Efficient Fair Data Exchange without In-Circuit Public-Key Operations

Dongwook Kim[1], Jihye Kim[2,3], and Hyunok Oh[1,3]

[1] Hanyang University, Seoul, Republic of Korea
{dongwook1214,hoh}@hanyang.ac.kr
[2] Kookmin University, Seoul, Republic of Korea
jihyek@kookmin.ac.kr
[3] Zkrypto Inc., Seoul, Republic of Korea

Abstract. Fair data exchange (FDE) ties payment to recovery of data matching an agreed commitment. Code-based FDE offers near-plaintext communication, but the sampled-ElGamal baseline incurs costly in-circuit checks when low redundancy requires many samples. We replace auxiliary ElGamal ciphertexts with direct KZG authentication and external witness linking. This leaves only field arithmetic in the consistency circuit and permits a BLS12-381 instantiation with EVM precompile support for settlement-key verification. We prove correctness, soundness, and zero knowledge under an external KZG setup in the algebraic group and random oracle models. Our implementation reduces aggregate sender computation from 179.6 to 2.75 seconds (65.3×) for a 2 MiB file at redundancy $\beta \approx 1.1$, relative to the sampled-ElGamal baseline. Bulk ciphertext remains about 1.1× the plaintext size. The corresponding EVM implementation reduces gas for successful seller settlement by a factor of 31.3.

Keywords: Fair data exchange · Verifiable encryption · zk-SNARKs · Reed–Solomon codes · Polynomial commitments

1 Introduction

Fair data exchange (FDE) lets mutually distrustful parties exchange data for payment: the buyer can recover the agreed data if and only if the seller is paid. For example, a client retrieving an archived file needs assurance of receiving the original data, while the server needs assurance of payment for delivering it.

A blockchain can escrow payment, but checking an entire off-chain file on-chain is expensive. FairSwap uses commitments and an on-chain dispute procedure [4]. Another approach uses *verifiable encryption under a committed key* (VECK) [5]. A file commitment identifies the agreed data; the seller sends a ciphertext and proves that releasing a valid key will let the buyer recover data matching that commitment. The buyer verifies the proof off-chain before locking payment in escrow, and the contract checks the revealed key before paying the

seller. Efficiency therefore requires low costs for data transmission, consistency proving, and on-chain settlement.

Moving verification off-chain leaves substantial computation and communication costs. In the ElGamal-based VECK of Tas et al., each file element is encrypted using exponential ElGamal and participates in a consistency check against the file's KZG polynomial commitment [5, 8]. The core consistency proof is succinct, but proving and verification involve costly group operations across the file and grow with its length. Using measurements from a smaller file, we estimate that preparing the ciphertext and proofs for a 2 MiB file takes approximately 786 seconds (Appendix B.7). Exponential ElGamal decryption recovers the plaintext only as a discrete logarithm. File elements must therefore be split into small chunks, each encrypted as its own group element, which further inflates the transmission overhead. With compressed BLS12-381 group encodings, a 2 MiB file requires 24 MiB of ciphertext before proofs, a 12× expansion.

Code-based FDE trades additional coding redundancy for fewer expensive consistency checks: Reed–Solomon coding allows recoverability of the committed file to be established, with negligible soundness error, by proving consistency at only R randomly sampled positions [7]. Our direct baseline, $\mathrm{VECK}^{\star}_{\mathrm{EL}}$, combines this strategy with a hash-derived symmetric keystream (*hash masking*): adding a key-derived field mask to each codeword entry replaces full-file ElGamal ciphertexts with one masked field element per entry. An ℓ-element file thus yields approximately $\beta\ell$ bulk ciphertext elements, where $\beta > 1$ is the coding redundancy.

To authenticate the sampled masked values against the file commitment, $\mathrm{VECK}^{\star}_{\mathrm{EL}}$ retains auxiliary ElGamal ciphertexts for those positions. These supply a common hidden representation for two proofs: a VECK proof authenticates the sample values against the file commitment, and a zero-knowledge SNARK checks that they equal the values recovered by unmasking with the committed key [7]. In the baseline construction, the SNARK checks the sampled ElGamal encryption relations individually, incurring $O(R)$ in-circuit elliptic-curve operations.

The remaining bottleneck. Lowering β saves bandwidth but reduces the number of incorrect entries the code can correct, so maintaining the same sampling soundness guarantee requires a larger R. Under our evaluation's security parameters, $\beta \approx 1.1$ uses $R = 2384$ samples. For a 2 MiB file, SNARK proving takes 179.1 seconds, or 99.7% of the baseline's 179.6 seconds of total sender computation (Table 1). The in-circuit relation also constrains the curve choice. In the baseline construction, the auxiliary ElGamal ciphertexts and settlement key share the file commitment's group. To keep the sampled group operations native to the SNARK field, the evaluated implementation uses a BLS12-377/BW6-761 two-chain. This choice also places the settlement key on BLS12-377, which lacks an EVM precompile. Consequently, successful seller settlement costs 31.3× more gas than in our BLS12-381 implementation (Table 3). The challenge is therefore to retain near-plaintext communication without paying the proving and settlement costs that in-circuit public-key operations impose.

Our approach. We replace the auxiliary ElGamal ciphertexts with KZG authentication of a blinded sample polynomial outside the circuit. The SNARK checks unmasking and polynomial consistency using only *field arithmetic*. External witness links bind its hidden evaluation and masking key to the KZG-authenticated value and settlement key, respectively. This preserves the baseline's coding and sampling parameters while removing its per-sample in-circuit encryption checks.

This direct connection must preserve zero knowledge. The astute reader may expect the simulator to forge an opening at an arbitrary evaluation using the KZG trapdoor, and then to invoke the zero knowledge of the SNARK that takes this evaluation as input. The file commitment and public KZG parameters are supplied independently of our construction, so the simulator receives neither the file nor the setup's secret trapdoor. We bypass this obstacle by opening the polynomial in group-encoded form, and connect the opening to the SNARK with a CP-Link. The evaluation then lies in the same group as the KZG parameters, which lets the simulator derive an accepting opening from the polynomial commitment and those parameters alone. Section 4.2 explains this simulation strategy and its role in the security proof.

The redesign yields two sources of computational savings. First, the circuit no longer checks a public-key encryption relation for each sample; its remaining unmasking and interpolation work uses field arithmetic and still depends on the sample count. Second, removing in-circuit ElGamal relaxes its restrictions on the proof field and key group. Our implementation can therefore use BLS12-381 for both the proof system and the key group, further reducing proving cost and enabling an EVM precompile for the settlement key check [17].

Contributions. Our contributions are threefold:

1. **Consistency without auxiliary ElGamal ciphertexts.** We construct a code-based VECK scheme that replaces the sampled ElGamal interface with direct KZG authentication and external witness linking. Its consistency circuit uses only field arithmetic, while preserving the baseline's coding, masking, and payment interface (Section 4).
2. **Security under an external KZG setup.** For honestly generated, degree-bounded reference commitments and externally supplied KZG parameters, we prove correctness, soundness, and zero knowledge in the AGM and ROM (Theorem 1). The bounds account for sampling, random-evaluation error, and transcript grinding; simulation requires no external KZG trapdoor.
3. **Efficiency at low redundancy.** On the same BW6-761 SNARK curve, the redesign reduces constraints by about 20× and proving time by 25× (Table 1). For a 2 MiB file at $\beta \approx 1.1$, our BLS12-381 implementation reduces aggregate sender computation from 179.6 to 2.75 s (65.3×), including the benefit of changing curves, with bulk ciphertext about 1.1× the plaintext size. Complete successful seller transactions use 31.3× less gas in the tested EVM configuration (Table 3).

2 Related Work

Fair exchange and verifiable encryption. Classical fair exchange is impossible in full generality without some trusted mechanism [1]; optimistic protocols minimize this trust by contacting a third party only on aborts or disputes [2, 3]. Blockchain-based FDE replaces the trusted third party with smart-contract enforcement, using either an on-chain dispute procedure, as in FairSwap [4], or a succinct payment condition based on VECK, as in Tas et al. [5]. Classical verifiable encryption proves that a public-key ciphertext encrypts a plaintext satisfying a specified relation [6]. VECK adapts this idea to fair data exchange by providing separate checks for ciphertext consistency with the agreed data commitment and for the validity of a subsequently released decryption key [5]. This interface supports symmetric-key encryption and allows payment to be conditioned on valid key release.

Code-based FDE. Khabbazian uses Reed–Solomon coding and Fiat–Shamir sampling to reduce consistency checking from the whole file to sampled positions [7]. $\mathrm{VECK}^{+}_{\mathrm{EL}}$ retains full-file exponential ElGamal, whereas $\mathrm{VECK}^{\star}_{\mathrm{EL}}$ uses hash masking for the bulk data and auxiliary ElGamal ciphertexts for the samples. We retain the latter's coding, masking, and sampling strategy, but replace its sampled-ElGamal interface with direct KZG subset authentication and external witness links.

Commit-and-prove linking. LegoSNARK supplies modular committed-witness proofs, and CP-Link connects their witnesses to external group relations [9, 11]; our implementation uses these tools. Artemis [18] and our work meet the same difficulty: evaluating a pre-committed polynomial without breaking zero knowledge. Artemis handles it with a hiding term; it sidesteps the difficulty by opening a masked aggregate of the external commitments. Their approach, however, assumes that those commitments are already hiding, whereas our reference commitment is a plain KZG commitment carrying no hiding randomness, so their argument does not apply directly. Our construction instead uses a group-encoded KZG opening together with CP-Link to preserve zero knowledge for a plain KZG commitment (Section 4.2 and Appendix A.7). This gives an FDE-specific composition of existing polynomial tests and linking tools, together with its security argument.

3 Exchange Model and Background

3.1 Fair Data Exchange and VECK

VECK—verifiable encryption under a committed key—proves that a ciphertext encrypts data consistent with a public commitment under a key bound to a public verification key [5]. In FDE, the buyer checks this proof off-chain before depositing payment; the contract releases the funds upon a valid key revelation.

Given an external crs, $\mathsf{Gen}(crs)$ generates public parameters. For a function F and committed witness w, $\mathsf{Enc}(F, C_w, w)$ outputs $(\mathsf{vk}, \mathsf{sk}, ct, \pi)$. The verifier checks ciphertext consistency with Ver_{ct} and the released key with Ver_{key}; $\mathsf{Dec}(F, \mathsf{sk}, ct)$ returns a value v or $\perp$. Here w is a file polynomial and F returns its designated evaluations. We use the correctness, soundness, and pre-key-release zero-knowledge games in Appendix A.1.

3.2 Notation

Let λ be the security parameter and $x \xleftarrow{\$} S$ denote uniform sampling. The bilinear groups G_1, G_2, G_T have prime order p; file symbols, masks, keys, challenge points, and circuit values lie in their scalar field $\mathbb{F}_p$. Write $[B] = \{1, \ldots, B\}$. A file contains ℓ evaluations $(\phi(i))_{i\in[\ell]}$ of $\phi \in \mathbb{F}_p[X]$ with degree at most $d = \ell - 1$. Encoding expands it to $m = \lceil \beta\ell \rceil$ positions for redundancy $\beta > 1$; R denotes the number of sampled positions.

For a finite ordered set $S \subseteq \mathbb{F}_p$, let $L_{i,S}$ be the Lagrange basis polynomial of degree at most $|S|-1$, equal to one at i and zero at the other points of S. Define

$$V_S(X) = \prod_{i\in S}(X - i), \qquad \phi_S(X) = \sum_{i\in S} \phi(i) L_{i,S}(X).$$

Thus ϕ_S agrees with ϕ on S and has degree below $|S|$.

3.3 Bilinear Groups and KZG Commitments

Let g_1, g_2 generate G_1, G_2, with pairing $e : G_1 \times G_2 \to G_T$. For maximum degree n, KZG setup [8] samples $\tau \xleftarrow{\$} \mathbb{F}_p$ and publishes $crs = (\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R})$, where $\mathcal{G}$ describes the groups. A polynomial ϕ of degree at most n has commitment $C_\phi = g_1^{\phi(\tau)}$. To open $\phi(\alpha) = y$, the prover sends $\pi_\alpha = g_1^{q_\alpha(\tau)}$, where $q_\alpha(X) = (\phi(X) - y)/(X - \alpha)$; verification checks

$$e(C_\phi / g_1^y, g_2) = e(\pi_\alpha, g_2^{\tau - \alpha}).$$

We use KZG extractable binding and evaluation soundness in the AGM, with error ϵ_{KZG}, under the n-DLOG and auxiliary-CRS assumptions of Theorem 1. Lemma 2 covers both ordinary and subset openings.

3.4 Commit-and-Prove SNARKs

A CP-SNARK proves that hidden committed values (u_j), together with an auxiliary witness ω, satisfy $\mathcal{R}(x, (u_j), \omega)$ [9]. We require completeness, knowledge soundness, and zero knowledge with simulated setup (Appendix A.2).

Its linking primitive CPLink proves that commitments under different keys contain the same value. We instantiate the LegoSNARK link with the Kiltz–Wee QA-NIZK for linear subspaces [9, 11]. Its AGM knowledge soundness under DL applies to witness-samplable subspace matrices, including our two single-base equality links. This interface connects circuit witnesses to external group elements without placing the group equations inside the circuit.

3.5 Reed–Solomon Redundancy

The RS codeword $(\phi(i))_{i\in[m]}$ extends the file's ℓ evaluations. Its redundancy permits recovery of the degree-at-most-d polynomial despite up to $t = \lfloor (m-\ell)/2 \rfloor$ incorrect entries [10]. For evaluations $y = (y_i)_{i\in S}$, $\mathrm{RS.Det}(S, y)$ returns zero exactly when y is consistent with a degree-at-most-d polynomial. $\mathrm{RS.Dec}(S', S, y)$ returns the evaluations on S' of the unique bounded-distance decoding, or $\perp$ on failure. After unmasking, decryption returns the file prefix if detection reports no error and otherwise invokes the decoder.

3.6 Fiat–Shamir Sampling

Masking, subset sampling, and point sampling use domain-separated calls to a random oracle H. Transcripts use an injective encoding: canonical fixed-width field and group elements and length-prefixed vectors. After the bulk ciphertext is fixed,

$$S_R = \mathsf{DeriveSubset}_H(C_\phi, \mathsf{vk}, \{ct_i\}_{i\in[m]}, m, R)$$

derives R distinct positions in $[m]$. $\mathsf{DerivePoint}_H$ derives α outside the sampled and two auxiliary positions after the auxiliary ciphertexts, sample commitment, and subset proof are fixed (Figure 2). Both procedures hash the transcript with a counter and use rejection sampling, binding each challenge to the preceding transcript.

Admissible parameters. For at most q_S subset-challenge queries, the grinding-aware sampling target and domain condition are

$$q_S \left(\frac{\beta+1}{2\beta}\right)^R \le 2^{-\lambda}, \qquad m = \lceil \beta\ell \rceil \ge R.$$

The domain condition applies to the encoded word, not the original file. We consider only parameters meeting both conditions. Theorem 1 gives the complete soundness bound; Appendix B.1 gives the equivalent lower bound on β.

4 Construction and Security

We now present our code-based VECK construction. Before an exchange, the parties identify the file by a reference KZG commitment C_ϕ and a declared degree bound d. The committed polynomial ϕ is extended to a Reed–Solomon codeword over $[m]$. The verifier checks a Fiat–Shamir-derived sample, while Reed–Solomon detection and decoding guarantee recovery within the decoding radius. The FDE functionality is relative to C_ϕ: it guarantees recovery of the committed field elements, not the provenance, quality, or application-level meaning of the asset.

Our core security model follows prior VECK work and treats C_ϕ as an honestly generated degree-d reference commitment fixed before the seller's per-sale transcript [5, 7]. This does not require a separate trusted party to create the

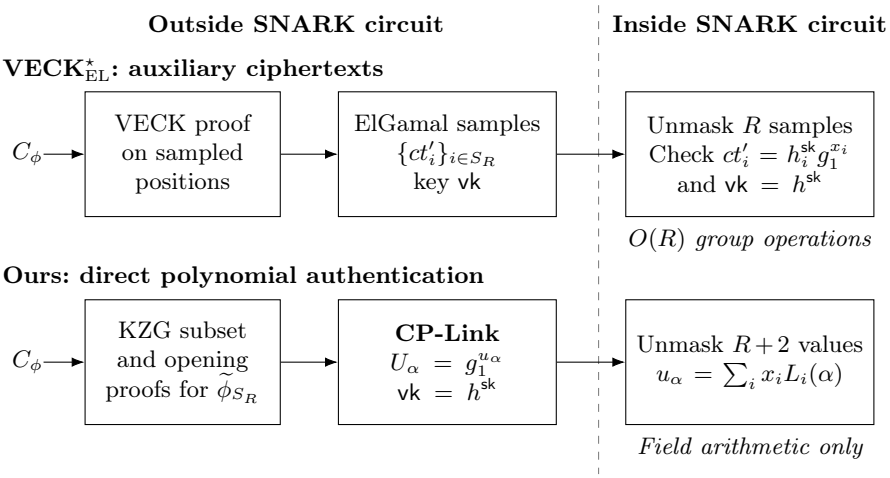


Fig. 1. Consistency-proof interfaces: group operations inside the circuit for the baseline [7] versus field arithmetic only for ours.

reference commitment. Once C_ϕ is fixed, the seller is modeled as *fully malicious*: it chooses $(\mathsf{vk}, \mathsf{sk}, ct, \pi)$ freely and may try to obtain payment while sending ciphertexts that do not decode to the committed file (Definition 2). If the seller also creates C_ϕ, the optional registration check of Appendix A.6 enforces the declared degree bound under a separate assumption. We further assume that the blockchain or payment contract faithfully executes key verification.

4.1 Overview

The consistency proof must show that unmasking the sampled ciphertexts with the settlement key yields the committed values: $ct_i - H(\mathsf{sk}, i) = \phi(i)$ for $i \in S_R$. The baseline connects these two sides through auxiliary ElGamal ciphertexts. We instead authenticate a sample polynomial directly with KZG and link it to a field-arithmetic circuit. Figure 1 shows which operations each construction places inside and outside the circuit, and Figure 2 gives the complete protocol.

Authenticating the samples. After deriving S_R from the masked codeword, the seller interpolates ϕ_{S_R} and blinds it as

$$\widetilde{\phi}_{S_R}(X) = \phi_{S_R}(X) + (t_1 + t_2 X) V_{S_R}(X), \qquad t_1, t_2 \xleftarrow{\$} \mathbb{F}_p.$$

Because V_{S_R} vanishes at the sampled positions, this preserves their values. The seller commits to $\widetilde{\phi}_{S_R}$ as C_{S_R} and supplies a KZG subset proof π_{S_R} certifying agreement with the file commitment on S_R. The blinding lets these checks authenticate the samples without revealing them.

Checking the masked values. The blinded polynomial has degree at most $R+1$, so we interpolate it over $S'_R = S_R \cup \{-1,-2\}$. The seller sends two additional masked evaluations, ct_{-1} and ct_{-2}, of this polynomial at the fixed positions outside $[m]$. Once the ciphertexts, C_{S_R}, and π_{S_R} are fixed, the transcript determines a random $\alpha \notin S'_R$. The circuit checks

$$ct_i = x_i + H(\mathsf{sk}, i) \quad (i \in S'_R), \qquad u_\alpha = \sum_{i \in S'_R} x_i L_{i,S'_R}(\alpha).$$

The verifier computes the Lagrange coefficients from S'_R and α as public inputs. Thus the circuit only checks unmasking and a weighted sum over field elements, keeping sk, x_i, and u_α private.

Linking outside the circuit. A KZG opening proof π_α authenticates the group encoding $U_\alpha = g_1^{\widetilde{\phi}_{S_R}(\alpha)}$. External CPLink proofs bind this encoding and vk to the *same* hidden u_α and sk used by the circuit, enforcing $U_\alpha = g_1^{u_\alpha}$ and $\mathsf{vk} = h^{\mathsf{sk}}$. Both group relations are proved outside the arithmetic circuit through its committed witness; they are stacked into one Kiltz–Wee instance (Appendix A.2). Section 4.2 explains why this interface also supports simulation under the external KZG setup.

4.2 Design rationale

We explain the settlement and simulation consequences of external linking, followed by the setup requirements of the combined construction.

Lowering the on-chain cost. The on-chain cryptographic predicate is the key check $\mathsf{vk} = h^{\mathsf{sk}}$, whose cost depends on the key group. In the $\mathrm{VECK}^{\star}_{\mathrm{EL}}$ implementation, sampled in-circuit ElGamal couples the consistency proof to a curve chain and places the key on BLS12-377. Delegating $\mathsf{vk} = h^{\mathsf{sk}}$ and $U_\alpha = g_1^{u_\alpha}$ to CPLink removes these group relations from our circuit, allowing our implementation to use BLS12-381 for both the proof system and the key group. Its EVM precompile lowers the key-check cost [17], as measured in Section 5.3. This advantage is relative to the sampled-ElGamal baseline, not exclusive to our choice of CP connector.

Simulating the fingerprint interface. The reference commitment and universal KZG CRS are external inputs, so our simulator receives neither the file polynomial nor the KZG trapdoor τ. This restriction concerns the external KZG setup; the simulator may still use the CP-SNARK's own simulated-setup state. After simulating the subset transcript, it has C_{S_R} but does not know the fingerprint scalar. Our group-encoded interface lets it construct the opening transcript without recovering that scalar. Specifically, it samples $y_2 \xleftarrow{\$} \mathbb{F}_p$ and sets

$$\pi_\alpha = g_1^{y_2}, \qquad U_\alpha = \frac{C_{S_R}}{(g_1^{\tau}/g_1^{\alpha})^{y_2}}.$$

Both values are computable from the public CRS and satisfy the KZG opening equation without revealing or computing $\log_{g_1} U_\alpha$. Validity alone does not establish zero knowledge. After the CP-proof and masking hybrids, the remaining

Gen$(crs) \to pp$: On input $crs = (\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R})$

1. Set $h := g_1$.
2. $crs_z \xleftarrow{\$}$ zk-CP-SNARK.KeyGen(g_1, h, R), including the link key.
3. Output $pp = (crs, crs_z, h)$.

Enc$(F_{[\ell]}, C_\phi, \phi(X)) \to (\mathsf{vk}, \mathsf{sk}, ct, \pi)$:

1. Sample $s \xleftarrow{\$} \mathbb{F}_p$.
2. Set $\mathsf{sk} := s$ and $\mathsf{vk} := h^s \in G_1$.
3. Compute $ct_i := \phi(i) + H(\mathsf{sk}, i)$ for all $i \in [m]$ (on an H-domain separate from $\mathsf{DeriveSubset}_H$ and $\mathsf{DerivePoint}_H$).
4. Derive $S_R := \mathsf{DeriveSubset}_H(C_\phi, \mathsf{vk}, \{ct_i\}_{i\in[m]}, m, R)$.
5. Define $\phi_{S_R}(X) := \sum_{i\in S_R} \phi(i)\, L_{i,S_R}(X)$.
6. Sample $t_1, t_2 \xleftarrow{\$} \mathbb{F}_p$ and compute $\widetilde{\phi}_{S_R}(X) := \phi_{S_R}(X) + (t_1 + t_2 X)\, V_{S_R}(X)$.
7. Compute $C_{S_R} := \mathsf{Commit}(\mathsf{crs}, \widetilde{\phi}_{S_R}(X))$.
8. Compute $\pi_{S_R} := \mathsf{Commit}\left(\mathsf{crs}, \frac{\phi(X) - \widetilde{\phi}_{S_R}(X)}{V_{S_R}(X)}\right)$.
9. Let $S'_R := S_R \cup \{-1, -2\}$, where the two auxiliary points are fixed outside $[m]$.
10. For each $i \in \{-1, -2\}$, compute $ct_i := \widetilde{\phi}_{S_R}(i) + H(\mathsf{sk}, i)$.
11. Derive $\alpha := \mathsf{DerivePoint}_H(C_\phi, \mathsf{vk}, \{ct_i\}_{i\in[m]\cup\{-1,-2\}}, m, R, C_{S_R}, \pi_{S_R}, S_R) \notin S'_R$.
12. Compute $u_\alpha,\ \pi_\alpha := \mathsf{KZG.Open}(\mathsf{crs}, \alpha, \widetilde{\phi}_{S_R}(X))$ and $U_\alpha := g_1^{u_\alpha}$.
13. Compute a zk-CP-SNARK proof π_Z for the arithmetic relation R_{mask} and two external CPLink relations. The full statement and witness, with the circuit and link checks distinguished, are:

    $\mathsf{stmt} = (h,\ S'_R,\ \{ct_i\}_{i\in S'_R},\ \{L_{i,S'_R}(\alpha)\}_{i\in S'_R},\ U_\alpha,\ \mathsf{vk})$,
    $\mathsf{wit} = (u_\alpha,\ \mathsf{sk},\ \{x_i\}_{i\in S'_R})$,
    Circuit: $\forall i \in S'_R : ct_i = x_i + H(\mathsf{sk}, i)$; $\quad u_\alpha = \sum_{i\in S'_R} x_i\, L_{i,S'_R}(\alpha)$;
    External links: $U_\alpha = g_1^{u_\alpha}$; $\quad \mathsf{vk} = h^{\mathsf{sk}}$.

14. Output $\big(\mathsf{vk}, \mathsf{sk}, ct = \{ct_i\}_{i\in[m]\cup\{-1,-2\}}, \pi = (C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha, \pi_Z)\big)$.

Ver$_{ct}(F_{[\ell]}, C_\phi, \mathsf{vk}, ct, \pi) \to 0/1$:

1. Parse π as $(C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha, \pi_Z)$ and compute the Fiat–Shamir challenge S_R and α.
2. Verify $e(C_\phi / C_{S_R}, g_2) = e(\pi_{S_R}, g_2^{V_{S_R}(\tau)})$.
3. Verify the opening proof π_α: $e(C_{S_R}/U_\alpha, g_2) = e(\pi_\alpha, g_2^{\tau-\alpha})$.
4. Compute $\{L_{i,S'_R}(\alpha)\}_{i\in S'_R}$ and verify the zk-CP-SNARK proof π_Z with public inputs.

Ver$_{key}(\mathsf{vk}, \mathsf{sk}) \to 0/1$: For $\mathsf{sk} = s \in \mathbb{F}_p$, return 1 if and only if $\mathsf{vk} = h^s$ in G_1.

Dec$(F_{[\ell]}, \mathsf{sk}, ct) \to \{\phi(i)\}_{i\in[\ell]} \cup \{\perp\}$:

1. Compute the unmasked evaluations $\{\phi'(i) := ct_i - H(\mathsf{sk}, i)\}_{i\in[m]}$.
2. If RS.Det$([m], \{\phi'(i)\}_{i\in[m]}) = 0$ (no error), then output $\{\phi'(i)\}_{i\in[\ell]}$.
3. Otherwise, return RS.Dec$([\ell], [m], \{\phi'(i)\}_{i\in[m]})$, including $\perp$ on failure.

Fig. 2. Our code-based VECK construction

blinding randomness makes the opening-proof exponent uniform conditioned on the subset transcript and challenge. Appendix A.7 uses this fact to match the joint transcript distribution. Conversely, for soundness, CP-SNARK knowledge extraction recovers the scalar u_α linked to U_α, allowing the KZG opening argument to apply. The same interface thus supports simulation without the scalar and extraction of it from an accepting prover.

Setup profile and the choice of proving system. Beyond the external universal KZG CRS shared by all schemes compared here, Gen samples the scheme-specific string crs_z carrying the CP-SNARK and linking keys. Our implementation uses LegoGroth16 with Kiltz–Wee linking [9, 11], retaining circuit-specific setup as in the Groth16-based direct baseline. Tas et al.'s Σ-protocol instead needs no additional trusted setup beyond its universal SRS [5]. A replacement backend must support both external witness links, satisfy the knowledge-soundness and zero-knowledge interface of Appendix A.2, and meet the τ-algebraic CRS compatibility condition of Definition 5. A universal-setup backend could avoid the circuit-specific ceremony, but its linking layer and security composition would need to be instantiated and its costs evaluated separately.

4.3 Security

Soundness follows from a spot-checking argument: passing the sampled check on enough positions forces agreement with a codeword within the decoding radius, and hence with the committed file. Here the sampled check is realized by KZG subset consistency together with the CP-SNARK. The only extra loss relative to the ideal sampled check is a Schwartz–Zippel term for the maximum degree n supported by the KZG CRS. Our security model follows the prior VECK constructions [5, 7], which are also analyzed in the AGM and ROM. Hash masking removes the DDH assumption of their ElGamal variants and the DCR assumption of their Paillier variant, leaving the assumptions listed below.

Theorem 1 (Security of the construction). *Work in the algebraic group model (AGM) and the random oracle model (ROM), and let q_S, q_P bound the adversary's subset- and point-challenge queries. Fix a file polynomial ϕ of degree at most d and its honestly generated reference commitment $C_\phi = \mathsf{Commit}(crs, \phi)$ before the sale-specific transcript is formed. Assume:*

(i) the n-discrete-logarithm problem for the KZG CRS is hard, so that the KZG scheme is extractable-binding and evaluation-sound for degree at most the CRS-supported bound n; we write ϵ_{KZG} for the aggregate advantage of the constant number of reductions to this problem used in Lemma 2;

(ii) the commit-and-prove SNARK with its CPLink instance for $U_\alpha = g_1^{u_\alpha}$ and $\mathsf{vk} = h^{\mathsf{sk}}$ (link key on bases (g_1, h)) is complete, knowledge-sound (error $\epsilon_{\mathsf{CP}}^{\mathsf{snd}}$), and zero-knowledge (error $\epsilon_{\mathsf{CP}}^{\mathsf{zk}}$, Definition 6);

(iii) the auxiliary CP-SNARK setup satisfies the τ-algebraic compatibility property of Definition 5, as realized by Corollaries 1–2;

(iv) the discrete-logarithm problem is hard in G_1 with advantage at most $\epsilon_{\mathsf{DL}}^{G_1}$;

(v) H is modeled as a random oracle; and

(vi) the Reed–Solomon detector RS.Det *and decoder* RS.Dec *are correct within the decoding radius* $t = \lfloor (m-\ell)/2 \rfloor$.

Assume also that $n \geq \max\{d, R+1\}$, $m \geq R$, $m < p-2$, *and* $R+2 < p$. *These conditions support both committed polynomials, make the sample and auxiliary points well defined, and leave a nonempty challenge domain for* α. *If the redundancy factor satisfies* $q_S\big((\beta+1)/(2\beta)\big)^R \leq 2^{-\lambda}$, *then the construction of Figure 2 is a secure VECK in the sense of Tas et al. [5]: it is correct, sound, and computationally zero-knowledge. Quantitatively, every probabilistic polynomial-time algebraic adversary respecting these grinding budgets has* soundness advantage*—the probability of producing an accepting transcript whose revealed key decrypts to a file different from the committed one—at most*

$$\epsilon_{\mathsf{KZG}} + \epsilon_{\mathsf{CP}}^{\mathsf{snd}} + q_P \frac{n}{p-R-2} + q_S \left(\frac{\beta+1}{2\beta}\right)^R,$$

where the two factors separately account for Fiat–Shamir grinding over S_R *and* α, *and* zero-knowledge distinguishing advantage *at most*

$$\epsilon_{\mathsf{CP}}^{\mathsf{zk}} + \epsilon_{\mathsf{DL}}^{G_1}.$$

Here $F(\phi) = (\phi(1), \ldots, \phi(\ell))$ denotes full-file retrieval. The core theorem uses the reference-commitment game of Definition 2; the adversary does not choose C_ϕ. Proposition 1 separately extends the soundness bound to a seller-generated commitment that passes the optional degree check.

Proof (Proof overview). The three VECK properties are proved in full in Appendix A (Lemmas 3–5) and assembled in Appendix A.8.

Correctness holds by inspection: the blinded $\widetilde{\phi}_{S_R}$ agrees with ϕ on S_R with degree at most $|S_R|+1$, so S'_R supplies the $|S_R|+2$ interpolation points, the two KZG checks and the CP-SNARK relation verify, and honest unmasking yields the exact codeword.

Soundness. CP-SNARK knowledge soundness extracts $\mathsf{sk}, u_\alpha, \{x_i\}_{i \in S'_R}$ satisfying the masking relation and both CPLink equations; here $U_\alpha = g_1^{u_\alpha}$ turns the group element into the *explicit* scalar opening to which KZG soundness applies (Lemma 2), and $\mathsf{vk} = h^{\mathsf{sk}}$ pins the extracted key to the $\log_h \mathsf{vk}$ used by the decryptor. A Schwartz–Zippel argument at α then forces $ct_i - H(\mathsf{sk}, i) = \phi(i)$ on S_R, and the Reed–Solomon sampling bound catches any accepted word beyond the decoding radius except with probability $((\beta+1)/2\beta)^R$ per sample; union-bounding over the q_S and q_P challenges gives the stated bound.

Zero-knowledge is a five-hybrid argument (Lemma 5). The simulator, which never learns τ, replaces π_Z by a simulated proof, the hash-masked word by uniform field elements (the only bad event, a prior query $H(\mathsf{sk}, \cdot)$, reduces to discrete log in G_1), and the KZG opening and subset transcripts by uniform valid pairing transcripts from the public CRS alone—using the group-element opening U_α, not the scalar u_α.

5 Evaluation

We evaluate sender computation, proof verification, communication, and successful EVM settlement. For the direct baseline and ours, sender time is the sum of per-stage medians over five runs. Sender timings exclude the pre-existing file commitment, SRS loading, setup, network transfer, and contract execution. Buyer decoding, settlement latency, and peak memory are not evaluated. Only the contextual VECK and $\text{VECK}^{+}_{\text{EL}}$ results contain extrapolations, described in Appendix B.7.

5.1 Experimental Setup

We implemented KZG in Rust/arkworks and the SNARK in Go/gnark, using Groth16, Poseidon2, and one stacked Kiltz–Wee link for U_α and vk.[4] Experiments use an Apple M3 Pro with eleven worker threads and 18 GB unified memory. Implementation versions and the artifact are given in Appendix B.1. The proof models the hash calls as random oracles; Poseidon2 is their concrete heuristic instantiation.

Comparison configurations. $\text{VECK}^{\star}_{\text{EL}}$, our direct baseline, uses BLS12-377 for KZG and sampled ElGamal and BW6-761 for the SNARK. We test ours with both layers on BW6-761 and with both on BLS12-381. The former holds only the *SNARK curve* fixed; its polynomial layer uses a larger field than the baseline, about 1.5× wider per element, so this comparison favors the baseline. VECK and $\text{VECK}^{+}_{\text{EL}}$ provide context for full-file ElGamal costs in Appendix B.7.

The reference case is $\ell = 2^{16}$, nominal $\beta = 1.1$, and $R = 2384$: a 2 MiB file for the direct baseline and ours on BLS12-381. Comparisons fix the number of file elements and samples, not the byte size across all curves; ours on BW6-761 uses wider elements. We fix $\lambda = 128$, $q_S, q_P \leq 2^{32}$, and the (β, R) pairs in Table 2. Displayed β values are rounded labels; encoding uses the actual $m = \lceil \beta\ell \rceil$ (Appendix B.1). The protocol requires $m \geq R$; the measured grid additionally requires $R < \ell$, which is not a protocol condition. At $\beta = 1.1$ it starts at $\ell = 2^{12}$, and at $\beta = 1.25$ at $\ell = 2^{11}$.

5.2 Circuit and Sender Costs

Separating circuit and curve effects. Table 1 summarizes the reference case. On the same BW6-761 SNARK curve, the redesign reduces constraints by about 20× and SNARK proving from 179.1 to 7.172 s (25.0×). Moving ours to BLS12-381 reduces proving further to 2.346 s, about 3.1×. The resulting 76× SNARK speedup therefore includes both effects. Across the four nominal redundancy settings $\beta \in \{2, 1.5, 1.25, 1.1\}$, the same-SNARK-curve constraint reduction remains near 20×; full SNARK resources, including circuit-specific setup and CRS costs, are in Table B.1.

[4] Our implementation is available at `https://github.com/dongwook1214/efficient-fde`.

Table 1. Measured reference-case costs at $\ell = 2^{16}$, nominal $\beta = 1.1$, $R = 2384$.

	$\text{VECK}^{\star}_{\text{EL}}$	Ours	Ours
KZG curve	BLS12-377	BW6-761	BLS12-381
SNARK curve	BW6-761	BW6-761	BLS12-381
Constraints	9,573,767	477,208	477,208
SNARK proving (s)	179.1	7.172	2.346
Aggregate sender (s)	179.607	8.118	2.749

Table 2. Aggregate sender computation (s) at $\ell = 2^{16}$.

Scheme (KZG / SNARK)	$\beta = 2$ $R = 386$	$\beta = 1.5$ $R = 609$	$\beta = 1.25$ $R = 1053$	$\beta = 1.1$ $R = 2384$
$\text{VECK}^{\star}_{\text{EL}}$ BLS12-377 / BW6-761	16.997	29.027	61.986	179.607
Ours BW6-761 / BW6-761	2.621	3.212	4.705	8.118
Ours BLS12-381 / BLS12-381	0.773	0.947	1.463	2.749

The corresponding aggregate sender costs are 179.607, 8.118, and 2.749 s. Thus the final BLS12-381 implementation yields the 65.3× sender reduction, distinct from the SNARK-only ratio. The BW6-761 sender comparison includes different KZG fields and is not an all-layers-matched experiment.

The cost of lowering redundancy. Table 2 lowers nominal β from 2 to 1.1 at fixed $\ell = 2^{16}$, increasing R from 386 to 2384. The baseline rises from 16.997 to 179.607 s; ours on BLS12-381 rises from 0.773 to 2.749 s. Our low-redundancy configuration therefore takes 6.2× less computation than the baseline's fastest tested setting, while using about 1.1 rather than 2 bulk symbols per file element. This comparison credits the baseline with plaintext-width ciphertext encoding without charging its additional modular-reduction circuit work (Section 5.3).

Where the improvement comes from. At the reference point, SNARK proving accounts for 179.096 s, or 99.7% of the baseline's sender aggregate; native sampled public-key operations take only 0.104 s. The baseline and ours on BLS12-381 spend similar time on interpolation and commitment (104.3 versus 102.2 ms) and KZG proving (221.9 versus 220.4 ms). The full breakdown is retained in Table B.2.

Scaling with file size. Figure 3 fixes nominal $\beta = 1.25$ and varies ℓ over the admissible measured grid. At this fixed sampling budget, our BLS12-381 configuration saves about 60.5 s of sender computation across $\ell = 2^{11}$–2^{20}. At the largest file size, the baseline takes about 65.96 s and ours 5.462 s. The contextual full-file ElGamal comparisons, which include extrapolated stages, are kept separate in Appendix B.7.

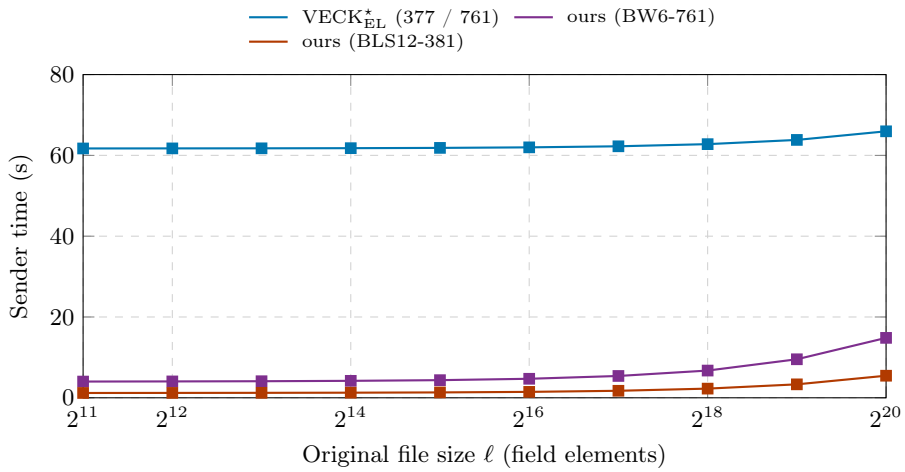


Fig. 3. Measured-component sender aggregates at nominal $\beta = 1.25$, $R = 1053$, over $\ell = 2^{11}$–2^{20}. The baseline uses BLS12-377 for KZG and BW6-761 for its SNARK; ours uses the indicated curve for both layers.

5.3 Settlement, Verification, and Communication

Successful settlement. Table 3 measures complete buyer `deposit` and seller `settle` transactions, including base and calldata gas. The seller call checks the released key and releases escrow. The BLS12-381 precompile reduces our seller call to 43,154 gas, versus 1,349,585 for the baseline's BLS12-377 implementation (31.3×); both calls together improve by 15.9×. These are successful-path costs for the tested state and keys, excluding deployment. The key-check component and one-time deployment costs are reported separately in Appendix B.6.

Table 3. Gas for successful buyer and seller transactions.

Operation Key curve	VECK$^{\star}_{\mathrm{EL}}$ BLS12-377	Ours BLS12-381	Reduction
Buyer call	44,370	44,278	1.0×
Seller call	1,349,585	43,154	31.3×
Sum of calls	1,393,955	87,432	15.9×

Core proof verification. As in the prior code-based construction [7], we time proof verification after the transcript and sampled entries are fixed, excluding parsing and the one-pass transcript hash. Ours on BLS12-381 takes 10.8–28.6 ms across the four settings, a 2.2–4.5× reduction; at $\beta = 1.1$, the baseline takes 127.7 ms and ours 28.6 ms. The same-SNARK-curve BW6-761 configuration

is slower than the baseline at $\beta \in \{2, 1.5, 1.25\}$ and faster at 1.1 (77.9 ms). Verification gains are therefore smaller than proving gains and do not remove linear transcript processing. Table B.4 gives the complete comparison.

Communication. Our BLS12-381 ciphertext, proof, and verification key occupy $32(m+2)+528$ bytes. At the 2 MiB reference point, the bulk expansion is about $1.1\times$ and the total is about $1.1003\times$, including the two auxiliary values and proof/key elements (Appendix B.5).

The baseline's plaintext uses 32-byte symbols but its implemented BW6-761 masks use 48-byte ciphertext symbols, giving bulk expansion about $1.6501\times$ at this point. We instead credit it with a normalized $32m$-byte bulk ciphertext, about $1.1\times$, without charging the extra modular-reduction circuit work. This favors the baseline: its reported timing and normalized communication are not a jointly measured compact implementation.

6 Conclusion

We replace sampled in-circuit ElGamal checks with external KZG authentication and witness linking, leaving only field arithmetic in the consistency circuit. The construction preserves the code-based recovery and sampling guarantees and supports simulation without the external KZG trapdoor. Our evaluation separates the benefit of the circuit redesign from the additional proving and settlement gains of the BLS12-381 instantiation. Together, these changes reduce the computational cost of low redundancy while retaining near-plaintext bulk communication under the reference-commitment model.

References

1. Henning Pagnia and Felix C. Gärtner. On the Impossibility of Fair Exchange Without a Trusted Third Party. Technical Report TUD-BS-1999-02, Darmstadt University of Technology, 1999.
2. N. Asokan, Victor Shoup, and Michael Waidner. Optimistic Fair Exchange of Digital Signatures. In *Advances in Cryptology – EUROCRYPT 1998*, Lecture Notes in Computer Science. Springer, 1998.
3. Juan A. Garay, Markus Jakobsson, and Philip D. MacKenzie. Abuse-Free Optimistic Contract Signing. In *Advances in Cryptology – CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466. Springer, 1999. doi: 10.1007/3-540-48405-1_29.
4. Stefan Dziembowski, Lisa Eckey, and Sebastian Faust. FairSwap: How to Fairly Exchange Digital Goods. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*, pages 967–984, 2018. doi: 10.1145/3243734.3243857.
5. Ertem Nusret Tas, István András Seres, Yinuo Zhang, Márk Melczer, Mahimna Kelkar, Joseph Bonneau, and Valeria Nikolaenko. Atomic and Fair Data Exchange via Blockchain. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24)*, pages 3227–3241, 2024. doi: 10.1145/3658644.3690248.
6. Jan Camenisch and Victor Shoup. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer, 2003. doi: 10.1007/978-3-540-45146-4_8.
7. Majid Khabbazian. Plaintext-Scale Fair Data Exchange. *arXiv preprint* arXiv:2506.14944, 2025. To appear in *Financial Cryptography and Data Security (FC 2026)*.
8. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-Size Commitments to Polynomials and Their Applications. In *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010. doi: 10.1007/978-3-642-17373-8_11.
9. Matteo Campanelli, Dario Fiore, and Anaïs Querol. LegoSNARK: Modular Design and Composition of Succinct Zero-Knowledge Proofs. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*, pages 2075–2092, 2019. doi: 10.1145/3319535.3339820.
10. 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. doi: 10.1137/0108018.
11. Eike Kiltz and Hoeteck Wee. Quasi-Adaptive NIZK for Linear Subspaces Revisited. In *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9056 of *Lecture Notes in Computer Science*, pages 101–128. Springer, 2015. doi: 10.1007/978-3-662-46803-6_4.
12. Jens Groth. On the Size of Pairing-Based Non-interactive Arguments. In *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2016. doi: 10.1007/978-3-662-49896-5_11.
13. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge. Cryptology ePrint Archive, Paper 2019/953, 2019.

14. Behzad Abdolmaleki and Daniel Slamanig. CRS-Updatable Asymmetric Quasi-Adaptive NIZK Arguments. In *Progress in Cryptology – INDOCRYPT 2022*, volume 13774 of *Lecture Notes in Computer Science*, pages 3–25. Springer, 2022.
15. Helger Lipmaa. Plonk is Simulation Extractable in ROM Under Falsifiable Assumptions. In *Theory of Cryptography – TCC 2025*, Lecture Notes in Computer Science. Springer, 2025. Cryptology ePrint Archive, Paper 2025/1759.
16. Lorenzo Grassi, Dmitry Khovratovich, and Markus Schofnegger. Poseidon2: A Faster Version of the Poseidon Hash Function. In *Progress in Cryptology – AFRICACRYPT 2023*, volume 14064 of *Lecture Notes in Computer Science*, pages 177–203. Springer, 2023. doi: 10.1007/978-3-031-37679-5_8.
17. Alex Vlasov, Kelly Olson, Alex Stokes, and Antonio Sanso. EIP-2537: Precompile for BLS12-381 Curve Operations. Ethereum Improvement Proposals, 2020. https://eips.ethereum.org/EIPS/eip-2537.
18. Hidde Lycklama, Alexander Viand, Nikolay Avramov, Nicolas Küchler, and Anwar Hithnawi. Artemis: Efficient Commit-and-Prove SNARKs for zkML. *arXiv preprint* arXiv:2409.12055v2, 2025. https://arxiv.org/abs/2409.12055v2.

A Complete Security Proofs

This appendix proves Theorem 1 in full: after fixing the security model (Appendix A.1) we prove an algebraic-group extraction lemma for the KZG layer (Appendix A.3), then correctness, soundness, and zero-knowledge (Appendices A.4–A.7), assembled in Appendix A.8.

We work throughout in the algebraic group model (AGM) and the random oracle model (ROM). In the AGM every group element an adversary outputs comes with a representation as a known exponent-linear combination of the elements it received; Lemma 2 uses these representations directly for the KZG layer, while the CP-SNARK and CPLink witnesses come from the knowledge extractors assumed in Theorem 1(ii), which our Groth16 and Kiltz–Wee instantiations provide in this model. We write τ for the (secret, external) KZG trapdoor, fix $h := g_1$, and write n for the CRS-supported degree bound and $p = |\mathbb{F}_p|$; recall $S'_R = S_R \cup \{-1, -2\}$ with $|S_R| = R$ and $|S'_R| = R + 2$. Throughout we assume the standing constraints of Theorem 1. In particular, the sample must fit in the encoded domain: $R \le m$. The often useful regime $R \le \ell$ is not required by the theorem.

A.1 Security model and assumptions

In the FDE instantiation the committed object is the file polynomial $w = \phi \in \mathbb{F}_p^{\le d}[X]$, its commitment is the honestly generated KZG commitment $C_\phi = \mathsf{Commit}(crs, \phi) = g_1^{\phi(\tau)}$, and the function to be encrypted is $F(\phi) = (\phi(1), \ldots, \phi(\ell))$. We use the VECK security notions of Tas et al. [5], which we restate here specialized to this F so that the appendix is self-contained.

The experiment fixes this reference commitment before the adversary generates the sale-specific key, ciphertext, and proof. It therefore models a malicious seller but not an adversarial committer. The commitment may originate from any authenticated application workflow; the game does not posit a trusted committer. It fixes which bounded-degree polynomial constitutes the promised data. Appendix A.6 gives the separate extension in which the seller registers a commitment of its choosing.

Definition 1 (Correctness). *For every ϕ of degree at most d,*

$$\Pr\left[\begin{array}{l|l} \mathsf{Ver}_{ct}(F, C_\phi, \mathsf{vk}, ct, \pi) = 1 \wedge & crs \leftarrow \mathsf{Setup}(1^\lambda);\ C_\phi \leftarrow \mathsf{Commit}(crs, \phi) \\ \mathsf{Ver}_{key}(\mathsf{vk}, \mathsf{sk}) = 1 & pp \leftarrow \mathsf{Gen}(crs) \\ & (\mathsf{vk}, \mathsf{sk}, ct, \pi) \leftarrow \mathsf{Enc}(F, C_\phi, \phi) \end{array}\right] = 1,$$

and $\mathsf{Dec}(F, \mathsf{sk}, ct) = F(\phi)$ with probability 1.

Definition 2 (Soundness). *A scheme is* sound *with advantage ϵ_{snd} if for every ϕ of degree at most d and every PPT (algebraic) adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{l|l} \mathsf{Ver}_{ct}(F, C_\phi, \mathsf{vk}, ct, \pi) = 1 \wedge & crs \leftarrow \mathsf{Setup}(1^\lambda);\ pp \leftarrow \mathsf{Gen}(crs) \\ \mathsf{Ver}_{key}(\mathsf{vk}, \mathsf{sk}) = 1 \ \wedge\ y \neq F(\phi) & C_\phi \leftarrow \mathsf{Commit}(crs, \phi) \\ & (\mathsf{sk}, \mathsf{vk}, ct, \pi) \leftarrow \mathcal{A}^H(pp, F, \phi, C_\phi) \\ & y \leftarrow \mathsf{Dec}(F, \mathsf{sk}, ct) \end{array}\right] \le \epsilon_{\mathsf{snd}}.$$

Definition 3 (Computational zero-knowledge). *A scheme is* computationally zero-knowledge *with advantage ϵ_{zk} if there is a PPT simulator* Sim *(with oracle control of H) such that for every ϕ of degree at most d and every PPT distinguisher $\mathcal{D}$,*

$$\Big|\Pr\big[\mathcal{D}^H(pp, C_\phi, \mathsf{vk}, ct, \pi) = 1\big] - \Pr\big[\mathcal{D}^H(\widetilde{pp}, C_\phi, \widetilde{\mathsf{vk}}, \widetilde{ct}, \widetilde{\pi}) = 1\big]\Big| \le \epsilon_{\mathsf{zk}},$$

where the real experiment samples $pp = (crs, crs_z, h)$ by Gen *and runs* Enc*. In the simulated experiment the external KZG crs and $h = g_1$ are identical, while $(\widetilde{crs}_z, \mathsf{st}) \leftarrow \mathsf{S}^{\mathsf{CP}}_{\mathsf{kg}}(g_1, h, R)$, $\widetilde{pp} := (crs, \widetilde{crs}_z, h)$, and $(\widetilde{\mathsf{vk}}, \widetilde{ct}, \widetilde{\pi}) \leftarrow \mathsf{Sim}(\widetilde{pp}, \mathsf{st}, F, C_\phi)$. The simulator receives public F but neither ϕ nor τ, and the view is taken before key revelation. Indistinguishability of the honest and simulated CP reference strings is included in the definition.*

Definition 4 (n-DLOG). *For $\tau \xleftarrow{\$} \mathbb{F}_p$, given the challenge $\big(\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R}\big)$, output τ. An algorithm's advantage $\epsilon_{n\text{-dl}}$ is its probability of success. The exposed powers match the KZG CRS used in Lemma 2: degree n in G_1 and degree R in G_2. We state this q-type challenge explicitly, rather than invoking a generic KZG assumption [8], because the AGM extraction lemma reduces a nonzero polynomial identity at the CRS trapdoor to recovering τ. The success probability is over the sampled τ, the group-generation randomness, and the algorithm's randomness.*

We isolate the property of the auxiliary (CP-SNARK) setup that Lemma 2 actually needs, keeping it separate from the proof-system interface of Theorem 1(ii).

Definition 5 (τ-algebraically compatible auxiliary CRS). *An auxiliary proof-system setup producing G_1-elements $crs_z = (Z_1, \ldots, Z_k)$ is* τ-algebraically compatible *with a KZG CRS $(\{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R})$ if there is a PPT setup simulator (equivalently, a representation algorithm) that outputs, for every Z_j, a polynomial $f_j \in \mathbb{F}_p[X]$ of degree at most $d_{\mathsf{aux}} \le n$ with*

$$Z_j = g_1^{f_j(\tau)},$$

such that the resulting joint public-parameter distribution (crs, crs_z) is identical to the real setup distribution. The f_j are known to the reduction; τ is not.

This one definition subsumes the independent, shared-SRS, and transparent setups, each realized by a distinct representation algorithm.

Corollary 1 (Independent setup). *Independently generated Groth16 and Kiltz–Wee parameters are τ-algebraically compatible with $d_{\mathsf{aux}} = 0$, provided the reduction generates the auxiliary setup itself and retains the exponent representation of each G_1-element: every $Z_j = g_1^{\zeta_j}$ for a known τ-free scalar ζ_j, i.e. $f_j \equiv \zeta_j$. A transparent setup falls in this case, the reduction programming the setup's random oracle (domain-separated from H) to place each $Z_j = g_1^{\zeta_j}$, which preserves the distribution.*

Corollary 2 (Shared SRS). *A shared-SRS proving system whose commitment key reuses the KZG powers is τ-algebraically compatible whenever every auxiliary G_1-element is accompanied by an efficiently computable representation in the KZG power basis $\{\tau^i\}_{i=0}^n$; then $d_{\mathsf{aux}} \le n$, and the reduction reuses the challenge CRS verbatim without knowing τ.*

Proof. In each case the stated representation algorithm returns polynomials f_j of degree at most $d_{\mathsf{aux}} \le n$ with $Z_j = g_1^{f_j(\tau)}$. Corollary 1: the reduction samples the auxiliary trapdoor, so each ζ_j is known and τ-free, and self-generation reproduces the real distribution exactly. Corollary 2: the supplied basis representation gives f_j directly, and reusing the challenge powers leaves the distribution unchanged. In both the joint distribution matches the real one, meeting Definition 5.

Assumptions. We use the assumptions of Theorem 1: n-DLOG (Definition 4) in the bilinear group (advantage $\epsilon_{n\text{-dl}}$), which underlies KZG extractable binding and evaluation soundness in the AGM, and ordinary discrete-logarithm hardness in G_1 (advantage $\epsilon_{\mathsf{DL}}^{G_1}$); the proof-system interface—knowledge soundness, equality linking, and zero-knowledge of the CP-SNARK and CPLink (errors $\epsilon_{\mathsf{CP}}^{\mathsf{snd}}, \epsilon_{\mathsf{CP}}^{\mathsf{zk}}$; for the link, see [11, 9]); τ-algebraic compatibility of the auxiliary CP CRS with the KZG CRS (Definition 5), which is exactly the representation property Lemma 2 requires and which the settings of Corollaries 1–2 realize; and the ROM for H. Since $h = g_1$, Lemma 2 reduces only to n-DLOG, so $\epsilon_{\mathsf{KZG}} \le c\epsilon_{n\text{-dl}}$ for a small constant c. The KZG CRS is external, with τ unknown to all reductions and to the simulator.

A.2 The proof-system interface

Assumption (ii) of Theorem 1 asks two things of the commit-and-prove layer: knowledge soundness and zero knowledge. We first define the property as a game and show how it composes from the link and the core proof. We then give the trusted-setup instantiation used in our implementation and discuss the additional work needed for a universal-setup alternative.

Definition 6 (Zero-knowledge with simulated setup). *Let $\Pi = (\mathsf{KeyGen}, \mathsf{Prove}, \mathsf{VerProof})$ prove a relation $\mathcal{R}$. Π is* zero-knowledge *with error ϵ^{zk} if there is a simulator pair $(\mathsf{S}_{\mathsf{kg}}, \mathsf{S}_{\mathsf{prv}})$ such that:*

(a) (simulated setup) $(\widetilde{crs}, \mathsf{td}) \leftarrow \mathsf{S}_{\mathsf{kg}}(pp)$ *yields a reference string $\widetilde{crs}$ perfectly, statistically, or computationally indistinguishable from an honest $crs \leftarrow \mathsf{KeyGen}(pp)$; and*

(b) (indistinguishability) *for every $(x, w) \in \mathcal{R}$, $(\widetilde{crs}, \mathsf{S}_{\mathsf{prv}}(\widetilde{crs}, \mathsf{td}, x))$ and $(crs, \mathsf{Prove}(crs, x, w))$ are within ϵ^{zk}.*

The next lemma is the modular core of the interface: it builds the property from the two ingredients we actually implement, isolating exactly where the committed witness enters.

Lemma 1 (Composition preserves the interface). *Realize the commit-and-prove layer from*

- *a* core *proof system* $\mathsf{CP}_{\mathsf{core}}$ *proving* $\mathcal{R}_{\mathsf{mask}}$ *over an internal witness commitment cc whose first coordinates are* (u_α, sk)*; and*
- *a* link $\mathsf{CP}_{\mathsf{link}}$ *selecting those coordinates of cc and binding them to the external commitments* $U_\alpha = g_1^{u_\alpha}$ *and* $\mathsf{vk} = h^{\mathsf{sk}}$ *on the bases* (g_1, h)*,*

sharing the commitment cc, which is computationally hiding and computationally binding under the joint public parameters, with binding advantage ϵ_{bind} *against the two extraction procedures below. If* $\mathsf{CP}_{\mathsf{core}}$ *and* $\mathsf{CP}_{\mathsf{link}}$ *are each complete, knowledge-sound (errors* $\epsilon^{\mathsf{snd}}_{\mathsf{core}}, \epsilon^{\mathsf{snd}}_{\mathsf{link}}$*), and zero-knowledge (Definition 6, errors* $\epsilon^{\mathsf{zk}}_{\mathsf{core}}, \epsilon^{\mathsf{zk}}_{\mathsf{link}}$*), then their composition satisfies Theorem 1(ii) with*

$$\epsilon^{\mathsf{snd}}_{\mathsf{CP}} \le \epsilon^{\mathsf{snd}}_{\mathsf{core}} + \epsilon^{\mathsf{snd}}_{\mathsf{link}} + \epsilon_{\mathsf{bind}}, \qquad \epsilon^{\mathsf{zk}}_{\mathsf{CP}} \le \epsilon^{\mathsf{zk}}_{\mathsf{core}} + \epsilon^{\mathsf{zk}}_{\mathsf{link}} + \epsilon_{\mathsf{hide}},$$

where ϵ_{hide} *is the hiding advantage of cc.*

Proof. Knowledge soundness. On an accepting composed proof $(cc, \pi_{\mathsf{core}}, \pi_{\mathsf{link}})$ the core extractor returns $((u_j), \omega)$ with $cc = \mathsf{Commit}((u_j); \rho)$ and $\mathcal{R}_{\mathsf{mask}}((u_j), \omega) = 1$; the link extractor returns openings of the *same* cc and of U_α, vk with equal selected coordinates. The two extractions may a priori return different openings of cc; a pair that disagrees on a selected coordinate is a binding break, which occurs with probability at most ϵ_{bind}. Otherwise they agree, so $U_\alpha = g_1^{u_\alpha}$ and $\mathsf{vk} = h^{\mathsf{sk}}$ for the extracted (u_α, sk); a union bound over the two extraction failures and this event gives the stated error.

Zero-knowledge. Set $\mathsf{S}^{\mathsf{CP}}_{\mathsf{kg}}$ to run $\mathsf{S}^{\mathsf{core}}_{\mathsf{kg}}$ and $\mathsf{S}^{\mathsf{link}}_{\mathsf{kg}}$ and return the pair of simulated strings $\widetilde{crs}$ with their simulation states td; the joint string is indistinguishable from the honest joint string by a two-step hybrid over clause (a) of the two components. For a well-formed composed statement $x = (\text{public inputs}, U_\alpha, \mathsf{vk})$, $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}$ samples a well-formed commitment $cc \leftarrow \mathsf{Commit}(\mathbf{0}; \rho)$, forms the sub-statements $x_{\mathsf{core}} = (\text{public inputs}, cc)$ and $x_{\mathsf{link}} = (cc, U_\alpha, \mathsf{vk})$, and outputs

$$\left(cc,\ \mathsf{S}^{\mathsf{core}}_{\mathsf{prv}}(\cdot, x_{\mathsf{core}}),\ \mathsf{S}^{\mathsf{link}}_{\mathsf{prv}}(\cdot, x_{\mathsf{link}})\right).$$

For clause (b), fix a true (x, w) and walk the hybrids: H_0 is the honest proof with $cc^* = \mathsf{Commit}((u_j); \rho)$; H_1 simulates the core setup and proof—the core statement is true, so Definition 6(b) for the core costs $\epsilon^{\mathsf{zk}}_{\mathsf{core}}$; H_2 simulates the link setup and proof—likewise $\epsilon^{\mathsf{zk}}_{\mathsf{link}}$; H_3 replaces cc^* by $\mathsf{Commit}(\mathbf{0}; \rho)$. In H_2 both sub-proofs are produced from the simulation states td and statements alone, so the opening of cc appears nowhere in the view; H_2 and H_3 differ only in the committed value, bounded by ϵ_{hide}. H_3 is the composed simulator. Summing the three steps gives the bound; completeness is immediate.

The implemented trusted-setup instantiation satisfies the interface and the compatibility requirement of assumption (iii). A universal-setup alternative is discussed afterward.

Corollary 3 (Trusted-setup instantiation: Groth16 + Kiltz–Wee). *Instantiate* $\mathsf{CP}_{\mathsf{core}}$ *with Groth16 in its committed-witness (LegoSNARK) form [12,*

9] and $\mathsf{CP}_{\mathsf{link}}$ *with the Kiltz–Wee QA-NIZK for linear subspaces [11] on the bases* (g_1, h)*. Then, in the AGM:*

(i) Knowledge soundness. *Groth16 is knowledge-sound under its q-type (q-DLOG) assumption; the Kiltz–Wee link is knowledge-sound for witness-samplable subspace matrices under discrete log, which our two single-base equality relations meet. Lemma 1 lifts these to the composition.*

(ii) Zero-knowledge. *Groth16 is perfectly zero-knowledge (its simulator uses the CRS trapdoor); the Kiltz–Wee QA-NIZK is perfectly zero-knowledge from its trapdoor. The LegoSNARK witness commitment is perfectly hiding (*$\epsilon_{\mathsf{hide}} = 0$*). By Lemma 1 the composition is zero-knowledge.*

(iii) Algebraic CRS compatibility. *Both strings are sampled independently of* τ*; the reduction generates them and keeps each exponent, so* $f_j \equiv \zeta_j$ *with* $d_{\mathsf{aux}} = 0$ *(Corollary 1).*

Hence this instantiation—the one benchmarked in Section 5—meets Theorem 1(ii)–(iii); the setup is trusted and circuit-specific. Its core proof is $2G_1 + G_2$*, and the committed witness element and one stacked Kiltz–Wee link add* $2G_1$*, so* $\pi_Z = 4G_1 + G_2$*.*

The interface is not tied to a trusted setup. A universal instantiation—PLONK over the external KZG SRS [13] for the core, and the CRS-updatable asymmetric QA-NIZK of Abdolmaleki and Slamanig [14] for the link—appears to satisfy the same three requirements, and would remove the circuit-specific ceremony. PLONK is knowledge-sound (indeed simulation-extractable in the ROM under a falsifiable q-DLOG-type assumption [15]) and zero-knowledge in the ROM—by the same work its simulator is even trapdoorless—while the QA-NIZK is knowledge-sound under BDH-KE and is perfectly zero-knowledge from its trapdoor. For τ-compatibility, PLONK reuses the KZG powers (Corollary 2), and the QA-NIZK string is algebraically independent of τ—the τ-free case of Corollary 1. We do not carry out this instantiation in full: under an updatable ceremony the reduction no longer samples the QA-NIZK string itself, so its τ-free representation must instead be recovered by extracting the update contributions—e.g. through BDH-KE—and this has to be checked against the concrete update procedure. We leave a complete universal instantiation to future work.

A.3 Algebraic extraction for the KZG layer

The next lemma is the rigorous form of the two KZG checks in Ver_{ct}. Part (a) is the subset-consistency guarantee; part (b) is evaluation soundness, stated so that it binds the *scalar* u_α once that scalar is made explicit (which, in the construction, is the role of the CPLink equation $U_\alpha = g_1^{u_\alpha}$).

Lemma 2 (KZG soundness in the AGM). *Let* $crs = (\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R})$ *with* $\tau \xleftarrow{\$} \mathbb{F}_p$*,* $h := g_1$*, and* $C_\phi = g_1^{\phi(\tau)}$ *for a known* ϕ *of degree at most* $d \leq n$*. Let* $\mathcal{A}$ *be an algebraic algorithm that outputs* $C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha \in G_1$ *(with*

representations), a set $S_R \subseteq [m]$ *of size* $R \le n$*, and a point* $\alpha \in \mathbb{F}_p \setminus S'_R$*, such that the two checks*

$$e(C_\phi / C_{S_R}, g_2) = e(\pi_{S_R}, g_2^{V_{S_R}(\tau)}), \qquad e(C_{S_R}/U_\alpha, g_2) = e(\pi_\alpha, g_2^{\tau-\alpha})$$

hold. Then there is a polynomial $\widehat{\phi}_{S_R} \in \mathbb{F}_p[X]$ *of degree at most* n *such that, except with probability* $\epsilon_{\mathsf{KZG}} \le c\,\epsilon_{n\text{-dl}}$ *for a small constant* c*,*

(a) $C_{S_R} = g_1^{\widehat{\phi}_{S_R}(\tau)}$ *and* $\widehat{\phi}_{S_R}(i) = \phi(i)$ *for all* $i \in S_R$*; and*
(b) for any scalar $y \in \mathbb{F}_p$ *known to the reduction with* $U_\alpha = g_1^y$*, we have* $\widehat{\phi}_{S_R}(\alpha) = y$*.*

Proof. Because $\mathcal{A}$ is algebraic, each output element is a known combination of the elements it received. The external G_1 inputs are the KZG powers $g_1^{\tau^i}$ for $0 \le i \le n$ and C_ϕ, itself such a combination; moreover $h = g_1$. The remaining inputs are the crs_z-elements of the CP-SNARK; by τ-algebraic compatibility (Definition 5, assumption (iii) of Theorem 1) the reduction holds, for each G_1-component Z_j of crs_z, a known polynomial f_j of degree at most $d_{\mathsf{aux}} \le n$ with $Z_j = g_1^{f_j(\tau)}$—without knowing τ—and produces a joint distribution matching the real one. Hence every G_1-element the adversary receives has a known polynomial representation in τ of degree at most n, so the AGM representations yield known polynomials A, B, P, each of degree at most n, such that

$$C_{S_R} = g_1^{A(\tau)}, \qquad \pi_{S_R} = g_1^{B(\tau)}, \qquad \pi_\alpha = g_1^{P(\tau)}.$$

Part (a). Taking the discrete logarithm of the first check to the base $e(g_1, g_2)$,

$$\phi(\tau) - A(\tau) = B(\tau) V_{S_R}(\tau).$$

Set the known polynomial

$$D(\mathsf{T}) = \phi(\mathsf{T}) - A(\mathsf{T}) - B(\mathsf{T}) V_{S_R}(\mathsf{T}),$$

whose degree is at most $n + R$. If $D \not\equiv 0$, the accepting check forces τ to be one of its roots, and we turn this into an n-DLOG solver. The reduction takes an n-DLOG challenge $\big(\{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R}\big)$ (Definition 4), embeds it verbatim as the KZG CRS, generates crs_z itself as just described, and runs $\mathcal{A}$. From the AGM representations it holds A and B as explicit polynomials, so it forms D and *factors it over* $\mathbb{F}_p$ *in the clear*; among the at most $n + R$ roots it returns the r with $g_1^r = g_1^\tau$, reading g_1^τ off the CRS. Factoring uses no group operations, and the reduction never forms a group element of degree exceeding n; so the challenge need only supply the powers already present in the KZG CRS, and n-DLOG—not $(n{+}R)$-DLOG—suffices despite $\deg D \le n + R$. Whenever $\mathcal{A}$ produces an accepting transcript with $D \not\equiv 0$ the reduction outputs τ, so that event has probability at most $\epsilon_{n\text{-dl}}$, which we fold into ϵ_{KZG}. Hence $D \equiv 0$, and $V_{S_R} \mid (\phi - A)$. Put $\widehat{\phi}_{S_R} := A$; then $C_{S_R} = g_1^{\widehat{\phi}_{S_R}(\tau)}$, and $V_{S_R}(i) = 0$ for $i \in S_R$ gives $\widehat{\phi}_{S_R}(i) = \phi(i)$.

Part (b). Let y satisfy $U_\alpha = g_1^y$ and be known to the reduction. Using $C_{S_R} = g_1^{A(\tau)}$ from part (a), the second check reads $A(\tau) - y = P(\tau)(\tau - \alpha)$. Define

$$E(\mathsf{T}) = A(\mathsf{T}) - y - P(\mathsf{T})(\mathsf{T} - \alpha), \qquad \deg E \le n + 1.$$

The same reduction rules out $E \not\equiv 0$: here $\deg E \le n + 1$, and the reduction factors E in the clear and returns its root equal to τ exactly as in part (a), again needing only the n-DLOG powers. Whenever $\mathcal{A}$ accepts with $E \not\equiv 0$ it solves n-DLOG, an event of probability at most $\epsilon_{n\text{-dl}}$, folded into ϵ_{KZG}. Thus $(\mathsf{T} - \alpha) \mid (A(\mathsf{T}) - y)$; evaluating at $\mathsf{T} = \alpha$ yields $\widehat{\phi}_{S_R}(\alpha) = A(\alpha) = y$.

Both parts bound their bad event by a single computational reduction to n-DLOG, so ϵ_{KZG} is the aggregate advantage of the constant number of such reductions and $\epsilon_{\mathsf{KZG}} \le c\,\epsilon_{n\text{-dl}}$. This is distinct from—and does not absorb—the Schwartz–Zippel sampling loss over the Fiat–Shamir challenges α and S_R, which is a statistical term (the $q_P\, n/(p - R - 2)$ and $q_S((\beta + 1)/2\beta)^R$ contributions in Theorem 1) and enters the soundness bound separately, not through this lemma. □

Remark 1. The opening check *alone* does not pin U_α to $g_1^{\widehat{\phi}_{S_R}(\alpha)}$: an algebraic prover may output $U_\alpha = g_1^{\widehat{\phi}_{S_R}(\tau) + (\tau - \alpha) r(\tau)}$ for any r and still open it. Part (b) becomes meaningful only once an *explicit* scalar y with $U_\alpha = g_1^y$ is available. In the construction this scalar is delivered by the CPLink equation $U_\alpha = g_1^{u_\alpha}$, which is therefore also essential to soundness, not only a zero-knowledge device.

A.4 Correctness

Lemma 3 (Correctness). *The construction of Figure 2 satisfies Definition 1.*

Proof. Fix ϕ of degree at most d and run Enc honestly. The key check holds since $\mathsf{vk} = h^{\mathsf{sk}}$ by construction. The blinded polynomial $\widetilde{\phi}_{S_R} = \phi_{S_R} + (t_1 + t_2 X) V_{S_R}$ has degree at most $R + 1$ (as $\deg \phi_{S_R} \le R - 1$ and $\deg V_{S_R} = R$) and, since V_{S_R} vanishes on S_R, satisfies $\widetilde{\phi}_{S_R}(i) = \phi_{S_R}(i) = \phi(i)$ for $i \in S_R$. Thus $\phi - \widetilde{\phi}_{S_R}$ vanishes on S_R, so $Q := (\phi - \widetilde{\phi}_{S_R})/V_{S_R}$ is a polynomial (degree $\le \max(d, R+1) - R \le n$) and $\pi_{S_R} = g_1^{Q(\tau)}$ gives the subset identity $\phi(\tau) - \widetilde{\phi}_{S_R}(\tau) = Q(\tau) V_{S_R}(\tau)$, i.e. $e(C_\phi / C_{S_R}, g_2) = e(\pi_{S_R}, g_2^{V_{S_R}(\tau)})$; likewise, with $u_\alpha = \widetilde{\phi}_{S_R}(\alpha)$ the quotient $q_\alpha = (\widetilde{\phi}_{S_R} - u_\alpha)/(X - \alpha)$ is a polynomial and $\pi_\alpha = g_1^{q_\alpha(\tau)}$ gives the opening identity $e(C_{S_R}/U_\alpha, g_2) = e(\pi_\alpha, g_2^{\tau - \alpha})$.

For the CP-SNARK relation set $x_i = \phi(i)$ $(i \in S_R)$ and $x_i = \widetilde{\phi}_{S_R}(i)$ $(i \in \{-1, -2\})$; then $ct_i = x_i + H(\mathsf{sk}, i)$ on S'_R, and as $\deg \widetilde{\phi}_{S_R} \le R + 1 = |S'_R| - 1$, Lagrange interpolation over the $R + 2$ points of S'_R is exact, giving $\sum_{i \in S'_R} x_i L_{i, S'_R}(\alpha) = \widetilde{\phi}_{S_R}(\alpha) = u_\alpha$ (using $\widetilde{\phi}_{S_R}(i) = \phi(i) = x_i$ on S_R). With $U_\alpha = g_1^{u_\alpha}$ and $\mathsf{vk} = h^{\mathsf{sk}}$ the witness $(u_\alpha, \mathsf{sk}, \{x_i\})$ satisfies R_{mask} and both CPLink relations, so π_Z verifies by completeness and all three checks of Ver_{ct} accept. Finally, honest unmasking gives $\phi'(i) = ct_i - H(\mathsf{sk}, i) = \phi(i)$ on $[m]$, the exact codeword, so RS.Det reports no error and Dec outputs $F(\phi)$. □

A.5 Soundness

Lemma 4 (Soundness). *Under the assumptions of Theorem 1, every PPT algebraic adversary with grinding budgets (q_S, q_P) has soundness advantage (Definition 2) at most*

$$\epsilon_{\mathsf{KZG}} + \epsilon^{\mathsf{snd}}_{\mathsf{CP}} + q_P \frac{n}{p-R-2} + q_S \left(\frac{\beta+1}{2\beta}\right)^R.$$

Proof. Let $\mathcal{A}$ play the game of Definition 2, outputting $(\mathsf{sk}^*, \mathsf{vk}, ct, \pi)$ with $\pi = (C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha, \pi_Z)$. We bound the probability of the event Win that $\mathsf{Ver}_{ct} = 1$, $\mathsf{Ver}_{key}(\mathsf{vk}, \mathsf{sk}^*) = 1$, and $y := \mathsf{Dec}(F, \mathsf{sk}^*, ct) \neq F(\phi)$. The verifier recomputes S_R and $\alpha \notin S'_R$ from the fixed transcript.

Step S1 (CP-SNARK extraction). Running the knowledge extractors of the core CP-SNARK and of the CPLink instance yields a witness $(\mathsf{sk}, u_\alpha, \{x_i\}_{i\in S'_R})$; except with probability $\epsilon^{\mathsf{snd}}_{\mathsf{CP}} := \epsilon^{\mathsf{snd}}_{\mathsf{ccSNARK}} + \epsilon^{\mathsf{snd}}_{\mathsf{CPLink}} + \epsilon_{\mathsf{bind}}$ it satisfies R_{mask} and the two linking equations, i.e.

$$ct_i = x_i + H(\mathsf{sk}, i)\ (i \in S'_R), \quad u_\alpha = \sum_{i\in S'_R} x_i\, L_{i,S'_R}(\alpha), \quad U_\alpha = g_1^{u_\alpha}, \quad \mathsf{vk} = h^{\mathsf{sk}}. \tag{1}$$

Step S2 (the extracted key is the decryption key). By construction (Gen) h is a nontrivial element of the prime-order group G_1, hence a generator, so $\log_h(\cdot)$ is a bijection. From $\mathsf{Ver}_{key}(\mathsf{vk}, \mathsf{sk}^*) = 1$ we have $\mathsf{vk} = h^{\mathsf{sk}^*}$, and from (1) $\mathsf{vk} = h^{\mathsf{sk}}$; therefore $\mathsf{sk} = \mathsf{sk}^*$. Thus the key extracted from π_Z is exactly the key that Dec uses to unmask, and $H(\mathsf{sk}, \cdot) = H(\mathsf{sk}^*, \cdot)$.

Step S3 (KZG extraction). Apply Lemma 2. Except with probability ϵ_{KZG}, there is $\widehat{\phi}_{S_R}$ of degree at most n with $\widehat{\phi}_{S_R}(i) = \phi(i)$ for $i \in S_R$ (part a); and, taking the explicit scalar $y := u_\alpha$ from (1) (for which $U_\alpha = g_1^{u_\alpha}$ holds, so the hypothesis of part b is met), $\widehat{\phi}_{S_R}(\alpha) = u_\alpha$ (part b).

Step S4 (interpolation and Schwartz–Zippel). Let $\psi(X) := \sum_{i\in S'_R} x_i\, L_{i,S'_R}(X)$, so $\deg \psi \le R+1$ and, by (1), $\psi(\alpha) = u_\alpha$. Combining with Step S3, $\psi(\alpha) = \widehat{\phi}_{S_R}(\alpha)$. Both ψ and $\widehat{\phi}_{S_R}$ are fixed before α is derived: although the proof π_Z from which $\{x_i\}$ are extracted is produced after α, the masking equation in (1) pins each $x_i = ct_i - H(\mathsf{sk}, i)$ to a value determined by the pre-α transcript and sk, so $\psi = \sum_i x_i L_{i,S'_R}$ does not depend on α; and $\widehat{\phi}_{S_R}$ is a function of C_{S_R}, which is an input to $\alpha = \mathsf{DerivePoint}_H(\ldots, C_{S_R}, \pi_{S_R}, S_R)$ and hence also fixed beforehand. Suppose $\psi \neq \widehat{\phi}_{S_R}$; then $\psi - \widehat{\phi}_{S_R}$ is a nonzero polynomial of degree at most n, with at most n roots in $\mathbb{F}_p$. Each invocation of $\mathsf{DerivePoint}_H$ returns a value uniform on $\mathbb{F}_p \setminus S'_R$ (size $p-R-2$), so a single derivation lands on a root with probability at most $n/(p-R-2)$. Since $\mathcal{A}$ obtains at most q_P candidate point derivations, a union bound gives $\Pr[\psi(\alpha) = \widehat{\phi}_{S_R}(\alpha) \wedge \psi \neq \widehat{\phi}_{S_R}] \le q_P n/(p-R-2)$. Except with this probability, $\psi = \widehat{\phi}_{S_R}$ as polynomials, whence for $i \in S_R$

$$x_i = \psi(i) = \widehat{\phi}_{S_R}(i) = \phi(i), \qquad \text{i.e.} \qquad ct_i - H(\mathsf{sk}, i) = \phi(i).$$

Step S5 (sampling to decoding). Let $c_i := ct_i - H(\mathsf{sk}, i)$ for $i \in [m]$; this is precisely the word Dec unmasks (Step S2). Let $C := \{i \in [m] : c_i \neq \phi(i)\}$ be its set of disagreements with the codeword $(\phi(1), \ldots, \phi(m))$. Step S4 shows $S_R \cap C = \emptyset$.

- If $|C| \leq t = \lfloor (m-\ell)/2 \rfloor$, then c lies within the decoding radius; by RS correctness RS.Det and (if needed) RS.Dec reconstruct ϕ on $[\ell]$, so $y = F(\phi)$ and Win fails.
- If $|C| > t$, a direct counting bound applies over the $m = \lceil \beta\ell \rceil$ codeword positions $[m]$ with correction radius $t = \lfloor (m-\ell)/2 \rfloor$ and corruption set C. Since $|C| \geq t+1$, the fraction of positions avoiding C is at most

$$\frac{m-|C|}{m} \leq \frac{m-t-1}{m} \leq \frac{m+\ell}{2m} \leq \frac{\beta+1}{2\beta},$$

  where $t = \lfloor (m-\ell)/2 \rfloor$ and $m = \lceil \beta\ell \rceil$. Thus a uniform R-subset of $[m]$ avoids C with probability at most $((\beta+1)/2\beta)^R$ (sampling without replacement only lowers this). In the ROM, $S_R = \mathsf{DeriveSubset}_H(\ldots)$ is computed from random-oracle outputs at the subset-derivation inputs $(C_\phi, \mathsf{vk}, ct, m, R, j)$, which—under the injective, domain-separated input encoding of H—are disjoint from the masking inputs (sk, i) whose outputs determine c and hence C; thus, conditioned on any fixed transcript, S_R is a uniform R-subset independent of C. An adversary fixes at most q_S candidate transcripts, so $\Pr[S_R \cap C = \emptyset \wedge |C| > t] \leq q_S((\beta+1)/2\beta)^R$.

Hence the event "Win with $|C| > t$" (an accepting transcript whose proven $S_R \cap C = \emptyset$ coincides with an undetected large corruption) has probability at most $q_S((\beta+1)/2\beta)^R$, and the event "Win with $|C| \leq t$" is impossible.

Collecting. Across S1–S5,

$$\Pr[\mathsf{Win}] \leq \epsilon_{\mathsf{CP}}^{\mathsf{snd}} + \epsilon_{\mathsf{KZG}} + q_P \frac{n}{p-R-2} + q_S \left(\frac{\beta+1}{2\beta}\right)^R,$$

which is the claimed bound. □

A.6 Optional extension to an untrusted committer

Theorem 1 fixes an honestly generated degree-d commitment (Definition 2), as in prior VECK schemes [5, 7]. If the same party both commits and runs Enc, it registers C_ϕ and a degree proof once, before any sale-specific transcript is formed. The same proof can be reused for later sales of that asset under the same public parameters and declared degree bound. Each buyer still verifies the proof unless it relies on an authenticated registration result; proof reuse alone does not eliminate this verification cost. The following optional check enforces the degree bound using one additional G_2 CRS element, one G_1 proof element per registered file, and one pairing equation. Its security is conditional on the following assumption, not on the core theorem's n-DLOG assumption.

Definition 7 (Shifted (n, d, R)-DLOG). *For $\tau \xleftarrow{\$} \mathbb{F}_p$, given*

$$\left(\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R}, g_2^{\tau^{n-d}}\right),$$

output τ. We denote the maximum success probability by $\epsilon_{\mathsf{shift}}$. This is a separate assumption. When $n - d > R$, it gives the adversary a shifted G_2 power not present in Definition 4; when $n - d \leq R$, that power is already included in the core challenge. We make no reduction from the former case to the core n-DLOG assumption.

Corollary 4 (KZG soundness under the augmented CRS). *Lemma 2 holds verbatim when the CRS is augmented with $g_2^{\tau^{n-d}}$ and the adversary additionally receives that element together with $\pi_{\deg} = g_1^{\tau^{n-d}\phi(\tau)}$, with ϵ_{KZG} replaced by an advantage $\epsilon_{\mathsf{KZG}}^{\mathsf{aug}} \leq c\,\epsilon_{\mathsf{shift}}$ for the same constant c.*

Proof. The reduction of Lemma 2 consumes its challenge only by embedding it verbatim as the CRS; it factors D and E in the clear and reads g_1^{τ} off the G_1 powers. Replace the n-DLOG challenge by a shifted (n, d, R)-DLOG challenge $\left(\mathcal{G}, \{g_1^{\tau^i}\}_{i=0}^{n}, \{g_2^{\tau^i}\}_{i=0}^{R}, g_2^{\tau^{n-d}}\right)$ (Definition 7), which is exactly the augmented CRS; the embedding step is unchanged, and a recovered τ now solves the shifted assumption.

Two observations complete the argument. First, the added element lies in G_2, while the AGM bookkeeping tracks representations of the G_1 elements the adversary receives and outputs; it therefore adds no new representation and leaves $\deg A, \deg B, \deg P \leq n$ intact. Second, $\pi_{\deg}$ is a G_1 element, but $\tau^{n-d}\phi(\tau)$ has degree at most n because $\deg \phi \leq d$, so the reduction can form it from the G_1 powers it already holds and its known representation keeps the degree bound. Every other step of Lemma 2 is untouched.

Proposition 1 (Degree enforcement for an untrusted committer). *Augment the CRS with $g_2^{\tau^{n-d}}$. Before the exchange, the committer publishes $\pi_{\deg} := g_1^{\tau^{n-d}\phi(\tau)}$ and the buyer checks*

$$e\left(C_\phi,\, g_2^{\tau^{n-d}}\right) = e(\pi_{\deg},\, g_2)\,.$$

Under shifted (n, d, R)-DLOG (Definition 7), any C_ϕ accepted by this check is an algebraic commitment to a polynomial of degree at most d, except with probability $\epsilon_{\mathsf{shift}}$. Consequently the soundness bound of Theorem 1 extends to an adversarially chosen C_ϕ that passes the check, with the additional term $\epsilon_{\mathsf{shift}}$ and with ϵ_{KZG} read as the augmented advantage $\epsilon_{\mathsf{KZG}}^{\mathsf{aug}}$ of Corollary 4.

Proof (Proof sketch). An honest committer can form $\pi_{\deg}$ only when $\tau^{n-d}\phi(\tau)$ has degree at most n, i.e. $\deg \phi \leq d$, since the CRS exposes g_1-powers only up to τ^n. For an algebraic committer, extract $C_\phi = g_1^{a(\tau)}$ and $\pi_{\deg} = g_1^{b(\tau)}$ with known polynomials a, b of degree at most n. The accepted pairing check means $a(\tau)\,\tau^{n-d} = b(\tau)$. If a had a monomial of degree above d, then $a(X)X^{n-d}$

would have a monomial of degree above n, which the degree-$\leq n$ polynomial $b(X)$ cannot cancel, so $D(X) := a(X)X^{n-d} - b(X) \not\equiv 0$; yet $D(\tau) = 0$, and recovering the roots of the known nonzero D yields τ, contradicting shifted (n, d, R)-DLOG. Hence $\deg a \leq d$ and the committed word lies in the code the decoder assumes. The remaining soundness argument is unchanged, as it already treats $(\mathsf{vk}, \mathsf{sk}, ct, \pi)$ as fully adversarial, its KZG step now invoking Corollary 4 so that the adversary is granted the augmented CRS throughout.

A.7 Zero-knowledge

Lemma 5 (Computational zero-knowledge). *Under the assumptions of Theorem 1, the construction satisfies Definition 3 with advantage at most* $\epsilon^{\mathsf{zk}}_{\mathsf{CP}} + \epsilon^{G_1}_{\mathsf{DL}}$.

Proof. The simulator. As is standard for non-interactive zero-knowledge, we prove the property in the model where the simulator also simulates the SNARK setup. Let $(\mathsf{S}^{\mathsf{CP}}_{\mathsf{kg}}, \mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}})$ be the simulator guaranteed by the zero-knowledge of the CP-SNARK (Theorem 1(ii)): $\mathsf{S}^{\mathsf{CP}}_{\mathsf{kg}}$ produces a reference string crs_z together with a simulation state $\mathsf{st}_{\mathsf{CP}}$, and $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}(\mathsf{st}_{\mathsf{CP}}, \cdot)$ produces, on true statements, proofs indistinguishable from honest ones. Below we also run $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}$ on statements that need not lie in the relation; there we use nothing about its output—it enters the argument only as a fixed randomized function of $\mathsf{st}_{\mathsf{CP}}$ and the public statement. Nothing below uses the internal form of $\mathsf{st}_{\mathsf{CP}}$. $\mathsf{Sim}(\widetilde{pp}, \mathsf{st}_{\mathsf{CP}}, F, C_\phi)$ receives the simulated CP reference string and state produced in Definition 3 and controls the random oracle H; the external KZG crs (hence τ) is given and is *not* simulated. Sim proceeds as follows. (1) Sample $\mathsf{sk} \xleftarrow{\$} \mathbb{F}_p$ and set $\mathsf{vk} = h^{\mathsf{sk}}$. (2) Sample $ct_i \xleftarrow{\$} \mathbb{F}_p$ for all $i \in [m] \cup \{-1, -2\}$. (3) Derive $S_R = \mathsf{DeriveSubset}_H(C_\phi, \mathsf{vk}, \{ct_i\}_{i\in[m]}, m, R)$ (which depends only on the ciphertext, not on C_{S_R}); sample $y_1 \xleftarrow{\$} \mathbb{F}_p$ and set $\pi_{S_R} = g_1^{y_1}$, $C_{S_R} = C_\phi \cdot g_1^{-y_1 V_{S_R}(\tau)}$, where $g_1^{V_{S_R}(\tau)} = \prod_i (g_1^{\tau^i})^{v_i}$ (for $V_{S_R}(X) = \sum_i v_i X^i$) is a public combination of CRS elements, well-defined since $\deg V_{S_R} = R \leq n$; then derive $\alpha = \mathsf{DerivePoint}_H(\ldots, C_{S_R}, \pi_{S_R}, S_R)$. (4) Sample $y_2 \xleftarrow{\$} \mathbb{F}_p$ and set $\pi_\alpha = g_1^{y_2}, U_\alpha = C_{S_R} \cdot g_1^{-y_2(\tau-\alpha)}$ (again a public CRS combination). (5) Set $\pi_Z \leftarrow \mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}(\mathsf{st}_{\mathsf{CP}}, \mathsf{stmt})$ for the public statement $(h, S'_R, \{ct_i\}_{i\in S'_R}, \{L_{i,S'_R}(\alpha)\}_{i\in S'_R}, U_\alpha, \mathsf{vk})$. (6) Output $(\mathsf{vk}, ct, (C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha, \pi_Z))$. Given the sampled ciphertext and the derived challenges, the KZG components use only C_ϕ, the public KZG CRS, and fresh randomness. The CP proof additionally uses $\mathsf{st}_{\mathsf{CP}}$ from the simulated CP setup. The full simulator never uses ϕ or the external KZG trapdoor τ.

We bound the distinguishing advantage through hybrids $H_0, \ldots, H_4$, with H_0 the real view (including pp) and H_4 the simulated view (including $\widetilde{pp}$). We use $\alpha \notin S'_R$, hence $V_{S_R}(\alpha) \neq 0$, as guaranteed by construction. No conditioning on $\tau \neq \alpha$ is needed: the opening-transcript argument below also covers $\tau = \alpha$.

$H_0 \to H_1$ *(simulate* π_Z*).* Replace the honest SNARK reference string crs_z by a simulated one (produced by $\mathsf{S}^{\mathsf{CP}}_{\mathsf{kg}}$, with state $\mathsf{st}_{\mathsf{CP}}$) and π_Z by $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}$ on the same statement; zero-knowledge of the CP-SNARK bounds both changes at once. At

H_0 the statement is true (it has the honest witness), so zero-knowledge applies directly and $|\Pr[\mathcal{D}(H_1)] - \Pr[\mathcal{D}(H_0)]| \le \epsilon^{\mathsf{zk}}_{\mathsf{CP}}$.

$H_1 \to H_2$ *(uniform ciphertext).* Replace every ct_i ($i \in [m] \cup \{-1,-2\}$) by a uniform field element and derive S_R from $\{ct_i\}_{i\in[m]}$ and α from the full resulting transcript; the KZG components $C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha$ remain the honest ones (computed from $\widetilde{\phi}_{S_R}$ and the blinders t_1, t_2 for the new S_R). Let Q be the event that no H-query with first coordinate sk (at any second input) is made by $\mathcal{D}$ or by $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}$—the latter matters only for a simulator that queries H at all; the Groth16 and Kiltz–Wee simulators make no such queries. Conditioned on Q, the pads $H(\mathsf{sk}, i)$ for $i \in [m] \cup \{-1,-2\}$ are uniform and independent in $\mathcal{D}$'s view—$\mathcal{D}$ sees only the ciphertext entries, not the simulator's own oracle evaluations—so the real $ct_i = \phi(i) + H(\mathsf{sk}, i)$ (and $ct_i = \widetilde{\phi}_{S_R}(i) + H(\mathsf{sk}, i)$ for $i \in \{-1,-2\}$) are uniform, identical to H_2; and on both sides $\pi_Z = \mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}(\mathsf{st}_{\mathsf{CP}}, \mathsf{stmt})$ with fresh coins, a fixed randomized function of view components whose joint law has just been matched—that the statement in H_2 need no longer lie in the relation is immaterial; thus $H_2 \equiv H_1$ given Q. We bound $\Pr[\neg Q]$ by a discrete-log reduction: on challenge (g_1, T), set $h := g_1$, $\mathsf{vk} := T$, and run the H_2 experiment, which needs neither sk nor τ; whenever any H-query in the experiment—by $\mathcal{D}$ or by $\mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}$—has the form $H(s', \cdot)$ with $g_1^{s'} = T$, output $s' = \log_{g_1} T$. Hence $\Pr[\neg Q] \le \epsilon^{G_1}_{\mathsf{DL}}$. *Crucially, after this step the ciphertext is independent of the blinders* (t_1, t_2)*:* the auxiliary entries ct_{-1}, ct_{-2}, which in H_0, H_1 carry $\widetilde{\phi}_{S_R}(-1), \widetilde{\phi}_{S_R}(-2)$ and are therefore correlated with u_α through (t_1, t_2), are now uniform; the only view components that still depend on (t_1, t_2) are the KZG transcripts $C_{S_R}, \pi_{S_R}, U_\alpha, \pi_\alpha$. This independence enables the exact distributional arguments in the next two transitions.

$H_2 \to H_3$ *(rewrite the opening transcript).* Replace (U_α, π_α) by $\pi_\alpha = g_1^{y_2}$ (y_2 uniform) and $U_\alpha = C_{S_R}\, g_1^{-y_2(\tau-\alpha)}$, and recompute $\pi_Z = \mathsf{S}^{\mathsf{CP}}_{\mathsf{prv}}(\mathsf{st}_{\mathsf{CP}}, \cdot)$ on the updated statement. We show $H_3 \equiv H_2$ by proving that the honest opening-proof exponent is uniform, including when $\tau = \alpha$.

Write $f := \phi_{S_R}$, $V := V_{S_R}$, and $a := t_1 + t_2\tau$. Since f agrees with ϕ on S_R, $g := (\phi - f)/V$ is a polynomial. For fixed public parameters and ciphertext, the subset pair depends on the blinders only through a:

$$C_{S_R} = g_1^{f(\tau)+aV(\tau)}, \qquad \pi_{S_R} = g_1^{g(\tau)-a}.$$

The map $(t_1, t_2) \mapsto (a, t_2)$ is invertible, so t_2 is uniform given a. By H_2, the ciphertext is independent of both blinders. The challenge α is derived from that ciphertext and the subset pair, which contain no further information about t_2. Thus t_2 remains uniform conditioned on this pre-opening transcript and α.

For a polynomial r, define the polynomial quotient $D_\alpha r(X) := (r(X) - r(\alpha))/(X - \alpha)$, with $D_\alpha r(\alpha) = r'(\alpha)$. The honest opening quotient is $Q_\alpha := D_\alpha \widetilde{\phi}_{S_R}$. Expanding $\widetilde{\phi}_{S_R} = f + (t_1 + t_2 X)V$ gives

$$Q_\alpha(\tau) = D_\alpha f(\tau) + a D_\alpha V(\tau) + t_2 V(\alpha). \tag{2}$$

This polynomial identity holds also at $\tau = \alpha$. Since $V(\alpha) \neq 0$, the last term makes $Q_\alpha(\tau)$ uniform conditioned on the pre-opening transcript and challenge.

Consequently, $\pi_\alpha = g_1^{Q_\alpha(\tau)}$ is uniform, and the polynomial opening identity determines

$$U_\alpha = C_{S_R}\,(g_1^{\tau}/g_1^{\alpha})^{-Q_\alpha(\tau)}.$$

This is exactly the conditional distribution sampled in H_3; when $\tau = \alpha$, both hybrids simply have $U_\alpha = C_{S_R}$. Finally, π_Z is the same fixed randomized function of the updated public statement and $\mathsf{st}_{\mathsf{CP}}$ in both hybrids. Hence the entire joint views agree: $H_3 \equiv H_2$.

$H_3 \to H_4$ *(rewrite the subset transcript).* Replace (C_{S_R}, π_{S_R}) by $\pi_{S_R} = g_1^{y_1}$ (y_1 uniform) and $C_{S_R} = C_\phi\, g_1^{-y_1 V_{S_R}(\tau)}$; since $\alpha = \mathsf{DerivePoint}_H(\ldots, C_{S_R}, \pi_{S_R}, S_R)$ takes these as inputs, re-derive α, refresh $U_\alpha = C_{S_R}\, g_1^{-y_2(\tau-\alpha)}$, and recompute π_Z likewise. It suffices to show the joint law of (C_{S_R}, π_{S_R}) is preserved, since then the induced laws of α, of the refreshed U_α, and of the rest of the view—again including π_Z—are unchanged. If $\tau \notin S_R$ (so $V_{S_R}(\tau) \neq 0$), the real pair has C_{S_R} uniform with $\pi_{S_R} = (C_\phi/C_{S_R})^{1/V_{S_R}(\tau)}$ forced, while H_4 has π_{S_R} uniform with C_{S_R} forced by the same relation; the bijection $C_{S_R} \leftrightarrow \pi_{S_R}$ (using $V_{S_R}(\tau) \neq 0$) carries one law to the other, so $H_4 \equiv H_3$. In the residual case $\tau \in S_R$ we have $V_{S_R}(\tau) = 0$: both hybrids fix $C_{S_R} = C_\phi$ (in H_3, $C_{S_R} = g_1^{\phi_{S_R}(\tau)} = g_1^{\phi(\tau)} = C_\phi$ because ϕ_{S_R} interpolates ϕ at $\tau \in S_R$; in H_4, $C_{S_R} = C_\phi g_1^0 = C_\phi$). For π_{S_R}: writing $\phi - \phi_{S_R} = V_{S_R}\, g$ (a polynomial identity, as $\phi \equiv \phi_{S_R}$ on S_R), the honest quotient is $(\phi - \widetilde{\phi}_{S_R})/V_{S_R} = g - (t_1 + t_2 X)$, so $\pi_{S_R} = g_1^{g(\tau)-a}$ with $a = t_1 + t_2\tau$ uniform and unconstrained (because $C_{S_R} = C_\phi$ regardless of a, and by H_2 the ciphertext no longer constrains (t_1, t_2)); in H_4, $\pi_{S_R} = g_1^{y_1}$ is uniform. So $H_4 \equiv H_3$ in either case. In H_4, the KZG components are computed from C_ϕ, the public KZG CRS, S_R, α, and fresh y_1, y_2, while π_Z uses the CP simulation state. Together with the sampled key and uniform ciphertext, this is exactly Sim.

Total. Summing the transition bounds,

$$\epsilon_{\mathsf{zk}} \le \epsilon_{\mathsf{CP}}^{\mathsf{zk}} + \underbrace{\epsilon_{\mathsf{DL}}^{G_1}}_{H_1 \to H_2:\, \Pr[\neg Q]} + \underbrace{0}_{H_2 \to H_3} + \underbrace{0}_{H_3 \to H_4} = \epsilon_{\mathsf{CP}}^{\mathsf{zk}} + \epsilon_{\mathsf{DL}}^{G_1}.$$

□

A.8 Proof of Theorem 1

Proof (Proof of Theorem 1). Correctness is Lemma 3. Soundness is Lemma 4, whose bound $\epsilon_{\mathsf{KZG}} + \epsilon_{\mathsf{CP}}^{\mathsf{snd}} + q_P n/(p - R - 2) + q_S((\beta+1)/2\beta)^R$ is exactly the soundness advantage stated in the theorem; under the parameter choice $q_S((\beta+1)/2\beta)^R \le 2^{-\lambda}$ the sampling term is negligible in λ. Zero-knowledge is Lemma 5. All three hold in the AGM and ROM under the stated assumptions, so the construction is a secure VECK in the sense of Definitions 1–3. □

B Extended Evaluation and Implementation Details

This appendix gives the full measurements and implementation details supporting Section 5. The main text uses the direct sampled-ElGamal baseline; the full-file ElGamal schemes here provide additional context.

B.1 Implementation and Parameter Details

We implemented KZG in Rust/arkworks and the SNARK in Go/gnark, following the stack of $\text{VECK}^{\star}_{\text{EL}}$ [7]. We implemented the linking layer for $U_\alpha = g_1^{u_\alpha}$ and $\mathsf{vk} = h^{\mathsf{sk}}$ as a single stacked Kiltz–Wee instance [11]. Experiments ran on an **Apple M3 Pro** with an **11-core CPU** and **18 GB** of unified memory, running **macOS 26.6.2**, with **Groth16** [12] and **Poseidon2** in the circuit [16]. The implementations use eleven worker threads, arkworks 0.4.x, Go 1.25.7, gnark 0.14.0, and gnark-crypto 0.19.0.

Timing methodology. For the direct baseline and ours, each stage time is the median of five wall-clock runs with no discarded warm-up. The sender aggregate sums these medians for RS encoding, masking, transcript hashing and sampling, polynomial and native public-key work, and SNARK/linking proofs; it is not an independent outer-timer measurement. All sender comparisons use the same interpolation optimizations. These timings describe the tested hardware and software configuration.

Parameters and encoding. We fix $\lambda = 128$ and $q_S, q_P \le 2^{32}$. The sample sizes R are 386, 609, 1053, and 2384. These correspond to nominal redundancy labels $\beta \in \{2, 1.5, 1.25, 1.1\}$. The implementation uses R-indexed values approximately 2.0000378, 1.4997596, 1.2500311, and 1.1000224, respectively, satisfying $q_S((\beta+1)/2\beta)^R \le 2^{-128}$. For our BLS12-381 implementation and the largest reported bounds $n \le 2^{20}$, $R \le 2384$, and $q_P \le 2^{32}$, the point-challenge soundness term satisfies $q_P n/(p - R - 2) < 2^{-200}$, below this sampling term. Throughout the evaluation, the displayed β values are rounded labels; encoding and communication use the actual $m = \lceil \beta \ell \rceil$.

For fixed (λ, R, q_S), the sampling target is equivalent to

$$\beta \ge \beta_{\min}(\lambda, R, q_S) := \frac{1}{2^{1-(\lambda+\log_2 q_S)/R} - 1},$$

provided $R > \lambda + \log_2 q_S$. Otherwise no finite $\beta > 1$ reaches the target, since the per-sample failure bound approaches $1/2$ as β grows.

Admissible file sizes. The construction requires $m \ge R$. At nominal $\beta = 1.1$, $\ell = 2^{10}, 2^{11}$ violate this condition, so the series starts at 2^{12}. The reported grid also restricts sampling to $R < \ell$. Thus it omits $\ell = 2^{10}, \beta = 1.25$, although that point satisfies $m \ge R$. This additional restriction selects a benchmark regime and is not a requirement of Theorem 1.

Table B.1. Directly measured SNARK resources at each nominal redundancy.

scheme (curve)	metric (sample size)	$\beta = 2$ (R=386)	$\beta = 1.5$ (R=609)	$\beta = 1.25$ (R=1053)	$\beta = 1.1$ (R=2384)
$\text{VECK}^{\star}_{\text{EL}}$ (BW6-761)	constraints	1,551,797	2,447,142	4,229,802	9,573,767
	setup time	303.1 s	512.1 s	942.3 s	2350 s
	proving time	16.68 s	28.72 s	61.61 s	179.1 s
	verification time	18.81 ms	20.26 ms	33.06 ms	108.2 ms
	CRS size	653.3 MiB	1.09 GiB	1.98 GiB	4.28 GiB
Ours (BW6-761)	constraints	77,602	122,203	211,004	477,208
	setup time	18.29 s	25.26 s	44.86 s	98.46 s
	proving time	1.882 s	2.506 s	3.911 s	7.172 s
	verification time	13.81 ms	17.86 ms	21.88 ms	39.59 ms
	CRS size	38.33 MiB	53.46 MiB	95.60 MiB	209.9 MiB
Ours (BLS12-381)	constraints	77,602	122,203	211,004	477,208
	setup time	5.21 s	7.45 s	13.13 s	28.98 s
	proving time	477.6 ms	667.0 ms	1.135 s	2.346 s
	verification time	2.99 ms	4.18 ms	5.98 ms	10.63 ms
	CRS size	22.81 MiB	32.47 MiB	57.71 MiB	127.4 MiB

B.2 Complete SNARK Resources

Setup and CRS costs. At the same $\beta = 1.1$ setting, holding BW6-761 fixed reduces setup from 2350 to 98.46 s and the circuit-specific CRS from 4.28 GiB to 209.9 MiB. Our BLS12-381 configuration further reduces these to 28.98 s and 127.4 MiB. These setup times are not charged to each exchange. Each configuration also needs a universal KZG CRS; for ours on BLS12-381, its compressed size is $48(n+1) + 96(R+1)$ bytes, about 48.2 MiB at $n = 2^{20}, R = 2384$, in addition to the circuit-specific material.

B.3 Sender Breakdown and Polynomial Work

Table B.2 expands the reference-case totals in Table 1. Its $\text{VECK}^{+}_{\text{EL}}$ column includes extrapolated encryption; the direct-baseline and ours columns are measured. Table B.3 gives the full polynomial and transcript sweep.

Polynomial and linking work. Both our configurations and the code-based baselines use the same interpolation optimizations: parallel polynomial work with rayon and fewer field inversions. For ours on BLS12-381 at the reference setting, interpolation and commitment fall from 1146.2 to 102.2 ms. All sender comparisons use the optimized routines; the SNARK circuits and their measurements are unchanged.

The subset proof uses one batched polynomial division, not a separate opening per sample. Across the redundancy settings, Table B.3 shows combined subset and fingerprint opening times of 147.2–220.4 ms, while interpolation and commitment grow from 5.3 to 102.2 ms. Thus polynomial work also remains sample-dependent. The stacked CP-Link proof takes 0.206 ms on BLS12-381

Table B.2. Sender-time breakdown (ms) at $\ell = 2^{16}$, $\beta = 1.1$, $R = 2384$; $\mathrm{VECK}^{+}_{\mathrm{EL}}$ encryption is extrapolated.

Stage	$\mathrm{VECK}^{+}_{\mathrm{EL}}$	$\mathrm{VECK}^{\star}_{\mathrm{EL}}$	Ours	Ours
KZG curve	BLS12-381	BLS12-377	BW6-761	BLS12-381
SNARK curve	—	BW6-761	BW6-761	BLS12-381
RS encoding	9.4	8.0	14.8	8.1
Masking/encryption	23,203.2	67.8	123.3	66.6
Transcript hashing and sampling	5.5	5.4	8.4	5.3
Subset interpolation and commitment	102.9	104.3	193.4	102.2
Native sampled public-key operations	28,662.8	103.5	—	—
KZG proof generation	314.2	221.9	605.6	220.4
SNARK proof generation	—	179,096.5	7,171.7	2,345.7
Linking proof generation	—	—	0.902	0.206
Aggregate sender computation	52,297.9	179,607.4	8,118.2	2,748.5

Table B.3. Polynomial and transcript costs (ms) of our BLS12-381 implementation at $\ell = 2^{16}$.

Stage	$\beta = 2$	$\beta = 1.5$	$\beta = 1.25$	$\beta = 1.1$
Interpolation and commitment	5.3	10.0	23.8	102.2
KZG proof generation	147.2	162.9	214.6	220.4
Transcript hashing and sampling	9.5	7.1	6.0	5.3

Table B.4. Core cryptographic verification aggregates (ms) at $\ell = 2^{16}$.

Scheme (KZG / SNARK)	$\beta = 2$	$\beta = 1.5$	$\beta = 1.25$	$\beta = 1.1$
$\mathrm{VECK}^{+}_{\mathrm{EL}}$ BLS12-381 / —	1747	2766	4733	10670
$\mathrm{VECK}^{\star}_{\mathrm{EL}}$ BLS12-377 / BW6-761	26.3	29.0	44.9	127.7
Ours BW6-761 / BW6-761	35.1	41.0	48.7	77.9
Ours BLS12-381 / BLS12-381	10.8	13.2	17.7	28.6

and 0.902 ms on BW6-761 at $\beta = 1.1$, already included in the sender totals. These external operations do not replace the removed circuit work with a comparable new bottleneck in the measured configurations.

Whole-transcript hashing and sample selection cost 5.3–9.5 ms, or 0.2–1.3% of our BLS12-381 sender aggregate on this grid. At $\beta = 1.1$, reusable RS encoding accounts for 8.1 ms; the remaining 2.740 s is per-exchange computation.

B.4 Complete Core Verification Results

Table B.4 compares verification time after the ciphertext transcript and sampled entries are fixed. The original VECK [5] is omitted because it samples no positions and its verification is linear in the file. Among the sampled schemes, $\mathrm{VECK}^{+}_{\mathrm{EL}}$ must check a $\Theta(\lambda)$-sized collection of range proofs, since its exponential ElGamal requires proving that each sampled plaintext lies in the brute-forceable decryption range—a substantial cost. $\mathrm{VECK}^{\star}_{\mathrm{EL}}$ avoids the range proofs, and with its sampled ElGamal checks batched into a single multi-scalar multiplication its verifier is already inexpensive. Our verifier instead checks the KZG subset and point-opening proofs together with the CP-SNARK and its external links.

Our BW6-761 configuration is slower to verify than the baseline at $\beta \in \{2, 1.5, 1.25\}$, despite its faster proving. At $\beta = 1.1$, its verification aggregate is 77.9 ms versus 127.7 ms. These configurations share a SNARK curve but use different KZG curves. Our BLS12-381 configuration takes 10.8–28.6 ms, a 2.2–4.5× reduction from the baseline across the four settings. The improvement in core verification is therefore smaller than the sender-side gain, and does not remove the receiver's linear-time transcript processing.

B.5 Communication Accounting

The bulk symbol count determines only part of the communication: the byte encoding of each symbol and the additional proof and key material also contribute to the total.

Baseline bytes. The baseline encodes each file symbol in 32 bytes using the scalar field of BLS12-377. Its masked ciphertext uses the scalar field of BW6-761, requiring 48 bytes per symbol. Thus its bulk ciphertext occupies $48m$ bytes for a 32ℓ-byte file, an expansion of $1.5m/\ell$, not m/ℓ. At $\ell = 2^{16}$ and nominal $\beta = 1.1$, this is 3,460,416 bytes, or about 1.6501×, before proof, auxiliary, and key material. Our BLS12-381 configuration uses 32-byte symbols for both the file and masked ciphertext.

For comparison, we normalize the baseline's bulk communication to $32m$ bytes, crediting it with the smaller plaintext-field encoding while omitting the additional circuit cost of modular reduction into that field. This favors the baseline: its reported computation and normalized communication do not describe a jointly measured compact implementation. The normalized baseline and our BLS12-381 configuration both have bulk expansion m/ℓ, about 1.1000× at this setting.

Our protocol encoding. The polynomial proofs contribute four G_1 elements: C_{S_R}, π_{S_R}, U_α, and π_α. The CP-SNARK proof π_Z adds four G_1 and one G_2 element. Including vk, this gives $9G_1 + G_2 = 528$ compressed bytes. The exchange also sends the $m + 2$ field elements of ct (including positions $-1, -2$). With 32-byte field serialization, the ciphertext, proof, and verification key together occupy $32(m + 2) + 528$ bytes. Relative to a 32ℓ-byte file, the total factor is $m/\ell + 592/(32\ell)$, approximately 1.1003 at $\ell = 2^{16}$ and nominal $\beta = 1.1$.

B.6 Settlement Components and Deployment

Curve-dependent cost. Our key check uses the BLS12-381 precompile, whose scalar multiplication costs 12,000 gas [17]. The baseline uses a width-4 fixed-base method with on-chain lookup tables, mixed Jacobian–affine addition, and a final inversion; its reported scalar-multiplication cost is 1,305,332 gas. This component is already inside the seller call, not an additional charge. The baseline's table data and address table were actually deployed, costing 26,545,516 gas once. That on-chain setup cost is excluded from the per-exchange figures, as is escrow-contract deployment. The 108.8× scalar-multiplication ratio becomes 31.3× for the complete seller call because the remaining transaction and escrow work is still required.

B.7 Contextual Baselines and Extrapolation

VECK and $\text{VECK}^{+}_{\text{EL}}$ apply exponential ElGamal to the entire file; the direct baseline applies ElGamal only to samples. At $\ell = 2^{16}$, the estimated VECK sender cost is about 786 s; Table B.2 gives the $\text{VECK}^{+}_{\text{EL}}$ reference-case breakdown. At $\ell = 2^{20}$ and nominal $\beta = 1.25$, the estimated totals are 12,575 s for VECK and 437 s for $\text{VECK}^{+}_{\text{EL}}$. These estimates are distinct from the measured-component curves of Figure 3. Figure B.1 plots the nominal $\beta = 1.25$ setting alongside VECK, which has no coding parameter.

Extrapolation procedure. For a payload of m symbols, the implementation times a prefix of $n_{\text{pre}} = \min(m, n_{\max})$ symbols and reports

$$t(m) = t_{\text{meas}}(n_{\text{pre}})\frac{m}{n_{\text{pre}}}, \qquad n_{\max} = \begin{cases} 2^{14}, & \text{VECK}, \\ 2^{16}, & \text{VECK}^{+}_{\text{EL}}. \end{cases}$$

Unlike the five-run stage medians of the direct comparison, each prefix timing is a single run reused across rows with the same prefix length. For VECK, scaling applies to encryption, full-file range proofs, and the MSM inside the DLEQ proof; for $\text{VECK}^{+}_{\text{EL}}$, it applies only to encryption. Its range proofs cover R sampled positions and remain directly measured. Other stages run at full size. These aggregates sum the reported stage costs, including extrapolated estimates where applicable.

For the 16,384-symbol VECK prefix, encryption takes 5,162.845 ms and range proving 191,276.140 ms. The 65,536-symbol $\text{VECK}^{+}_{\text{EL}}$ encryption prefix takes 21,093.088 ms. The largest scale factors on the reported grid are 64× and approximately 32×, respectively. Across fully measured rows, per-element encryption cost varies by ±0.59% for VECK and ±1.17% for $\text{VECK}^{+}_{\text{EL}}$; VECK's range-proof cost varies by ±2.48%. These observations motivate the linear estimate but do not make extrapolated totals full-run measurements.

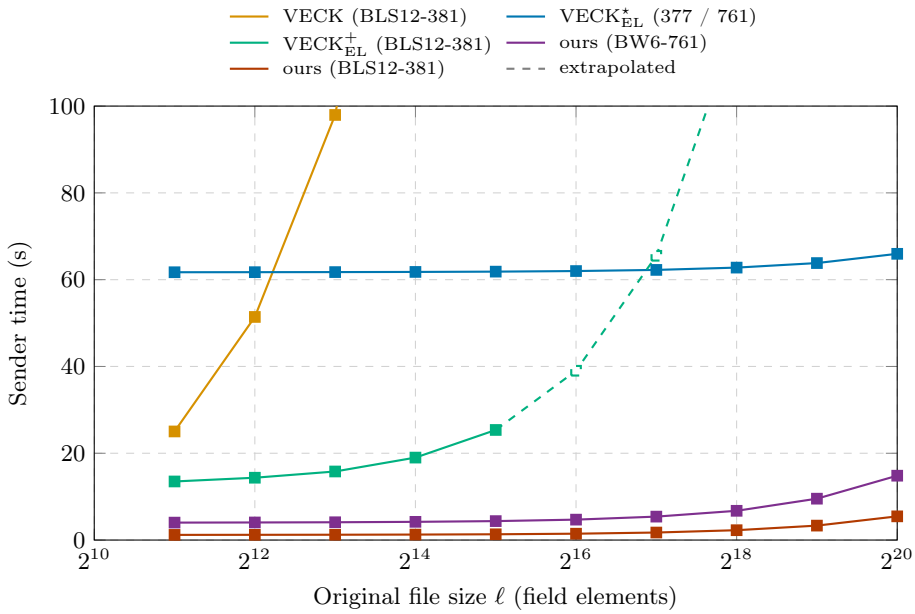


Fig. B.1. Contextual sender scaling at nominal $\beta = 1.25$. Solid points are measured; dashed segments include extrapolated stages.