CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Client and Server Verifiable Additive Homomorphic Secret Sharing

Enforcing Clients to Act Honestly in Sever Verifiable Additive Homomorphic Secret Sharing by including a Range proofs

Master's thesis in Computer science and engineering

Hanna Ek

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Client and Server Verifiable Additive Homomorphic Secret Sharing

Enforcing Clients to Act Honestly in Sever Verifiable Additive Homomorphic Secret Sharing by including a Range proofs

Hanna Ek

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

A Chalmers University of Technology Master's thesis template for LaTeX
Enforcing Clients to Act Honestly in Sever Verifiable Additive Homomorphic Secret Sharing by including a Range proofs
Hanna Ek

Supervisor: Georgia Tsaloli and Katerina Mitrokotsa, Department of Computer Science and Engineering
Examiner: Katerina Mitrokotsa, Department of Computer Science and Engineering

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Description of the picture on the cover page (if applicable)

Typeset in LaTeX
Gothenburg, Sweden 2021

A Chalmers University of Technology Master's thesis template for LaTeX
Enforcing Clients to Act Honestly in Sever Verifiable Additive Homomorphic Secret Sharing by including a Range proofs
Hanna Ek
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Abstract text about your project in Computer Science and Engineering.

Keywords: Computer, science, computer science, engineering, project, thesis.

Acknowledgements

Here, you can say thank you to your supervisor(s), company advisors and other people that supported you during your project.

Name Familyname, Gothenburg, March 2021

Contents

List of Figures

List of Tables

1 Introduction

1.1 Contribution

1.2 Organisation

2
Theory

This chapter will present the theory behind the client and server verifiable additive homomorphic secret sharing construction described in section 3. First the preliminaries are described, including notation, theorems/definitions, assumptions and cryptographic preliminaries /concepts. Then the VAHSS [12] [11] construction, which this reports aims to extend to include a verification of clients input Finally two range proof constructions, refereed to as *signature-based range proof* and *bulletproof*, is described.

2.1 Preliminaries

Here relevant background for the cryptographic constructions presented later is given.

Notation and setup

To make the text more comprehensible notation that is used throughout the paper is introduced and defined here.

Let $\mathbb{F} = \mathbb{Z}_p$ denote a finite field, where p is a large prime and let $\mathbb{G}$ denote a unique subgroup of order q. Define $g \in \mathbb{G}$ to be a group generator and $h \in \mathbb{G}$ a group element such that $log_g\, h$ is unknown.

The variable $x \in \mathbb{F}$ will consistently denote a secret, i.e the value is not know to all parties in the protocol, and the variable $R \in \mathbb{F}$ will denote a randomly chosen value that hides the secret x. The notation $y \in_R \mathbb{Y}$, means that an element y in chosen at random from the set $\mathbb{Y}$.

Definitions, Theorems and Assumptions

The discrete logarithm assumption and q-strong Diffie Hellman assumption define below does not hold in the presence of quantum computers. All cryptographic constructions presented in this paper relies on one or both of these two assumptions, hence the security is not guaranteed post quantum.

Definition 1 (**Pseudorandom Function (PRF)**).

Definition 2 (**Euler's totient function**). *The function $\Phi(n)$ is defined as the counter of the number of integers that are relative primes to n in the set $\{1, ..., n\}$. Note if n is a prime number $\phi(n) = n - 1$.*

Theorem 1 (**Euler's Theorem**)**.** *For all integers x and n that are co-prime it holds that: $x^{\Phi(n)} = 1\,(mod\ n)$, where $\Phi(n)$ is Euler's totient function.*

From Theorem 1 it follows that for arbitrary y it holds that $x^{y\Phi(n)} = 1\,(\text{mod n})$.

Assumption 1 (**Discrete logarithmic assumption**)**.** *Let $\mathbb{G}$ be a group of prime order q, a generator $g \in \mathbb{G}$ and an arbitrary element $y \in \mathbb{G}$, it is infeasible to find $x \in \mathbb{Z}_q$, such that $y = g^x$*

Assumption 2 (**q-strong Diffie Hellman Assumption**)**.** *Given a group $\mathbb{G}$, a random generator $g \in \mathbb{G}$ and powers $g^x, ..., g^{x_q}$, for $x \in_R \mathbb{F}$ and $q = |\mathbb{G}|$. It is then infeasible for an adversary to find $(c, g^{\frac{1}{x+c}})$, where $c \in \mathbb{F}$.*

Homomorphic Secret Sharing

Secret sharing, first mentioned in [10], hides a secret x by splitting it into shares, where any subset $\mathcal{S}$ of shares smaller than a threshold τ, i.e $|\mathcal{S}| < \tau$, reviles no information about the original value of x. Let a secret x be split into m shares denoted x_i s.t $i \in \{1, ..., m\}$, then in order to reconstruct the value x at least τ shares has to be combined, this is called a (τ, m)-threshold scheme. Here the threshold is equal to the number of shares, $\tau = m$. Further in this paper additive secret sharing scheme is considered, this means that to reconstruct the secret at least τ shares are added, $x = \sum_{i=1}^{\tau} x_i$,

Homomorphic hash functions

Let $\mathcal{H}$ be a cryptographic hash function, $\mathcal{H} : \mathbb{F} \mapsto \mathbb{G}$. Any such function should satisfy the following two properties:

- **Collision-resistant** It should be hard to find $x, x' \in \mathbb{F}$ such that $x \neq x'$ and $\mathcal{H}(x) = \mathcal{H}(x')$.
- **One-Way** It should be computationally hard to find $\mathcal{H}^{-1}(x)$.

A homomorphic hash function should also satisfy the following property:

- **Homomorphism** For any $x, x' \in \mathbb{F}$ it should hold that $\mathcal{H}(x \circ x') = \mathcal{H}(x) \circ \mathcal{H}(x')$. Where $\circ$ is either "$+$" or "$*$".

A such function satisfying the thee properties is $\mathcal{H}_1(x) : \mathbb{F} \mapsto \mathbb{G}$ and $\mathcal{H}_1(x) = g^x$ [13].

Pedersen Commitment scheme

Define a commitment to a secret $x \in \mathbb{F}$ as $\mathbb{E}(x, R) = g^x h^R$, where $R \in_R \mathbb{F}$, this commitment is known as *Pedersen commitment* and originally presented in [9]. This commitment satisfies the following theorem;

Theorem 2. *For any $x \in \mathbb{F}$ and for $R \in_R \mathbb{F}$, it follows that $\mathbb{E}(x, R)$ is uniformly distributed in $\mathbb{G}$. If we have two commits satisfying $\mathbb{E}(x, R) = \mathbb{E}(x', R')$ $x \neq x'$ and*

$x \neq x'$ then it must hold that $R \neq R' \mod q$ and

$$log_g(h) = \frac{x - x'}{R' - R} \mod N. \tag{2.1}$$

Proof. The statements of the theorem follows from solving for $log_g(h)$ in $\mathbb{E}(x, R) = \mathbb{E}(x', R')$ □

Theorem 2 implies that if someone knows the discrete logarithm of h with respect to g, i.e $log_g(h)$, this person is able to provide two equal commits, $\mathbb{E}(x, R) = \mathbb{E}(x', R')$ such that $x \neq x'$. However the $log_g h$ is assumed to be unknown hence it is not possible to construct two equal commits hiding different secrets. This means that the Pedersen commitment scheme is computational binding under the discrete logarithm assumption, it is also perfectly hiding of the secret x [9].

Further note that Pedersen commitment is homomorphic. Hence for arbitrary messages $x_1, x_2 \in \mathbb{F}$, random values $R_1, R_2 \in_R \mathbb{F}$ and the commits $C_i = \mathbb{E}(x_i, R_i), i \in \{1, 2\}$, it holds that $C_1 \cdot C_2 = \mathbb{E}(x_1 + x_2, R_1 + R_2)$.

A final remark is the similarity between the hash function $\mathcal{H}_1$ and the Pedersen commitment $\mathbb{E}$, the hash function can be seen as a generalisation of the Pedersen commitment. This will be used to including verification of client in the VAHSS construction [12].

Vector Pedersen Commitment scheme

Bilinear mapping

Zero knowledge proof

Zero-knowledge proofs (ZKP) was first presented in [7]. A ZKP consist of two parties: *Prover* & *Verifier* and satisfies the properties in Definition 3. After successfully performing a ZKP the prover has convinced the verifier that a certain statement of a secret x is true without having relieved any other information about x. This is done by providing a witness w of the statement. In this paper ZKP that ensures proof of knowledge (PoK) is of interest, this means that the verifier is now only convinced that the statement is true but also that the prover knows the value of secret x. Further this paper will study zero knowledge range proof (ZKRP) where the statement that the prover convinces the verifier of is that the value of secret belongs to a predetermined interval.

Definition 3. *A ZKP should fulfill the three properties:*

- ***Completeness***
- ***Soundness***
- ***Zero-knowledge***

Fiat-Shamir heuristic

Fiat-Shamir heuristic [1] can be used to convert an interactive protocol into non interactive, here it will be used to construct non-interactive ZKP. A non interactive ZKP requires no communication between the prover and verifier during the construction of the proof. In Interactive constructions the verifier sends a challenge $c \in_R \mathbb{F}$

to the prover that is included in the proof in order to convince the verifier that the prover did not cheat. The Fiat-Shamir heuristic replaces the random challenge sent by the verifier with the output of a hash-function of the partial-proof up to this point. The Fiat-Shamir heuristic converts an interactive ZKP to non-interactive plus preserves security and full zero-knowledge relying on the random oracle model (ROM).

2.2 Verifiable additive homomorphic secret sharing

This section will describe a verifiable additive homomorphic secret sharing (VAHSS) Lets assume n clients/data providers and m servers, to simplify notation define the two sets $\mathcal{N} = \{1, ..., n\}$ and $\mathcal{M} = \{1, ..., m\}$. Let c_i and x_i for $i \in \mathcal{N}$ denote the clients (data providers) and their respective data. Denote the servers by $s_j,\ j \in \mathcal{M}$. The idea of VAHSS is that each client split their secret x_i into m shares, denoted x_{ij} and sends one share to each server. The servers receives shares from all n clients and computes the partial output $y_j = \sum_{i=1}^{n} x_{ij}$ and publishes the result. The final result is the sum of all partial results, $y = \sum_{j=1}^{m} y_j$ and can then be computed by any party. In verifiable additive homomorphic secret sharing a proof σ that verifies that $y = \sum_{j=1}^{n} y_j = \sum_{j=1}^{m} \left(\sum_{i=1}^{n} x_{ij} \right) = \sum_{i=1}^{n} \left(\sum_{j=1}^{m} x_{ij} \right) = \sum_{i=1}^{n} x_i$ is generated and published. This allows any party to verify the correctness of the severs computations. Remark that the individual secrets x_i is never revealed in the protocol.

Construction

A construction of VAHSS was presented in [12] and later implemented by [11]. The construction consists of the six PPT (probabilistic polynomial time) algorithms: **ShareSecret**, **PartialEval**, **PartialProof**, **FinalEval**, **FinalProof** and **Verify**. The clients/data providers executed the step **ShareSecret**, the servers **PartialEval** and **PartialProof** and the last three steps can run by anyone. A full discription of the construction and all six algorithms is seen in Construction 1.

To obtain a secret sharing protocol such that any true subset of shares reviles no information about the secret the construction makes use the following polynomial. For each client, c_i, let $\theta_{i1}, ..., \theta_{im} \in \mathbb{F} \backslash \{0\}$ and $\lambda_{i1}, ..., \lambda_{im} \in \mathbb{F}$ such that the following property for polynomial p_i holds,

$$p_i(0) = \sum_{j=1}^{m} \lambda_{ij} p_i(\theta_{ij}). \tag{2.2}$$

Note that is the step **ShareSecret** the shares are put to $x_{ij} = \lambda_{ij} p_i(\theta_{ij})$ and the polynomial $p_i(X)$ is a t-degree polunomial defined as $p_i(X) = x_i + \sum_{k=1}^{t} a_k X^k$, thus $\sum_{j=1}^{m} x_{ij} = \sum_{j=1}^{m} \lambda_{ij} p_i(\theta_{ij}) = p_i(0) = x_i$. Which shows that the proposed shares x_{ij} does adds to the secret x_i if all shares are in the sum and the secret is hidden else.

Construction 1 : Verifiable additive homomorphic secret sharing

Goal: Construct and share the sum $\sum_{i=1}^{n} x_i$, where x_i is a secret value known by client c_i, where $i \in \mathcal{N}$ without any client needing to revealing their individual secret. The servers, used to sharing the secrets, computations are verified so they must be honest.

- **ShareSecret** $(1^\lambda, i, x_i) \to (\tau_i, \{x_{ij}\}_{j\in\mathcal{M}})$
  Pick uniformly at random $\{a_i\}_{i\in\{1,..,t\}} \in \mathbb{F}$ and a t-degree polynomial p_i on the form $p_i(X) = x_i + a_1X + ... + a_tX^t$. Let $\mathcal{H} : x \mapsto g^x$, be a collision-resistant homomorphic hash function. Let $R_i \in \mathbb{F}$ be the output of a PRF. Where it is required that $R_n \in \mathbb{F}$ satisfies $R_n = \phi(N)\lceil\frac{\sum_{i=1}^{n-1} R_i}{\phi(N)}\rceil - \sum_{i=1}^{n-1} R_i$. Compute $\tau_i = \mathcal{H}(x_i + R_i)$, and put $x_{ij} = \lambda_{i,j}p_i(\theta_{ij})$. Output τ_i and $x_{i,j}$ for $j \in \mathcal{M}$.
- **PartialEval** $(j, \{x_{ij}\}_{i\in\mathcal{N}}) \to y_j$
  Compute and output $y_j = \sum_{i=1}^{n} x_{ij}$.
- **PartialProof** $(j, \{x_{ij}\}_{i\in\mathcal{N}}) \to \sigma_j$
  Compute and output $\sigma_j = \prod_{i=1}^{n} g^{x_{ij}} = g^{\sum_{i=1}^{n} x_{ij}} = g^{y_j} = \mathcal{H}(y_j)$.
- **FinalEval** $(\{y_j\}_{j\in\mathcal{M}}) \to y$
  Compute and output $y = \sum_{j=1}^{m} y_j$.
- **FinalProof** $(\{\sigma_j\}_{j\in\mathcal{M}}) \to \sigma$
  Compute and output $\sigma = \prod_{j=1}^{m} \sigma_j = \prod_{j=1}^{m} g^{y_j} = g^{\sum_{j=1}^{m} y_j} = g^y = \mathcal{H}(y)$.
- **Verify** $(\{\tau_i\}_{i\in\mathcal{N}}, \sigma, y) \to \{0, 1\}$
  Compute and output $\sigma = \prod_{i=1}^{n} \tau_i \wedge \prod_{i=1}^{n} \tau_i = \mathcal{H}(y)$.

Correctness, Security and Verifiability

A HSS/additive-HSS construction should satisfy two requirements: *Correctness* and *Security*. A verifiable additive HSS should also satisfy *Verifiability*. The requirements are defined as:

- **Correctness** It must hold that $\Pr\Big[\textbf{Verify}(\{\tau_i\}_{i\in\mathcal{N}}, \sigma, y) = 1\Big] = 1$. This means that with probability 1 the output y from FinalEval is accepted given all parties where honest and the protocol were executed correctly.
- **Security** Let T define the set of corrupted servers such that $|T| < m$, i.e at least one server is honest. Denote a PPT adversary by $\mathcal{A}_1$ and let the $\mathrm{Adv}(1^\lambda, \mathcal{A}, T) := \Pr[b' = b] - 1/2$ be the advantage of $\mathcal{A} = \{\mathcal{A}_1, \mathcal{D}\}$ in guessing b in the following experiment:
  1. The adversary $\mathcal{A}_1$ gives (i, x_i, x_i') to the challenger, where $i \in [n], x_i \neq x_i'$ and $|x_i| = |x_i'|$.
  2. The challenger picks a bit $b \in \{0, 1\}$ uniformly at random chooses and computes $\textbf{ShareSecret}(1^\lambda, i, \hat{x}_i) = (\widehat{\text{share}}_{i1}, ..., \widehat{\text{share}}_{im}, \tau_i)$, where $\hat{\mathbf{x}}_i$ is such that $\hat{x}_i = \begin{cases} x_i, & \text{if } b = 0 \\ x_i' & \text{else} \end{cases}$.
  3. Given the shares from the corrupted servers T and $\hat{\tau}_i$ the adversary distinguisger outputs a guess $b' \leftarrow \mathcal{D}((\widehat{\text{share}}_{ij})_{j|s_j\in T}, \hat{\tau}_i)$.

  A VAHSS-construction is t-secure if for all $T \subset \{s_1, ..., s_m\}$ with $|T| < t$ it holds that $\mathrm{Adv}(1^\lambda, \mathcal{A}, T) < \varepsilon(\lambda)$ for some negligible $\varepsilon(\lambda)$.
- **Verifiability** Let $\mathcal{A}$ denote any PPT adversary and T denote the set of cor-

rupted servers with $T \leq m$. The verifiability property requires that any $\mathcal{A}$ who can modify the input shares to all servers $s_j \in T$ can cause a wrong value to be excepted as $y = f(x_1, ..., x_n)$ with negligible probability.

The VHASS in Construction 1 satisfies the correctness, security and verifiability requirements defined above, this is stated in Theorem 3

Theorem 3. *Construction 1 satisfies the correctness, security and verifiability requirements described above.*

Proof. See section 4.1 in [11]. □

2.3 Constructions for verifying clients input

Range proofs allows a prover to convince a verifier that the value of a secret is in an allowed range, zero knowledge range proofs (ZKRP) does this with out revealing any other information about the secret. Here ZKRP constructed to prove the following statement about a secret x is considered:

$$\{(g, h \in \mathbb{G}, C; x, R \in \mathbb{F}) \; : \; C = g^x h^R \wedge x \in \{\text{"}\textit{predetermined allowed range}\text{"}\} \tag{2.3}$$

Note that in the above statement it is assumed that x is the secret in a Pedersen commitment, which is not required for range proofs however only such range proof will be studied in this paper. The range which x is proved to belong to may vary between different constructions and will be more precisely defied below for the separate constructions.

Let's denote the two parties prover and verifier as $\mathcal{P}$ respectively $\mathcal{V}$. After successfully performing a range proof $\mathcal{P}$ has convinced $\mathcal{V}$, that the secret x in a Pedersen commitment C is in an predetermined allowed range (or set) without $\mathcal{V}$ learning anything else about x.

There exists several constructions for range proofs and this paper will only investigate two for potential extensions of the VAHSS-construction described above to ensure clients honesty. Before presenting these two a XXX will be given to motivate the choice of these two ZKRP. (TODO)Square based range proofs [2] Another construction which could be used to construct a prove that a value is in an allowed range is function secret sharing [3]

In the subsections below theory and construction of *Signature based range proofs* and *bulletproofs* are presented. Both range proofs satisfies the three conditions *completeness*, *soundness* and *zero-knowledge* stated in Definition 3 and proves a statement on the form given in equation (2.3).

2.3.1 Signature-based range proof

First the zero knowledge set membership (ZKSM) is described and then extended to a ZKRP, originally presented in [5]. Both the ZKSM and ZKRP constructions presented are modified compared to the original construction according to the Fiat-Shamir heuristic to be non-interactive.

The idea behind the ZKSM (and also the later derived ZKRP) is that for each element in the allowed set Φ there exist a public commitment, denoted $A_i \, \forall i \in \Phi$. These commitments are made public in the set-up phase by the verifier. The prover who aims to prove that the secret hidden by a pre published Pedesen commitment, denoted C, is in the allowed set Φ chooses the commitment representing the the secret x, i.e A_x. Then hides this choice by raising A_x to a random value $\tau \in_R \mathbb{F}$, this gives $V = A_x^{\tau}$, and publishes V. Then the prover has to convince the verifier that 1) the published value V is indeed equal to A_x^{τ} where A_x is from the allowed set 2) the secret in the Pedersen commitment C is the same as the secret hidden by V. In Construction 2 a detailed description of the ZKSM algorithms that both constructs the public commitments and convinces the verifier of the above statements is given.

Construction 2 : Non interactive set membership proof

Goal: Given a Pedersen commitment $C = g^x h^R$ and a set Φ, prove that the secret x in the commitment belongs to the set Φ without revealing anything else about x.

- **SetUp** $(g, h, \Phi) \to (y, \{A_i\}_{i\in\Phi})$
  Pick uniformly at random $\chi \in_R \mathbb{F}$. Define $y = g^{\chi}$ and $A_i = g^{\frac{1}{\chi+i}} \, \forall i \in \Phi$, output y and $\{A_i\}_{i\in\Phi}$.
- **Prove** $(g, h, C, \Phi) \to \; \mathit{proof}_{SM} = (V, a, D, z_x, z_\tau, z_R)$
  Pick uniformly at random $\tau \in_R \mathbb{F}$, choose from the set $\{A_i\}$ the element A_x and calculate $V = A_x^{\tau}$. Pick uniformly random three values $s, t, m \in_R \mathbb{F}$. Put $a = e(V,g)^{-s} e(g,g)^t$ ($e(\cdot,\cdot)$ is a bilinear mapping as described above), $D = g^s h^m$, and $c = \text{Hash}(V, a, D)$. Finally compute $z_x = s - xc$, $z_R = m - Rc$ and $z_\tau = t - \tau c$ then construct and publish $\mathit{proof}_{SM} = (V, a, D, z_x, z_tau, z_R)$.
- **Verify** $(g, h, C, \mathit{proof}) \to \{0,1\}$ Check if $D \overset{?}{=} C^c h^{z_R} g^{z_x} \wedge a \overset{?}{=} e(V,g)^c e(V,g)^{-z_x} e(g,g)^{z_\tau}$. If the equality holds the prover has convinced the verifier that $x \in \Phi$ return 1 otherwise return 0.

The ZKSM construction can be turned into a efficient zero knowledge range proof by rewriting the secret x in base u such that,

$$x = \sum_{j=0}^{l-1} x_j u^j.$$

Optimal choice of the two parameters u, l is described in [5]. Using this notation it follows that if $x_j \in [0, u) \, \forall j \in \mathbb{Z}_l$, then $x \in [0, u^l)$. A remark is that the subscript j goes though the number $[0, l-1]$ and not $[0, l]$. This has been wrongly notated [5, 8] and therefore an explicit proof of this is given in Appendix A. Construction 3 is a modification of construction 2 into a non interactive zero knowledge range proof using the above decomposition of the secret x.

This ZKRP construction can be generalised to prove membership to an arbitrary interval $[a, b]$ where $a > 0$ and $b > a$, by showing that $x \in [a, a + u^l)$ and $x \in [b - u^l, b)$, since then must hold that $x \in [a, b]$. Figure 2.1 illustrates the intuition and correctness of the statement. Proving $x \in [a, a+u^l)$ and $x \in [b-u^l, b)$ can easily be transferred into proving $x - a \in [0, u^l)$ and $x - b + u^l \in [0, u^l)$, since both a, b are

Construction 3 : Non interactive range proof

Goal: Given a Pedersen commitment $C = g^x h^R$ and two parameters u, l, prove that the secret $x = \sum_{j=0}^{l} x_j u^j$ belongs to the interval $[0, u^l)$ without revealing anything else about x.

- **SetUp** $(g, h, u, l) \rightarrow (y, \{A_i\}_{i \in \mathbb{Z}_u})$
  Pick uniformly at random $\chi \in_R \mathbb{F}$. Define $y = g^\chi$ and $A_i = g^{\frac{1}{\chi+i}}\ \forall i \in \mathbb{Z}_u$, output y and $\{A_i\}$.
- **Prove** $(g, h, C, u, l) \rightarrow\ proof_{RP} = (\{V_j\}, \{a_j\}, D, \{z_{x_j}\}, \{z_{\tau_j}\}, z_R)^1$
  For every $j \in \mathbb{Z}_l$: pick uniformly at random $\tau_j \in_R \mathbb{F}$ and compute $V_j = A_{x_j}^{\tau_j}$. Then pick uniformly at random three more values $s_j, t_j, m_j \in_R \mathbb{F}$ and compute $a_j = e(V_j, g)^{-s_j} e(g, g)^{t_j}$ for all $j \in \mathbb{Z}_l$ and $D = \prod_{j \in \mathbb{Z}_l} (g^{u^j s_j}) h^{m_j}$ Given this let $c = \text{Hash}(\{V_j\}, \{a_j\}, D)$. Then for all $j \in \mathbb{Z}_l$ compute $z_{x_j} = s_j - x_j c$,$z_{\tau_j} = t_j - \tau_j c$ tand $z_R = m - Rc$, where $m = \sum_{j \in \mathbb{Z}_l} m_j$. Finally output the proof: $proof_{RP}$=$(\{\text{V}_j\}, \{a_j\}, D, \{z_{x_j}\}, \{z_{\tau_j}\}, z_R)$
- **Verify** $(g, h, C, proof) \rightarrow \{0, 1\}$
  Check if $D \overset{?}{=} C^c h^{z_R} \prod_{j \in \mathbb{Z}_l} (g^{u^j z_{x_j}}) \wedge a_j \overset{?}{=} e(V_j, y)^c e(V_j, g)^{-z_{x_j}} e(g, g)^{z_{\tau_j}}$ for all $j \in \mathbb{Z}_l$. If the equality holds the prover has convinced the verifier that $x \in [0, u^l)$ return 1 otherwise return 0.

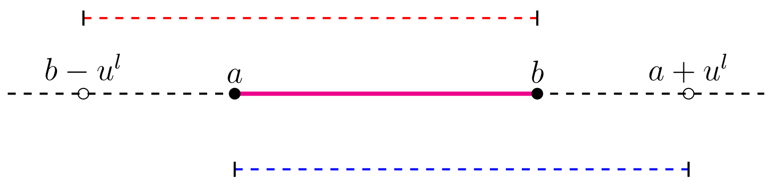

Figure 2.1: Illustration of generalisation to arbitrary intervals $[a, b]$ for range proofs

public. Therefore to prove a secret is in an arbitrary interval the steps **Prove** and **Verify** in construction 3 will have to be executed twice. Plus the **Verify** algorithm has to be modified to include a `AND` operation to verify that $x - a \in [0, u^l)$ *and* $x - b + u^l \in [0, u^l)$. In [6] an optimised implementation is presented reducing the complexity with a factor 2. This rather small reduction is important when a verifier is required check the range of multiple clients secrets, which is the case in VAHSS where it is done once for each client.

2.3.2 Bulletproofs

Bulletproof is a range proof, logarithmic in the size of the range. The proof relies on the inner product argument which allows a prove to convince a verifier that he knows the opening $\boldsymbol{s}, \boldsymbol{q} \in \mathbb{F}^n$ to a Pedersen vector commitment $P_v = \boldsymbol{g}^{\boldsymbol{s}} \boldsymbol{h}^{\boldsymbol{q}}$ such that the inner product of $\boldsymbol{s}, \boldsymbol{q}$ is equal to a known value, c. This can be done with a proof of size $log\ n$, compare to the trivial solution of publishing $\boldsymbol{s}, \boldsymbol{q}$ which is a proof of size n.

Notation and SetUp

The description and construction or bulletproofs requires additional notation which will be presented here. First let lowercase bold font variables denote vectors, i.e $\mathbf{a} \in \mathbb{F}^n$ is a vector with element $a_1, .., a_n \in \mathbb{F}$, and uppercase bold font variables denote matrices, i.e $\mathbf{A} \in \mathbb{F}^{n \times m}$ is a matrix and a_{ij} the element of $\mathbf{A}$ at row i and column j. Given this notation denote scalar multiplication with a vector as $\mathbf{b} = c \cdot \mathbf{a} \in \mathbb{F}^n$, where $c \in \mathbb{F}$ and $b_i = c \cdot a_i$, $i \in \{1, ..., n\}$. Denote the euclidean inner product of two vectors as $\langle \mathbf{a}, \mathbf{b} \rangle$ and Hadamard product as $\mathbf{a} \circ \mathbf{b}$.

Further consider vector polynomials $p(X)$ of degree d on the form $p(X) = \sum_{i=0}^{d} \mathbf{p_i} \cdot X^i \in \mathbb{F}^n[X]$, where the coefficients $\mathbf{p_i} \in \mathbb{F}^n$. The inner product of two vector polynomials, $l(X), r(X)$ is defined as,

$$\langle l(X), r(X) \rangle = \sum_{i=0}^{d} \sum_{j=0}^{n} \langle l_i, r_j \rangle \cdot X^{i+j} \in \mathbb{F}[X].$$

The following is equivalent: evaluating two polynomials at x then taking the inner product versus taking the inner product polynomial at x.

Let $\mathbf{a}||\mathbf{b}$ denote the concatenation of two vectors. Python notation will be used to denote sections of vectors such that $\mathbf{a}_{[:l]} = (a_1, ..., a_l)$ and $\mathbf{a}_{[l:]} = (a_{l+1}, ..., a_n)$ for $l \in [1, n]$.

For $k \in \mathbb{F}^*$ let $\mathbf{k}^n = (1, k, k^2, ..., k^{n-1})$, i.e the vector containing the n fist powers of k.

Let $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ and remember that $\mathbf{a} \in \mathbb{F}^n$ then define $C = \mathbf{g}^{\mathbf{a}} = \prod_{i=1}^{n} g_i^{a_i} \in \mathbb{G}$, where C can be interpreted as a commitment to the vector $\mathbf{a}$. In this section the two vectors $\mathbf{g}, \mathbf{h}$ will be considered to be generators of the space $\mathbb{G}^n$.

Remark that in here n denotes the dimension of the room not the number of clients as earlier, further remark that the dimension of the room is the length of the bit representation of the secret in the Pedersen vector commitment considered below or if not the bit representation is padded with zeros.

Both the construction of the inner product argument and the bullet proof the parameters $g, h, \mathbf{g}, \mathbf{h}, u$ are assumed to be pre-shared and known by both verifier and prover. The assumptions about g, h are before. Further the two vectors $\mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ are assumed to be independent generators of the space $\mathbb{G}^n$. The variable $u \in \mathbb{G}$ is such that there is no known discrete logarithm relation among $\mathbf{g}, \mathbf{h}$. In order to ensure the fairness and correctness of the parameters $g, h, \mathbf{g}, \mathbf{h}, u$ they can be assumed to be chosen by some trusted third party. Another possibility that drops the assumption of a trusted setup is to use the *Nothing Up My Sleeve* (NUMS) stategy, [8].

Inner product argument

The bulletproof construction is based on the inner product argument which will be closer presented in this section. Remember that the inner product argument is a argument of knowledge of $\mathbf{s}, \mathbf{q}$ in a Pedersen vector commitment $P_v = \mathbf{g}^{\mathbf{s}} \mathbf{h}^{\mathbf{q}}$ satisfying a given inner product denoted c. More formally the argument is a proof system of the statement,

$$\{(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n,\ P_v \in \mathbb{G},\ c \in \mathbb{F};\ \mathbf{s}, \mathbf{q} \in \mathbb{F}^n) :\ P_v = \mathbf{g}^{\mathbf{s}}\mathbf{h}^{\mathbf{q}} \wedge\ c = \langle \mathbf{s}, \mathbf{q} \rangle\}$$

Which can be shown to be equivalent to a proof of the statement,

$$\{(\mathbf{g}, \mathbf{h} \in \mathbb{G}^n,\ u, P_v \in \mathbb{G};\ \mathbf{s}, \mathbf{q} \in \mathbb{F}^n) :\ P_v = \mathbf{g}^{\mathbf{s}}\mathbf{h}^{\mathbf{q}}u^{\langle \mathbf{s}, \mathbf{q} \rangle}\}. \tag{2.4}$$

A logarithmic sized proof of the above inner product statement is presented in Construction 4. The construction presented is modified compared to the one presented in [4] to be non-interactive using the Fiat-Shamir heuristic.

Construction 4 : Inner-product argument

Goal: Given a Pedersen vector commitment $P_v = \boldsymbol{g}^{\boldsymbol{s}}\boldsymbol{h}^{\boldsymbol{q}}$ and a value c prove that the two vectors $\boldsymbol{s}, \boldsymbol{q}$ satisfies $\langle \boldsymbol{s}, \boldsymbol{q} \rangle = c$.

- **Prove** $(\mathbf{g}, \mathbf{h}, u, P_v, c, \mathbf{s}, \mathbf{q}) \rightarrow \mathit{proof}_{IP}$
  Let $y = \text{Hash}_{IP}(\mathbf{g}, \mathbf{h}, P_v, c) \in \mathbb{F}^*$ and compute $P'_v = u^{y \cdot c}P$. Let $\mathbf{l}, \mathbf{r}$ be two empty vectors. Run the recursive algorithm **GenerateProof**$(\mathbf{g}, \mathbf{h}, u^{x \cdot c}, P_v, c, \mathbf{s}, \mathbf{q}, \mathbf{l}, \mathbf{r})$ use the output $(g', h', u', P'_v, s', q', \mathbf{l}, \mathbf{r})$ to construct the inner product proof $\text{proof}_{IP} = (\mathbf{g}, \mathbf{h}, u', P_v, s', q', \mathbf{l}, \mathbf{r})$ and output proof_{IP}.
- **GenerateProof**$(\mathbf{g}, \mathbf{h}, u, P_v, \mathbf{s}, \mathbf{q}, \mathbf{l}, \mathbf{r}) \rightarrow (g, h, u, P_v, s, q, \mathbf{l}, \mathbf{r})$
  - If the dimension of the vectors $\mathbf{g}, \mathbf{h}, \mathbf{s}, \mathbf{q}$ drop the bold font and publish the proof $\mathit{proof}_{IP} = (g, h, P_v, u, s, q, \mathbf{l}, \mathbf{r})$.
  - Otherwise: Let $n' = n/2$ and define $c_L = \langle \boldsymbol{s}_{[:,n']}, \boldsymbol{q}_{[n',:]} \rangle$ and $c_R = \langle \mathbf{s}_{[n',:]}, \mathbf{q}_{[:,n']} \rangle$. Then use these variables to calculate $L = \mathbf{g}_{[n':]}^{\mathbf{s}_{[:n']}}\mathbf{h}_{[:n']}^{\mathbf{q}_{[n':]}}u^{c_L}$ and $R = \mathbf{g}_{[:n']}^{\mathbf{s}_{[n':]}}\mathbf{h}_{[n':]}^{\mathbf{q}_{[:n']}}u^{c_R}$. Append $L, R \in \mathbb{G}$ to the vectors $\mathbf{l}$ resp $\mathbf{r}$. Now update $y = \text{Hash}_{BP}(L, R)$, and update $\mathbf{g}' = \mathbf{g}_{[:n']}^{y^{-1}}\mathbf{g}_{[n':]}^{y}$, $\mathbf{h}' = \mathbf{h}_{[:n']}^{y}\mathbf{h}_{[n':]}^{y^{-1}}$ and the commitment $P'_v = L^{y^2}PR^{y^{-2}}$. Finally update the vectors $\mathbf{s}, \mathbf{q}$ to $\mathbf{s}' = \mathbf{s}_{[:n']}y + \mathbf{s}_{[n':]}y^{-1}$ and $\mathbf{q}' = \mathbf{q}_{[:n']}y^{-1} + \mathbf{q}_{[n':]}y$. Run the algorithm recursively, **GenerateProof**$(\mathbf{g}', \mathbf{h}', u, P'_v, \mathbf{s}', \mathbf{q}', \mathbf{l}, \mathbf{r})$ with the updated variables. Note that the vectors $\mathbf{g}, \mathbf{h}, \mathbf{s}, \mathbf{q}$ now have the dimension $n' = n/2$, hence performing the recursion until one-dimensional vectors will require $log\, n$ iterations.
- **Verify** $(\mathit{proof}_{IP} = (\mathbf{g}, \mathbf{h}, u, P_v, s, q, \mathbf{l}, \mathbf{r})) \rightarrow \{0, 1\}$
  For $i \in \{0, log(n)\}$ put $n = n/2$ and $y = \text{Hash}(\boldsymbol{l}[i], \boldsymbol{r}[i])$, then update the vectors $\boldsymbol{g}$ and $\boldsymbol{h}$ as well as the variable P'_v according to, $\boldsymbol{g}' = \boldsymbol{g}_{[:,n]}^{y^{-1}}{\boldsymbol{g}'_{[n,:]}}^{y}$, $\boldsymbol{h} = \boldsymbol{h}_l[:,n]^x\boldsymbol{h}_{[n,:]}^{y^{-1}}$ and $P'_v = L^{y^2}PR^{y^{-2}}$. After iterating over all i the dimension of the vectors $\boldsymbol{g}, \boldsymbol{h}$ is one and we can drop the bold font. Compute $c = \langle s, q \rangle$ and accept if $P'_v = g^s h^r u^c$.

Remark that the inner product argument is sound but not zero-knowledge since information about two exponentials $\mathbf{s}, \mathbf{q}$ is relieved.

Inner product rang proof

Lets present a logarithmic sized range proof called *bulletproof*, based, on the inner product argument. This construction allows a prover, given a Pedersen commitment

$C = g^x h^R$ to convince a verifier that the secret x belongs to the interval $[0, 2^n)$. By convincing the verifier that $\boldsymbol{x} \in \{0,1\}^n$ is the binary representation of the secret x, or equivalently that $x = \langle \boldsymbol{x}, \mathbf{2}^n \rangle$ and that the prover knows $\boldsymbol{x}$. This can be shown to be done by proving the following statement;

$$\Big\langle \boldsymbol{x} - z \cdot \mathbf{1}^n, \boldsymbol{y}^n \circ (\bar{\boldsymbol{x}} + z \cdot \mathbf{1}^n) + z^2 \cdot \mathbf{2}^n \Big\rangle = z^2 \cdot x + \delta(y, z), \tag{2.5}$$

where $\bar{\boldsymbol{x}}$ is the component-wise complement of $\boldsymbol{x}$ and $\delta(y,z) = (z - z^2) \cdot \langle \mathbf{1}^n, \boldsymbol{y}^n \rangle - z^3 \langle \mathbf{1}^n, \mathbf{2}^n \rangle \in \mathbb{F}$. The values z and y are either chosen at random from the set $\mathbb{F}$ by the verifier in an interactive construction or are the output of a hash function in a non-interactive construction. Here a non-interactive construction will be considered.

Directly using a inner product argument presented in Construction 4 to prove the statement in equation (2.5) would leak information about x, since information about the two vectors $\boldsymbol{x}, \bar{\boldsymbol{x}}$ is revealed, i.e the binary representation of x. Hence two new vectors $\boldsymbol{s}_1, \boldsymbol{s}_2$ are introduced and will serve as blinding vectors and help construct a zero-knowledge range proof even if the inner product argument is not a zero knowledge construction. Given this idea, the inner product in (2.5) is tweaked to include the two blinding vectors and the new statement is to prove the inner product of is,

$$\begin{aligned}
t(X) &= \langle l(X), r(X) \rangle = t_0 + t_1 \cdot X + t_2 \cdot X^2 \\
l(X) &= \boldsymbol{x} - z \cdot \mathbf{1}^n + \boldsymbol{s}_1 \cdot X \\
r(X) &= \boldsymbol{y}^n \circ (\bar{\boldsymbol{x}} + z \cdot \mathbf{1}^n + \boldsymbol{s}_2 \cdot X) + z^2 \cdot \mathbf{2}^n,
\end{aligned}$$

Note that $t_0 = z^2 + \delta(y,z)$ which is equal to the right hand side of equation (2.5). Further it holds that $t_1 = \langle \boldsymbol{x} - z \cdot \mathbf{1}^n, \boldsymbol{y}^n \circ \boldsymbol{s_R} \rangle + \langle \boldsymbol{s_L}, \boldsymbol{y}^n \circ (\boldsymbol{a_R} + z \cdot \mathbf{1}^n) + z \cdot \mathbf{2}^n \rangle$ and $t_2 = \langle \boldsymbol{s_L}, \boldsymbol{y}^n \circ \boldsymbol{s_R} \rangle$. Given these vectors and innner product Construction 5 gives a non interactive zero knowledge range proof that the secret x belongs to the interval $[0, 2^n]$.

An optimised version of bulletproof where one prover wishes to verify the range of several commitments reduces the proof size from growing multiplicatively in the number of commits to additive. More precisely lets a assume a prover wants to prove the range of k commits a naive implementation would lead to a proof size of $k \cdot log_2 n$, but an optimised implementation reduces this to $log_2 n + 2log_2 k$. With a similar approach as presented for the signature based range proofs, illustrated in Figure 2.1, bulletproof can also be generalised to arbitrary ranges $[a, b]$, where $a > 0$ and $b > a$. This would then increase the proof size with the additive term $2log_2 2 = 2$.

Construction 5 : Bulletproof

Goal: Given a Pedersen commitment $C = g^x h^R$ and a number n, prove that the secret x in the commitment belongs to the range $[0, 2^n)$ without revealing anything else about x.

- **Prove** $(g, h, \mathbf{g}, \mathbf{h}, P, n, x, R, u) \to \ \mathit{proof}_{RP}$
  Let $\boldsymbol{x}$ denote the binary representation of the secret x in the commitment P and $\bar{\boldsymbol{x}}$ the component-wise complement such that $\boldsymbol{x} \circ \bar{\boldsymbol{x}} = 0$. Construct the commitment $A = h^\alpha \boldsymbol{g}^{\boldsymbol{x}} \boldsymbol{h}^{\bar{\boldsymbol{x}}}$, where $\alpha \in_R \mathbb{F}$. Then chose the two blinding vectors $\boldsymbol{s_R}, \boldsymbol{s_L} \in_R \mathbb{F}^n$ and the value $\rho \in_R \mathbb{F}$ and compute the commitment $S = h^\rho \boldsymbol{g}^{\boldsymbol{s_L}} \boldsymbol{h}^{\boldsymbol{s_R}}$. Let $y = \text{Hash}(A, S)$, $z = \text{Hash}(A, S, y)$ and $\tau_1, \tau_2 \in_R \mathbb{F}$. Now the it is possible to construct t_1, t_2 defined above. Given this let $T_1 = g^{t_1} h^{\tau_1}$ and $T_2 = g^{t_2} h^{\tau_2}$, next let $X = \text{Hash}(T_1, T_2)$. Now construct the two vectors for the inner product argument: $\boldsymbol{l} = \boldsymbol{x} - z \cdot \mathbf{1}^n - \boldsymbol{s_L} \cdot X$, $\boldsymbol{r} = \boldsymbol{y}^n \circ (\bar{\boldsymbol{x}} + z \cdot \mathbf{1}^n + \boldsymbol{s_R} \cdot X) + z^2\, X$ and calculate the inner product $\hat{t} = \langle \boldsymbol{l}, \boldsymbol{r} \rangle$. Finally compute $\tau_X = \tau_2 x^2 + \tau_1 X + z^2 R$ and $\mu = \alpha + R\, X$. Now use the inner product argument to prove that $\hat{t}$ is indeed the inner product of the two vectors $\mathbf{l}, \mathbf{r}$ using the commitment $P_v = \boldsymbol{g}^{\boldsymbol{l}} \boldsymbol{h}^{\boldsymbol{r}}$, run the algorithm **Prove** defined Construction 4 with the input $(\boldsymbol{g}, \boldsymbol{h}, u, P_v, \hat{t}, \boldsymbol{l}, \boldsymbol{r})$ to construct such a proof denoted proof_{IP}. Combine and publish the proof: $\mathit{proof}_{RP} = (\tau_X, \mu, \hat{t}, P, A, S, T_1, T_2, P_v, \mathit{proof}_{IP})$.
- **Verify** $(g, h, C, \mathit{proof}_{RP}) \to \{0, 1\}$
  Compute the three hash functions $y = \text{Hash}(A, S)$, $z = \text{Hash}(A, S, y)$ and $X = \text{Hash}(T_1, T_2)$. Then given y, z, X compute $h'_i = h_i^{y^{-i+1}}$ for all $i \in \{1, ..., n\}$, $P_l = P \cdot h\mu$ and $P_r = A \cdot S^x \boldsymbol{g}^{-z} \boldsymbol{h'}^{z\boldsymbol{y}^n + z^2 \cdot \mathbf{2}^n}$. Then check if the following equalities hold: $P_l \overset{?}{=} P_r \wedge g^{\hat{t}} h^{\tau_X} \overset{?}{=} P^{z^2}\ g^{\delta(y,z)}\ T_1^x\ T_2^{x^2}$ and if the output of **Verify** in Construction 4 on the input (proof_{IP}) is 1. If all three criterion is fulfilled then the secret x in the commitment P is in the range $[0, 2^n]$.

3 Methods

Based on the theory presented in the previous chapter . Evaluate and choose one to implement, do implementation and construct proofs

3.1 Comparison of range proofs

In this section the different constructions for verifying clients honesty presented in section 2.3 will be analysed and compared in order to evaluate their suitability to combine with the VAHSS scheme described in Construction 1 to verify clients honesty. First a theoretical analysis of each range proofs will given and then a prototype analysis where the range proofs are compares.

The aspects that will be considered in the evaluation of the range proofs and their compatibility with the VAHSS construction is presented is the below list;

- Proof size (communication complexity)
- Computation complexity (for setup, prover verifier)
- Flexibility of range

Remark that all of the range proof considered aim to prove that the secret in a Pedersen commitment is in an allowed range. Thus to combine any of the range proofs with the VAHSS construction, the clients needs beyond previously computed and published values also publish a Pedersen commitment of their secret x_i. This is investigated further in section 3.2. Since the adaptation of the VAHSS construction is the same independent of the considered range proof it is not relevant in the evaluation of range proofs expect in terms of runtime.

The considerable difference between the bulletproof and the signature based range proofs makes the comparison between them not straightforward. Signature based range proofs requires bilinear mappings unlike bulletproofs, bilinear mappings are relative expensive operation compared to for example group exponentials which are dominating the computational complexity for bulletproofs. Therefore it is not straightforward to compare them in aspects of number of operations performed and an explicit comparison will only be made with respect to runtime. But first the theoretical performance of the range proofs will be discussed individually.

3.1.1 Theoretical analysis: Signature-based set membership and range proof

First lets discuss the communication complexity and proof size starting with the signature based set membership . This construction allows for a $\mathcal{O}(1)$- size proof that a committed value belongs to a given set Φ. In order to construct such a proof $n = |\Phi|$ digital signatures needs to be known by both prover and verifier, one signature for each elements in Φ. This signatures are usually shared by the verifier in the Setup phase. Sharing the digital signatures of the elements in the set Φ becomes intractable when the set is large. A large set in this context would be a set consisting of a few hundred elements since the verifier has to publish n digital signatures in the SetUp phase.

The signature based range proof reduces this to only needing to publish u digital signatures to prove a commitment is in the range $[0, u^l]$ in the SetUp phase. In the algorithm **Prove** in Construction 3 the prover sends $l + 1$ elements from the group $\mathbb{G}_1$, l elements from the group $\mathbb{G}_T$ and $2l + 1$ field elements. Comparing to the algorithm **Prove** in Construction 2 where the prover sends two elements from the group $\mathbb{G}_1$, one elements from the group $\mathbb{G}_T$ and three field elements. For the ZKRP the communication complexity depends on the choice of u, l. Asymptotic analysis gives a communication complexity $\mathcal{O}(\frac{k}{log\, k - log\, log\, k})$, where $l = \frac{k}{log\, u}$ and u put to $u = \frac{k}{log\, k}$ Here k satisfies $u^l \geq 2^{k-1}$.

For ZKSM the communicational complexity for the proof is lower then for the ZKRP, given $l > 1$. In some practical applications the digital signatures shared in the setup phase can be assumed to be pre shared, for example in applications where Φ is used many times. This leads that ZKSM is to prefer over ZKRP in such applications or when Φ is a relative small.

Next consider the computational complexity for algorithms **Prove** and **Verify** in the ZKSM and ZKRP. constructions In the set membership construction both the prover and verifier has to perform one bilinear paring and two exponentials over the group $\mathbb{G}$. While in the range proof construction the prover need to perform l bilinear mappings and $5l$ exponentials to prove a secret is in the range $[0, u^l)$ and additionally $3l$ exponentials for arbitrary ranges $[a, b]$. The verifier need to ?? Discuss on meeting.

An advantage of the set membership construction is that it can prove membership of non continuous sets. An example could be that the set Φ represents all odd numbers in a certain interval and then the prover can insure the verifier that the secret is an odd number in a given range. This is an illustrative example of the flexibility of set membership proofs compared to range proofs.

3.1.2 Theoretical analysis: Bulletproof

First the communication and computational complexity of the inner product argument which is used in the bulletproof is considered. Then based on this the bulletproofs will be analysed.

The inner product argument as described in Construction 4, compared to the naive approach, reduces the communication complexity for proving the statement in equation (2.4) from linear to logarithmic size in terms of the vecotrs length. More precisely the prover has to send $2\lceil log_2 n \rceil$ group elements and 2 field elements to the verifier when proving the statement, thus the commutation complexity id of order $\mathcal{O}(log_2 n)$, where n is the length of the vectors.

The computational effort for the inner product argument is dominated by $8n$

Table 3.1: Communication Complexity TODO

	Proof Size	Set up
Signature based SM	$\mathcal{O}(1)$	$\mathcal{O}(n)$
Signature based RP	$\mathcal{O}(l)$	$\mathcal{O}(u)$
Bulletproof	$\mathcal{O}(log_2\, n)$	$\mathcal{O}(1)$

Table 3.2: Operations of proof construction TODO

	exponentials		Fiels Operations		Bilinear mapping	
	Prover	Verifier	Prover	Verifier	Prover	Verifier
Signature based SM	b1	b2	c1	c2	1	d
Signature based RP	b1	b2	$5l$		l	d
Bulletproof	b1	b2	c1	c2	d	d2

group exponentiations for the prover and $4n$ group exponentiations for the verifier. In a non-interactive construction this can be optimised such that the verifier instead perform only one multidimensional-exponent of size $2n + 2log_2n + 1$. This leads to a significant speed up of the verification of the argument.

Using the inner product argument to build bullet proofs result in a communication complexity of $2\lceil log_2n\rceil + 4$ group elements and 5 field elements, where n is such that a secret is proved to be in the range $[0, 2^n)$. A remark is that in a bulletproof construction the range always has to be an exponent on 2, if the length of the binary representation of the secret is not a two-exponent this can be solved with padding. IWhen extending the bulletproof to prove a secret is in an arbitrary range $[a, b]$ the communication complexity is increased by an additive term of size 2.

3.1.3 Prototype Analysis TODO

3.2 Additive homomorphic secret sharing with verification of both clients and severs

The VAHSS constructions discussed in section 2.2 assumes honest clients but verifiers that the servers where honest. The aim of this paper is to present an extended this VAHSS construction to verify both client and servers honesty. To achieve this range proofs has been studied, extending the VAHSS construction with a range proof forces potential malicious client to submit a value still within the allowed

Table 3.3: Operations of proof construction TODO

	exponentials		Fiels Operations		Bilinear mapping	
	Prover	Verifier	Prover	Verifier	Prover	Verifier
Signature based SM	b1	b2	c1	c2	1	d
Signature based RP	b1	b2	$5l$		l	d
Bulletproof	b1	b2	c1	c2	d	d2

range. This means that even if a client submits a false value the amount of impact this haves on the final output y is limited by the size of the range.

The considered range proofs emanate from a Pedersen commitment hiding a secret and then generate a zero knowledge proof that the secret belongs to an pre-specified interval. In order to combine the VAHSS construction with a range proof a link between the shares hiding the secret generated in the algorithm **ShareSecret** in the VAHSS construction and the Pedersen commitment in the range proof need to be establoshed to convince the verifier that the shares represents a secret that is in the allowed range, without revealing the secret. Individual shares themselves does not contain information about the secret and therefore a Pedersen commitment hiding a share would not be useful use in a range proof. The clients except from the shares also publishes the checksum τ_i for the secret x_i, more precisely the definition of $\tau_i = g^{x_i+R_i}$, where R_i chosen uniformly at random. This checksum is indeed equal to a Pedersen commitment where $g = h$. Performing a range proof of τ_i assuming $g = h$ would be sound. However if $g = h$ the computationally hiding property of a Pedersen commitment would not hold since $log_g(h) = log_g(g) = 1$ which leads to that the LHS in equation (2.1) is equal to 1. Therefore to construct two commits $\mathbb{E}(x, R)$ and $\mathbb{E}(x', R')$ such that $\mathbb{E}(x, R) = \mathbb{E}(x', R')$ but $x =' $ it is sufficient to solve,

$$1 = \frac{x - x'}{R - R'} \, mod\, N \implies x' = \frac{x}{R' - R} \, mod\, N.$$

In other words it is straightforward to create a false commitment hence also a false range proof. Lets instead investigate modifying the checksum τ_i to a Pedersen commitment. Let the clients compute and output $\pi_i = g^{x_i} h^{R_i}$, where x_i, R_i, g, h are as defined above instead of τ_i as before. Now a range proof can easily be constructed for the commitment π_i. Below it will be shown that Theorem 3 still hold after replacing τ_i with π_i.

In Construction 6 the extended VAHSS is described in detail. Compared to the construction presented n [12] the algorithms **ShareSecret** and **Verify** has been modified and the algorithm **ConstructRangeProof** has been added. More precisely in the algorithm **ShareSecret** the client computes the commitment π_i instead of the checksum τ_i as motivated above. The algorithm **ConstructRangeProof** constructs a range proof (or set membership proof) denoted RP_i of the commitment π_i. It is not specified which range proof construction that is used since will not affect the rest of construction as long as the verification algorithm used to verify range proof is the algorithm **Verify** is the same as used to construct the proof. This algorithm will be run by the client. The algorithm **Verify** contains an call to the verification algorithms corresponding to the one used to generate the range proof RP_i and an additional `AND` operator.

Theorem 4. *The client and server verifiable AHSS presented in Construction 6 satisfies the same correctness, security and requirements as Construction 1 as well as the verifiability requirements:*

- ***Verifiability Servers*** *Let $\mathcal{A}$ denote any PPT adversary and T denote the set of corrupted servers with $T \leq m$. Note that if $|T| = m$, the verifiability property holds but not the security property. The verifiability property requires that any $\mathcal{A}$ who can modify the input shares to all servers $s_j \in T$ can cause a*

Construction 6 : Client and Server Verifiable additive homomorphic secret sharing

Goal: Construct and share the sum $\sum_{i=1}^{n} x_i$, where x_i is a secret value known by client c_i, where $i \in \mathcal{N}$ without any client needing to revealing their individual secret. Servers and clients computations are verified.

- **ShareSecret** $(1^\lambda, i, x_i) \to (\pi_i, \{x_{ij}\}_{j\in\mathcal{M}})$
  Pick uniformly at random $\{a_i\}_{i\in\{1,..,t\}} \in \mathbb{F}$ and a t-degree polynomial p_i on the form $p_i(X) = x_i + a_1 X + ... + a_t X^t$. Let $P : x, y \to g^x h^y$ be a Pedersen commitment function. Let $R_i \in \mathbb{F}$ be the output of a PRF. Where it is required that $R_n \in \mathbb{F}$ satisfies $R_n = \phi(N)\lceil \frac{\sum_{i=1}^{n-1} R_i}{\phi(N)} \rceil - \sum_{i=1}^{n-1} R_i$. Compute $\pi_i = P(x_i, R_i)$, and $x_{ij} = \lambda_{i,j} p_i(\theta_{ij})$.
  Output π_i and $\{x_{i,j}\}_{j\in\mathcal{M}}$.
- **ConstructRangeProof** $x \mapsto Proof_{RP}$
  Construct a range proof, denoted RP_i, for π_i to the range $[0, B]$ using Construction 2, 3 or 5. All required parameters and setup is assumed to be pre-shared and known by all parties.
- **PartialEval** $(j, \{x_{ij}\}_{i\in\mathcal{N}}) \to y_j$
  Compute and output $y_j = \sum_{i=1}^{n} x_{ij}$.
- **PartialProof** $(j, \{x_{ij}\}_{i\in\mathcal{N}}) \to \sigma_j$
  Compute and output $\sigma_j = \prod_{i=1}^{n} g^{x_{ij}} = g^{\sum_{i=1}^{n} x_{ij}} = g^{y_j} = H(y_j)$.
- **FinalEval** $(\{y_j\}_{j\in\mathcal{M}}) \to y$
  Compute and output $y = \sum_{j=1}^{m} y_j$.
- **FinalProof** $(\{\sigma_j\}_{j\in\mathcal{M}}) \to \sigma$
  Compute and output $\sigma = \prod_{j=1}^{m} \sigma_j = \prod_{j=1}^{m} g^{y_j} = g^{\sum_{j=1}^{m} y_j} = g^y = H(y)$.
- **Verify** $(\{\pi_i\}_{i\in\mathcal{N}}, x, y) \to \{0, 1\}$
  Compute and output $\sigma = \prod_{i=1}^{n} \pi_i \wedge \prod_{i=1}^{n} \pi_i = H(y) \wedge$ **Verify**(RP_i). Where **Verify** is the verification step of the range proof used by the client to construct RP_i.

wrong value to be excepted as $y = f(x_1, ..., x_n)$ with negligible probability.

- ***Verifiability Clients***

Proof. The proof of security is the same as in [12] since the pedersen commitment is perfectly hiding. For proving the correctness it is sufficient to show that $\sigma = \prod_{i=1}^{n} \pi_i \wedge \prod_{i=1}^{n} \pi_i = \mathcal{H}(y)$. Both y and σ are the same as in construction as in [11]. Hence by construction:

$$y = \sum_{j=1}^{m} y_j = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_{ij} p_i(\theta_{ij}) = \sum_{i=1}^{n} \overbrace{\left(\sum_{j=1}^{m} \lambda_{ij} p_i(\theta_{ij}) \right)}^{p_i(0)} = \sum_{i=1}^{n} p_i(0) = \sum_{i=1}^{n} x_i, \tag{3.1}$$

and for σ it holds that:

$$\sigma = \prod_{j=1}^{m} \sigma_j = \prod_{j=1}^{m} g^{y_j} = g^{\sum_{j=1}^{m} y_j} = g^y = \mathcal{H}(y)$$

For the π_i, whose construction has been modified compared to [11] we have:

$$\prod_{i=1}^{n} \pi_i = \prod_{i=1}^{n} \mathbb{E}(x_i, R_i) = \prod_{i=1}^{n} g^{x_i} h^{R_i} = g^{\sum_{i=1}^{n} x_i} h^{\sum_{i=1}^{n} R_i} \overset{(3.1)}{=} g^y h^{\sum_{i=1}^{n-1} R_i + R_n} =$$
$$= g^y h^{\phi(N) \left\lceil \frac{\sum_{i=1}^{n-1} R_i}{\phi(N)} \right\rceil} \overset{*}{=} g^y = \mathcal{H}(y) \quad \text{*- } \textit{since } h \textit{ is co-prime to } N.$$

The proof of ***Verifiability Severs*** is the same as in [12] and the proof of ***Verifiability Clients*** follows from the properties of the range proof. □

3.3 Implementation

The implementations is done in Golang. Specify all parameters used for the implementation.

4
Results

4.1 Runtime and complexity

5 Conclusion

5.1 Discussion

Limit, only considerd range proof using pedersen commitment scheme.

FFS for intervals: Need comunication between servers. We do not want

5.2 Conclusion

Bibliography

[1] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, page 62–73, New York, NY, USA, 1993. Association for Computing Machinery.

[2] F. Boudot. Efficient proofs that a committed number lies in an interval. In B. Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 431–444, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[3] E. Boyle, N. Gilboa, and Y. Ishai. Function secret sharing: Improvements and extensions. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1292–1303, New York, NY, USA, 2016. Association for Computing Machinery.

[4] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334, 2018.

[5] J. Camenisch, R. Chaabouni, and a. shelat. Efficient protocols for set membership and range proofs. In J. Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008*, pages 234–252, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[6] R. Chaabouni, H. Lipmaa, and A. Shelat. Additive combinatorics and discrete logarithm based range protocols. In R. Steinfeld and P. Hawkes, editors, *Information Security and Privacy*, pages 336–351, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[7] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[8] E. Morais, T. Koens, C. van Wijk, and A. Koren. A survey on zero knowledge range proofs and applications. *SN Applied Sciences*, 1(946), 2019.

[9] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 129–140, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

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

[11] G. Tsaloli, G. Banegas, and A. Mitrokotsa. Practical and provably secure distributed aggregation: Verifiable additive homomorphic secret sharing. *Cryptography*, 4(3), 2020.

[12] G. Tsaloli and A. Mitrokotsa. Sum it up: Verifiable additive homomorphic secret sharing. In J. H. Seo, editor, *Information Security and Cryptology – ICISC 2019*, pages 115–132, Cham, 2020. Springer International Publishing.

[13] H. Yao, C. Wang, B. Hai, and S. Zhu. Homomorphic hash and blockchain based

authentication key exchange protocol for strangers. In *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)*, pages 243–248, 2018.

A

Proof of range in signature based range proof

Let $x = \sum_{j=0}^{l-1} x_j u^j$, where x_j is an integer and $x_j \in [0, u)$, u, l are integers and $j \in [0, l-1](= \mathbb{Z}_l)$. Then it holds that $x \in [0, u^l)$.

$$x = \sum_{j=0}^{l-1} x_j u^j \leq \sum_{j=0}^{l-1} (u-1)u^j = \sum_{j=0}^{l-1} u^{j+1} - \sum_{j=0}^{l-1} u^j = (u-1)\sum_{j=0}^{l-1} u^j = (u-1)\frac{u^l - 1}{u-1} = u^l - 1 < u^l$$

Hence the statement is proved and it is trivial to see that if $j \in [0, l]$ the value of x could exceed u^l.