara CA, USA, August 15-19, 1999)*, Lecture Notes in Computer Science, pages 148–164, Germany, 1999. Springer.

[18] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, Nov. 1979.

[19] L. R. Welch and E. Berlekamp. Error correction for algebraic block codes, December 1986. U.S. Patent 4,633,470.