

Universally Composable Reverse Firewalls for Oblivious Linear Evaluation and Applications

Suvradip Chakraborty¹, Kiran Deep Ghosh², Satrajit Ghosh³, Subha Kar⁴, Divyam Katiyar³, Sougata Mandal^{5,6}, Soumit Pal⁴, and Amlan Sinha^{5,7}

¹ VISA Research

² Ruhr University Bochum, Germany

³ Indian Institute of Technology, Kharagpur, India

⁴ Indian Statistical Institute, Kolkata, India

⁵ Institute for Advancing Intelligence (IAI), TCG CREST, Kolkata, India

⁶ Ramakrishna Mission Vivekananda Educational and Research Institute, Belur, India

⁷ Academy of Scientific and Innovative Research (AcSIR), Ghaziabad, India

`suvradip1111@gmail.com, kiran.ghosh@ruhr-uni-bochum.de, satrajit@cse.iitkgp.ac.in, subhakar9732@gmail.com, katiyardivya0@gmail.com, sougatamandal2014@gmail.com, soumitpal378@gmail.com, amlan.sinha.168@tcgcrest.org`

Abstract. Secure multi-party computation (MPC) traditionally assumes that parties execute protocols on trustworthy machines. This assumption fails under machine subversion, where compromised devices may leak sensitive information through manipulated protocol transcripts. Reverse Firewalls (RFs), introduced by Mironov and Stephens-Davidowitz, counter this threat by externally sanitizing messages in a functionality-preserving manner, without trusting the internal execution environment. Prior RF constructions are limited to Boolean-circuit-based MPC using oblivious transfer (OT) and garbled circuits (GC). In contrast, many high-performance MPC protocols are predominantly arithmetic, relying on Oblivious Linear Evaluation (OLE) as a core primitive. Achieving subversion resilience in this setting requires sanitizing OLE, a problem that has remained open.

We initiate a systematic study of subversion-resilient arithmetic MPC in the universally composable (UC) framework against semi-honest adversaries. We formalize a sanitizable OLE functionality $\mathcal{F}_{\text{SOLE}}$, prove a two-round impossibility for a natural class of additive homomorphic encryption (AHE)-based protocols, and present a UC-secure three-round construction from key-malleable AHE. To recover the full expressiveness of plain OLE, we further introduce *wrapped sanitizing OLE* under a covert adversary model, yielding an efficient UC-secure two-round protocol. Both constructions are instantiated using a key-malleable AHE scheme from class groups that we construct.

Using OLE as a black box, we obtain the first subversion-resilient UC-secure protocols for Oblivious Polynomial Evaluation (OPE), Oblivious

Polynomial Addition (OPA), and Private Set Intersection (PSI), achieving optimal communication for PSI. A full end-to-end implementation demonstrates that RF protection incurs only a $1.5\text{--}3\times$ overhead, showing that practical subversion-resilient MPC is attainable.

Keywords: OLE, PSI, Subversion Resilience, Universal Composability

Table of Contents

1	Introduction	3
1.1	Our Contribution	4
1.2	Technical Overview	6
2	Related Works	11
3	Preliminaries	12
3.1	Notation	12
3.2	Relevant UC Functionalities	12
3.3	Reverse Firewalls via Sanitizable Functionalities	13
3.4	Corruption Model	14
3.5	Subversion Resilient UC Paradigm	15
4	Sanitizing Oblivious Linear Evaluation	16
4.1	Sanitizable OLE Functionality	16
4.2	Impossibility of 2-Round Sanitizable OLE	17
4.3	3-Round Protocol Realizing Sanitizable OLE	21
4.4	2-Round Sanitizing OLE Protocol	24
4.5	Instantiation with Class Group	26
5	Applications of Sanitizing OLE	27
5.1	Oblivious Polynomial Evaluation	27
5.2	Oblivious Polynomial Addition	28
6	Sanitizing Private Set Intersection	29
6.1	Subversion-Resilient PSI from OPE	29
6.2	Subversion-Resilient PSI from OPA	31
7	Experimental Results	32
A	Background on Universal Composability	38
B	Missing Proofs in section 4	39
C	Randomized Equality	55
D	Correctness and Security of Class Group based Key Malleable AHE	56
E	OPE and OPA Protocol Specifications	57
F	Missing Proofs in section 5	58
G	PSI from OPE	65
H	PSI from OPA	67
I	Proof of Theorem 11	70
J	Sanitizing PSI from OPA	72
K	Sanitizing Two Sided PSI	73

1 Introduction

Secure multi-party computation (MPC) enables a set of mutually distrusting parties to jointly compute any function over their private inputs, without revealing those inputs to one another. The security guarantees of such protocols rest on a fundamental assumption: that each party’s machine faithfully executes the prescribed instructions. However, this assumption breaks down entirely in the presence of a *subverted machine* – one that has been compromised by an adversary and may covertly leak sensitive information through the protocol transcript.

This threat became starkly concrete in the post-Snowden era, which revealed that hardware and software could be silently backdoored to exfiltrate sensitive data, often entirely without the user’s knowledge [MS15]. Addressing this concern, [MS15] introduced the notion of a **Reverse Firewall (RF)**, subsequently developed in a line of follow-up works. An RF is positioned between the user’s machine and the outside world, intercepting and sanitizing all incoming and outgoing messages during a cryptographic protocol in a functionality-preserving manner. More precisely, a Reverse Firewall achieves three core objectives:

- **Functionality Preservation:** Assuming the user’s machine is operating correctly, the RF should not alter the functionality of the protocol.
- **Security Preservation:** If the underlying protocol is secure assuming an uncompromised implementation, the RF maintains the same security guarantees regardless of the machine’s behavior.
- **Exfiltration Prevention:** If the user’s machine is compromised, the RF ensures that no sensitive information can be leaked to the external world.

To achieve practical subversion resilience in secure computation, it is essential to construct reverse firewalls (RFs) for the core building blocks from which larger protocols are composed. Prior works have developed RFs for variants of Oblivious Transfer (OT) and garbled circuits [MS15, CGS23], as well as RF-protected MPC protocols [CDN20, CGPS21] based on GMW-style generic compilers [GMW87] in the stand-alone setting. Chakraborty, Magri, Nielsen, and Venturi [CMNV22] later extended the RF framework to the universal composability (UC) setting, introducing the notion of subversion-resilient UC (srUC), which guarantees that a protocol’s subversion resilience is preserved under arbitrary composition with other protocols. Subsequent work [CMMV24] further developed the srUC framework and constructed an efficient subversion-resilient password-authenticated key exchange (PAKE) protocol in this setting by introducing a sanitizable OT functionality and sanitizing the OT-based PAKE construction of Canetti et al. [CDVW12].

Despite this progress, an important practical limitation remains. Existing RF techniques based on OT and garbled circuits are naturally tailored to *Boolean circuits*, and evaluating arithmetic circuits over large fields using these approaches incurs substantial overhead. Modern high-performance MPC protocols, however, are predominantly arithmetic in nature. Achieving practical subversion resilience for such protocols therefore requires sanitizing the arithmetic analogue

of OT, namely **Oblivious Linear Evaluation (OLE)** [NP99, IPS09a, GNN17, ADI⁺17, BEPU⁺20, BDM22], which serves as a fundamental primitive in arithmetic MPC protocols [BDOZ11, DPSZ12, DGN⁺17]. In an OLE protocol, the sender holds $(a, b) \in \mathbb{F}^2$, the receiver holds $x \in \mathbb{F}$, and the receiver learns only the value $ax + b$, while the sender learns nothing about x . In this sense, OLE can be viewed as the arithmetic counterpart of OT. Despite its central role in arithmetic secure computation, the problem of constructing subversion-resilient OLE protocols has remained unexplored.

Our constructions further enable, for the first time, subversion-resilient realizations of several important higher-level arithmetic primitives, including Oblivious Polynomial Evaluation (OPE), Oblivious Polynomial Addition (OPA), and Private Set Intersection (PSI). Since these protocols admit constructions using only *black-box* access to OLE, they are particularly well-suited to modular sanitization through reverse firewalls. We summarize our contributions next.

1.1 Our Contribution

We initiate a systematic study of MPC protocols for arithmetic circuits in RF setting. Specifically, we make the following contributions:

- **Sanitizable OLE.** We formalize a sanitizable OLE functionality $\mathcal{F}_{\text{SOLE}}$ in which both firewalls independently blind the sender’s affine coefficients. We show that realizing sanitizable OLE with 2-round protocol is impossible for a class of OLE protocols based on additive homomorphic encryption (AHE), and circumvent this impossibility by constructing a srUC-secure 3-round OLE protocol from key-malleable AHE. As an application, we obtain a randomized equality protocol requiring only two symmetric calls to $\mathcal{F}_{\text{SOLE}}$, improving upon the $2n$ invocations of the underlying subversion-resilient OT required in the sanitizable OT-based instantiation of [CMNV22], where n is the shared-key length of the randomized equality functionality.
- **Wrapped Sanitizing OLE.** The input masking enforced by $\mathcal{F}_{\text{SOLE}}$ breaks the input-output correlation between the sender and the receiver, rendering sanitizable OLE unsuitable for building protocols that would otherwise be achievable from plain OLE. To recover usability, we consider a relaxed notion of wrapped sanitizing OLE, where the adversary is modeled as covert: it may attempt to exfiltrate information by manipulating the protocol transcript, but is not permitted to alter the honest parties’ inputs. Under this model, we construct a 2-round wsrUC-secure wrapped sanitizing OLE protocol from key-malleable AHE.
- **Concrete Instantiation from Key-Malleable AHE.** We instantiate both the 3-round protocol realizing sanitizable OLE and the 2-round wrapped sanitizing OLE protocols from key-malleable AHE. Key-malleability is crucial in these constructions, as one party must sample and transmit its public key to the other, which a subverted implementation could manipulate to exfiltrate information. We propose a key-malleable AHE scheme from class groups and use it to instantiate both the protocols.

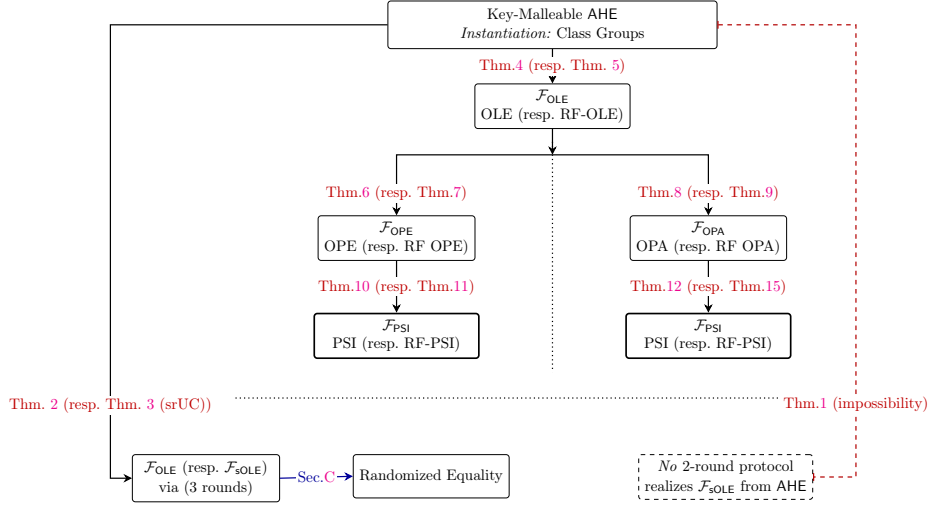


Fig. 1: A visual description of contributions in this paper. RF- $\{\text{OLE}, \text{OPE}, \text{PSI}\}$ stands for Sanitizing- $\{\text{OLE}, \text{OPE}, \text{PSI}\}$. All the functionalities are realized using wsrUC except $\mathcal{F}_{\text{sOLE}}$, which is realized using srUC.

- **Applications of RF-OLE.** We consider cryptographic primitives that can be constructed from OLE in a black-box manner – specifically, Oblivious Polynomial Evaluation (OPE), Oblivious Polynomial Addition (OPA), and Private Set Intersection (PSI). Since OPE and OPA admit black-box constructions from OLE, and PSI can in turn be built from either OPE or OPA as a black box, sanitizing OLE naturally lifts to reverse firewalls for all three primitives. To the best of our knowledge, these are the first universally composable subversion-resilient instantiations of OPE, OPA, and PSI. Moreover, our subversion-resilient PSI construction from RF-OPA achieves optimal communication complexity.
- **Implementation and Benchmarks.** We implement all protocols in C++ using the BICYCL class-group library [BCIL23]. Our experiments demonstrate that the reverse-firewalled variants incur an overhead of roughly 1.5–3× over their direct counterparts, confirming that subversion resilience can be achieved at a modest and practical cost.

Scope and Outlook. We focus on AHE-based OLE protocols, as AHE malleability enables a modular approach to RF construction and underlies many practically efficient MPC protocols. We view this work as an initial step, and expect future research to extend these techniques to other OLE instantiations and arithmetic primitives. A visual description of the contribution is given in the Figure 1.

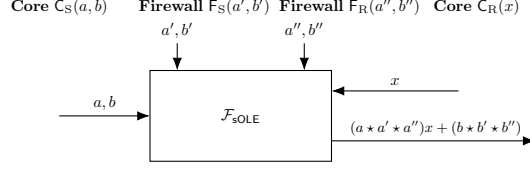


Fig. 2: Our sanitizable OLE functionality $\mathcal{F}_{\text{sOLE}}$, with $\star$ denoting an appropriate blinding operation for the input domain.

1.2 Technical Overview

We give a high-level walk-through of the technical contributions of the paper. Our development follows a modular path: starting from a sanitizing realization of OLE at the bottom, we lift it through OPE and OPA, and finally compose them into two subversion-resilient PSI protocols based on OPE and OPA. Additionally, we gave an impossibility result about 2-round sanitizable OLE.

Sanitizable OLE Functionality. The plain OLE functionality $\mathcal{F}_{\text{OLE}}$ lets a sender S choose a pair $(a, b) \in \mathbb{F}^2$ defining an affine map $f_{a,b}(X) = aX + b$ and a receiver R choose an evaluation point $x \in \mathbb{F}$, only the receiver gets the output $ax + b$, and sender learns nothing. This regular functionality is susceptible to subversion attacks. A (specious) sender can encode internal state into the pair (a, b) and route this state to a (specious) receiver through the output, while a subverted receiver can pick a triggering value of x that activates a hidden behavior on the sender’s side as soon as the input is processed.

Motivating from sanitizable OT in [CMMV24] our sanitizable OLE variant $\mathcal{F}_{\text{sOLE}}$ lets *both* firewalls blind the sender’s affine coefficients. Each firewall submits its own mask, and $\mathcal{F}_{\text{sOLE}}$ progressively folds these masks into (a, b) via a generic blinding operation “ $\star$ ”, which can be instantiated as addition, multiplication, or any other operation compatible with the primitive used in the instantiation. The receiver’s input x is *not* blinded as blinding (a, b) alone already neutralizes any triggering risk on the sender’s side. The net effect is that what reaches the receiver is indistinguishable from a fresh affine evaluation under honest inputs, no matter how the sender’s core sampled (a, b) . This is depicted in Figure 2.

Impossibility of 2-Round Sanitizable OLE. We prove that no 2-round AHE-based protocol can srUC-realize the natural sanitizable OLE functionality $\mathcal{F}_{\text{sOLE}}$, in which both firewalls independently apply uniform blinding to the sender’s coefficients (a, b) , producing effective coefficients $\hat{a} = a \star a' \star a''$ and $\hat{b} = b \star b' \star b''$.

At a high level, the impossibility stems from the limited algebraic expressiveness available in two rounds. Since sanitization must occur solely through the homomorphic interface, each firewall can only apply affine transformations to the underlying plaintext. Concretely, the ciphertext evolves as follows: the receiver R encrypts its input x ; the receiver-side firewall F_R applies its mask (a'', b'') , yielding plaintext $a''x + b''$; the sender S homomorphically evaluates the affine map $a(\cdot) + b$, obtaining $aa''x + ab'' + b$; finally, the sender-side fire-

wall F_S applies its mask (a', b') , resulting in $aa'a''x + aa'b'' + a'b + b'$. After the return-trip sanitization $z \mapsto \gamma z + \delta$ performed by F_R , the receiver outputs $\gamma(aa'a''x + aa'b'' + a'b + b') + \delta$. Correctness requires this expression to equal the ideal output $y_{\text{ideal}} = aa'a''x + bb'b''$ as a polynomial identity in (x, a, b) . However, this is impossible. Matching the coefficient of ax forces $\gamma = 1$, yet the cross-terms $aa'b''$ and $a'b$ appearing in the real-world execution have no counterpart in the ideal functionality. Eliminating these terms would require γ or δ to depend on values such as a , b , or a' . This is impossible because F_R has access to none of them: the sender's coefficients (a, b) remain hidden inside the ciphertext by IND-CPA security, while the mask a' is sampled locally by F_S , and the two firewalls share no randomness. Consequently, the receiver-side firewall cannot disentangle the unwanted cross-terms on the return path. Hence, two rounds do not provide sufficient algebraic coordination to realize sanitizable OLE in this setting.

Motivated by the impossibility result above, we move to a three-round protocol and show that sanitizable OLE can indeed be realized in this setting.

The 3-Round Protocol Realizing Sanitizable OLE. Our three-round OLE protocol proceeds as follows. In round 1 the sender relays encryptions of its inputs a and b under its own AHE public key. In round 2 the receiver uses the additive homomorphism to compute, under that same key, an encryption of $a \cdot x + b + r$, where r is a fresh additive mask the receiver chose to hide $a \cdot x + b$ from the sender. In round 3 the sender decrypts and forwards the plaintext value back to the receiver, who simply subtracts r and obtains $a \cdot x + b$. Due to the mask r the sender remains oblivious about receiver's output.

In order to instantiate $\mathcal{F}_{\text{SOLE}}$, we sanitize this 3-round OLE protocol. We rely on three modular transformations: *maul* the public key into a fresh-looking key, *align* a ciphertext from one public key to its mauled version, and *rerandomize* a ciphertext to randomize any information embedded in the ciphertext. These operations are provided by our key-malleable AHE abstraction in [Definition 4](#).

In round 1, each firewall sequentially applies its own additive mask (a' and a'' respectively) to the encryption of a via homomorphic addition, align the ciphertext under a freshly mauled public key, and also rerandomize the encryption of b . As a result, the receiver gets an encryption of $\tilde{a} = a + a' + a''$ and b under a mauled public key that is statistically independent of the sender's original key.

In round 2, the receiver samples a random r , encrypt it under the mauled public key, homomorphically evaluates the encryption of $\tilde{a}x + b + r$, and the resulting ciphertext travels back through each RF in reverse order, with each firewall applying the inverse alignment and rerandomization, until it arrives at the sender encrypted under its original key. On this return trip each firewall also samples a fresh mask of its own (r'' and r' respectively), encrypts it under the public key it currently holds, and adds it to the ciphertext by homomorphic addition.

In round 3, the sender decrypts and transmits $\tilde{a}x + b + r + r'' + r'$ in the clear. Each RF then removes the return-trip mask it added in round 2 and additively applies its own plaintext mask (b' and b'' respectively) to the outgoing message. The receiver strips r and recovers $(a + a' + a'')x + (b + b' + b'')$. The correctness

of this construction follows from the separation of masking phases: the a -masks are injected *before* the homomorphic evaluation, while the b -masks are applied *after* decryption, and the return-trip masks r' and r'' cancel because each firewall strips in the clear exactly what it added inside the ciphertext, thus realizing the $\mathcal{F}_{\text{sOLE}}$ correctly.

Application: Randomized Equality from Sanitizable OLE. The Randomized Equality functionality $\mathcal{F}_{\text{RE}}$ lets an initiator I and responder R, each holding a password, derive a shared high-entropy random key if and only if their passwords match; otherwise each gets independent random keys. As an application of sanitizable OLE, we revisit the randomized equality protocol from [CMMV24]. Recall that, oblivious transfer (OT) lets a sender input (x_0, x_1) and a receiver input a choice bit σ , learning only x_σ . In randomized equality, each party picks two random strings and acts as OT sender, while using its password bit as the receiver's choice in the symmetric OT. Each party XORs the string it sent on its own password bit with the string it received; the results agree if passwords match. For an n -bit password, n OTs handle each bit pairwise, and symmetry between sender and receiver roles doubles this to $2n$ OTs.

Since we are considering OLE over a large field, each party's password can be hashed into a single field element and handled by two symmetric sanitizable OLE calls with the roles reversed. Let w^I and w^R for the hashed passwords held by I and R. In the plain protocol, I inputs $(a^I, 0)$ and R inputs its password w^R , so that R learns $w^R a^I$; the second call swaps the roles and let I learn $a^R w^I$. Adding the locally computable term $w^I a^I$ (resp. $w^R a^R$), the parties obtain $w^I(a^I + a^R)$ and $w^R(a^I + a^R)$ on their own, which agree exactly when the two passwords match. Sanitization is achieved by having both firewalls mask the inputs of I and R in each sOLE call, removing any triggering value or subliminal leakage. The $2n$ sanitizable OT invocations of the prior construction thus collapse into just 2 calls to sanitizable OLE, yielding an efficient instantiation of $\mathcal{F}_{\text{RE}}$.

2-Round Sanitizing OLE. Before going into the details of the construction of the sanitizing OLE, let us highlight the key difference between the *sanitizable OLE* functionality $\mathcal{F}_{\text{sOLE}}$ and the *wrapped sanitizing OLE* functionality. $\mathcal{F}_{\text{sOLE}}$ sanitizes *inputs*: each firewall independently blinds the sender's affine coefficients, preventing triggering values and input-embedded leakage, but destroying the correlation between the sender's and receiver's inputs. Consequently, it cannot be realized in two rounds using AHE-based OLE protocols. In contrast, wrapped sanitizing OLE sanitizes only the *transcript*: honest inputs remain unchanged, while the firewalls ensure that every message on the wire appears freshly sampled. This comes at the cost of a weaker adversary, covert rather than fully malicious subversion that exfiltrates only through the transcript without tampering with honest inputs. In return, input-output correlation is preserved and a two-round construction becomes possible.

The AHE-based two-round OLE is straightforward: the receiver encrypts x under its public key; the sender homomorphically evaluates $ax + b$ and returns the result; the receiver decrypts to obtain $ax + b$. Sanitization is achieved by having both firewalls ensure that all in-transit message is indistinguishable from

freshly sampled values. In the forward direction, each firewall successively mauls the public key and re-randomizes and re-aligns the ciphertext, scrubbing any subliminal payload a subverted receiver may have embedded. In the return direction, the firewalls reverse this process – re-aligning then re-randomizing – restoring the ciphertext to the original key for correct decryption. The construction realizes the wrapped $\mathcal{F}_{\text{OLE}}$ functionality, neutralizing every subliminal channel a subverted implementation could exploit, and yields an efficient 2-round sanitizing OLE that serves as the basic building block for all higher-level constructions in this work.

Instantiating Key-Malleable AHE from Class Groups. All our AHE-based constructions use a single primitive: a *key-malleable additively homomorphic encryption* scheme. Beyond the standard $\text{KeyGen}, \text{Enc}, \text{Dec}$ and homomorphic-evaluation interface, we require four key-malleability properties: *(KM1) key correctness*—mauling a public key admits a matching secret-key transformation that still decrypts; *(KM2) ciphertext translation*—mapping a ciphertext under a key to one under its mauled neighbour; *(KM3) key rerandomization*, the technical heart, requiring the mauled key to be statistically close to a fresh KeyGen output and carry no information about its predecessor; and *(KM4) encryption alignment*, like *(KM2)* but aligning between the mauled and original keys in either direction. Our protocols use *(KM4)* only. We instantiate this abstraction in the class-group framework of [CL15], working in the easy-DL subgroup $\mathbb{F}_{\text{CL}} = \langle f \rangle$ of prime order q inside a class group $\mathbb{G}_{\text{CL}}$. An ElGamal-style ciphertext is $(R, E) = (g_q^r, f^m \text{pk}^r)$ under $\text{pk} = g_q^{\text{sk}}$; decryption recovers f^m by group inversion and solves discrete log in $\mathbb{F}_{\text{CL}}$. Key-mauling is multiplicative: MaulPK samples a fresh exponent r' and sets $\text{pk}' = \text{pk} \cdot g_q^{r'}$, with the matching transformation adding r' to sk . Translation and alignment multiply (resp. divide) E by $R^{r'}$, and *(KM3)* follows from statistical uniformity of $g_q^{r'}$ in the easy-DL subgroup. Efficient scalar multiplication of a ciphertext by a field element lets the sender homomorphically evaluate $aX + b$ on the receiver's encryption. Plugging this instantiation into our generic constructions yields sanitizing 3-round and 2-round OLE protocols.

Sanitizing OPE from Sanitized OLE. In OPE the sender holds a polynomial $\mathbf{p}(X)$ of degree d and the receiver holds $\alpha \in \mathbb{F}$; the receiver learns $\mathbf{p}(\alpha)$ and the sender learns nothing. We use the Horner-style construction of Naor and Pinkas [NP06]: the parties maintain additive shares of the Horner suffix, advancing one coefficient per $\mathcal{F}_{\text{OLE}}$ call. The sender masks its share with a fresh field element at each step, making the receiver's view of every OLE output information-theoretically uniform and simulation straightforward. Lifting to the subversion-resilient setting routes each of the d OLE calls through the 2-round wrapped sanitizing OLE, which independently scrubs any subliminal channel a corrupted core might exploit. Since all Horner bookkeeping is local and no algebraic message is exchanged at the OPE layer beyond the OLE sub-transcripts, sanitation lifts cleanly from the sub-OLEs, yielding a subversion-resilient OPE.

PSI from OPE and its Sanitizing Version. A standard one-sided PSI from OPE has Alice encode her set as the polynomial whose roots are her elements,

and Bob call OPE once per element in his set, adding it to the intersection iff the OPE returns zero. Correctness follows from the root-encoding. OPE security hides Alice’s polynomial beyond a single value per query, uniform over the field when the query lies outside the intersection. Thus n OPE calls suffice for a receiver set of size n . To lift this to a subversion-resilient protocol, we again proceed modularly: replace each ideal OPE call with our sanitizing OPE. The key structural fact is that one-sided OPE-based PSI carries *no* PSI-layer messages beyond the OPE sub-transcripts—Alice only feeds her polynomial to each OPE, and Bob assembles the output locally. Every channel a corrupted core could exploit is therefore already a channel of the underlying sanitizing OPE, and sanitation lifts directly. The result is, to our knowledge, *the first subversion-resilient PSI construction* in the cryptographic reverse firewall model.

OPA from OLE and its Sanitizing Version. The OPA functionality takes a sender’s polynomial $\mathbf{p}_A$ of degree at most $2d$ and a random polynomial $\mathbf{r}_A$ of degree d and a receiver’s polynomial $\mathbf{p}_B$ of degree at most d , and delivers $\mathbf{p}_A + \mathbf{p}_B \mathbf{r}_A$ to the receiver, this multiplier prevents the receiver from cancelling its input to recover $\mathbf{p}_A$. Following [GN19], we realize OPA from plain OLE by evaluating each polynomial at $2d + 1$ public points, running one OLE per point, and letting the receiver interpolate the resulting degree- $2d$ output. Lifting to a sanitizing OPA is modular, mirroring the OPE case: replace each of the $2d + 1$ OLE calls with the 2-round wrapped sanitizing OLE. No algebraic message is exchanged at the OPA layer beyond the OLE sub-transcripts—evaluations and interpolation are local to each core so sanitation of the OPA inherits directly from that of its sub-OLEs.

Sanitizing PSI from Sanitized OPA. Our PSI construction follows the polynomial-encoding paradigm and is realized through a single invocation of the OPA functionality. Each party encodes its input set as a polynomial $\mathbf{p}_A$ and $\mathbf{p}_B$, whose roots are exactly the elements of the corresponding set. The sender additionally masks its polynomial using independently sampled random polynomials $\mathbf{r}_A$ and $\mathbf{r}'_A$. Both party run OPA with the input $(\mathbf{r}'_A \cdot \mathbf{p}_A, \mathbf{r}_A)$ and $\mathbf{p}_B$. Through the OPA functionality, the receiver obtains the randomized polynomial $\mathbf{p}_\cap = \mathbf{r}'_A \cdot \mathbf{p}_A + \mathbf{r}_A \cdot \mathbf{p}_B$, which evaluates to zero on every element in the set intersection and, except with negligible probability, is nonzero on all other elements. The receiver therefore recovers the intersection by evaluating $\mathbf{p}_\cap$ on its own inputs. The protocol is communication efficient, requires only a single OPA invocation, and derives its security from the underlying OPA functionality. The sanitizing PSI protocol is obtained by replacing the standard OPA invocation in the PSI construction with the sanitizing OPA protocol. Since the PSI protocol requires only a single OPA invocation and no further interaction, the resulting construction directly inherits the sanitization and security guarantees of the underlying OPA protocol.

In two-sided PSI both parties receive the intersection. The two-sided PSI protocol is obtained by extending the one-sided PSI construction so that the receiver additionally sends the computed intersection to the sender. In the subversion-resilient setting, the extra communication is made leak-free by sanitizing the

underlying message transmission protocol. By composing the firewalls of sanitizing OPA with those of sanitized message transmission, we obtain a modular construction of sanitized two-sided PSI.

2 Related Works

Oblivious Evaluation and Private Set Intersection. Naor and Pinkas [NP99] introduced OLE as a natural algebraic extension of oblivious transfer (OT) [EGL85, Rab05]. Since OLE enables secure multiplication over arithmetic domains, it has become a fundamental building block for secure arithmetic computation [IPS09b, GIP⁺14, DGN⁺17], much as OT serves as a basic primitive for evaluating Boolean circuits [GMW87, Kil88]. Vector OLE (VOLE) [ADI⁺17] extends OLE to vectors and has become an important tool for efficient correlation generation.

Oblivious polynomial evaluation (OPE) [NP99] generalizes OLE by allowing a party to securely evaluate a polynomial on a private input. OPE can be constructed using black-box access to OLE [NP99, NP06, GNN17]. It also provides a natural approach to private set intersection (PSI), but a direct construction incurs quadratic overhead in the set sizes. Ghosh and Nilges [GN19] introduced oblivious polynomial arithmetic (OPA), enabling PSI from a linear number of OLE calls and achieving optimal communication in the balanced setting.

Private set intersection (PSI) [Mea86] enables two or more parties to compute the intersection of their private sets. Its broad applicability in privacy-preserving systems has led to numerous constructions based on different cryptographic primitives [FNP04, KS05, HL08, KKRT16, HV17, GN19, PSTY19, PRTY20, RR22], as well as several variants targeting different settings and functionality [GS19, GRS22, GS23, CFR23, vBP24].

In this paper our goal is to design subversion-resilient PSI protocols. For the balanced setting, we build on the OLE-based PSI protocol of Ghosh and Nilges [GN19]; for the unbalanced setting, we use an OPE-based approach. Since both constructions ultimately reduce to OLE, a reverse firewall for OLE provides a common foundation for both settings.

Reverse Firewalls and Subversion Resistance. Reverse firewalls (RFs), introduced by Mironov and Stephens-Davidowitz [MS15], aim to protect cryptographic protocols against subverted implementations. Subsequent work has developed RF techniques for a range of cryptographic settings [AMV15, CMY⁺16, DMSD16, BBF⁺20, CGS23, CMMV24]. More recent applications include interactive proof systems [GMV20], MPC protocols [CDN20, CGPS21], and real-world secure messaging systems such as Signal [DMSDT25]. Chakraborty et al. [CMNV22] further extended the original framework to universally composable (UC) subversion-resistant cryptography.

Despite this progress, subversion resilience has not been studied for oblivious evaluation primitives such as OLE, OPE, and OPA. In particular, existing reverse-firewall constructions do not provide a subversion-resilient treatment of these primitives or of the PSI protocols built from them. Our work fills this

gap by developing reverse-firewall constructions for OLE and using them as a common foundation for subversion-resilient OPE, OPA, and PSI.

3 Preliminaries

3.1 Notation

Throughout the paper, λ denotes the security parameter and $\text{negl}(\lambda)$ a negligible function in λ . For a finite set or distribution S , we write $x \xleftarrow{\$} S$ for uniform sampling from S , and more generally $x \leftarrow \mathcal{D}$ when x is sampled according to a distribution $\mathcal{D}$. We denote by $\mathbb{Z}$ and $\mathbb{N}$ the sets of integers and natural numbers, respectively. Let $\mathbb{F}$ be a finite field and $\mathbb{F}[X]$ the ring of univariate polynomials over $\mathbb{F}$; for $f \in \mathbb{F}[X]$, $\deg(f)$ denotes its degree. For sets P, Q , we write $P \cap Q$ for their intersection. Finally, $X \stackrel{\mathcal{C}}{\approx} Y$ denotes computational indistinguishability of ensembles X and Y , while $X \equiv Y$ denotes identical or statistical indistinguishability, depending on the context. All other notation is defined where needed throughout the paper.

3.2 Relevant UC Functionalities

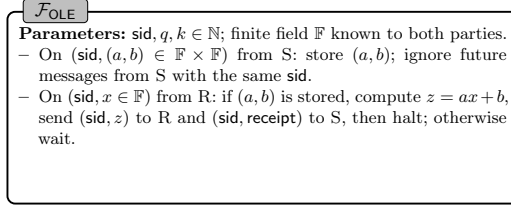
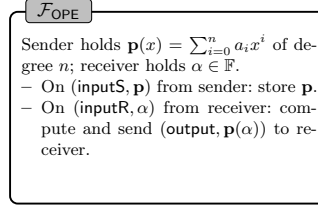
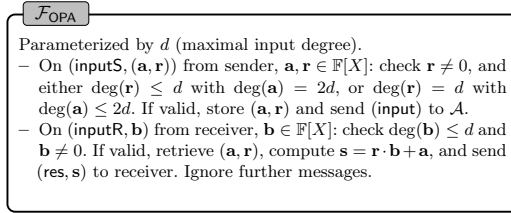
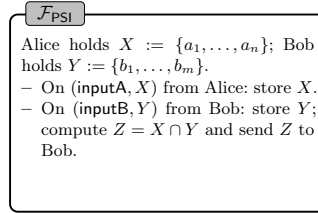
We describe the relevant ideal functionalities that are used throughout this paper. In Appendix A, we present a discussion on the UC framework of [Can00].

Oblivious Linear Evaluation (OLE). OLE is a two-party primitive for secure arithmetic computation over a finite field $\mathbb{F}$. The sender A holds $(a, b) \in \mathbb{F}^2$, defining the affine map $f_{a,b}(X) = aX + b$, while the receiver B holds $x \in \mathbb{F}$ and obtains only $y = ax + b$. Security ensures B learns nothing about a, b beyond $ax + b$, and A learns nothing about x . OLE is the arithmetic analogue of oblivious transfer, widely used as a building block for secure computation over fields [DGN⁺17, Gil99, BEPU⁺20]. The ideal functionality $\mathcal{F}_{\text{OLE}}$ is in Figure 3.

Oblivious Polynomial Evaluation (OPE). OPE allows two parties to privately evaluate a polynomial held by one party at a point held by the other [NP99, NP06]. The sender A inputs $\mathbf{p} \in \mathbb{F}[X]$ and the receiver B inputs $\alpha \in \mathbb{F}$; B obtains $\mathbf{p}(\alpha)$ while A gets no output. Security ensures B learns nothing about $\mathbf{p}$ beyond $\mathbf{p}(\alpha)$, and A learns nothing about α . OPE is a standard building block for secure arithmetic and polynomial-based protocols such as PSI. The ideal functionality $\mathcal{F}_{\text{OPE}}$ is given in Figure 4.

Oblivious Polynomial Addition (OPA). OPA allows two parties to compute a randomized sum of their input polynomials without disclosing them individually. Following Ghosh and Nilges [GN19], the sender holds $\mathbf{a}, \mathbf{r} \in \mathbb{F}[X]$ (where $\mathbf{r}$ is a randomizing polynomial) and the receiver holds $\mathbf{b} \in \mathbb{F}[X]$. The receiver obtains $\mathbf{s}(X) = \mathbf{a}(X) + \mathbf{r}(X)\mathbf{b}(X)$; the sender gets no output. The random multiplier $\mathbf{r}$ prevents the receiver from recovering $\mathbf{a}$ by subtracting $\mathbf{b}$. The ideal functionality $\mathcal{F}_{\text{OPA}}$ is in Figure 5.

Private Set Intersection (PSI). PSI is a fundamental two-party computation task in which Alice holds $X = \{a_1, \dots, a_n\}$ and Bob holds $Y = \{b_1, \dots, b_m\}$

**Fig. 3:** OLE Functionality $\mathcal{F}_{OLE}$.**Fig. 4:** OPE Functionality $\mathcal{F}_{OPE}$.**Fig. 5:** OPA Functionality $\mathcal{F}_{OPA}$.**Fig. 6:** PSI Functionality $\mathcal{F}_{PSI}$.

(elements in random order), and Alice learns only the intersection $Z := X \cap Y$ while Bob obtains no output. Security ensures Alice learns nothing about Y beyond elements in Z , and Bob learns nothing about X . When $n = m$, this is *balanced PSI*; when $m \ll n$, it is *unbalanced PSI*. The ideal functionality $\mathcal{F}_{PSI}$ is given in Figure 6.

3.3 Reverse Firewalls via Sanitizable Functionalities

We follow the UC modeling of reverse firewalls introduced in [CMNV22]. In this model, each logical party P_i is represented by two separate UC parties: a *core* C_i and a *firewall* F_i , with distinct party identifiers (pid, C) and (pid, F) , respectively. We use pid to refer to the logical party as a whole, while C_i and F_i denote its two components. This separation is needed so that the usual UC composition theorem can be applied in the presence of reverse firewalls.

Since C_i and F_i are distinct parties, they do not communicate directly. Instead, their interaction is mediated by a *sanitizable ideal functionality* $\mathcal{G}$. The functionality $\mathcal{G}$ provides an input-output interface IO_i connected to the core and a sanitation interface S_i connected to the firewall. Whenever $\mathcal{G}$ receives a message from C_i , it may forward the message, or suitable information about it, to F_i through S_i . The firewall may then modify the message before $\mathcal{G}$ processes or delivers it. This modification is called *sanitization*. This is illustrated in Figure 13 of Appendix A.3.

Thus, the firewall sanitizes the communication of the core without learning or holding the party's secret inputs. The exact form of sanitization is not fixed by the framework; rather, it is specified by the particular functionality $\mathcal{G}$. In this way, reverse firewalls are modeled as separate components that can inspect

and sanitize the core’s communication while still fitting naturally into the UC framework and its composition theorem.

3.4 Corruption Model

We adopt the corruption model of [CMNV22]. Each party P_i splits into two independent components: a *core* C_i holding private state and generating messages, and a *firewall* F_i sanitizing all communication to and from the core. The two may be corrupted independently.

The core admits three modes: *honest* (follows the protocol), *malicious* (arbitrary adversarial control), and *specious*—a subverted $\tilde{C}_i$ that is computationally indistinguishable from C_i under any efficient black-box test, yet may leak via subliminal channels or misbehave on a triggering input. Specious corruption models undetectable subversion that survives all audits. The firewall admits *honest*, *semi-honest* (follows the protocol but leaks its state), and *malicious*.

Since UC functionalities act at the party level, the joint status of (C_i, F_i) is mapped to a single party-level corruption via the *corruption translation table* (Table 1) from [CMNV22]. Notably, a specious core with an honest firewall is treated as honest—precisely the guarantee reverse firewalls aim to provide. An honest core with a malicious firewall yields an *isolate* corruption: a rogue firewall severing the core’s channel, which collapses to standard malicious (or man-in-the-middle in the unauthenticated setting) and is subsumed accordingly.

Core C_i	Firewall F_i	Party $P_i = (C_i, F_i)$
Honest	Semi-Honest	Honest
Specious	Honest	Honest
Honest	Malicious	Isolate
Malicious	Malicious	Malicious

Table 1: Corruption translation table, as in [CMNV22].

A central security notion tied to this model is that of *strong sanitation*. A firewall F_i is said to be strongly sanitizing if no efficient environment can distinguish a protocol execution in which core C_i is speciously corrupted (with F_i honest) from one in which C_i is entirely incorruptible (with F_i honest). As established in [CMNV22], whenever the firewalls are strongly sanitizing, the specious/honest combination is effectively equivalent to the honest/honest combination for the purposes of UC security, justifying the entry in Table 1.

Definition 1 (Strong sanitation [CMNV22]). Let (C_i, F_i) be the core and firewall pair of party P_i in the $\mathcal{G}$ -hybrid model, and let $\hat{C}_i$ denote the corresponding incorruptible, honest core. Let Π_i be the execution where party i runs (C_i, F_i) , and let Π'_i be the execution where party i runs $(\hat{C}_i, F_i)$, with all other parties being dummy parties. We say that F_i is strongly sanitizing if, for every polynomial-time environment $\mathcal{Z}$ that does not corrupt F_i but may speciously corrupt C_i , the two executions are computationally indistinguishable: $\text{Exec}_{\mathcal{Z}, \Pi_i, \mathcal{A}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{Z}, \Pi'_i, \mathcal{A}}$, where adversary $\mathcal{A}$ is a dummy adversary.

3.5 Subversion Resilient UC Paradigm

The subversion resilient UC (srUC) framework distinguishes two types of ideal functionalities. A *sanitizable functionality* interacts with both components of each party $P_i = (C_i, F_i)$: the core via an I/O interface IO_i , the firewall via a sanitation interface S_i . A *regular functionality* is a standard UC functionality, unaware of the core/firewall decomposition. Regular functionalities remain useful for tasks like coin tossing or key exchange, but lack sanitation interfaces. The framework therefore considers their wrapped version $\text{Wrap}(\mathcal{F})$, which translates joint core/firewall corruption into a party-level status and forwards every output for P_i to both C_i and F_i preventing trivial distinguishing attacks against real protocols run by separate components. This yields two notions: *wsrUC security* for *sanitizing* protocols realizing wrapped regular functionalities (Definition 2), and *srUC security* for protocols realizing sanitizable functionalities (Definition 3), both in the $\mathcal{G}$ -hybrid model. We will use “sanitizing” and “sanitized” to convey the same meaning.

Definition 2 (Wrapped subversion-resilient UC security [CMNV22]).

Let $\mathcal{F}$ be a regular ideal functionality for the parties $\{P_1, \dots, P_n\}$. Let Π be a sanitizing protocol described by a collection of cores $\{C_1, \dots, C_n\}$ together with their corresponding firewalls $\{F_1, \dots, F_n\}$, and let $\mathcal{G}$ denote the sanitizable ideal functionality available to Π as in Figure 13. We say that Π wsrUC-realizes $\mathcal{F}$ in the $\mathcal{G}$ -hybrid model if, in the presence of only specious environments and specious adversaries, the protocol Π UC-realizes the wrapped functionality $\text{Wrap}(\mathcal{F})$ in the $\mathcal{G}$ -hybrid model.

Definition 3 (Subversion-resilient UC security [CMNV22]). Let $\mathcal{F}$ be a sanitizable ideal functionality for the parties $\{P_1, \dots, P_n\}$. Consider a sanitizing protocol Π specified by cores $\{C_1^{\mathcal{F}}, \dots, C_n^{\mathcal{F}}\}$ and corresponding firewalls $\{F_1^{\mathcal{F}}, \dots, F_n^{\mathcal{F}}\}$. Let $\mathcal{G}$ be a sanitizable ideal functionality to which Π has access via the input-output and sanitation interfaces. We say that Π srUC-realizes $\mathcal{F}$ in the $\mathcal{G}$ -hybrid model if $\mathcal{F}$ is well-formed and sanitizable, and, against only specious environments and specious adversaries, the protocol Π UC-realizes $\mathcal{F}$ in the $\mathcal{G}$ -hybrid model.

Lemma 1 (Reduction via Strong Sanitation [CMNV22]). Let Π be a protocol such that, for every party $P_i = (C_i, F_i)$, the firewall F_i is strongly sanitizing. Then, to prove subversion-resilient UC security of Π , it suffices to consider only the following corruption patterns:

1. C_i is MALICIOUS and F_i is MALICIOUS;
2. C_i is HONEST and F_i is SEMIHONEST;
3. C_i is HONEST and F_i is MALICIOUS.

Furthermore, if the ideal functionality treats an ISOLATE party as MALICIOUS, then the third case is also subsumed by the first one. Hence, it is enough to prove security only for the cases where either both C_i and F_i are MALICIOUS, or C_i is HONEST and F_i is SEMIHONEST.

Sanitizable Authenticated Transmission (SAT). The functionality $\mathcal{F}_{\text{SAT}}$ provides the communication backbone of the srUC framework by mediating all communication between protocol cores through reverse firewalls. Concretely, when a core sends a message, it is first delivered to its local firewall through a sanitation interface, allowing the firewall to sanitize, rerandomize, or modify the message before forwarding it to the recipient’s firewall, which then delivers it to the receiving core. Thus, authentication is guaranteed only at the firewall level, while malicious firewalls may still alter or inject messages before they reach the cores. The functionality also supports a setup phase distributing correlated public parameters, while leaking firewall state to the adversary to ensure that firewalls hold no secrets. Overall, $\mathcal{F}_{\text{SAT}}$ captures a communication model in which reverse firewalls systematically eliminate subliminal channels and leakage attempts from potentially subverted cores, thereby enabling the construction of subversion-resilient protocols in the UC framework. All our sanitizing protocols are written in $\mathcal{F}_{\text{SAT}}$ -hybrid model even if it is not specified explicitly. We have written the functionality of SAT in Appendix A.2

4 Sanitizing Oblivious Linear Evaluation

4.1 Sanitizable OLE Functionality

We define the sanitizable OLE functionality $\mathcal{F}_{\text{sOLE}}$ in Figure 7. The plain $\mathcal{F}_{\text{OLE}}$ lets a specious sender encode hidden state into $(a, b) \in \mathbb{F}^2$ and route it to a corrupted receiver via the output $a \cdot x + b$, and symmetrically lets a specious receiver pick x as a triggering value activating hidden behavior in the sender’s code. In $\mathcal{F}_{\text{sOLE}}$, both firewalls blind the sender’s inputs via an operation “ $\star$ ” by sending BLIND message to $\mathcal{F}_{\text{sOLE}}$ with their blinding masks. The sender’s firewall F_S sanitizes (a, b) , washing out subliminal information from the core’s randomness, and F_R does the same on the inbound side. The receiver thus obtains $\hat{a} \cdot x + \hat{b}$. The receiver’s input x is not blinded: in OLE the receiver chooses its own point, and triggering risk on the sender’s side is already neutralized by blinding (a, b) . The output is indistinguishable to both cores.

The functionality is parameterized by $\star : \mathbb{F}^2 \rightarrow \mathbb{F}$, instantiable as additive, multiplicative, or any operation compatible with $\mathbb{F}$. The sender’s core feeds (a, b) via the input-output interface IO_S , the receiver’s core feeds x via IO_R , and the receiver’s output is released only after both firewalls have had the chance to blind—ensuring no sanitation step is silently bypassed. Until delivery, $\mathcal{F}_{\text{sOLE}}$ continues to accept BLIND messages. Each firewall gets one effective BLIND per session, with subsequent commands ignored. F_S ’s blinding inputs (a', b') updates the record to $(\tilde{a}, \tilde{b}) = (a \star a', b \star b')$. Then, F_R ’s blinding inputs (a'', b'') updates it to $(\hat{a}, \hat{b}) = (\tilde{a} \star a'', \tilde{b} \star b'')$. F_R may act only after F_S , with the usual exemption when the sender is malicious, according to the corruption translation Table 1, reflecting that F_R is the last sanitation layer before the receiver’s core.

Distinction between $\mathcal{F}_{\text{sOLE}}$ and $\mathcal{F}_{\text{OLE}}$. Note that, $\mathcal{F}_{\text{sOLE}}$ is not a drop-in replacement for OLE. Its firewalls add their masks into (a, b) inside the functionality, so the receiver obtains $\hat{a}x + \hat{b}$. This suits exactly those protocols in which the

sender's coefficients are randomness the core samples itself: Randomized equality is such a case. OPE, OPA and PSI are not. There (a, b) encode the actual polynomial, and blinding would destroy the structure of the output. For them we realize $\text{Wrap}(\mathcal{F}_{\text{OLE}})$, whose wrapper only translates corruptions and leaves $ax + b$ intact.

4.2 Impossibility of 2-Round Sanitizable OLE

We first define the notion of *key-malleable additive homomorphic encryption* (AHE) scheme. Then, we prove that no 2-round OLE protocol based on key-malleable AHE scheme srUC realizes $\mathcal{F}_{\text{SOLE}}$.

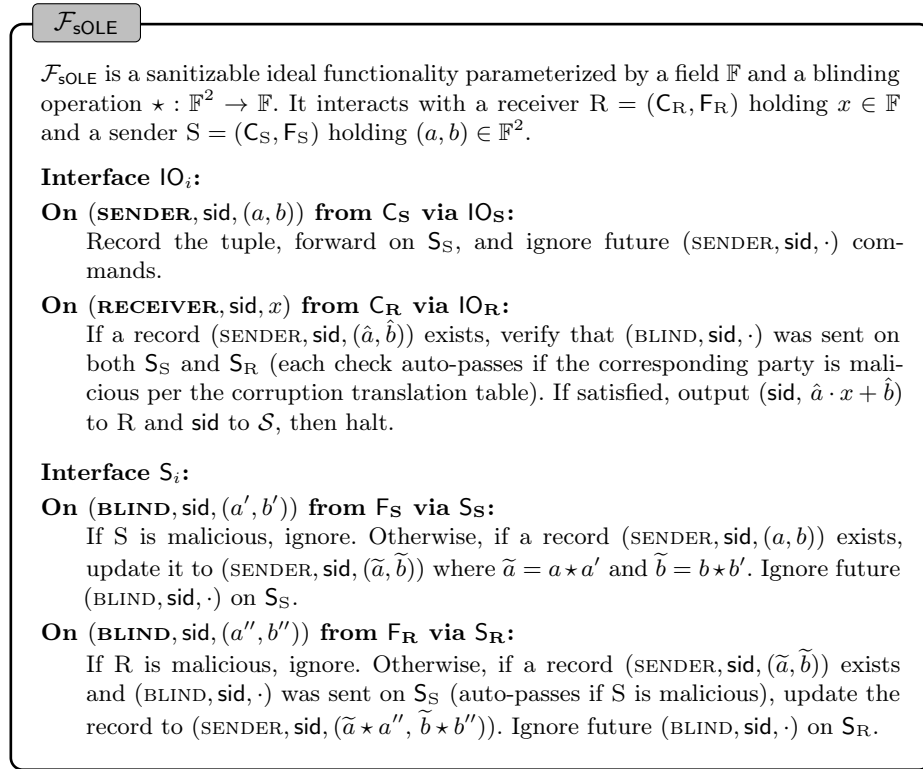


Fig. 7: The Sanitizable OLE Functionality.

Definition 4 (Key-Malleable AHE). Let $\mathcal{M} = (\mathcal{M}, \oplus, 0_{\mathcal{M}})$ be a finite abelian group, called the plaintext group, whose group operation, identity, and inversion are computable in time $\text{poly}(\lambda)$ given a succinct description of $\mathcal{M}$. A key-malleable AHE scheme over $\mathcal{M}$ consists of eight PPT algorithms (KeyGen, Enc,

Dec , HomOp , MaulPK , MaulSK , MaulCT , AlignEnc) together with a ciphertext rerandomization algorithm ReRand , satisfying:

- Correctness: For any $m \in \mathcal{M}$ and $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$, $\Pr[\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, m)) = m] \geq 1 - \text{negl}(\lambda)$.
- Additive Homomorphism: There exists HomOp on ciphertexts s.t. for any $m_1, m_2 \in \mathcal{M}$ and $c_i = \text{Enc}(\text{pk}, m_i)$, $\text{Dec}(\text{sk}, \text{HomOp}(c_1, c_2)) = m_1 \oplus m_2$.
- IND-CPA Security: Standard chosen-plaintext indistinguishability over messages in $\mathcal{M}$.
- Ciphertext Rerandomization: The PPT Rerand , with access to pk , satisfies for all $m \in \mathcal{M}$: if $c = \text{Enc}(\text{pk}, m)$, then $\text{Dec}(\text{sk}, \text{Rerand}(\text{pk}, c)) = m$, and $(c, \text{Rerand}(\text{pk}, c)) \stackrel{c}{\approx} (c, \text{Rerand}(\text{pk}, \text{Enc}(\text{pk}, 0)))$.

Key-malleability properties. For $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda, \mathcal{M})$ and $(\text{pk}', \tau) \leftarrow \text{MaulPK}(\text{pk})$:

- (**KM1**) Key Correctness: $\text{sk}' := \text{MaulSK}(\text{sk}, \tau)$ satisfies $\Pr[\text{Dec}(\text{sk}', \text{Enc}(\text{pk}', m)) = m] \geq 1 - \text{negl}(\lambda)$ for all $m \in \mathcal{M}$.
- (**KM2**) Ciphertext Alignment: for any $c \leftarrow \text{Enc}(\text{pk}, m)$, $\text{MaulCT}(c, \tau) \equiv \text{Enc}(\text{pk}', m)$.
- (**KM3**) Key Rerandomization: the marginal distribution of pk' from $\text{MaulPK}(\text{pk})$ for $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}$ is statistically close to a fresh public key: $\{\text{pk}' : (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}, (\text{pk}', \tau) \leftarrow \text{MaulPK}(\text{pk})\} \equiv \{\text{pk}^* : (\text{pk}^*, \text{sk}^*) \leftarrow \text{KeyGen}\}$. Moreover, (pk, pk') is statistically close to (pk, pk^*) with pk^* independently sampled, i.e., pk' carries no information about pk .
- (**KM4**) Encryption Alignment: For any $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda, \mathcal{M})$, $(\text{pk}', \tau) \leftarrow \text{MaulPK}(\text{pk})$, and $m \in \mathcal{M}$, let $\tilde{c} \leftarrow \text{Enc}(\text{pk}', m)$. Then $\text{AlignEnc}(\tilde{c}, \text{pk}, \tau) \equiv \text{Enc}(\text{pk}, m)$.

Efficient Addition and Scalar Multiplication. If $\mathcal{M}$ is additionally a commutative ring, the scheme supports efficient scalar multiplication if there exists a PPT algorithm CMult s.t. for any $c = \text{Enc}(\text{pk}, m)$ and $\alpha \in \mathcal{M}$, $\text{Dec}(\text{sk}, \text{CMult}(c, \alpha)) = \alpha \cdot m$, where “ $\cdot$ ” is ring multiplication in $\mathcal{M}$, and $\text{CMult}(c, \alpha)$ runs in time $\text{poly}(\lambda)$ independent of the magnitude of α . For two elements $m_1, m_2 \in \mathcal{M}$, we often write $m_1 \oplus m_2$ as $\text{ADD}(m_1, m_2)$.

Remark 1 (On Encryption Alignment). We define both MaulCT and AlignEnc as **KM2** and **KM4**, respectively. Although we emphasize that **KM2** can be achieved using **KM4** and vice versa. Another way to look at AlignEnc is that, it can be thought of as the inverse direction of MaulCT along the same mauling pair (pk, pk') . While $\text{MaulCT}(\cdot, \tau)$ pushes a ciphertext from pk to pk' , $\text{AlignEnc}(\cdot, \text{pk}, \tau)$ pulls back a ciphertext under pk' back to a ciphertext under pk of the same plaintext, statistically close to a fresh encryption. For any $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda, \mathcal{M})$, any $\text{pk}' \leftarrow \text{MaulPK}(\text{pk}, \tau)$, and any $m \in \mathcal{M}$, let $\tilde{c} \leftarrow \text{Enc}(\text{pk}', m)$. Then

$$\text{AlignEnc}(c, \text{pk}', \tau) \equiv \text{Enc}(\text{pk}', m) \equiv \text{MaulCT}(c, \tau).$$

For the simplicity of notation, we will only use AlignEnc throughout the paper in the subsequent protocols we design.

Impossibility of 2-Round Sanitizable OLE from AHE We show that no 2-round AHE-based OLE protocol can *srUC*-realize this natural sanitizable

OLE functionality defined in Figure 7. The impossibility already appears in the weakest setting considered by the srUC framework: a *specious* environment and honest firewalls. In particular, the problem is not caused by malicious behavior or firewall subversion. Rather, the limitation is inherent to the protocol structure itself. Informally, in two rounds, independently sampled firewall randomness cannot reliably sanitize the sender’s coefficients inside a single ciphertext.

Main ideas behind the obstruction. The proof tracks how the plaintext evolves through the protocol. In any 2-round AHE-based OLE protocol, firewalls can only manipulate ciphertexts via the scheme’s homomorphic operations, so each induces an affine transformation on the plaintext. The ciphertext undergoes four successive transformations: (1) R encrypts x ; (2) F_R blinds it, yielding an encryption of $a''x + b''$; (3) S homomorphically evaluates $a(\cdot) + b$, giving $aa''x + ab'' + b$; (4) F_S blinds again, producing $aa'a''x + aa'b'' + a'b + b'$. On the return trip, F_R has one final chance to apply an affine transformation $z \mapsto \gamma z + \delta$ before decryption.

The core difficulty: the real plaintext does not match the ideal output. Following $\mathcal{F}_{\text{TOT}}$ from [CMMV24, Section 3.1], $\mathcal{F}_{\text{SOLE}}$ applies independent multiplicative masks, so with firewall masks (a', b') and (a'', b'') the sanitized coefficients are $\hat{a} = aa'a''$, $\hat{b} = bb'b''$, and the receiver obtains $y_{\text{ideal}} = aa'a'' \cdot x + bb'b''$ *fully factorized*: the masking of a never interacts with that of b . The real protocol, by contrast, accumulates mixed terms like $aa'b''$ and $a'b$. Matching real to ideal becomes a polynomial equality in (x, a, b) , forcing (γ, δ) into incompatible conditions. Two independent obstructions emerge:

1. The cross-term $aa'b''$ appears in the real output but not the ideal one. Removing it would require the final firewall transformation to depend on a , which is hidden by IND-CPA security.
2. The cross-term $a'b$ similarly appears only in the real output. Eliminating it would require F_R to know both b (hidden by encryption) and a' (known only to F_S , whose randomness is independent of F_R under srUC).

Conceptually the ideal functionality assumes perfectly factorized masking, while homomorphic affine composition necessarily mixes masks and secrets. Undoing the mixed terms requires information no single firewall possesses, and with only two rounds there is no way for firewalls to coordinate enough algebraic information.

Semantics of the blinding. Both firewalls blind the sender’s coefficients multiplicatively: $a \mapsto a \cdot a' \cdot a''$ and $b \mapsto b \cdot b' \cdot b''$, giving $\hat{a} = a \cdot a' \cdot a''$ and $\hat{b} = b \cdot b' \cdot b''$, so the receiver obtains $\hat{a} \cdot x + \hat{b} = aa'a'' \cdot x + bb'b''$.

When the firewalls’ contributions are dropped. $\mathcal{F}_{\text{SOLE}}$ disregards F_i ’s contribution if the entire party P_i is malicious per the corruption translation table of [CMNV22, Table 1]. This is reasonable under static corruption: each component’s status is fixed in advance, and a malicious party’s joint behavior need not be simulated internally.

The Class of 2-Round AHE-Based Protocols. We now formalize the class of protocols against which the impossibility is stated. Following [CMNV22, Definition 2], a *sanitizing protocol* Π for an n -party functionality consists of n cores

$\{C_1, \dots, C_n\}$ and n firewalls $\{F_1, \dots, F_n\}$, where each F_i mediates the communication of C_i on both inbound and outbound channels (with secure channels between a core and its own firewall).

Before formalizing the protocol class, we make precise the entropy requirement on the firewalls' blinding masks. Without some lower bound on the entropy of the masks, a firewall could trivially sample $(a', b') = (1, 0)$ (the identity transformation), in which case there would be no sanitation to speak of and the impossibility would be vacuous.

Definition 5 (Well-spread mask distribution). *A mask distribution $\mathcal{D}$ over the ring $\mathcal{R}$ is well-spread if its min-entropy is super-logarithmic in the security parameter, i.e.,*

$$H_\infty(\mathcal{D}) := -\log \max_{r \in \mathcal{R}} \Pr_{r^* \leftarrow \mathcal{D}}[r^* = r] = \omega(\log \lambda).$$

Equivalently, $\max_{r \in \mathcal{R}} \Pr_{r^* \leftarrow \mathcal{D}}[r^* = r] = \text{negl}(\lambda)$. A pair of mask distributions $(\mathcal{D}_a, \mathcal{D}_b)$ over $\mathcal{R}^2$ is well-spread if both marginals are well-spread.

The well-spreadness condition is the weakest non-triviality assumption possible: each firewall mask must be sampled with enough randomness that no specific value is predictable. This holds automatically if masks are sampled uniformly from $\mathcal{R}$ (which has super-polynomial cardinality), or from any distribution with super-polynomially large, roughly balanced support. The condition is necessary because the impossibility relies on mask unpredictability—a corrupted firewall that always samples $(1, 0)$ acts as no firewall at all, leaving no obstruction to overcome.

Definition 6 (2-round AHE-based sanitizing protocol). *A sanitizing protocol Π for OLE in the $\mathcal{F}_{\text{SAT}}$ -hybrid model is 2-round AHE-based with bilateral plaintext-masking firewalls if there exists an AHE scheme $(\text{KeyGen}, \text{Enc}, \text{Dec}, \text{Eval})$ over plaintext ring $\mathcal{R}$, and well-spread mask distributions $\mathcal{D}_S, \mathcal{D}_R$ over $\mathcal{R}^2$ (in the sense of Definition 5), such that:*

- (i) **Round 1 (receiver $\rightarrow$ sender):** C_R computes $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and $c_1 \leftarrow \text{Enc}(\text{pk}, x)$, and sends (pk, c_1) to F_R . F_R samples masks $(a'', b'') \leftarrow \mathcal{D}_R$, applies the homomorphic affine transformation $z \mapsto a''z + b''$ to c_1 (and possibly mauls pk), and forwards the result to F_S .
- (ii) **Round 2 (sender $\rightarrow$ receiver):** Upon receiving the round-1 message, C_S computes a ciphertext c_2 whose encoded plaintext is $a \cdot P_1 + b$, where P_1 is the plaintext underlying the round-1 ciphertext it received. C_S sends c_2 to F_S . F_S samples masks $(a', b') \leftarrow \mathcal{D}_S$, applies the homomorphic affine transformation $z \mapsto a'z + b'$ to c_2 , and forwards the result to F_R .
- (iii) **Receiver output:** F_R may apply an additional homomorphic affine transformation $z \mapsto \gamma z + \delta$ to the inbound round-2 ciphertext before forwarding it to C_R , who decrypts to obtain y .
- (iv) **Firewall independence:** F_S and F_R share no state or random coins; in particular, $\mathcal{D}_S$ and $\mathcal{D}_R$ are sampled independently.

This class captures the natural 2-round AHE-based template: the receiver encrypts, the sender homomorphically evaluates, and the firewalls sanitize by applying plaintext-level affine transformations to the in-transit ciphertexts. The plaintext-level affine form of the firewall operations is forced by the AHE: the only plaintext operations available via the homomorphic interface are affine. The protocol may otherwise maul public keys, rerandomized ciphertext noise, or perform any other ciphertext-level sanitization.

Theorem 1 (Impossibility of 2-Round Sanitizable OLE from AHE).

Let $\mathcal{R}$ be a finite ring with super-polynomial cardinality. Let Π be any 2-round AHE-based sanitizing protocol with bilateral plaintext-masking firewalls in the sense of [Definition 6](#). Then Π does not $\text{srUC-realize } \mathcal{F}_{\text{SOLE}}$. In particular, there exist a PPT environment $\mathcal{Z}$ and a PPT adversary $\mathcal{A}$ such that for every PPT simulator $\mathcal{S}$, we obtain $\left| \Pr[\text{Exec}_{\Pi, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} = 1] - \Pr[\text{Exec}_{\mathcal{F}_{\text{SOLE}}, \mathcal{S}, \mathcal{Z}} = 1] \right| \geq 1 - \text{negl}(\lambda)$, where $\mathcal{Z}$ uses only specious corruptions of cores and honest corruptions of firewalls.

The obstruction is a *correctness* obstruction: the real and ideal executions diverge even under honest firewalls, so no simulator can reconcile them. The proof of this theorem is postponed to [Appendix B.2](#)

4.3 3-Round Protocol Realizing Sanitizable OLE

In light of the impossibility result of the previous section, we move our attention to the 3-round setting. The extra round buys exactly the room that two rounds could not afford: a return trip on which firewalls can clean up the in-transit values without having to undo any algebraic mixing of the sender's inputs with their masks.

A Regular 3-Round OLE Protocol. Before turning to the sanitized protocol, we propose the standard non-sanitized three-round template, in which an AHE scheme is used in a natural way. The sender S publishes encryptions of its inputs. The receiver R homomorphically evaluates the affine map at its own input while adding a fresh random mask and storing it in its state. Then the sender decrypts and returns the masked value. The receiver removes the mask. Construction in [Figure 8](#) formalizes this idea.

Protocol: $\Pi_{\text{OLE}}^{(3)}$

Let AHE be a key-malleable AHE scheme as in [Definition 4](#). The protocol run between the sender S with input (a, b) and the receiver R with input x .

Round-1: The sender S runs $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$. Compute: $\langle a \rangle \leftarrow \text{Enc}(\text{pk}, a)$ and $\langle b \rangle \leftarrow \text{Enc}(\text{pk}, b)$. Send $(\text{pk}, \langle a \rangle, \langle b \rangle)$ to R .

Round-2: The receiver R does $r \xleftarrow{\$} \mathcal{M}$ and compute $\langle r \rangle \leftarrow \text{Enc}(\text{pk}, r)$, then compute $t = \text{CMult}(\langle a \rangle, x)$, $u = \text{HomOp}(t, \langle b \rangle)$, and $y = \text{HomOp}(u, \langle r \rangle)$. Send y to S

Round-3: The sender S decrypts $z = \text{Dec}(\text{sk}, y)$ and send z to R.
Output: R outputs $\text{ADD}(z, -r)$.

Fig. 8: 3-Round OLE Protocol.

Correctness of the protocol is immediate: by additive homomorphism, the message decrypted by the sender carries the desired OLE output plus the receiver's own mask, which the receiver then removes. As for security, the protocol UC-realizes the regular OLE functionality $\mathcal{F}_{\text{OLE}}$ against semi-honest adversaries, under the standard IND-CPA assumption on the underlying encryption scheme. We formally state [Theorem 2](#), the proof is deferred to [Appendix B.5](#).

Theorem 2. *If the key-malleable AHE is IND-CPA secure, then the protocol $\Pi_{\text{OLE}}^{(3)}$ of UC-realizes the ideal functionality $\mathcal{F}_{\text{OLE}}$ against semi-honest adversary.*

Sanitizing 3-Round OLE Protocol. The sanitizing OLE protocol introduces two layers of cleanup. In the first round, both firewalls maul the receiver's public key into a fresh, independent one and re-randomize the in-transit ciphertexts using the key-malleable properties of AHE defined in the [Definition 4](#). No subliminal information that the sender's core may have planted into its key or into its encryptions can survive this cleanup. In the second and third rounds, each firewall additively masks the OLE output with a fresh random value, and then strips that mask on its way back to the receiver. The protocol is depicted in [Figure 9](#). We formally state the security [Theorem 3](#). The proof is postponed to [Appendix B.7](#).

Protocol: $\widehat{\Pi}_{\text{OLE}}^{(3)}$

The protocol is executed in the $\mathcal{F}_{\text{SAT}}$ -hybrid model between a sender $S = (C_S, F_S)$ and a receiver $R = (C_R, F_R)$. Let AHE be a key malleable, additive homomorphic encryption scheme as in [Definition 4](#). The protocol runs between the sender S with input (a, b) and the receiver R with input x .

Round 1:

- C_S runs $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$ and encrypts a and b : $\langle a \rangle \leftarrow \text{Enc}(\text{pk}, a)$ and $\langle b \rangle \leftarrow \text{Enc}(\text{pk}, b)$ and sends $(\text{pk}, \langle a \rangle, \langle b \rangle)$ to F_S .
- F_S samples $a' \xleftarrow{\$} \mathbb{F}$ and compute $\langle a' \rangle \leftarrow \text{Enc}(\text{pk}, a')$. Then does $(\widetilde{\text{pk}}, \tau) \leftarrow \text{MaulPK}(\text{pk})$. Then it computes $\langle \widetilde{a} \rangle = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(\langle a \rangle, \langle a' \rangle), \widetilde{\text{pk}}, \tau))$ and $\langle \widetilde{b} \rangle = \text{Rerand}(\text{AlignEnc}(\langle b \rangle, \text{pk}, \tau))$. Forward $(\widetilde{\text{pk}}, \langle \widetilde{a} \rangle, \langle \widetilde{b} \rangle)$ to F_R .
- F_R samples $a'' \xleftarrow{\$} \mathbb{F}$ and subsequently computes $\langle \widetilde{a''} \rangle \leftarrow \text{Enc}(\widetilde{\text{pk}}, a'')$ and $(\widehat{\text{pk}}, \rho) \leftarrow \text{MaulPK}(\widetilde{\text{pk}})$ and $\langle \widehat{a} \rangle = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(\langle \widetilde{a} \rangle, \langle \widetilde{a''} \rangle), \widehat{\text{pk}}, \rho))$ and $\langle \widehat{b} \rangle = \text{Rerand}(\text{AlignEnc}(\langle \widetilde{b} \rangle, \widetilde{\text{pk}}, \rho))$. Forward $(\widehat{\text{pk}}, \langle \widehat{a} \rangle, \langle \widehat{b} \rangle)$ to C_R .

Round 2:

- C_R samples $r \xleftarrow{\$} \mathcal{M}$ and computes $\langle \widehat{r} \rangle \leftarrow \text{Enc}(\widehat{\text{pk}}, r)$. Evaluate $t = \text{CMult}(\langle \widehat{a} \rangle, x)$, and $u = \text{HomOp}(t, \langle \widehat{b} \rangle)$. Compute $\widehat{y} = \text{HomOp}(u, \langle \widehat{r} \rangle)$. Send $\widehat{y}$ to F_R .

- F_R samples $r'' \xleftarrow{\$} \mathcal{M}$ and computes $\widehat{\langle r'' \rangle} \leftarrow \text{Enc}(\widehat{\text{pk}}, r'')$. It masks the reply under $\widehat{\text{pk}}$ as $\widehat{y}^* = \text{HomOp}(\widehat{y}, \widehat{\langle r'' \rangle})$. Then it computes $\widetilde{y} = \text{Rerand}(\text{AlignEnc}(\widehat{y}^*, \widehat{\text{pk}}, \rho))$ and forwards $\widetilde{y}$ to F_S .
- F_S samples $r' \xleftarrow{\$} \mathcal{M}$ and computes $\widetilde{\langle r' \rangle} \leftarrow \text{Enc}(\widetilde{\text{pk}}, r')$. It masks the reply under $\widetilde{\text{pk}}$ as $\widetilde{y}^* = \text{HomOp}(\widetilde{y}, \widetilde{\langle r' \rangle})$. Then it computes $y = \text{Rerand}(\text{AlignEnc}(\widetilde{y}^*, \widetilde{\text{pk}}, \tau))$ and forwards y to C_S .

Round 3:

- C_S decrypts and obtains $z = \text{Dec}(\text{sk}, y)$. Then it relays z to F_S .
- F_S samples $b' \xleftarrow{\$} \mathbb{F}$, removes its own mask r' and applies its plaintext mask: $\widetilde{z} = \text{ADD}(\text{ADD}(z, -r'), b')$. It sends $\widetilde{z}$ to F_R .
- F_R samples $b'' \xleftarrow{\$} \mathbb{F}$, removes its own mask r'' and applies its plaintext mask: $\widehat{z} = \text{ADD}(\text{ADD}(\widetilde{z}, -r''), b'')$. It sends $\widehat{z}$ to C_R .

Output: C_R outputs: $\text{ADD}(\widehat{z}, -r)$

Fig. 9: 3-Round Sanitizing OLE Protocol $\widehat{\Pi}_{\text{OLE}}^{(3)}$ Realizing $\mathcal{F}_{\text{SOLE}}$

Theorem 3. *If the key-malleable AHE is IND-CPA secure, then $\widehat{\Pi}_{\text{OLE}}^{(3)}$ srUC-realizes $\mathcal{F}_{\text{SOLE}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model against semi-honest adversary.*

Proof Overview of Theorem 3. It will be sufficient to write simulator for any adversary $\mathcal{A}$ that corrupting one party and firewall of the other party; along with showing that the firewalls are strongly sanitizing. *Corrupt sender.* $\mathcal{S}$ knows (a, b) , masks (a', b') , (a'', b'') of firewall and the return-trip masks $r', r'' \in \mathcal{M}$ sampled inside the two firewalls. $\mathcal{S}$ forwards them to $\mathcal{F}_{\text{SOLE}}$ and through BLIND commands. $\mathcal{S}$ simulates only the round-2 ciphertext y : sample $z^* \xleftarrow{\$} \mathbb{F}$, set $y \leftarrow \text{Enc}(\text{pk}_c, z^*)$, run firewall code, and emit $z = z^* + r' + r''$ as the round-3 plaintext. The real plaintext $(a + a' + a'')x + b + r$ is uniform because r is, and the sender decrypts it shifted by $r' + r''$; distributions match. *Corrupt receiver.* $\mathcal{S}$ knows x, r , both firewall masks together with their return-trip masks r', r'' , not (a, b) . It queries $\mathcal{F}_{\text{SOLE}}$ on the receiver's interface, obtains the target output $\widehat{y} = \widehat{a}x + \widehat{b}$. $\mathcal{S}$ generates a fresh key pair, sets $\langle a \rangle \leftarrow \text{Enc}(\text{pk}, 0)$ and $\langle b \rangle \leftarrow \text{Enc}(\text{pk}, 0)$ i.e., two fresh encryptions of 0, runs the firewall code through rounds 1 and 2, and emits $z \leftarrow \widehat{y} - b' - b'' + r + r' + r''$ as the round-3 plaintext. Substitution gives $z = (a + a' + a'')x + b + r + r' + r''$, the exact real-world decryption; the firewall chain then strips r' and r'' and yields $\widehat{z} = \widehat{y} + r$, so the receiver's local computation $\text{ADD}(\widehat{z}, -r) = \widehat{y}$ matches the ideal. *Strongly sanitizing firewalls.* Each firewall touches the wire at three points. Outbound key: MaulPK output is statistically fresh by (KM3), destroying subliminal patterns in pk . Outbound/inbound ciphertexts: ReRand + AlignEnc make every aligned ciphertext statistically close to a fresh encryption of the same plaintext, eliminating both leakage and trigger conditions on the bit-level encoding; on the return trip the fresh mask $r' \xleftarrow{\$} \mathcal{M}$ additionally blinds the plaintext carried by the reply. Round-3 plaintext: the firewall first removes its

own mask r' , and then the fresh additive mask $b' \xleftarrow{\$} \mathbb{F}$ overwrites z information-theoretically. The symmetric argument handles F_R .

Application: Randomized Equality from Sanitizable OLE. As an application of $\mathcal{F}_{\text{SOLE}}$ we discuss the possibility of Randomized Equality built from $\mathcal{F}_{\text{SOLE}}$. Each party hashes its password into the field, $w^I = \text{Hash}(\text{pw}^I)$, $w^R = \text{Hash}(\text{pw}^R)$, with $|\mathbb{F}| \geq 2^n$, and samples a uniform scalar $a^I, a^R \xleftarrow{\$} \mathbb{F}$. Two symmetric $\mathcal{F}_{\text{SOLE}}$ calls (with swapped roles) are made, C_I obtains $v^I = (a^R + a' + a'')w^I + b' + b''$ and C_R obtains $v^R = (a^I + a' + a'')w^R$. Each party finalizes

$$\begin{aligned} \text{skey}^I &:= a^I w^I + v^I = (a^I + a^R + a' + a'')w^I + b' + b'' \\ \text{skey}^R &:= a^R w^R + v^R = (a^I + a^R + a' + a'')w^R + b' + b''. \end{aligned}$$

When $w^I = w^R$ the keys match; otherwise the difference is $(a^I + a^R + a' + a'')(w^I - w^R)$ which is uniform in $\mathbb{F}^*$ since at least one of a^I, a^R is uniform and a', a'' are uniform firewall masks. A detailed discussion on Randomized Equality is postponed to Appendix C.

4.4 2-Round Sanitizing OLE Protocol

OLE underlies a wide range of cryptographic protocols. Its *sanitizable* counterpart, however, has limited reach: the firewalls' two-sided masking breaks the sender-receiver correlation that higher-level protocols rely on, making derandomization and composition syntactically awkward. Broader applications of $\mathcal{F}_{\text{SOLE}}$ are left as a future direction. We therefore study a restricted setting, where a *sanitizing* realization of the regular OLE functionality suffices.

Regular 2-Round OLE Protocol. The starting point of this section will be realizing the regular OLE functionality $\mathcal{F}_{\text{OLE}}$ in two rounds. It will incur less communication cost when used in other protocols. We start by recalling the familiar 2-round template. We begin with the textbook 2-round OLE protocol: Sender S with (a, b) and receiver R with x participates in the protocol with access to a AHE scheme. In Round-1, R runs $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$. Compute $\langle x \rangle \leftarrow \text{Enc}(\text{pk}, x)$. Send $(\text{pk}, \langle x \rangle)$ to S. In Round-2, S encrypts b with pk : $\langle b \rangle \leftarrow \text{Enc}(\text{pk}, b)$ then computes: $y = \text{HomOp}(a\langle x \rangle, \langle b \rangle)$ homomorphically and send y to R. Finally, R outputs $\text{Dec}(\text{sk}, y)$. This 2-round generic OLE construction $\Pi_{\text{OLE}}^{(2)}$ is depicted in Figure 17. The protocol $\Pi_{\text{OLE}}^{(2)}$ correctly UC-realizes $\mathcal{F}_{\text{OLE}}$ against semi-honest adversaries under the IND-CPA assumption on the underlying encryption scheme, as captured by Theorem 4 below, proof of which is obvious, for the sake of completeness the proof is deferred to the Appendix B.9.

Theorem 4. *If the key-malleable AHE scheme is IND-CPA secure, then the protocol $\Pi_{\text{OLE}}^{(2)}$ UC-realizes $\mathcal{F}_{\text{OLE}}$ against semi-honest corruptions.*

Sanitizing 2-Round OLE Protocol. We lift this to the subversion-resilient setting. We do not realize $\mathcal{F}_{\text{SOLE}}$, which is impossible in two rounds. Instead, we realize $\mathcal{F}_{\text{OLE}}$ in the wsrUC sense, where the wrapper absorbs the firewall layer and

firewalls neutralize subliminal channels rather than inject independent masking. The sanitized protocol layers key-mauling and ciphertext re-randomization on the plain template, applied symmetrically on the forward and return trips. In round one, the receiver's and sender's firewalls successively maul the public key and align the ciphertext with each new key; by key-malleability, every mauled key is statistically independent of its predecessor, washing out any subliminal information the receiver's core planted in its key or ciphertext. In round two, the firewalls reverse the process, aligning the sender's reply back to the receiver's original key and rerandomize the ciphertext to wash out any injected covert randomness.

Protocol: $\widehat{\Pi}_{\text{OLE}}^{(2)}$

Let AHE be a key-malleable additive homomorphic encryption scheme as in [Definition 4](#). The protocol is executed in the $\mathcal{F}_{\text{SAT}}$ -hybrid model between a sender $S = (C_S, F_S)$ and a receiver $R = (C_R, F_R)$.

Round 1: $C_R: (pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$ and compute $\langle x \rangle = \text{Enc}(pk, x)$. Send $(pk, \langle x \rangle)$ to F_R . Then, F_R computes $(pk', \rho) \leftarrow \text{MaulPK}(pk)$ and $\langle x \rangle' = \text{Rerand}(\text{AlignEnc}(\langle x \rangle, pk', \rho))$. Forward $(pk', \langle x \rangle')$ to F_S . Finally, F_S computes $(\widetilde{pk}, \tau) \leftarrow \text{MaulPK}(pk')$ and $\widetilde{\langle x \rangle} = \text{Rerand}(\text{AlignEnc}(\langle x \rangle', \widetilde{pk}, \tau))$. Send $(\widetilde{pk}, \widetilde{\langle x \rangle})$ to C_S .

Round 2: C_S computes $\widetilde{\langle b \rangle} \leftarrow \text{Enc}(\widetilde{pk}, b)$, $t = \text{CMult}(\widetilde{\langle x \rangle}, a)$, and $\widetilde{z} = \text{HomOp}(t, \widetilde{\langle b \rangle})$. Send $\widetilde{z}$ to F_S . Then, F_S computes $z' = \text{Rerand}(\text{AlignEnc}(\widetilde{z}, pk', \rho))$. Forward z' to F_R . Finally, F_R computes $z = \text{Rerand}(\text{AlignEnc}(z', pk, \gamma))$. Relay z to C_R .

Output: The core C_R outputs $\text{Dec}(sk, z)$.

Fig. 10: 2-Round Sanitizing OLE Protocol

We formally state, [Theorem 5](#), the proof is postponed to [Appendix B.11](#).

Theorem 5. *If key-malleable AHE is IND-CPA secure, then the $\widehat{\Pi}_{\text{OLE}}^{(2)}$ protocol wsrUC -realizes $\mathcal{F}_{\text{OLE}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model.*

Proof Overview of Theorem 5. We will show that the protocol $\widehat{\Pi}_{\text{OLE}}^{(2)}$ realizes $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ for both corruption cases. The adversary corrupts one party and the firewall of the other party. *Corrupt sender.* A three-hybrid argument moves from the real encryption $\text{Enc}(pk', x)$ to $\text{Enc}(pk', \widetilde{x})$ for uniform $\widetilde{x}$, via one IND-CPA hop on the freshly distributed key pk' produced by MaulPK . *Corrupt receiver.* $\mathcal{S}$ obtains $z^* = ax + b$ from $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ and re-encrypts $\widetilde{z}' \leftarrow \text{Enc}(pk', z^*)$. In the real protocol, the sender's $\widetilde{z}$ is computed homomorphically as $\text{HomOp}(\text{CMult}(\widetilde{\langle x \rangle}, a), \text{Enc}(pk_f, b))$, which by the homomorphic correctness of AHE plus the fresh randomness contributed by $\text{Enc}(pk_f, b)$ is distributed as a fresh encryption of $ax + b$ under pk_f . Applying MaulCT and ReRand on the return trip yields a ciphertext statistically close to a fresh $\text{Enc}(pk', ax + b)$, exactly matching the simulator's choice. *Strongly sanitizing firewalls.* By definition, a SPECIOUS core is computationally indistinguishable from the honest one

under any efficient black-box test. (KM3) in [Definition 4](#) makes every MaulPK output statistically close to a fresh KeyGen output. (KM2)/(KM4) in [Definition 4](#) combined with ReRand make every transformed ciphertext statistically close to a fresh encryption of the same plaintext.

4.5 Instantiation with Class Group

Building on the foundations developed in [\[CL15\]](#), [\[CCL⁺19\]](#), and [\[KMM⁺24\]](#), we briefly review the background of class-groups of imaginary quadratic fields before revisiting the encryption scheme.

Group Structure. The class group framework considers a finite abelian group $\widehat{\mathbb{G}}_{\text{CL}}$ of unknown order $q \cdot \widehat{s}$, where q is a publicly known prime, while $\widehat{s}$ is secret and co-prime to q . Let $\mathbb{F}_{\text{CL}} = \langle f \rangle$ denote a cyclic subgroup of $\widehat{\mathbb{G}}_{\text{CL}}$ of order q . The set $\widehat{\mathbb{G}}_{\text{CL}}^q = \{g^q : g \in \widehat{\mathbb{G}}_{\text{CL}}\}$, obtained as the image of the q -power map, forms a subgroup of $\widehat{\mathbb{G}}_{\text{CL}}$ of order $\widehat{s}$. As $\widehat{s}$ and q are co-prime, the group $\widehat{\mathbb{G}}_{\text{CL}}$ admits the decomposition $\widehat{\mathbb{G}}_{\text{CL}} \simeq \mathbb{F}_{\text{CL}} \times \widehat{\mathbb{G}}_{\text{CL}}^q$. We further assume that $U \in \mathbb{Z}$ is a known upper bound on $\widehat{s}$. Rather than working directly with $\widehat{\mathbb{G}}_{\text{CL}}$, we consider a cyclic subgroup $\mathbb{G}_{\text{CL}}$ of $\widehat{\mathbb{G}}_{\text{CL}}$ of order $q \cdot s$, where s is an unknown divisor of $\widehat{s}$. Consequently, $\mathbb{F}_{\text{CL}}$ also serves as a cyclic subgroup of $\mathbb{G}_{\text{CL}}$, and the corresponding q -power image $\mathbb{G}_{\text{CL}}^q = \{g^q : g \in \mathbb{G}_{\text{CL}}\}$ forms a cyclic subgroup of order s . Since q and s are likewise co-prime, we obtain the direct product decomposition $\mathbb{G}_{\text{CL}} \simeq \mathbb{F}_{\text{CL}} \times \mathbb{G}_{\text{CL}}^q$. We additionally assume that both s and $\widehat{s}$ are odd, and that each of $s, \widehat{s}, q$ grows exponentially with respect to the security parameter λ . Moreover, despite the group orders being unknown, we assume that group operations such as exponentiation and inversion can still be performed efficiently.

Algorithmic Framework. The feasibility of this framework stems from the fact that although the discrete logarithm problem is computationally hard in the groups $\mathbb{G}_{\text{CL}}, \mathbb{G}_{\text{CL}}^q$ and their corresponding q -power subgroups $\widehat{\mathbb{G}}_{\text{CL}}^q, \mathbb{G}_{\text{CL}}^q$, it can be solved efficiently in $\mathbb{F}_{\text{CL}}$. More specifically, the following two efficient algorithms are available: (i) $(U, \mathbb{G}_{\text{CL}}, \mathbb{F}_{\text{CL}}, f, g_q, \mathcal{D}, \mathcal{D}_q, \rho) \leftarrow \text{CGen}(1^\lambda, 1^{\lambda_{st}}, q)$. Given the computational security parameter λ , the statistical security parameter λ_{st} , and a prime q , this algorithm outputs the class group parameters together with the randomness ρ employed during their generation. For simplicity, we also include the specifications of the distributions $\mathcal{D}$ and $\mathcal{D}_q$, which are defined below. (ii) $x \leftarrow \text{Solve}(X = f^x, U, q, \widehat{\mathbb{G}}_{\text{CL}}, \mathbb{F}_{\text{CL}}, g_q, f)$. This algorithm deterministically and efficiently computes the discrete logarithm in the group $\mathbb{F}_{\text{CL}}$.

Sampling Distributions. We consider two efficiently samplable distributions $\mathcal{D}$ and $\mathcal{D}_q$ over $\mathbb{Z}$ such that $\{g^x \mid x \leftarrow \mathcal{D}\}$ and $\{g_q^x \mid x \leftarrow \mathcal{D}_q\}$ produce distributions over $\mathbb{G}_{\text{CL}}$ and $\mathbb{G}_{\text{CL}}^q$, respectively, which are statistically close (within distance $2^{-\lambda_{st}}$, for a statistical security parameter λ_{st} , typically set to 40 in practice) to uniform distributions over $\mathbb{G}_{\text{CL}}$ and $\mathbb{G}_{\text{CL}}^q$ respectively.

Scheme: CG.AHE	
CG.KeyGen(pp_{CG}): – Sample $\text{sk} \leftarrow \mathcal{D}_q$. – Sample $\text{pk} := g_q^{\text{sk}}$. – Output (sk, pk) .	ReRand(pk, c) for $c = (R, E)$: – Sample $r \leftarrow \mathcal{D}_q$. – $c_0 = \text{CG.Enc}(\text{pk}, 0) = (g_q^r, \text{pk}^r)$. – Output $\text{HomOp}(c, c_0)$.
CG.Enc(pk, m) for $m \in \mathbb{Z}_p$: – Sample $r \leftarrow \mathcal{D}_q$. – $R = g_q^r, E = f^m \text{pk}^r$. – Output $c := (R, E)$.	MaulPK(pk) for $\text{pk} = g_q^{\text{sk}}$: – Sample $r' \leftarrow \mathcal{D}_q$. – $\alpha = g_q^{r'}$. – Output $\text{pk}' := \text{pk} \cdot \alpha, \tau = r'$.
CG.Dec(sk, c): – Parse $c = (R, E)$; $M = E \cdot (R^{\text{sk}})^{-1}$. – Solve $m = \text{Solve}(\text{pp}_{\text{CG}}, M)$. – Output m .	MaulSK(sk, τ) for $\text{sk} \leftarrow \mathcal{D}_q, \tau = r'$: – Output $\text{sk}' := \text{sk} + r'$.
HomOp(c_1, c_2): – Parse $c_1 = (R_1, E_1), c_2 = (R_2, E_2)$. – $\text{HomOp}((R_1, E_1), (R_2, E_2)) := (R_1 R_2, E_1 E_2)$.	MaulCT(c, τ) for $c = (R, E), \tau = r'$: – $E' = E \cdot R^{r'}$. – $c'_0 = \text{CG.Enc}(\text{pk}', 0)$. – Output $\tilde{c} = \text{HomOp}(c, c'_0)$.
CMult(c, α): – Parse $c = (R, E)$. – $\text{CMult}((R, E), \alpha) := (R^\alpha, E^\alpha)$.	AlignEnc($\tilde{c}, \text{pk}, \tau$) for $\tilde{c} = (\tilde{R}, \tilde{E}), \tau = r'$: – $E' = \tilde{E} \cdot \tilde{R}^{-r'}$. – $c_0 \leftarrow \text{CG.Enc}(\text{pk}, 0)$. – Output $c' = \text{HomOp}((\tilde{R}, E'), c_0)$.

Fig. 11: Key-malleable AHE from Class Group

We discuss the correctness and security guarantee of Class Group based Key-malleable AHE in Appendix D. The OLE protocols, $\Pi_{\text{OLE}}^{(2)}$, and $\Pi_{\text{OLE}}^{(3)}$ along with their sanitizing variants, can be instantiated by the construction stated in Figure 11 in a natural way. Instantiation with post-quantum encryption schemes based on Lattices is not obvious, we leave it as a possible future work.

5 Applications of Sanitizing OLE

5.1 Oblivious Polynomial Evaluation

In the OPE setting, the party A holds a degree- d polynomial $\mathbf{p}(X) = \sum_{j=0}^d a_j X^j \in \mathbb{F}[X]$, while the party B holds an evaluation point $\alpha \in \mathbb{F}$. At the end of the protocol, B learns only $\mathbf{p}(\alpha)$, whereas A obtains no output. Following the ideas of [NP99, Gil01], our protocol Π_{OPE} in Figure 18 realizes OPE via a Horner-style decomposition using d sequential invocations of $\mathcal{F}_{\text{OLE}}$. The protocol maintains additive shares $s_i^A, s_i^B \in \mathbb{F}$ satisfying the invariant $s_i^A + s_i^B = a_d \alpha^i + a_{d-1} \alpha^{i-1} + \dots + a_{d-i}$ for every $i \in [d-1]$. In the i -th invocation of OLE, party A inputs $(A_i, B_i) = (s_{i-1}^A, a_{d-i} - \rho_i)$, where $\rho_i \xleftarrow{\$} \mathbb{F}$, while party B inputs α and receives $y_i = s_{i-1}^A \alpha + a_{d-i} - \rho_i$, after which it updates $s_i^B \leftarrow y_i + \alpha s_{i-1}^B$. In the final invocation, A inputs $(A_d, B_d) = (s_{d-1}^A, a_0)$ and B

inputs α , obtaining $y_d = A_d\alpha + B_d$ and outputting $y_d + \alpha s_{d-1}^B = \mathbf{p}(\alpha)$. We modify the final $\mathcal{F}_{\text{OLE}}$ invocation so that Π_{OPE} uses only black-box access to $\mathcal{F}_{\text{OLE}}$, without requiring to send any additional messages to the receiver. For readability, we defer the complete protocol specification to Appendix E.1. We formally state the security of our OPE protocol in Theorem 6, the proof is deferred to Appendix F.2.

Theorem 6. *The Protocol Π_{OPE} UC-realizes $\mathcal{F}_{\text{OPE}}$ in $\mathcal{F}_{\text{OLE}}$ -hybrid model against semi-honest adversary.*

Sanitizing OPE. The protocol $\hat{\Pi}_{\text{OPE}}$ is obtained from Π_{OPE} by replacing each of the d ideal $\mathcal{F}_{\text{OLE}}$ calls with an invocation of the wsrUC-secure OLE protocol $\hat{\Pi}_{\text{OLE}}^{(2)}$. For $i \in [d]$, in the i -th OLE sub-instance: Party A's core acts as a sender (S) and feeds the input (A_i, B_i) , where $(A_i, B_i) = (s_{i-1}^A, a_{d-i} - \rho_i)$ for $i \in [d-1]$, and $(A_d, B_d) = (s_{d-1}^A, a_0)$. Bob's core acts as a receiver (R) and feeds the input α . Let F_S^i and F_R^i denote the sender- and receiver-side firewalls of the i -th $\hat{\Pi}_{\text{OLE}}$ sub-protocol. We define the parallel composition of the per-instance firewalls as

$$F_A^{\text{OPE}} := F_S^1 \diamond F_S^2 \diamond \cdots \diamond F_S^d, \quad F_B^{\text{OPE}} := F_R^1 \diamond F_R^2 \diamond \cdots \diamond F_R^d,$$

where “ $\diamond$ ” denotes parallel composition: the composed firewall has one sanitization interface S^i per OLE sub-instance i . On input addressed to instance i , F_A^{OPE} (resp. F_B^{OPE}) behaves exactly as F_S^i (resp. F_R^i), without sharing any state across instances. Equivalently, F_A^{OPE} and F_B^{OPE} run independent copies of the per-instance firewalls with the d OLE sub-protocols. each sub-firewall sanitizes its own sub-transcript, and the OPE-level view of the environment is the joint view across the d instances. The complete proof of Theorem 7 is postponed to Appendix F.3.

Theorem 7. *There exists a protocol $\hat{\Pi}_{\text{OPE}}$ that wsrUC-realizes $\mathcal{F}_{\text{OPE}}$ in $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model.*

Proof Overview of Theorem 7. Define hybrids $H_0, \dots, H_d$ where H_i replaces the first i invocations of $\hat{\Pi}_{\text{OLE}}^{(2)}$ inside $\hat{\Pi}_{\text{OPE}}$ by ideal $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ calls. Theorem 5 gives $H_{i-1} \stackrel{c}{\approx} H_i$ for each i ; chaining yields $H_0 \stackrel{c}{\approx} H_d$. Up to wrapper bookkeeping, H_d is an execution of Π_{OPE} in the $\mathcal{F}_{\text{OLE}}$ -hybrid model, for which Theorem 6 already supplies a UC simulator. The composed simulator is the obvious concatenation. *Strong sanitation of firewalls* lifts by decomposing the composite OPE core into d parallel OLE sub-cores: a SPECIOUS composite induces SPECIOUS marginals on the sub-cores, per-instance strong sanitation comes from Theorem 5, and a d -step hybrid, as the parallel composition $\diamond$ shares no state across instances, yields joint strong sanitation.

5.2 Oblivious Polynomial Addition

We recall the OPA functionality in Figure 5. Party A holds a polynomial $\mathbf{a}$ of degree at most $2d$ and party B holds $\mathbf{b}$ of degree at most d ; then B learns

$\mathbf{r} \cdot \mathbf{b} + \mathbf{a}$ for a fresh random masking polynomial $\mathbf{r}$ of degree at most d chosen by the sender, and nothing else about $\mathbf{a}$. The protocol Π_{OPA} realizes oblivious polynomial addition in the $\mathcal{F}_{\text{OLE}}$ -hybrid model between A holding $\mathbf{a}(X) \in \mathbb{F}[X]$ with $\deg(\mathbf{a}) \leq 2d$ and B holding $\mathbf{b}(X) \in \mathbb{F}[X]$ with $\deg(\mathbf{b}) \leq d$, over distinct public points $\alpha_1, \dots, \alpha_{2d+1} \in \mathbb{F}$. Party A samples a random masking polynomial $\mathbf{r}(X) \xleftarrow{\$} \mathbb{F}^{\leq d}[X]$ and, for each $i \in [2d+1]$, computes $a_i := \mathbf{a}(\alpha_i)$ and $r_i := \mathbf{r}(\alpha_i)$, invoking $\mathcal{F}_{\text{OLE}}$ with sender input (r_i, a_i) . In parallel, for each $i \in [2d+1]$, party B computes $b_i := \mathbf{b}(\alpha_i)$ and invokes $\mathcal{F}_{\text{OLE}}$ with receiver input b_i , obtaining $c_i = r_i b_i + a_i$. Finally, B interpolates and outputs the unique polynomial $\mathbf{c}(X) \in \mathbb{F}[X]$ of degree at most $2d$ satisfying $\mathbf{c}(\alpha_i) = c_i$ for all $i \in [2d+1]$. The mask $\mathbf{r}$ prevents party B from subtracting its own input to recover $\mathbf{a}$. The protocol is depicted in Figure 19. We formally state Theorem 8, the proof is postponed to Appendix F.5.

Theorem 8. *The protocol Π_{OPA} UC-realizes the $\mathcal{F}_{\text{OPA}}$ in the $\mathcal{F}_{\text{OLE}}$ -hybrid model.*

Sanitizing OPA. We lift Π_{OPA} to the subversion-resilient setting. Since Π_{OPA} uses $\mathcal{F}_{\text{OLE}}$ as its only cryptographic primitive, it suffices to replace each of the $2d+1$ ideal calls with a sanitizing OLE protocol and compose the per-instance firewalls in parallel following the same template for $\widehat{\Pi}_{\text{OPE}}$. As all $\mathcal{F}_{\text{OLE}}$ sub-calls are independent, no shared state is needed: each sub-firewall sanitizes its own sub-transcript, and the OPA-level view of the environment is the joint view across the $2d+1$ instances.

Protocol. The protocol $\widehat{\Pi}_{\text{OPA}}$ is obtained from Π_{OPA} by replacing each of the $2d+1$ ideal $\mathcal{F}_{\text{OLE}}$ calls with an invocation of the wsrUC-secure OLE protocol $\widehat{\Pi}_{\text{OLE}}$ in Figure 10. For $i \in [2d+1]$, the party A as a sender in OLE feeds $(r_i, a_i) := (\mathbf{r}(\alpha_i), \mathbf{a}(\alpha_i))$ and party B as a receiver feeds $b_i := \mathbf{b}(\alpha_i)$ into the i -th internal OLE instance. Let $\mathbf{F}_S^i$ and $\mathbf{F}_R^i$ denote the sender- and receiver side firewalls of that i -th OLE instance. We define the *parallel composition* of these per-instance firewalls as,

$$\mathbf{F}_A^{\text{OPA}} := \mathbf{F}_S^1 \diamond \mathbf{F}_S^2 \diamond \dots \diamond \mathbf{F}_S^{2d+1}, \quad \mathbf{F}_B^{\text{OPA}} := \mathbf{F}_R^1 \diamond \mathbf{F}_R^2 \diamond \dots \diamond \mathbf{F}_R^{2d+1},$$

where “ $\diamond$ ” denotes parallel composition: the composed firewall has one sanitization interface $\mathbf{S}^i$ per OLE instance i , and on input addressed to instance i behaves exactly as $\mathbf{F}_S^i$ (resp. $\mathbf{F}_R^i$), without exchanging any state across instances. Equivalently, $\mathbf{F}_A^{\text{OPA}}$ and $\mathbf{F}_B^{\text{OPA}}$ run independent copies of the per-instance firewalls with the $2d+1$ OLE sub-protocols. The proof of the Theorem 9 is postponed to Appendix F.6.

Theorem 9. *The protocol $\widehat{\Pi}_{\text{OPA}}$ that wsrUC-realizes $\mathcal{F}_{\text{OPA}}$ in $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model.*

6 Sanitizing Private Set Intersection

6.1 Subversion-Resilient PSI from OPE

PSI from OPE. One-sided Private Set Intersection obtained from OPE as follows: the party A encodes her set $S_A = \{\gamma_1, \dots, \gamma_m\}$ as the vanishing polynomial

$\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$, and for each element $\beta_j \in S_B$ the parties invoke one instance of $\mathcal{F}_{\text{OPE}}$ to compute $r_j \cdot \mathbf{p}_A(\beta_j)$, where $r_j \xleftarrow{\$} \mathbb{F} \setminus \{0\}$ is sample randomly in each invocation; the party B adds β_j to the intersection set if and only if $r_j \cdot \mathbf{p}_A(\beta_j) = 0$. We call this protocol Π_{PSI} . We state the relevant results and discuss the proofs formally in Appendix G. In particular, the proof of [Theorem 10](#) is available in Appendix G.2.

Theorem 10. *The protocol Π_{PSI} UC-realizes $\mathcal{F}_{\text{PSI}}$ in the $\mathcal{F}_{\text{OPE}}$ -hybrid model.*

Sanitizing PSI from Sanitized OPE. We lift Π_{PSI} to the subversion-resilient setting by replacing each of the n $\mathcal{F}_{\text{OPE}}$ calls with the sanitizing OPE protocol $\widehat{\Pi}_{\text{OPE}}$, with party-level firewalls obtained as the parallel composition of the per-instance OPE firewalls. Since the only cross-party messages are OPE sub-transcripts, strong sanitation of the PSI firewalls reduces to that of the underlying OPE firewalls.

Protocol. We obtain $\widehat{\Pi}_{\text{PSI}}$ from Π_{PSI} by replacing each $\mathcal{F}_{\text{OPE}}$ call with $\widehat{\Pi}_{\text{OPE}}$, indexed by $j \in [n]$. In j -th OPE instance, party A is acting as the OPE sender with input $\mathbf{p}_A(X)$ and party B acts OPE receiver with input β_j . Let $F_S^{(j)}, F_R^{(j)}$ denote the sender- and receiver-side firewalls of the j -th $\widehat{\Pi}_{\text{OPE}}$ invocation. The PSI firewalls are

$$F_A^{\text{PSI}} := F_S^{(1)} \diamond \dots \diamond F_S^{(n)}, \quad F_B^{\text{PSI}} := F_R^{(1)} \diamond \dots \diamond F_R^{(n)},$$

where “ $\diamond$ ” is parallel composition: one sanitization interface S^j per sub-instance, behaving as $F_S^{(j)}$ (resp. $F_R^{(j)}$) on inputs addressed to j -th OPE call, with no state shared across instances. The one-sided OPE-based PSI exchanges no further communication at the PSI layer beyond the OPE sub-transcripts: party A feeds $\mathbf{p}_A$ to every instance and receives $\perp$, while B feeds $\beta_1, \dots, \beta_n$ and locally assembles the intersection from $z_1, \dots, z_n$. The complete proof of [Theorem 11](#) is postponed to Appendix I.

Theorem 11. *The protocol $\widehat{\Pi}_{\text{PSI}}$ that wsrUC-realizes $\mathcal{F}_{\text{PSI}}$ in $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}})$ -hybrid model.*

Proof Overview for [Theorem 11](#). The one-sided OPE-based PSI exchanges *no additional message at the PSI layer beyond the OPE sub-transcripts*. Every channel a subverted PSI core could exploit is therefore already a channel of some underlying OPE, so sanitation lifts modularly. (i) *UC realization.* Define hybrids $H_0, \dots, H_n$ in which the first j invocations of $\widehat{\Pi}_{\text{OPE}}$ are replaced by $\text{Wrap}(\mathcal{F}_{\text{OPE}})$. [Theorem 7](#) gives $H_{j-1} \stackrel{c}{\approx} H_j$ for each j , so $H_0 \stackrel{c}{\approx} H_n$; by the corruption translation [Table 1](#), H_n is statistically Π_{PSI} in the $\mathcal{F}_{\text{OPE}}$ -hybrid model, which UC-realizes $\mathcal{F}_{\text{PSI}}$ by [Theorem 10](#). Hence, chaining yields the simulator. (ii) *Strong sanitation.* By construction $F_A^{\text{PSI}} = F_S^{(1)} \diamond \dots \diamond F_S^{(n)}$ and $F_B^{\text{PSI}} = F_R^{(1)} \diamond \dots \diamond F_R^{(n)}$, with no shared state. A specious PSI core projects to a specious marginal on each OPE sub-instance; each marginal is neutralized by the corresponding per-instance OPE firewall, strongly sanitizing by [Theorem 7](#). A n -step hybrid lifts marginal indistinguishability to joint view.

Complexity of OPE based PSI. Since Π_{OPE} evaluates a degree- d polynomial via d calls to $\mathcal{F}_{\text{OLE}}$, and party A's polynomial $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$ has degree $m = |S_A|$, the PSI protocol uses $mn = |S_A| \cdot |S_B|$ $\mathcal{F}_{\text{OLE}}$ calls, becomes $O(n^2)$ for balanced sets. This is fine in the unbalanced setting but motivates the OPA-based construction of the next section for balanced PSI.

6.2 Subversion-Resilient PSI from OPA

Obtaining PSI from OPA. There has been a development by Ghosh and Nilges [GN19] on obtaining two-sided PSI from OPA. Their protocol uses two OPA invocations along with an additional communication round. We develop a one-sided PSI protocol using only one invocation of OPA and no additional communication rounds. The protocol $\Pi_{\text{PSI}}^{\text{opa}}$ realizes $\mathcal{F}_{\text{PSI}}$ from a single $\mathcal{F}_{\text{OPA}}$ call. Each party encodes its size- n set as the polynomial vanishing on it: Party A builds $\mathbf{p}_A$ with roots S_A , and party B builds $\mathbf{p}_B$ with roots S_B . A samples random $\mathbf{r}_A, \mathbf{r}'_A$ of degree n , forms $\mathbf{q}_A = \mathbf{p}_A \cdot \mathbf{r}'_A$, and inputs $(\mathbf{q}_A, \mathbf{r}'_A)$ to $\mathcal{F}_{\text{OPA}}$ as sender; B inputs $\mathbf{p}_B$ as receiver and obtains $\mathbf{p}_\cap = \mathbf{q}_A + \mathbf{r}_A \cdot \mathbf{p}_B$. Any $\beta \in S_B \cap S_A$ zeroes both summands, so $\mathbf{p}_\cap(\beta) = 0$; for $\beta \notin S_A$, the random $\mathbf{r}_A$ masks the value. B outputs $S_\cap = \{\beta \in S_B : \mathbf{p}_\cap(\beta) = 0\}$. We formally describe our protocol in Figure 21. The proof of Theorem 12 follows from the proof of Theorem 8, thus postponed to Appendix H.2.

Theorem 12. *The protocol $\Pi_{\text{PSI}}^{\text{opa}}$ UC-realizes $\mathcal{F}_{\text{PSI}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model against semi-honest adversary.*

Sanitizing PSI from Sanitized OPA. The PSI protocol $\Pi_{\text{PSI}}^{\text{opa}}$ (formally, in Figure 21) invokes the $\mathcal{F}_{\text{OPA}}$ functionality once and no additional communication rounds are required between the parties. Only, a local computation of party B remains after $\mathcal{F}_{\text{OPA}}$ invocation. Thus, to sanitize PSI based on OPA we will replace the standard OPA invocation $\mathcal{F}_{\text{OPA}}$ in the $\Pi_{\text{PSI}}^{\text{opa}}$ with the sanitizing OPA protocol $\hat{\Pi}_{\text{OPA}}$ derived in the previous section. So, the sanitizing PSI protocol becomes a direct application of sanitizing OPA protocol. Let us call this protocol $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$. The proof of correctness and security directly follows from that of OPA. For the sake of completeness, we provide the formal analysis of $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ in Appendix J. The proof of Theorem 15 is postponed to Appendix J.1.

Theorem 13. *The protocol $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ wsrUC-realizes $\mathcal{F}_{\text{PSI}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}})$ -hybrid model.*

Two-Sided PSI from OPA. Given the one-sided PSI protocol $\Pi_{\text{PSI}}^{\text{opa}}$, a two-sided PSI protocol $\Pi_{2\text{-PSI}}^{\text{opa}}$ is obtained by having B forward the intersection $S_\cap$ to A. For the subversion-resilient version, we build on $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ and ensure that this additional communication round does not leak information, by routing it through a simple sanitizable message transmission protocol. The firewalls of OPA composed with those of message transmission then yield firewalls for $\Pi_{2\text{-PSI}}^{\text{opa}}$, giving a subversion-resilient two-sided PSI. A formal treatment is deferred to Appendix K.

7 Experimental Results

All protocols were implemented as a prototype⁸ in C++ using the BICYCL library [BCIL23], which provides efficient ideal class-group arithmetic over imaginary quadratic fields. The encryption scheme was instantiated at a 128-bit security level.

Setup. Benchmarks were run on two LAN-connected machines (1 Gbps, 0.2 ms RTT): the receiver on an Intel Core i9-14900K (32 threads) and the sender on an Intel Xeon w5-2565X (36 threads), with each party and its RF co-located on the same machine.

The parameter N defines the problem size: the number of instances in Batched OLE, the sender’s polynomial degree in OPE and OPA, and each party’s set size in PSI. We benchmark over $N \in \{2^{10}, 2^{11}, \dots, 2^{15}\}$.

Table 2 reports total execution times across the parameter sweep; Figure 12 illustrates the key trend. The relative RF overhead decreases as the local computational complexity of the protocol increases: at $N = 2^{15}$, the RF layer imposes a $2.8\times$ latency increase on Batched OLE, but only $1.25\times$ on PSI, confirming that subversion resilience is achievable at a modest and practical cost.

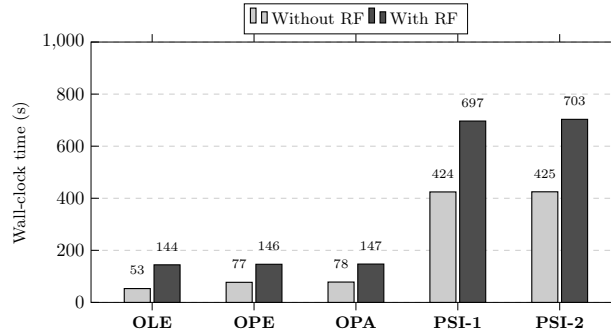


Fig. 12: Wall-clock time (sec), regular vs. RF protocol variants at $N = 2^{15}$. PSI-1: one-sided PSI from OPA; PSI-2: two-sided PSI from OPA.

Parallelism and Transport. Cryptographic work is parallelized via a persistent worker pool (28 threads on the receiver, 32 on the sender). Ciphertext traffic is split across eight TCP connections (network lanes); batched protocols process ciphertexts in chunks of 128 OLE instances to bound memory and overlap computation with I/O.

Acknowledgment. The authors thank the anonymous reviewers for their valuable comments and suggestions, which helped improve the manuscript. The authors also acknowledge the use of AI assistance for polishing the grammar and

⁸ https://github.com/Vaisampayna/rf_class_groups

Table 2: Total time (in sec) for all protocol variants, for $N = 2^{10}$ to 2^{15} .

Protocol	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}
Oblivious Linear Evaluation						
Batched OLE	2.73	4.21	7.3	13.56	26.73	53.18
Batched RF OLE	6.89	13.10	21.27	39.20	74.36	144.32
3-Round OLE	3.88	6.46	11.59	21.81	42.30	84.81
3-Round RF OLE ($\mathcal{F}_{\text{SOLE}}$)	10.27	18.32	34.75	66.80	131.55	260.80
Oblivious Polynomial Evaluation						
OPE	3.85	6.23	10.92	20.41	39.09	77.16
RF OPE	6.87	12.13	21.10	30.01	74.32	146.45
Oblivious Polynomial Addition						
OPA	5.07	7.40	12.12	21.49	40.37	78.18
RF OPA	8.22	13.73	23.03	38.20	76.67	147.27
Private Set Intersection						
PSI (one-sided)	11.42	20.82	39.80	77.70	154.14	424.31
RF PSI (one-sided)	23.50	39.17	80.36	145.29	288.53	696.52
PSI (two-sided)	11.46	20.85	39.82	77.76	154.49	424.84
RF PSI (two-sided)	26.13	45.19	81.79	158.84	297.65	703.10

language of the manuscript. This work originated from discussions among the authors at the First Crypto Winter School 2025, held at IIT Bhilai, India. Satrajit Ghosh acknowledges the support of SRIC, IIT Kharagpur, which partially funded this work through the FSRG grant and the 50-100 Academia-Industry Handshake initiative.

References

- ADI⁺17. Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 223–254. Springer, Heidelberg, August 2017.
- AMV15. Giuseppe Ateniese, Bernardo Magri, and Daniele Venturi. Subversion-resilient signature schemes. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 364–375. ACM, 2015.
- BBF⁺20. Angèle Bossuat, Xavier Bultel, Pierre-Alain Fouque, Cristina Onete, and Thyla van der Merwe. Designing reverse firewalls for the real world. In Liqun Chen, Ninghui Li, Kaitai Liang, and Steve A. Schneider, editors, *Computer Security - ESORICS 2020 - 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14-18, 2020, Proceedings, Part I*, volume 12308 of *Lecture Notes in Computer Science*, pages 193–213. Springer, 2020.
- BCIL23. Cyril Bouvier, Guilhem Castagnos, Laurent Imbert, and Fabien Laguillaumie. I want to ride my BICYCL : BICYCL implements cryptography in class groups. *J. Cryptol.*, 36(3):17, 2023.

- BDM22. Pedro Branco, Nico Döttling, and Paulo Mateus. Two-round oblivious linear evaluation from learning with errors. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part I*, volume 13177 of *LNCS*, pages 379–408. Springer, Heidelberg, March 2022.
- BDOZ11. Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 169–188. Springer, Heidelberg, May 2011.
- BEPU⁺20. Carsten Baum, Daniel Escudero, Alberto Pedrouzo-Ulloa, Peter Scholl, and Juan Ramón Troncoso-Pastoriza. Efficient protocols for oblivious linear function evaluation from ring-LWE. In Clemente Galdi and Vladimir Kolesnikov, editors, *SCN 20*, volume 12238 of *LNCS*, pages 130–149. Springer, Heidelberg, September 2020.
- Can00. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Paper 2000/067, 2000.
- CCL⁺19. Guilhem Castagnos, Dario Catalano, Fabien Laguillaumie, Federico Savasta, and Ida Tucker. Two-party ecDSA from hash proof systems and efficient instantiations. In *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III*, pages 191 – 221, 2019.
- CDN20. Suvradip Chakraborty, Stefan Dziembowski, and Jesper Buus Nielsen. Reverse firewalls for actively secure MPCs. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 732–762. Springer, Heidelberg, August 2020.
- CDVW12. Ran Canetti, Dana Dachman-Soled, Vinod Vaikuntanathan, and Hoeteck Wee. Efficient password authenticated key exchange via oblivious transfer. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 449–466. Springer, Heidelberg, May 2012.
- CFR23. Anrin Chakraborti, Giulia Fanti, and Michael K. Reiter. Distance-aware private set intersection. In *Proceedings of the 32nd USENIX Conference on Security Symposium, SEC ’23, USA, 2023*. USENIX Association.
- CGPS21. Suvradip Chakraborty, Chaya Ganesh, Mahak Pancholi, and Pratik Sarkar. Reverse firewalls for adaptively secure MPC without setup. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 335–364. Springer, Heidelberg, December 2021.
- CGS23. Suvradip Chakraborty, Chaya Ganesh, and Pratik Sarkar. Reverse firewalls for oblivious transfer extension and applications to zero-knowledge. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part I*, volume 14004 of *Lecture Notes in Computer Science*, pages 239–270. Springer, 2023.
- CL15. Guilhem Castagnos and Fabien Laguillaumie. Linearly homomorphic encryption from DDH. In Kaisa Nyberg, editor, *CT-RSA 2015*, volume 9048 of *LNCS*, pages 487–505. Springer, Heidelberg, April 2015.
- CLT18. Guilhem Castagnos, Fabien Laguillaumie, and Ida Tucker. Practical fully secure unrestricted inner product functional encryption modulo p . In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*,

- volume 11273 of *LNCS*, pages 733–764. Springer, Heidelberg, December 2018.
- CMMV24. Suvradip Chakraborty, Lorenzo Magliocco, Bernardo Magri, and Daniele Venturi. Key exchange in the post-snowden era: Universally composable subversion-resilient PAKE. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part V*, Lecture Notes in Computer Science, pages 101–133. Springer, 2024.
- CMNV22. Suvradip Chakraborty, Bernardo Magri, Jesper Buus Nielsen, and Daniele Venturi. Universally composable subversion-resilient cryptography. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 272–302. Springer, Heidelberg, May / June 2022.
- CMY⁺16. Rongmao Chen, Yi Mu, Guomin Yang, Willy Susilo, Fuchun Guo, and Mingwu Zhang. Cryptographic reverse firewall via malleable smooth projective hash functions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 844–876. Springer, Heidelberg, December 2016.
- DGN⁺17. Nico Döttling, Satrajit Ghosh, Jesper Buus Nielsen, Tobias Nilges, and Roberto Trifiletti. TinyOLE: Efficient actively secure two-party computation from oblivious linear function evaluation. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 2263–2276. ACM Press, October / November 2017.
- DMSD16. Yevgeniy Dodis, Ilya Mironov, and Noah Stephens-Davidowitz. Message transmission with reverse firewalls—secure communication on corrupted machines. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016*, pages 341–372, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
- DMSDT25. Yevgeniy Dodis, Bernardo Magri, Noah Stephens-Davidowitz, and Yiannis Tselekounis. Guarding the signal: Secure messaging with reverse firewalls. Cryptology ePrint Archive, Paper 2025/1172, 2025.
- DPSZ12. Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multi-party computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 643–662. Springer, Heidelberg, August 2012.
- EGL85. Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, June 1985.
- FNPO4. Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In Christian Cachin and Jan L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 1–19, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
- Gil99. Niv Gilboa. Two party RSA key generation. In Michael J. Wiener, editor, *CRYPTO’99*, volume 1666 of *LNCS*, pages 116–129. Springer, Heidelberg, August 1999.
- Gil01. Niv Gilboa. *Topics in private information retrieval*. PhD thesis, Technion - Israel Institute of Technology, Israel, 2001.
- GIP⁺14. Daniel Genkin, Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and Eran Tromer. Circuits resilient to additive attacks with applications to secure computation. In David B. Shmoys, editor, *46th ACM STOC*, pages 495–504. ACM Press, May / June 2014.

- GMV20. Chaya Ganesh, Bernardo Magri, and Daniele Venturi. Cryptographic reverse firewalls for interactive proof systems. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *ICALP 2020*, volume 168 of *LIPIcs*, pages 55:1–55:16. Schloss Dagstuhl, July 2020.
- GMW87. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, New York, NY, USA, 1987. Association for Computing Machinery.
- GN19. Satrajit Ghosh and Tobias Nilges. An algebraic approach to maliciously secure private set intersection. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 154–185. Springer, Heidelberg, May 2019.
- GNN17. Satrajit Ghosh, Jesper Buus Nielsen, and Tobias Nilges. Maliciously secure oblivious linear function evaluation with constant overhead. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 629–659. Springer, Heidelberg, December 2017.
- GRS22. Gayathri Garimella, Mike Rosulek, and Jaspal Singh. Structure-aware private set intersection, with applications to fuzzy matching. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 323–352. Springer, Heidelberg, August 2022.
- GS19. Satrajit Ghosh and Mark Simkin. The communication complexity of threshold private set intersection. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 3–29. Springer, Heidelberg, August 2019.
- GS23. Satrajit Ghosh and Mark Simkin. Threshold private set intersection with better communication complexity. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *Public-Key Cryptography – PKC 2023*, pages 251–272, Cham, 2023. Springer Nature Switzerland.
- HL08. Carmit Hazay and Yehuda Lindell. Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. In Ran Canetti, editor, *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 155–175. Springer, 2008.
- HV17. Carmit Hazay and Muthuramakrishnan Venkitasubramaniam. Scalable multi-party private set-intersection. In Serge Fehr, editor, *PKC 2017, Part I*, volume 10174 of *LNCS*, pages 175–203. Springer, Heidelberg, March 2017.
- IPS09a. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 294–314. Springer, 2009.
- IPS09b. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 294–314. Springer, Heidelberg, March 2009.
- Kil88. Joe Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press, May 1988.
- KKRT16. Vladimir Kolesnikov, Ranjit Kumaresan, Mike Rosulek, and Ni Trieu. Efficient batched oblivious PRF with applications to private set intersection. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel,

- Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 818–829. ACM, 2016.
- KMM⁺24. Aniket Kate, Easwar Vivek Mangipudi, Pratyay Mukherjee, Hamza Saleem, and Sri Aravinda Krishnan Thyagarajan. Non-interactive vss using class groups and application to dkg. In *CCS '24: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 4286 – 4300, 2024.
- KS05. Lea Kissner and Dawn Xiaodong Song. Privacy-preserving set operations. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 241–257. Springer, Heidelberg, August 2005.
- Mea86. Catherine Meadows. A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In *1986 IEEE Symposium on Security and Privacy*, pages 134–134, 1986.
- MS15. Ilya Mironov and Noah Stephens-Davidowitz. Cryptographic reverse firewalls. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 657–686. Springer, Heidelberg, April 2015.
- NP99. Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *31st ACM STOC*, pages 245–254. ACM Press, May 1999.
- NP06. Moni Naor and Benny Pinkas. Oblivious polynomial evaluation. *SIAM J. Comput.*, 35(5):1254–1281, May 2006.
- PRTY20. Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from paxos: Fast, malicious private set intersection. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 739–767. Springer, 2020.
- PSTY19. Benny Pinkas, Thomas Schneider, Oleksandr Tkachenko, and Avishay Yanai. Efficient circuit-based PSI with linear communication. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 122–153. Springer, Heidelberg, May 2019.
- Rab05. Michael O. Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptol. ePrint Arch.*, 2005:187, 2005.
- RR22. Srinivasan Raghuraman and Peter Rindal. Blazing fast PSI from improved OKVS and subfield VOLE. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 2505–2517. ACM Press, November 2022.
- vBP24. Aron van Baarsen and Sihang Pu. Fuzzy private set intersection with large hyperballs. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part V*, volume 14655 of *Lecture Notes in Computer Science*, pages 340–369. Springer, 2024.

A Background on Universal Composability

A.1 The Universal Composability Framework: A Brief Overview

The Universal Composability (UC) framework [Can00] provides a rigorous paradigm for defining and analyzing the security of cryptographic protocols. A protocol Π specifies the code executed by each party $P_1, \dots, P_n$, where parties may invoke certain *ideal functionalities* $\mathcal{G}$. Formally, each party is modeled as an interactive Turing machine parameterized by a party identifier pid , which determines the corresponding party code to be executed.

A party P_i invokes an ideal functionality by specifying the functionality $\mathcal{G}$, a session identifier sid , and an input x . The tuple $(\text{sid}, \text{pid}, x)$ is then delivered to $\mathcal{G}$, which is instantiated if it does not already exist. The execution also involves an *adversary* $\mathcal{A}$, who may corrupt parties through special corruption commands according to the underlying corruption model. In addition, an *environment* $\mathcal{Z}$ provides inputs to the parties, observes their outputs, and can communicate freely with the adversary during the execution.

A protocol Π is said to *UC-realize* an ideal functionality $\mathcal{F}$ if there exists an efficient *simulator* $\mathcal{S}$ such that no efficient environment $\mathcal{Z}$ can distinguish a real execution of Π from an ideal execution involving $\mathcal{F}$ and $\mathcal{S}$. Formally, this is written as

$$\exists \mathcal{S} \forall \mathcal{Z} : \text{Exec}_{\Pi, \mathcal{A}, \mathcal{Z}} \approx \text{Exec}_{\mathcal{F}, \mathcal{S}, \mathcal{Z}},$$

where $\mathcal{A}$ denotes the dummy adversary, which acts as a transparent proxy for the environment.

A key feature of the UC framework is its *composition theorem*. Suppose Π UC-realizes $\mathcal{F}$ in the $\mathcal{G}$ -hybrid model, and Γ UC-realizes $\mathcal{G}$. Let $\Pi^{\mathcal{G} \rightarrow \Gamma}$ denote the composed protocol obtained by replacing every call to $\mathcal{G}$ in Π with an invocation of Γ . The composition theorem guarantees that $\Pi^{\mathcal{G} \rightarrow \Gamma}$ UC-realizes $\mathcal{F}$. Thus, protocol components can be analyzed independently and then securely composed, even in the presence of arbitrary concurrent executions.

A.2 Sanitizable Authenticated Transmission (SAT)

A useful abstraction in the paper [CMNV22] is the notion of sanitizable authenticated transmission, formalized via the functionality $\mathcal{F}_{\text{SAT}}$. At a high level, this functionality models message transmission where all communication between protocol cores is mediated by reverse firewalls, which are responsible for sanitizing messages before they are delivered. Concretely, when a core sends a message, it is first exposed to its local firewall, which may modify or re-randomize it; only the sanitized version is then forwarded (via the adversary-controlled delivery interface) to the recipient's firewall, and finally passed to the receiving core. This ensures that any subliminal channels introduced by a potentially subverted core can be neutralized at the firewall layer. Importantly, while the functionality guarantees authentication between firewalls (i.e., firewalls know the sender identity), it does not guarantee that cores receive truthful metadata, since a malicious firewall may alter or inject messages arbitrarily. The functionality also includes

a setup phase distributing correlated parameters (e.g., common reference string) to cores and firewalls, with firewall state being adversarially visible to prevent it from holding secrets. Overall, $\mathcal{F}_{\text{SAT}}$ captures a communication model where correctness and authenticity are preserved at the firewall level, while enabling systematic sanitation of all transmitted data, thereby forming the backbone for constructing subversion-resilient protocols in the UC framework.

A.3 UC Modeling of Reverse Firewalls

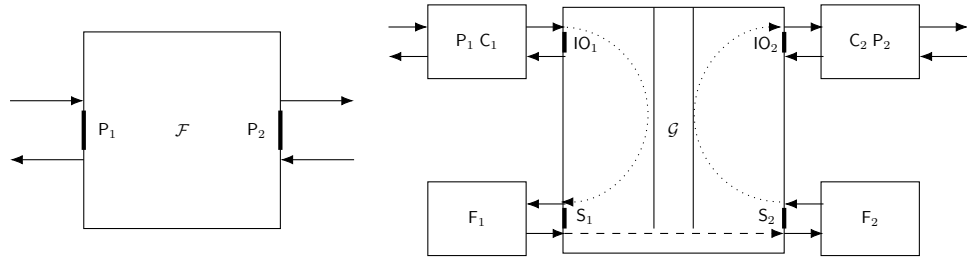


Fig. 13: Implementing a functionality $\mathcal{F}$ via a sanitizing protocol $\Pi = (\mathcal{C}, \mathcal{F})$ using a sanitizable functionality $\mathcal{G}$. Each party consists of a core and a firewall, where the firewall sanitizes all communication through the sanitation interface of $\mathcal{G}$.

B Missing Proofs in section 4

B.1 Schematic Diagram of 3-Round OLE Protocol Realizing Sanitizable OLE

B.2 Proof of Theorem 1

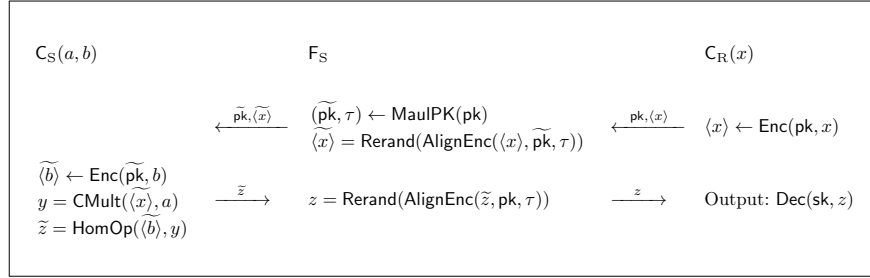
Proof. We exhibit an environment $\mathcal{Z}$ that distinguishes the real and ideal executions with probability $1 - \text{negl}(\lambda)$.

The environment. $\mathcal{Z}$ samples $x, a, b \in \mathcal{R}$ uniformly and delivers x to C_R and (a, b) to C_S on the appropriate IO_i interfaces. All four components are honest. The adversary $\mathcal{A}$ is the dummy adversary. After the protocol terminates, $\mathcal{Z}$ inspects R 's output y , reads the blinding masks (a', b', a'', b'') from the simulator's S_i messages (visible in the srUC transcript), computes $y_{\text{ideal}} := aa'a''x + bb'b''$, and outputs 1 iff $y = y_{\text{ideal}}$.

Ideal world. By the S_i composition rules, the functionality produces $\hat{a} = aa'a''$ and $\hat{b} = bb'b''$, so R outputs exactly y_{ideal} and $\mathcal{Z}$ outputs 1 with probability 1.

$\mathcal{F}_{\text{SAT}}$

- Initially sample $((v_S, w_S), (v_R, w_R)) \leftarrow \text{setup}(1^\lambda)$ and output v_S on IO_S , v_R on IO_R , w_S on S_S , and w_R on S_R . Leak w_S and w_R to the adversary $\mathcal{S}$.
- On input (SEND, a, R) on IO_S , output (SEND, a, R) on S_S . To keep the description simple, we assume honest parties send the same a at most once. Adding fresh message identifiers can be used for this in an implementation.
- On input (SEND, b, R) on S_S , leak (SEND, S, b, R) to the adversary $\mathcal{S}$ and store (SEND, S, b, R) .
- On input $(\text{DELEVER}, (\text{SEND}, S, b, R))$ from the adversary $\mathcal{S}$, where (SEND, S, b, R) is stored, delete this tuple and output $(\text{receive}, S, b)$ on S_R .
- On input $(\text{receive}, S, c)$ on S_R , output $(\text{Receive}, S, c)$ on IO_R .

Fig. 14: The sanitizable authenticated transmission functionality.**Fig. 15:** Schematic Diagram of Sanitizing 2-round OLE Protocol

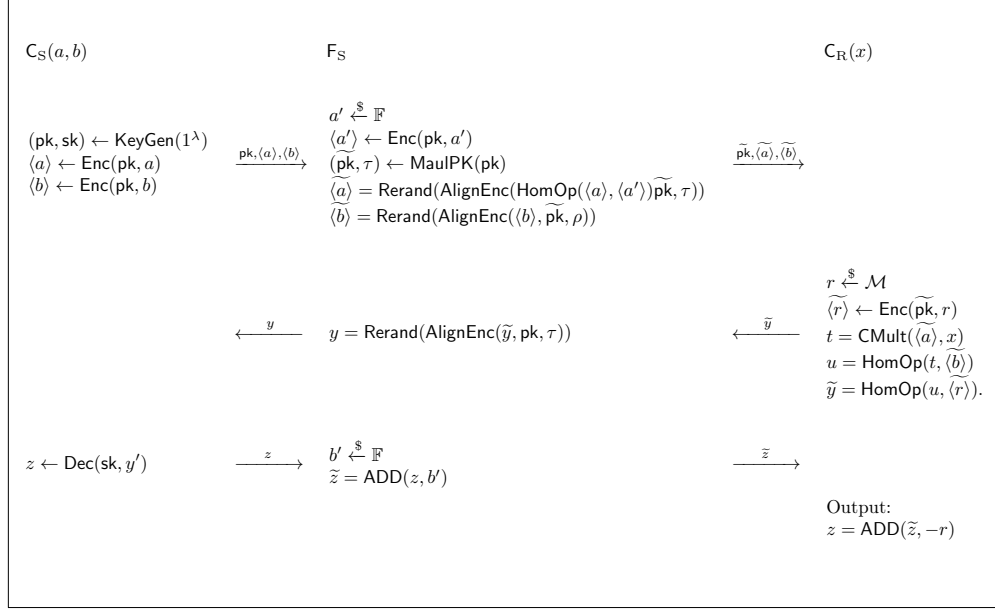
Real world. Tracing the plaintext through the four protocol steps yields

$$P_3 = a' \cdot (a \cdot (a''x + b'') + b) + b' = aa'a''x + aa'b'' + a'b + b'.$$

On the return trip, F_R applies a homomorphic affine transformation $z \mapsto \gamma z + \delta$ (the most general operation it can perform via the AHE), yielding the receiver's output

$$y_{\text{Real}} = \gamma(aa'a''x + aa'b'' + a'b + b') + \delta.$$

The mismatch. For $\mathcal{Z}$ to output 1 in the real world, we need $y_{\text{Real}} = y_{\text{Ideal}}$. Equating the two and viewing the equation as a polynomial identity in the formal indeterminates (x, a, b) over $\mathcal{R}$ (with the firewall masks held fixed), we

**Fig. 16:** Schematic Diagram of Sanitizable 3-Round OLE Protocol

match coefficients:

$$ax \text{ coefficient : } \gamma a' a'' = a' a'' \implies \gamma = 1, \quad (1)$$

$$ab'' \text{ coefficient : } \gamma a' = 0, \quad (2)$$

$$b \text{ coefficient : } \gamma a' = 0, \quad (3)$$

$$\text{constant : } \gamma b' + \delta = bb'b''. \quad (4)$$

Combining (1) and (2) forces $a' = 0$. But by the well-spreadness of $\mathcal{D}_S$ (Definition 5, invoked in Definition 6), a' has min-entropy $\omega(\log \lambda)$, so

$$\Pr[a' = 0] \leq 2^{-\omega(\log \lambda)} = \text{negl}(\lambda).$$

Moreover, (4) requires δ to depend on b i.e., the sender's secret coefficient, hidden from F_R by IND-CPA security of the AHE. Either obstruction alone shows that $\Pr[y_{\text{Real}} = y_{\text{ideal}}] = \text{negl}(\lambda)$ over the joint randomness of the firewalls and the IND-CPA game.

Conclusion. $\mathcal{Z}$ outputs 1 with probability 1 in the ideal world and with probability $\text{negl}(\lambda)$ in the real world. No simulator can close this gap. $\square$

Specious adversaries suffice. The impossibility holds against the weakest adversarial setting in the srUC framework: a specious environment with honest firewalls. We did not need to invoke malicious corruption of any component.

This is because the obstruction is a *correctness* obstruction: the protocol fails to compute the right function even when no party is acting adversarially. In particular, the impossibility extends *a fortiori* to specious corruptions of the cores: a specious core can only do less than an honest core (in terms of preserving correctness under firewall blinding), so the impossibility holds for the strong sanitation regime as well.

Composition. A virtue of casting the result in the srUC framework is that it lifts to arbitrary compositions. Any larger protocol that uses $\mathcal{F}_{\text{sOLE}}$ as a hybrid functionality inherits the impossibility: there is no way to instantiate $\mathcal{F}_{\text{sOLE}}$ with a 2-round AHE-based sub-protocol that respects the bilateral plaintext-masking design pattern. Conversely, if a higher-level protocol can be designed to use a weaker variant of $\mathcal{F}_{\text{sOLE}}$ (e.g., one without bilateral plaintext masking), it may still admit a 2-round subversion-resilient implementation.

B.3 2-Round OLE Protocol

Protocol: $\Pi_{\text{OLE}}^{(2)}$

Let AHE be a key-malleable, additive homomorphic encryption scheme as in [Definition 4](#). The protocol run between the sender S with input (a, b) and the receiver R with input x .

- **Round-1:** The receiver R runs $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$. Compute $\langle x \rangle \leftarrow \text{Enc}(\text{pk}, x)$. Send $(\text{pk}, \langle x \rangle)$ to S.
 - **Round-2:** Upon receiving $(\text{pk}, \langle x \rangle)$ from R the sender S compute: $t = \text{CMult}(\langle x \rangle, a)$, $u = \text{Enc}(\text{pk}, b)$, and $y = \text{HomOp}(t, u)$. Send y to R
- Output:** R outputs $\text{Dec}(\text{sk}, y)$.

Fig. 17: 2-Round OLE Protocol.

B.4 Proof of [Lemma 2](#)

Lemma 2. *The protocol $\Pi_{\text{OLE}}^{(3)}$ is correct.*

Proof. By the scalar-ciphertext multiplication and homomorphic operation properties of AHE, the Round-2 ciphertext satisfies

$$y = \text{HomOp}\left(\text{HomOp}(\text{CMult}(\langle a \rangle, x), \langle b \rangle), \langle r \rangle\right) = \text{Enc}(\text{pk}, a \cdot x + b + r).$$

By correctness of AHE, the sender obtains $z = \text{Dec}(\text{sk}, y) = a \cdot x + b + r$ in Round 3. The receiver then outputs $\text{ADD}(z, -r) = a \cdot x + b$, which is the correct OLE output. $\square$

B.5 Proof of Theorem 2

Proof. Fix any PPT environment $\mathcal{Z}$ and any PPT semi-honest adversary $\mathcal{A}$ statically corrupting a single party $I \in \{S, R\}$. We exhibit a simulator $\mathcal{S}_I$ such that

$$\text{Exec}_{\Pi_{\text{OLE}}, \mathcal{A}, \mathcal{Z}}^{(3)} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_I, \mathcal{Z}}.$$

We treat the two corruption cases separately. Each hybrid is described by how it generates the components of the joint view $(\text{View}_{\mathcal{A}}, z_{\text{out}})$.

Case 1: Corrupt Sender

The corrupted party $\mathcal{A}_S$ produces $(\text{pk}, \langle a \rangle, \langle b \rangle)$ in round 1, receives y in round 2, and produces z in round 3. The simulator must generate the round-2 message y without knowing the receiver's input x .

Simulator $\mathcal{S}_S$

1. Receive $\mathcal{A}_S$'s round-1 message $(\text{pk}, \langle a \rangle, \langle b \rangle)$ and read (a, b) from $\mathcal{A}_S$'s input tape.
2. Submit $(\text{sid}, (a, b))$ to $\mathcal{F}_{\text{OLE}}$.
3. Sample $u \xleftarrow{\$} \mathcal{M}$ and send $\tilde{y} \leftarrow \text{Enc}(\text{pk}, u)$ to $\mathcal{A}_S$ as the round-2 message.
4. Receive $\mathcal{A}_S$'s round-3 message z and discard it.

H_0 . This is the real world. The honest receiver, on the round-1 message $(\text{pk}, \langle a \rangle, \langle b \rangle)$ produced by $\mathcal{A}_S$, samples $r \xleftarrow{\$} \mathcal{M}$ and replies with the honest evaluation $y = \text{HomOp}(\text{HomOp}(\text{CMult}(\langle a \rangle, x), \langle b \rangle), \langle r \rangle)$. On round-3 input z , the receiver outputs $\text{ADD}(z, -r)$.

H_1 . y is replaced by a fresh encryption $\tilde{y} \leftarrow \text{Enc}(\text{pk}, \tilde{u})$ for $\tilde{u} \xleftarrow{\$} \mathcal{M}$ sampled independently. The receiver's output is delivered as $a \cdot x + b$ (directly by $\mathcal{F}_{\text{OLE}}$ in the ideal experiment).

$H_0 \stackrel{c}{\approx} H_1$. The only difference is the Round-2 ciphertext: an encryption of an x -dependent plaintext (which is itself uniformly random due to the mask r) versus an encryption of an independent uniform plaintext $\tilde{u}$. We claim that

$$y = \text{HomOp}(\text{HomOp}(\text{CMult}(\langle a \rangle, x), \langle b \rangle), \langle r \rangle) \stackrel{c}{\approx} \tilde{y} = \text{Enc}(\text{pk}, \tilde{u})$$

directly by IND-CPA security of AHE. A distinguisher D with advantage ε between H_1 and H_2 yields an IND-CPA adversary $\mathcal{B}$ with the same advantage. $\mathcal{B}$ runs $\mathcal{Z}$ and $\mathcal{A}_S$ internally, obtaining the public key pk from $\mathcal{A}_S$'s Round-1 message, extracting (a, b) from $\mathcal{A}_S$'s input tape and x from $\mathcal{Z}$. It samples $r \xleftarrow{\$} \mathcal{M}$ and $\tilde{u} \leftarrow \mathcal{M}$, submits the challenge pair $(m_0, m_1) := (ax + b + r, \tilde{u})$ to its IND-CPA challenger against the key pk produced by $\mathcal{A}_S$, and receives $c^* \leftarrow \text{Enc}(\text{pk}, m_\beta)$. $\mathcal{B}$ delivers c^* to $\mathcal{A}_S$ as the Round-2 message, computes the receiver's output algebraically as $ax + b$, and outputs D 's decision bit. The joint view matches H_1 if $\beta = 0$ and H_2 if $\beta = 1$, so $\mathcal{B}$'s advantage equals ε .

$H_1 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_S, \mathcal{Z}}$. Inspection of $\mathcal{S}_S$ confirms that H_1 is the ideal-world execution.

Chaining: $\text{Exec}_{\Pi_{\text{OLE}}^{(3)}, \mathcal{A}, \mathcal{Z}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_S, \mathcal{Z}}$.

Case 2: Corrupt Receiver

The corrupted party $\mathcal{A}_R$ receives $(\text{pk}, \langle a \rangle, \langle b \rangle)$ in round 1, produces y in round 2, and receives z in round 3. Here the simulator generates $(\text{pk}, \langle a \rangle, \langle b \rangle)$ and z without knowing (a, b) , using only $z^* = a \cdot x + b$ obtained from $\mathcal{F}_{\text{OLE}}$ and the masking scalar r extracted from $\mathcal{A}_R$'s random tape. Since $\mathcal{A}_R$ does not hold sk , IND-CPA security of AHE is available and is used to switch $\langle a \rangle, \langle b \rangle$ to encryptions of 0.

Simulator $\mathcal{S}_R$

1. Read x from $\mathcal{A}_R$'s input tape and the masking scalar $r \in \mathcal{M}$ from $\mathcal{A}_R$'s random tape.
2. Submit (sid, x) to $\mathcal{F}_{\text{OLE}}$ and receive $z^* = a \cdot x + b$.
3. Sample $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and discard sk . Send $(\text{pk}, \text{Enc}(\text{pk}, 0), \text{Enc}(\text{pk}, 0))$ to $\mathcal{A}_R$ as the round-1 message.
4. Receive $\mathcal{A}_R$'s round-2 message y .
5. Send $\tilde{z} \leftarrow z^* + r$ to $\mathcal{A}_R$ as the round-3 message.

We define the sequence of hybrids as follows,

H_0 . This is the real world. $(\text{pk}, \langle a \rangle, \langle b \rangle) = (\text{pk}, \text{Enc}(\text{pk}, a), \text{Enc}(\text{pk}, b))$; on receiving the honestly-produced y from $\mathcal{A}_R$, the sender returns $z = \text{Dec}(\text{sk}, y)$.

H_1 . Identical to H_0 , except z is computed algebraically as $z \leftarrow a \cdot x + b + r$, where r is read from $\mathcal{A}_R$'s random tape. Decryption with sk is no longer used.

H_2 . Identical to H_1 , except $\langle a \rangle$ is replaced by $\text{Enc}(\text{pk}, 0)$.

H_3 . This is the ideal world. Identical to H_2 , except $\langle b \rangle$ is replaced by $\text{Enc}(\text{pk}, 0)$.

$H_0 \equiv H_1$. Semi-honest $\mathcal{A}_R$ computes y as a fresh $\text{Enc}(\text{pk}, ax+b+r)$ (homomorphic freshness). By correctness, $\text{Dec}(\text{sk}, y) = ax+b+r = (ax+b)+r$. *This hop removes sk from the computation of z , enabling the IND-CPA reductions below.*

$H_1 \stackrel{c}{\approx} H_2$. The only difference is the plaintext underlying $\langle a \rangle$: real a versus 0. By IND-CPA security of AHE, ciphertexts under pk hide their plaintexts, so $\text{Enc}(\text{pk}, a) \stackrel{c}{\approx} \text{Enc}(\text{pk}, 0)$.

$H_2 \stackrel{c}{\approx} H_3$. Symmetric: embed the IND-CPA challenge as $\langle b \rangle$ with challenge pair $(0, b)$.

$H_3 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_R, \mathcal{Z}}$. By definition of $\mathcal{F}_{\text{OLE}}$, $z^* = ax+b$, so $z^* + r = (ax+b)+r$. The receiver's output, computed by $\mathcal{A}_R$ as $\text{ADD}(z, -r)$, equals $ax+b$ in both H_3 and the ideal world.

Chaining: $\text{Exec}_{\Pi_{\text{OLE}}, \mathcal{A}, \mathcal{Z}} \equiv H_1 \stackrel{c}{\approx} H_2 \stackrel{c}{\approx} H_3 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_R, \mathcal{Z}}$.

This completes the proof of both corruption cases, and hence of [Theorem 2](#). $\square$

B.6 Proof of Lemma 3

Lemma 3. *The Protocol $\widehat{\Pi}_{\text{OLE}}^{(3)}$ is correctly realizes the functionality $\mathcal{F}_{\text{sOLE}}$.*

Proof. Write $\hat{a} = (a + a' + a'')$ and $\hat{b} = (b + b' + b'')$. As in the original protocol,

$$\text{HomOp}(u, \langle \widehat{r} \rangle) = \text{Enc}(\widehat{\text{pk}}, \hat{a}x + b + r).$$

In the second round the firewalls now add their own masks homomorphically: F_R adds r'' under $\widehat{\text{pk}}$ and then F_S adds r' under $\widehat{\text{pk}}$. Thus, by the correctness of HomOp, Rerand and AlignEnc we obtain that

$$y = \text{Enc}(\text{pk}, \hat{a}x + b + r + r' + r'').$$

Eventually, in the third round we obtain that $\hat{a}x + b + r + r' + r'' = \text{Dec}(\text{sk}, y)$. Each firewall then strips the mask it added and applies its plaintext mask, first F_S with $(-r', b')$ and then F_R with $(-r'', b'')$. The masks r' and r'' cancel, so we obtain $\hat{z} = \hat{a}x + \hat{b} + r$. At last the receiver obtains $\hat{a}x + \hat{b}$ after removing the random mask r it applied in round 2. Thus the protocol correctly realizes the functionality $\mathcal{F}_{\text{sOLE}}$. $\square$

B.7 Proof of Theorem 3

Proof. To simplify notation, we write $\widehat{\Pi} := \widehat{\Pi}_{\text{OLE}}^{(3)}$ and $\widehat{\mathcal{G}} = \mathcal{F}_{\text{sOLE}}$. Recall that, by the definition of subversion resilience, we need to show that $\widehat{\Pi}$ UC-realizes, realizes $\widehat{\mathcal{G}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model, and that $\widehat{\mathcal{G}}$ can be written as well formed sanitizing ideal functionality. Towards this we first build a simulator communicating with $\widehat{\mathcal{G}}$ that simulates an execution of $\widehat{\Pi}$ for the cases of one party semi-honestly corrupt and the firewall of the other party is semi-honestly corrupt. Next, we show that the firewalls for both the sender and receiver are strongly sanitizing.

Lemma 4. *For every PPT adversary $\mathcal{A}$ that corrupts one party semi-honestly and the firewall of the other party semi-honestly in an execution of $\widehat{\Pi}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model, there exists a PPT simulator $\mathcal{S}$ such that for every PPT environment $\mathcal{Z}$,*

$$\text{Exec}_{\widehat{\Pi}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{G}, \mathcal{S}, \mathcal{Z}}.$$

Proof. Following the corruption translation table (Table 1), we have two symmetric cases: the corrupted party may be either the sender or the receiver. We describe a simulator for each.

Case 1: Sender semi-honestly corrupted. Here C_S, F_S are semi-honest, F_R is semi-honest, and C_R is honest. Since all three semi-honest components follow the prescribed algorithms, the simulator $\mathcal{S}$ learns their inputs and random tapes: the sender's input $(a, b) \in \mathbb{F}^2$ on IO_S , the sender's firewall mask $(a', b') \in \mathbb{F}^2$ on S_S , the receiver's firewall mask $(a'', b'') \in \mathbb{F}^2$ on S_R , and the return-trip masks $r', r'' \in \mathcal{M}$ sampled inside the two firewalls. The simulator proceeds as follows.

1. Upon activation of C_S with input (a, b) , send $(\text{SENDER}, \text{sid}, (a, b))$ to $\mathcal{F}_{\text{SOLE}}$ on IO_S .
2. Upon activation of F_S , forward $(\text{BLIND}, \text{sid}, (a', b'))$ to $\mathcal{F}_{\text{SOLE}}$ on S_S .
3. Upon activation of F_R , forward $(\text{BLIND}, \text{sid}, (a'', b''))$ to $\mathcal{F}_{\text{SOLE}}$ on S_R .
4. *Simulating the honest receiver's outbound message y .* Without access to C_R 's input x , the simulator samples $z^* \xleftarrow{\$} \mathbb{F}$ uniformly and sets

$$y \leftarrow \text{Enc}(\widehat{\text{pk}}, z^*),$$

using $\widehat{\text{pk}}$ as produced inside the simulated run of F_R . The simulator then executes the protocol code of F_R and F_S on y , which masks it with r'' and then with r' , to compute $\tilde{y}$ and the value $\bar{y}$ delivered to C_S , and emits $z = z^* + r' + r''$ as the Round-3 message of (the simulated) C_S on the wire to F_S . Subsequent computations (the mask removals and additive blindings $\tilde{z} = \text{ADD}(\text{ADD}(z, -r'), b')$ and $\hat{z} = \text{ADD}(\text{ADD}(\tilde{z}, -r''), b'')$) follow the protocol code of F_S and F_R .

Indistinguishability. In the real execution, the plaintext underlying y equals $(a + a' + a'') \cdot x + b + r$, where $r \xleftarrow{\$} \mathbb{F}$ is the mask sampled by C_R in Round 2, and the value decrypted by C_S is this plaintext shifted by $r' + r''$. Because r is uniform and independent of all other variables, both are uniform over $\mathbb{F}$. The simulator's z^* is also uniform over $\mathbb{F}$ and it emits $z = z^* + r' + r''$; hence the value z observed by the corrupted sender side has the same distribution in the two executions. After the chain of (KM4)-alignments and rerandomizations, $\bar{y}$ is statistically close to a fresh $\text{Enc}(\text{pk}, z^* + r' + r'')$ in the simulation and to a fresh $\text{Enc}(\text{pk}, (a + a' + a'')x + b + r + r' + r'')$ in the real execution; the marginal distributions over $\bar{y}$ coincide. All other messages in the corrupted parties' view are produced by deterministic firewall code on values that are identically distributed in both worlds. The honest receiver's output is delivered directly by $\mathcal{F}_{\text{SOLE}}$ in the ideal world, and equals $\hat{a} \cdot x + \hat{b}$ where $\hat{a} = a + a' + a''$ and $\hat{b} = b + b' + b''$, which by the correctness of $\widehat{\Pi}$ (Lemma 3) is exactly the value C_R would output in the real protocol. Thus

$$\text{Exec}_{\widehat{\Pi}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} \equiv \text{Exec}_{\widehat{g}, \mathcal{S}, \mathcal{Z}}.$$

Case 2: Receiver semi-honestly corrupted. Here C_R, F_R are semi-honest, F_S is semi-honest, and C_S is honest. The simulator $\mathcal{S}$ learns the receiver's input $x \in \mathbb{F}$, the receiver's mask $r \in \mathbb{F}$ chosen in Round 2, the firewalls' masks $(a', b'), (a'', b'') \in \mathbb{F}^2$, and their return-trip masks $r', r'' \in \mathcal{M}$. The simulator does not know (a, b) . It proceeds as follows.

1. Upon activation of C_R with input x , send $(\text{RECEIVER}, \text{sid}, x)$ to $\mathcal{F}_{\text{SOLE}}$ on IO_R . *Do not yet deliver the response.*
2. Upon activation of F_S , forward $(\text{BLIND}, \text{sid}, (a', b'))$ to $\mathcal{F}_{\text{SOLE}}$ on S_S .
3. Upon activation of F_R , forward $(\text{BLIND}, \text{sid}, (a'', b''))$ to $\mathcal{F}_{\text{SOLE}}$ on S_R . The functionality $\mathcal{F}_{\text{SOLE}}$ now returns $\hat{y} := \hat{a} \cdot x + \hat{b}$, with $\hat{a} = a + a' + a''$ and $\hat{b} = b + b' + b''$, to be delivered to C_R ; the simulator records $\hat{y}$.

4. *Simulating the honest sender's outbound messages.* The simulator runs $\text{KeyGen}(1^\lambda)$ to obtain $(\text{pk}_S, \text{sk}_S)$ and sets

$$\langle a \rangle \leftarrow \text{Enc}(\text{pk}_S, 0_{\mathbb{F}}), \quad \langle b \rangle \leftarrow \text{Enc}(\text{pk}_S, 0_{\mathbb{F}}),$$

relaying $(\text{pk}_S, \langle a \rangle, \langle b \rangle)$ on the wire from C_S to F_S .

5. *Simulating C_S 's Round-3 message.* After running the protocol code of the (simulated) firewalls through Rounds 1 and 2, the simulator outputs, on the wire from C_S to F_S :

$$z \leftarrow \hat{y} - b' - b'' + r + r' + r''.$$

The remaining messages $\tilde{z} = \text{ADD}(\text{ADD}(z, -r'), b')$ and $\hat{z} = \text{ADD}(\text{ADD}(\tilde{z}, -r''), b'')$ are produced by the firewall code; note that $\hat{z} = \hat{y} + r$, so C_R 's computed output $\text{ADD}(\hat{z}, -r) = \hat{y}$, matching the value $\mathcal{F}_{\text{OLE}}$ already returned to C_R .

Indistinguishability. We argue via a single hop based on the IND-CPA security of AHE. Consider the hybrid H obtained from the real execution by replacing the plaintexts of $\langle a \rangle$ and $\langle b \rangle$ with encryption of $0_{\mathbb{F}}$. The view of any PPT environment in H is computationally indistinguishable from its view in the real execution by IND-CPA, since none of the corrupted components knows sk (which is held only by the honest C_S). All downstream ciphertexts $\widetilde{\langle a \rangle}, \widetilde{\langle b \rangle}, \widehat{\langle a \rangle}, \widehat{\langle b \rangle}, \bar{y}$ derived from $\langle a \rangle, \langle b \rangle$ through key-malleable operations $\text{HomOp}, \text{AlignEnc}, \text{Rerand}$ remain IND-CPA secure (they are functions of the public parameters and IND-CPA-secure ciphertexts).

It remains to verify that the field element z produced in the simulation matches that of H . In H , C_S decrypts $\bar{y}$ to obtain $z_H = (a + a' + a'') \cdot x + b + r + r' + r''$; in the simulation, the simulator emits $z_S = \hat{y} - b' - b'' + r + r' + r''$. Substituting $\hat{y} = (a + a' + a'')x + (b + b' + b'')$ yields

$$z_S = (a + a' + a'')x + b + r + r' + r'' = z_H.$$

Hence $z_S = z_H$ identically. All other messages in the simulation agree with H by construction (the simulator executes the same firewall code on the same masks). Therefore the simulation is computationally indistinguishable from H , and consequently from the real execution:

$$\text{Exec}_{\hat{\Pi}, A, Z}^{\mathcal{F}_{\text{SAT}}} \stackrel{c}{\approx} \text{Exec}_{\hat{G}, S, Z}.$$

□

Lemma 5. *The sender's firewall F_S in $\hat{\Pi}_{\text{OLE}}^{(3)}$ is strongly sanitizing.*

Proof. We show that no environment Z that allows specious corruptions of the core C_S but keeps F_S honest is able to distinguish an execution of $\hat{\Pi}$ with an incorruptible C_S from one in which the corruption $\bar{C}_S$ is accepted. There are

three places in which $\widetilde{C}_S$ interacts with the outside world: the outbound Round-1 message $(\text{pk}, \langle a \rangle, \langle b \rangle)$, the inbound Round-2 message y (which C_S decrypts), and the outbound Round-3 message z . We argue that F_S neutralizes leakage in each direction.

Round 1 (outbound leakage). The specious core $\widetilde{C}_S$ may generate the keypair (pk, sk) and the ciphertexts $\langle a \rangle, \langle b \rangle$ from a biased distribution in order to encode a subliminal signal. The firewall replaces pk by $\widetilde{\text{pk}} \leftarrow \text{MaulPK}(\text{pk}, \tau)$. By property (KM3) of [Definition 4](#), $\widetilde{\text{pk}}$ is statistically close to a freshly sampled public key whose marginal distribution carries no information about pk . The ciphertexts are then replaced by $\widetilde{\langle a \rangle}$ and $\widetilde{\langle b \rangle}$

$$\widetilde{\langle a \rangle} = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(\langle a \rangle, \langle a' \rangle), \widetilde{\text{pk}}, \tau)), \quad \widetilde{\langle b \rangle} = \text{Rerand}(\text{AlignEnc}(\langle b \rangle, \widetilde{\text{pk}}, \tau)),$$

with $a' \leftarrow \mathbb{F}$ sampled inside F_S . By [Definition 4](#) and the freshness guarantee of Rerand , $\widetilde{\langle a \rangle}$ and $\widetilde{\langle b \rangle}$ are statistically close to fresh encryptions $\text{Enc}(\widetilde{\text{pk}}, a + a')$ and $\text{Enc}(\widetilde{\text{pk}}, b)$ respectively, regardless of how $\langle a \rangle, \langle b \rangle$ were originally produced. If any of $\text{pk}, \langle a \rangle, \langle b \rangle$ cannot be parsed correctly, the message is dropped. Hence the outbound tuple carries no subliminal information.

Round 2 (inbound triggering). The only inbound message C_S receives is y (the ciphertext returned by F_S after masking the value $\widetilde{y}$ obtained from F_R and applying $\text{Rerand} \circ \text{AlignEnc}$). This is the channel through which a potentially adversarial receiver might attempt to inject a triggering input. The firewall samples $r' \xleftarrow{\$} \mathcal{M}$ and computes

$$y = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(\widetilde{y}, \widetilde{\langle r' \rangle}), \text{pk}, \tau)).$$

By (KM4) of [Definition 4](#) together with Rerand , the ciphertext y is statistically close to a fresh $\text{Enc}(\text{pk}, m + r')$, where m is the plaintext underlying $\widetilde{y}$. The bit-level encoding of y is therefore uniformly distributed over the ciphertext space and, since r' is fresh and independent of m , so is the field element that C_S decrypts. Hence any trigger condition on the encoding of y or on its decryption fires with probability $\text{negl}(\lambda)$. If $\widetilde{y}$ is not a well-formed ciphertext, the message is dropped.

Round 3 (outbound leakage). The specious core may attempt to leak via the field element $z = \text{Dec}(\text{sk}, y)$ that it forwards to F_S . The firewall samples $b' \xleftarrow{\$} \mathbb{F}$, strips its own mask r' and outputs $\widetilde{z} = \text{ADD}(\text{ADD}(z, -r'), b')$ to F_R . Since b' is a fresh element of $\mathbb{F}$ drawn independently of z , the value $\widetilde{z}$ is uniformly distributed over $\mathbb{F}$ regardless of z . Any subliminal payload embedded in z by $\widetilde{C}_S$ is therefore information-theoretically destroyed.

Combining the three steps, the joint view of $\mathcal{Z}$ in the execution with $\widetilde{C}_S$ is statistically close to its view in the execution with an incorruptible C_S . $\square$

Lemma 6. *The receiver's firewall F_R in $\widehat{\Pi}_{\text{OLE}}^{(3)}$ is strongly sanitizing.*

Proof. The argument is symmetric to that of [Lemma 5](#). C_R interacts with the outside world through the inbound Round-1 message $(\widehat{pk}, \langle \widehat{a} \rangle, \langle \widehat{b} \rangle)$, the outbound Round-2 message y , and the inbound Round-3 message $\widehat{z}$.

Round 1 (inbound triggering). The firewall computes $(\widehat{pk}, \rho) \leftarrow \text{MaulPK}(\widetilde{pk})$ and then

$$\langle \widehat{a} \rangle = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(\langle \widetilde{a} \rangle, \langle \widetilde{a''} \rangle), \widehat{pk}, \rho)), \quad \langle \widehat{b} \rangle = \text{Rerand}(\text{AlignEnc}(\langle \widetilde{b} \rangle, \widehat{pk}, \rho)),$$

with $a'' \xleftarrow{\$} \mathbb{F}$ chosen inside F_R . By (KM3) the key $\widehat{pk}$ is statistically close to a fresh public key; by (KM2)/(KM4) and *Rerand*, $\langle \widehat{a} \rangle$ and $\langle \widehat{b} \rangle$ are statistically close to fresh encryptions of $a + a' + a''$ and b under $\widehat{pk}$. Hence the tuple delivered to C_R is uniformly distributed independently of any subliminal pattern that the (possibly malicious) sender's side might have planted in the incoming $(\widetilde{pk}, \langle \widetilde{a} \rangle, \langle \widetilde{b} \rangle)$. Malformed inbound messages are dropped.

Round 2 (outbound leakage). The specious core $\widetilde{C}_R$ might produce the ciphertext y from a biased distribution. The firewall samples $r'' \xleftarrow{\$} \mathcal{M}$ and returns $\widetilde{y} = \text{Rerand}(\text{AlignEnc}(\text{HomOp}(y, \langle r'' \rangle), \widetilde{pk}, \rho))$ to F_S . By (KM4) and *Rerand*, $\widetilde{y}$ is statistically close to a fresh encryption of $\text{Dec}(\widehat{sk}, y) + r''$ under $\widetilde{pk}$, so its bit-level encoding is uniform and, since r'' is fresh, so is its plaintext; it carries no signaling information.

Round 3 (inbound triggering). The firewall samples $b'' \xleftarrow{\$} \mathbb{F}$, strips its own mask r'' and delivers $\widehat{z} = \text{ADD}(\text{ADD}(\widetilde{z}, -r''), b'')$ to C_R . Since b'' is drawn fresh from $\mathbb{F}$, the value $\widehat{z}$ is uniformly distributed over $\mathbb{F}$ irrespective of $\widetilde{z}$; thus no specific triggering field element can be steered to C_R except with probability $\text{negl}(\lambda)$.

Combining the three steps, the view of any environment $\mathcal{Z}$ in the execution with $\widetilde{C}_R$ is statistically close to its view with an incorruptible C_R . $\square$

By the corruption translation table ([Table 1](#)) and [Lemma 1](#), in the semi-honest setting it suffices to handle two corruption configurations: the one where the corrupted party has both its core and its firewall semi-honestly corrupted (and the firewall of the other party is semi-honestly corrupted while its core is honest), and the one where the honest party's core is SPECIOUS. By [Lemma 5](#) and [6](#), both firewalls are strongly sanitizing. Hence by [Lemma 1](#) the SPECIOUS-core and HONEST-firewall configuration is computationally indistinguishable from the HONEST-core and HONEST-firewall configuration, which is itself subsumed by the HONEST-core and SEMIHONEST-firewall configuration. We conclude that $\widehat{\Pi}_{OLE}^{(3)}$ srUC-realizes $\mathcal{F}_{OLE}$ in the $\mathcal{F}_{SAT}$ -hybrid model under semi-honest corruptions. $\square$

B.8 Proof of [Lemma 7](#)

Lemma 7. *The protocol $\Pi_{OLE}^{(2)}$ is correct.*

Proof. In the protocol, the receiver computes $\langle x \rangle = \text{Enc}(\text{pk}, x)$ and sends $(\text{pk}, \langle x \rangle)$ it to the sender. The sender computes: $t = \text{CMult}(\langle x \rangle, a) = \text{Enc}(\text{pk}, ax)$ followed by $u = \text{Enc}(\text{pk}, b)$ by the homomorphic property. Finally the sender compute $y = \text{HomOp}(t, u) = \text{Enc}(\text{pk}, ax + b)$. The receiver decrypts: $\text{Dec}(\text{sk}, y) = ax + b$, where the last equality follows from the correctness and additive homomorphism of AHE. Hence the receiver obtains the correct output. $\square$

B.9 Proof of Theorem 4

Proof. For each non-trivial corruption $I \in \{\text{S}, \text{R}\}$ we exhibit a PPT simulator $\mathcal{S}_I$ such that for every PPT environment $\mathcal{Z}$

$$\text{Exec}_{\Pi_{\text{OLE}}^{(2)}, \mathcal{A}, \mathcal{Z}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_I, \mathcal{Z}}.$$

We argue each case by an explicit hybrid sequence over the transcript $\tau = (\text{pk}, \langle x \rangle, y)$.

Case 1: Corrupted Sender. The corrupted party $\mathcal{A}_\text{S}$ sees $(\text{pk}, \langle x \rangle)$ and produces y . The simulator must generate $(\text{pk}, \langle x \rangle)$ without knowing the receiver's input x .

Simulator $\mathcal{S}_\text{S}$

1. Sample $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and $\tilde{x} \xleftarrow{\$} \mathcal{M}$.
2. Send $(\text{pk}, \text{Enc}(\text{pk}, \tilde{x}))$ to $\mathcal{A}_\text{S}$ as the simulated round-1 message.
3. Read (a, b) from $\mathcal{A}_\text{S}$'s input tape and submit $(\text{sid}, (a, b))$ to $\mathcal{F}_{\text{OLE}}$.
4. Receive $\mathcal{A}_\text{S}$'s round-2 message y and discard it.

We now show that for every PPT environment $\mathcal{Z}$, the two executions $\text{Exec}_{\Pi_{\text{OLE}}^{(2)}, \mathcal{A}, \mathcal{Z}}$ and $\text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_\text{S}, \mathcal{Z}}$ are indistinguishable. Consider the following series of hybrid experiments.

- H_0 . This is the real world. $\mathcal{A}_\text{S}$ receives $(\text{pk}, \langle x \rangle)$ with $\langle x \rangle = \text{Enc}(\text{pk}, x)$; the honest receiver outputs $\text{Dec}(\text{sk}, y)$.
- H_1 . Identical to H_0 , except the receiver's output is computed algebraically as $a \cdot x + b$, rather than by decrypting y . The rest of the transcript shown to $\mathcal{A}_\text{S}$ is unchanged from H_0 .
- H_2 . This is the ideal world. Identical to H_1 , except $\langle x \rangle$ is replaced by $\text{Enc}(\text{pk}, \tilde{x})$ for a fresh uniform $\tilde{x} \xleftarrow{\$} \mathcal{M}$.

$\text{H}_0 \equiv \text{H}_1$. The view of $\mathcal{A}_\text{S}$ is sampled identically in both hybrids. Since $\mathcal{A}_\text{S}$ is semi-honest, y is the honest evaluation and, by correctness of AHE together with the homomorphism of HomOp and CMult , $\text{Dec}(\text{sk}, y) = a \cdot x + b$, which coincides with the value computed in H_1 . Hence the joint distributions are identical.

$\text{H}_1 \stackrel{c}{\approx} \text{H}_2$. The only difference between the two experiments is that the plaintext encrypted under pk switches from x to an independent uniform $\tilde{x}$. Any PPT distinguisher with advantage ε yields an IND-CPA adversary with the same advantage: on receiving challenge $c^* \leftarrow \text{Enc}(\text{pk}, m_\beta)$ for $(m_0, m_1) = (x, \tilde{x})$, simulate

the rest of the joint view (in particular, $z = ax + b$ is computed algebraically without invoking sk), and output the distinguisher's bit. If $\beta = 1$ the joint view matches H_1 ; if $\beta = 0$, it matches H_2 . Hence the two hybrids are computationally indistinguishable. Hence, $H_1 \stackrel{c}{\approx} H_2$.

$H_2 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_S, \mathcal{Z}}$ by inspection of $\mathcal{S}_S$: the simulated round-1 message is exactly $(\text{pk}, \text{Enc}(\text{pk}, \tilde{x}))$, and the receiver's output $a \cdot x + b$ is delivered by $\mathcal{F}_{\text{OLE}}$.

Combining, we obtain $\text{Exec}_{\Pi_{\text{OLE}, \mathcal{A}_S, \mathcal{Z}}^{(2)}} = H_0 \equiv H_1 \stackrel{c}{\approx} H_2 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_S, \mathcal{Z}}$.

Case 2: Corrupted Receiver. The corrupted party $\mathcal{A}_R$ produces $(\text{pk}, \langle x \rangle)$ in round 1 and receives y in round 2. The simulator must produce y without knowing the sender's input (a, b) , using only the output value $z^* = a \cdot x + b$ obtained from $\mathcal{F}_{\text{OLE}}$.

Simulator $\mathcal{S}_R$

1. Receive $\mathcal{A}_R$'s round-1 message $(\text{pk}, \langle x \rangle)$ and read x from $\mathcal{A}_R$'s input tape.
2. Submit (sid, x) to $\mathcal{F}_{\text{OLE}}$ and obtain $z^* = a \cdot x + b$.
3. Send $\tilde{y} \leftarrow \text{Enc}(\text{pk}, z)$ to $\mathcal{A}_R$.

We define the following sequence hybrids,

H_0 . This is the real world. The sender returns $y = \text{HomOp}(\text{CMult}(\langle x \rangle, a), \text{Enc}(\text{pk}, b))$; $\mathcal{A}_R$ outputs $\text{Dec}(\text{sk}, y)$.

H_1 . This is the ideal world. Identical to H_0 , except y is replaced by $\tilde{y} \leftarrow \text{Enc}(\text{pk}, a \cdot x + b)$, a fresh encryption of $a \cdot x + b$.

$H_0 \equiv H_1$. Apply, in order, the equality guarantees of CMult and HomOp :

$$\begin{aligned} \text{CMult}(\langle x \rangle, a) &= \text{Enc}(\text{pk}, a \cdot x) \\ \text{HomOp}(\text{Enc}(\text{pk}, a \cdot x), \text{Enc}(\text{pk}, b)) &= \text{Enc}(\text{pk}, a \cdot x + b), \end{aligned}$$

where the second relation uses that $\text{Enc}(\text{pk}, b)$ is sampled with independent fresh coins by the honest sender. Hence in H_0 the ciphertext y is distributed as a fresh encryption of $a \cdot x + b$ under pk , identically to $\tilde{y}$ in H_1 . The receiver's output agrees: $\text{Dec}(\text{sk}, \tilde{y}) = a \cdot x + b$ by correctness of AHE.

$H_1 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_R, \mathcal{Z}}$ by inspection of $\mathcal{S}_R$. Therefore we obtain,

$\text{Exec}_{\Pi_{\text{OLE}, \mathcal{A}_R, \mathcal{Z}}^{(2)}} = H_0 \equiv H_1 = \text{Exec}_{\mathcal{F}_{\text{OLE}}, \mathcal{S}_R, \mathcal{Z}}$. This completes the proof. $\square$

B.10 Proof of Lemma 8

Lemma 8. *The Protocol $\widehat{\Pi}_{\text{OLE}}^{(2)}$ is correct.*

Proof. $\mathcal{C}_R$ encrypts x under pk . The firewalls $\mathcal{F}_R$ and $\mathcal{F}_S$ successively apply MaulPK to change the public key to pk' and then $\widetilde{\text{pk}}$, and then rerandomize and align the ciphertext so that $\mathcal{C}_S$ receives $\langle \widetilde{x} \rangle$ encrypting x under $\widetilde{\text{pk}}$. Then, $\mathcal{C}_S$ computes,

$$\widetilde{z} = \text{HomOp}(\text{CMult}(\langle \widetilde{x} \rangle, a), \text{Enc}(\widetilde{\text{pk}}, b)) = \text{Enc}(\widetilde{\text{pk}}, a \cdot x + b),$$

by the homomorphic properties of AHE. The firewall F_S aligns $\tilde{z}$ back under pk' : $z' \leftarrow \text{Rerand}(\text{AlignEnc}(\tilde{z}, pk', \rho))$. By [Definition 4](#), $\text{AlignEnc}(\tilde{z}, pk', \rho)$ is a ciphertext under pk' encrypting $a \cdot x + b$, and Rerand preserves this plaintext. The firewall F_R then aligns z' back under pk and obtaining $z \leftarrow \text{Rerand}(\text{AlignEnc}(z', pk, \gamma))$, which is a ciphertext under pk encrypting $a \cdot x + b$. $\square$

B.11 Proof of [Theorem 5](#)

To simplify notation, write $\hat{\Pi} := \hat{\Pi}_{\text{OLE}}^{(2)}$ and $\mathcal{G} = \mathcal{F}_{\text{OLE}}$. We first construct a simulator covering the corruption pattern “HONEST core and SEMIHONEST firewall” on the unaffected party, and then show that the firewalls F_S and F_R are strongly sanitizing.

Lemma 9. *For every PPT adversary $\mathcal{A}$ corrupting the core and firewall of at most one party semi-honestly in an execution of $\hat{\Pi}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model, there exists a PPT simulator $\mathcal{S}$ such that for every PPT environment $\mathcal{Z}$,*

$$\text{Exec}_{\hat{\Pi}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{G}), \mathcal{S}, \mathcal{Z}}.$$

Proof. There are two corruption cases: corrupted (C_S, F_S) and corrupted (C_R, F_R) . We treat each of them separately.

Case 1: Corrupted Sender. On reading (a, b) from C_S 's tape, $\mathcal{S}_S$ forwards $(\text{sid}, (a, b))$ to $\mathcal{G}$ and simulates the honest receiver side

1. Sample $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$, $\tilde{x} \xleftarrow{\$} \mathcal{M}$.
2. Set $(pk', \gamma) \leftarrow \text{MaulPK}(pk)$ and $\langle x \rangle' \leftarrow \text{Rerand}(\text{AlignEnc}(\text{Enc}(pk, \tilde{x}), pk', \gamma))$.
3. Submit $(pk', \langle x \rangle')$ on F_R 's sanitization interface to $\mathcal{A}$.
4. Receive $\mathcal{A}$'s round-2 message z' and discard it.

H_0 . The honest C_R, F_R produce $(pk', \langle x \rangle')$ with

$$\langle x \rangle' = \text{Rerand}(\text{AlignEnc}(\text{Enc}(pk, x), pk', \gamma))$$

the output is $\text{Dec}(sk, z)$.

H_1 . Replace $\langle x \rangle'$ by a fresh $\text{Enc}(pk', x)$, and replace the receiver's output by the algebraic value $a \cdot x + b$. By properties of AlignEnc and Rerand ,

$$\text{Rerand}(\text{AlignEnc}(\text{Enc}(pk, x), pk', \gamma)) \equiv \text{Enc}(pk', x);$$

and by correctness of the inner OLE evaluation $\text{Dec}(sk, z) = a \cdot x + b$. Hence $H_0 \stackrel{c}{\approx} H_1$.

H_2 . This is the ideal world. Replace the encrypted plaintext x by $\tilde{x} \xleftarrow{\$} \mathcal{M}$; this is the view produced by $\mathcal{S}_S$, with the output delivered by $\mathcal{G}$.

$H_1 \stackrel{c}{\approx} H_2$. The only difference between H_1 and H_2 is the plaintext encrypted under $\mathbf{pk}'$. Given a distinguisher D with advantage ε , construct an IND-CPA adversary $\mathcal{B}$: on receiving the challenge key $\mathbf{pk}^*$, $\mathcal{B}$ sets $\mathbf{pk}' := \mathbf{pk}^*$ (this matches the honest distribution of $\mathbf{pk}'$ by (i) above), extracts x from $\mathcal{Z}$, samples $\tilde{x} \stackrel{\$}{\leftarrow} \mathcal{M}$, submits the challenge pair $(m_0, m_1) := (x, \tilde{x})$, and uses the challenge ciphertext c^* as the round-1 message $\langle x \rangle'$. The receiver's output is computed algebraically as $a \cdot x + b$. The joint view matches H_1 when the challenge bit is 1 and H_2 when it is 0. This is precisely the reduction used in Case 1 of [Theorem 4](#), with $\mathbf{pk}$ replaced by $\mathbf{pk}'$.

Case 2: Corrupted Receiver. The simulator $\mathcal{S}_R$ reads x from C_R 's tape, observes $\mathcal{A}$ run the honest receiver side to produce $(\mathbf{pk}, \mathbf{sk}, \gamma, \mathbf{pk}', \langle x \rangle, \langle x \rangle')$ on its tapes, queries $\mathcal{G}$ on (sid, x) , and obtains $z^* = a \cdot x + b$. It then simulates the sender-side return message:

1. Sample $\tilde{z}' \leftarrow \text{Enc}(\mathbf{pk}', z^*)$ with fresh randomness.
2. Submit $\tilde{z}'$ on F_S 's sanitization interface; the firewall then proceeds honestly to produce z delivered to C_R .

H_0 . This is the real world. C_S computes $\tilde{z} \leftarrow \text{HomOp}(\text{CMult}(\langle \tilde{x} \rangle, a), \text{Enc}(\mathbf{pk}, b))$, and F_S outputs $z' = \text{Rerand}(\text{AlignEnc}(\tilde{z}, \mathbf{pk}', \rho))$.

H_1 . This is the ideal world. Replace z' by a fresh $\text{Enc}(\mathbf{pk}', a \cdot x + b)$.

By the homomorphism and correctness of AHE together with the distribution-preserving properties of CMult and HomOp , the ciphertext $\tilde{z}$ produced by C_S in H_0 is distributed as a fresh encryption of $a \cdot x + b$ under $\mathbf{pk}$ (because $\text{Enc}(\mathbf{pk}, b)$ contributes fresh independent randomness). Applying properties of AlignEnc and Rerand produces an output independent of its input among encryptions of the same plaintext), $\text{Rerand}(\text{AlignEnc}(\tilde{z}, \mathbf{pk}', \rho))$ is computationally indistinguishable from $\text{Enc}(\mathbf{pk}', a \cdot x + b)$. Hence $H_0 \stackrel{c}{\approx} H_1 = \text{Exec}_{\mathcal{G}, S_R, \mathcal{Z}}$. The receiver's final output, computed by $\mathcal{A}$ as $\text{Dec}(\mathbf{sk}, \text{Rerand}(\text{AlignEnc}(\tilde{z}', \mathbf{pk}, \gamma)))$, equals $a \cdot x + b$ by correctness of AHE. This is exactly the simulation strategy of Case 2 of [Theorem 4](#) lifted through the firewall layer by properties of Rerand and AlignEnc . $\square$

Lemma 10. *The sender's firewall F_S is strongly sanitizing.*

Proof. Fix a PPT environment $\mathcal{Z}$ that does not corrupt F_S but mounts $(\text{SPECIOUS}, \tilde{C}_S)$ on the sender's core C_S , and let $\mathcal{A}$ be the dummy adversary, so that $\mathcal{Z}$ plays the receiver side. We must show

$$\text{Exec}_{\hat{\Pi}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} \stackrel{c}{\approx} \text{Exec}_{\hat{\Pi}', \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}}$$

where $\hat{\Pi}'$ replaces $\tilde{C}_S$ by an incorruptible core $\hat{C}_S$. The round-1 pair $(\mathbf{pk}, \langle x \rangle)$ delivered to the sender's core, and the round-2 ciphertext $z' = \text{Rerand}(\text{AlignEnc}(\tilde{z}, \mathbf{pk}', \rho))$.

For the round-1 message, the firewall's input $(\text{pk}', \langle x \rangle')$ is produced entirely by the receiver side, which is played by $\mathcal{Z}$ and acts before any sender message arrives; its distribution therefore has no dependence on which sender core is plugged in.

The honest firewall samples a fresh $\rho \xleftarrow{\$} \{0, 1\}^n$ together with fresh internal randomness for Rerand , then deterministically computes $\widetilde{\text{pk}} \leftarrow \text{MaulPK}(\text{pk}', \rho)$ and $\widetilde{\langle x \rangle} = \text{Rerand}(\text{AlignEnc}(\langle x \rangle', \widetilde{\text{pk}}, \rho))$. With identically distributed inputs, $(\widetilde{\text{pk}}, \widetilde{\langle x \rangle})$ is identically distributed in the two executions.

For the round-2 message, write $\widetilde{z}_{\widetilde{\mathcal{C}}}$ and $\widetilde{z}_{\widehat{\mathcal{C}}}$ for the outputs of $\widetilde{\mathcal{C}}_S$ and $\widehat{\mathcal{C}}_S$ on input (a, b) and round-1 message $(\widetilde{\text{pk}}, \widetilde{\langle x \rangle})$. Speciousness of $\widetilde{\mathcal{C}}_S$, instantiated with any tester that runs F_S honestly to generate the round-1 message and then feeds it together with (a, b) to the target core, gives

$$(\widetilde{\langle x \rangle}, \widetilde{\text{pk}}, \widetilde{z}_{\widetilde{\mathcal{C}}}) \stackrel{c}{\approx} (\widetilde{\langle x \rangle}, \widetilde{\text{pk}}, \widetilde{z}_{\widehat{\mathcal{C}}}).$$

The firewall transforms $\widetilde{z}$ via the PPT map $\Phi(\widetilde{z}; \text{pk}', \rho) = \text{Rerand}(\text{AlignEnc}(\widetilde{z}, \text{pk}', \rho))$, which depends on $(\widetilde{z}, \text{pk}', \rho)$ and fresh Rerand randomness. Since computational indistinguishability is preserved under PPT transformations, applying Φ to both sides yields indistinguishability of z' . Combining the two messages, $\mathcal{Z}$'s joint view is computationally indistinguishable across the two executions. $\square$

Lemma 11. *The receiver's firewall F_R in the protocol is strongly sanitizing.*

Proof. Fix a PPT environment $\mathcal{Z}$ that does not corrupt F_R but mounts $(\text{SPECIOUS}, \widetilde{\mathcal{C}}_R)$ on the receiver's core, and let $\mathcal{A}$ be the dummy adversary, so $\mathcal{Z}$ plays the sender side and additionally observes the receiver's final output. We must show

$$\text{Exec}_{\widehat{\Pi}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}} \stackrel{c}{\approx} \text{Exec}_{\widehat{\Pi}', \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}}.$$

where $\widehat{\Pi}'$ replaces $\widetilde{\mathcal{C}}_R$ by an incorruptible $\widehat{\mathcal{C}}_R$. Three quantities produced or routed by F_R enter $\mathcal{Z}$'s view: the round-1 message $(\text{pk}', \langle x \rangle')$, the round-2 ciphertext z delivered to the receiver's core, and the receiver's final output y_{out} . For the round-1 message, the firewall receives $(\text{pk}, \langle x \rangle)$ from the receiver's core. Speciousness of $\widetilde{\mathcal{C}}_R$, applied with the tester that samples a random input x , delivers it, and records the emitted public key and ciphertext, gives $(x, \text{pk}, \langle x \rangle)|_{\widetilde{\mathcal{C}}_R} \stackrel{c}{\approx} (x, \text{pk}, \langle x \rangle)|_{\widehat{\mathcal{C}}_R}$. The firewall then samples fresh $\gamma \xleftarrow{\$} \{0, 1\}^n$ together with fresh Rerand randomness and deterministically computes $\text{pk}' \leftarrow \text{MaulPK}(\text{pk}, \gamma)$ and $\langle x \rangle' = \text{Rerand}(\text{AlignEnc}(\langle x \rangle, \text{pk}', \gamma))$. Property (KM3) ensures that for uniform γ , the key pk' is statistically close to a fresh KeyGen output and statistically independent of pk ; property (KM2) followed by Rerand ensures that $\langle x \rangle'$ is statistically close to a fresh $\text{Enc}(\text{pk}', x)$. Hence $(\text{pk}', \langle x \rangle')$ is statistically close to the same distribution in both executions.

For the round-2 message, $z = \text{Rerand}(\text{AlignEnc}(z', \text{pk}, \gamma))$ is a deterministic PPT function of (z', pk, γ) and fresh Rerand randomness. Inside $\mathcal{Z}$'s sender simulation, z' is computed from $\mathcal{Z}$'s local randomness and the round-1 message $(\text{pk}', \langle x \rangle')$;

conditioning on the same realization of $(pk', \langle x \rangle')$, which is statistically close across the two experiments by the previous paragraph, and using that γ and pk are sampled identically, z is statistically close in $\widehat{\Pi}$ and $\widehat{\Pi}'$.

For the final output, in $\widehat{\Pi}'$ the incorruptible core decrypts to $y_{\text{out}} = \text{Dec}(\text{sk}, z) = a \cdot x + b$ by correctness of AHE and plaintext-preservation of AlignEnc. Speciousness of $\widetilde{C}_R$ on the same input distribution z then yields $y_{\text{out}}|_{\widetilde{C}_R} \stackrel{c}{\approx} y_{\text{out}}|_{\widehat{C}_R}$. Combining all three contributions, $\mathcal{Z}$'s view in the two executions is computationally indistinguishable. $\square$

C Randomized Equality

Recall the $\mathcal{F}_{\text{RE}}$ functionality from [CMMV24]. We instantiate randomized equality in the $\mathcal{F}_{\text{sOLE}}$ -hybrid model, following the template of [CMMV24] but replacing $\mathcal{F}_{\text{sOT}}$ with our sanitizable OLE functionality of Figure 7. Because each invocation of $\mathcal{F}_{\text{sOLE}}$ accepts inputs from a field $\mathbb{F}$ with $|\mathbb{F}| \geq 2^n$, a password pw^I is first hashed into the field $\mathbb{F}$ as $w^I = H(pw^I)$ and viewed as a n -bit element. Then w^I is processed as a *single* field element. Hence, our construction requires only *two* symmetric invocations of $\mathcal{F}_{\text{sOLE}}$, in contrast to the $2n$ sOT invocations needed in [CMMV24, Fig. 4].

Protocol $\widehat{\Pi}_{\text{RE}}$. Let $w^I, w^R \in \mathbb{F}$ denote the hash of passwords of the initiator $I = (C_I, F_I)$ and the responder $R = (C_R, F_R)$, respectively. The cores sample uniform masking scalars $a^I, a^R \xleftarrow{\$} \mathbb{F}$ and invoke $\mathcal{F}_{\text{sOLE}}$ twice, with swapped roles:

- *First invocation.* C_I plays the sender with input $(a^I, 0)$, and C_R plays the receiver with input w^R . After sanitation by both firewalls via BLIND masks (a', b') and (a'', b'') , the functionality delivers

$$v^R = (a^I + a' + a'')w^R + b' + b'' \quad \text{to} \quad C_R.$$

- *Second invocation.* The roles are reversed: C_R plays the sender with input $(a^R, 0)$, and C_I plays the receiver with input w^I . The functionality delivers

$$v^I = (a^R + a' + a'')w^I + b' + b'' \quad \text{to} \quad C_I.$$

Each core then locally derives its session key by adding the contribution of its own mask against its own password:

$$\begin{aligned} \text{skey}^I &:= a^I w^I + v^I = (a^I + a^R + a' + a'')w^I + b' + b'', \\ \text{skey}^R &:= a^R w^R + v^R = (a^I + a^R + a' + a'')w^R + b' + b''. \end{aligned}$$

Lemma 12 (Correctness). *If $w^I = w^R$, then $\text{skey}^I = \text{skey}^R$. If $w^I \neq w^R$, then skey^I and skey^R are uniform over $\mathbb{F}$.*

Proof. Substituting the values delivered by $\mathcal{F}_{\text{sOLE}}$,

$$\begin{aligned} \text{skey}^I &:= a^I w^I + v^I = (a^I + a^R + a' + a'')w^I + b' + b'' \\ \text{skey}^R &:= a^R w^R + v^R = (a^I + a^R + a' + a'')w^R + b' + b''. \end{aligned}$$

When $w^I = w^R = w$, both keys equal $a^I w^I + v^I = (a^I + a^R + a' + a'')w^I + b' + b''$. When $w^I \neq w^R$, we have $\text{skey}^I - \text{skey}^R = (a^I + a^R + a' + a'')(w^I - w^R)$; since at least one of a^I, a^R is uniform and unknown to the peer and a', a'' are uniform, $(w^I - w^R) \in \mathbb{F}^*$, the difference is uniform in $\mathbb{F}$. $\square$

The protocol Π_{RE} can be shown secure in $\mathcal{F}_{\text{sOLE}}$ -hybrid model. We leave any further developments on Randomized Equality from $\mathcal{F}_{\text{sOLE}}$ as an avenue for future work.

Theorem 14. *The protocol Π_{RE} wsrUC-realizes $\mathcal{F}_{\text{RE}}$ in $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{sOLE}})$ -hybrid model.*

Proof (Proof Sketch). The argument follows the template of the proof of [CMMV24, Theorem 3, Appendix C.3], with the two batches of n sanitizable OTs collapsed to two symmetric calls to $\mathcal{F}_{\text{sOLE}}$ over a single field element (the hashed password). By the corruption translation Table 1 together with Lemma 1, it suffices to argue (i) strong sanitation of F_I and F_R , and (ii) the existence of a UC simulator against malicious corruption of either party. For (i), in each $\mathcal{F}_{\text{sOLE}}$ invocation the honest firewall submits a uniform blinding mask (a', b') (resp. (a'', b'')) on its S_S/S_R interface, and $\mathcal{F}_{\text{sOLE}}$ folds this mask multiplicatively into the sender's coefficients before they are consumed. Hence any subliminal payload that a specious core may have embedded in its $\mathcal{F}_{\text{sOLE}}$ input is destroyed inside the functionality, while any triggering value reaching the specious core through the $\mathcal{F}_{\text{sOLE}}$ output is uniform over $\mathbb{F}$. Because the peer firewall's mask is unknown to the corrupted side. So trigger inputs fire only with probability $1/(|\mathbb{F}| - 1) = \text{negl}(\lambda)$ (this is the OLE-level analogue of the " ρ_b^R blinding" argument of [CMMV24, Lemma 8]). For (ii), the simulator $\mathcal{S}$ plays the hybrid $\mathcal{F}_{\text{sOLE}}$ toward the adversary $\mathcal{A}$ and thereby learns the corrupted party's password w^P from its receiver-input to the $\mathcal{F}_{\text{sOLE}}$ call in which it plays the receiver; $\mathcal{S}$ forwards w^P to $\mathcal{F}_{\text{RE}}$, locally assembles the key $\text{skey}_P = v^P + a^P w^P$ exactly as the corrupted core would, and issues a matching NEWKEY query to $\mathcal{F}_{\text{RE}}$ on behalf of both parties. By correctness of $\mathcal{F}_{\text{RE}}$, the assembled key matches the value $\mathcal{F}_{\text{RE}}$ delivers to the honest peer whenever $w^I = w^R$, and the difference is uniform in $\mathbb{F}$ otherwise (since at least one of a^I, a^R is uniform and unknown to $\mathcal{A}$, and a', a'' are uniform firewall masks). Treating ISOLATE as MALICIOUS corruption of the peer and combining the two configurations yields

$$\text{Exec}_{\Pi_{\text{RE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{sOLE}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{RE}}), \mathcal{S}, \mathcal{Z}}$$

which is the wsrUC guarantee of Theorem 14. $\square$

D Correctness and Security of Class Group based Key Malleable AHE

Hardness assumption on class groups. All the assumptions presented below are based on a common setup. Given the computational security parameter λ , the statistical security parameter λ_{st} , and a prime q , let the public parameters be generated as $\text{pp}_{\text{CG}} := (U, q, \mathbb{G}_{\text{CL}}, \mathbb{F}_{\text{CL}}, \mathbb{G}_{\text{CL}}^q, g_q, f, \mathcal{D}, \mathcal{D}_q, \rho) \leftarrow \text{CGen}(1^\lambda, 1^{\lambda_{st}}, q)$.

Definition 7 (Hard subgroup membership assumption [CLT18]). Draw $x \xleftarrow{\$} \mathcal{D}_q$ and $u \xleftarrow{\$} \mathbb{Z}_q$. Let $b \xleftarrow{\$} \{0, 1\}$ be sampled uniformly at random. If $b = 0$, set $h^* \leftarrow g_q^x$, whereas if $b = 1$, set $h^* \leftarrow f^u \cdot g_q^x$. We say that the hard subgroup membership assumption holds in the class group framework if, for every PPT adversary $\mathcal{A}$, we obtain $|\Pr[b = b^* \mid b^* \leftarrow \mathcal{A}(\text{pp}_{\text{CG}}, h^*)^{\text{Solve}(\cdot)}] - \frac{1}{2}| \leq \text{negl}(\lambda)$.

Correctness follows from the ElGamal-type structure of the class group encryption. For a ciphertext $c = (R, E) = (g_q^r, f^m \text{pk}^r)$ under $\text{pk} = g_q^{\text{sk}}$, decryption computes $M = E \cdot (R^{\text{sk}})^{-1} = f^m \text{pk}^r \cdot (g_q^{\text{sk}})^{-1} = f^m$, and hence $\text{Solve}(\text{pp}_{\text{CG}}, M)$ recovers $m \in \mathbb{Z}_p$ from the easy-DL subgroup $\langle f \rangle$. The homomorphic operations are also correct, since $(R_1, E_1) \boxplus (R_2, E_2)$ decrypts to $m_1 + m_2 \bmod p$, while $\text{CMult}((R, E), \alpha)$ decrypts to $\alpha m \bmod p$. Rerandomization preserves the plaintext because it adds an encryption of 0. The key-malleability procedures are correct as well: if $\text{pk}' = \text{pk} \cdot g_q^{r'} = g_q^{\text{sk}+r'}$, then $\text{sk}' = \text{sk} + r'$ decrypts ciphertexts under pk' ; moreover, $\text{MaulCT}(c, \tau)$ transforms $c = (R, E)$ into an encryption of the same message under pk' , since $E' = E \cdot R^{r'} = f^m \text{pk}^r g_q^{r r'} = f^m (\text{pk}')^r$, and AlignEnc performs the inverse transformation by multiplying by $\tilde{R}^{-r'}$, thereby bringing a ciphertext under pk' back to a ciphertext under pk , followed by fresh re-randomization. Security follows as in the Castagnos Laguillaumie scheme [CL15]: the encryption is IND-CPA-secure under the DDH assumption in the underlying class group G ; equivalently, any adversary distinguishing encryptions of two messages can be turned into an adversary distinguishing a Diffie–Hellman tuple from a random tuple in G . The paper [CL15] also shows that one-wayness under chosen-plaintext attack is equivalent to the hardness of the Lift Diffie–Hellman problem, and hence to the Partial Discrete Logarithm problem, while the class group instantiation relies on DDH in a non-maximal imaginary quadratic order and not on integer factorization. The additional operations ReRand , MaulPK , MaulSK , MaulCT , and AlignEnc do not harm CPA security: MaulPK produces a rerandomized public key distributed as a fresh key, and MaulCT and AlignEnc output ciphertexts statistically close to fresh encryptions of the same plaintext under the corresponding keys. Thus the scheme remains CPA-secure under the same DDH assumption even in the presence of these key-malleability properties.

E OPE and OPA Protocol Specifications

E.1 OPE Protocol

Protocol: Π_{OPE}

Let $\mathcal{F}_{\text{OLE}}$ be an ideal OLE functionality. The protocol runs between party A with input $(a_0, a_1, \dots, a_d) \in \mathbb{F}^{d+1}$ defining the polynomial $\mathbf{p}(X) = \sum_{j=0}^d a_j X^j$ and party B with input $\alpha \in \mathbb{F}$. Bob obtains $\mathbf{p}(\alpha)$, while A obtains $\perp$.

Party A:

1. Initialize $s_0^A \leftarrow a_d$

2. For each $i = 1, 2, \dots, d-1$: Sample a fresh mask $\rho_i \xleftarrow{\$} \mathbb{F}$. Set $A_i \leftarrow s_{i-1}^A$ and $B_i \leftarrow a_{d-i} - \rho_i$. Invoke $\mathcal{F}_{\text{OLE}}$ with sender input (A_i, B_i) . Set $s_i^A \leftarrow \rho_i$.
 3. For the d -th OLE party A provide $\mathcal{F}_{\text{OLE}}$ inputs $A_d = s_{d-1}^A = \rho_{d-1}$ and $B_d = a_0$.
- Party B:**
1. Initialize $s_0^B \leftarrow 0$.
 2. For each $i = 1, 2, \dots, d-1$: Invoke $\mathcal{F}_{\text{OLE}}$ with receiver input α . Receive $y_i = A_i \alpha + B_i = s_{i-1}^A \alpha + a_{d-i} - \rho_i$. Locally compute $s_i^B \leftarrow y_i + \alpha s_{i-1}^B$, equivalently, $s_i^B = \alpha(s_{i-1}^A + s_{i-1}^B) + a_{d-i} - \rho_i$.
 3. For the d -th OLE call B provides α to $\mathcal{F}_{\text{OLE}}$.
 4. B receives $y_d = A_d \alpha + B_d$ from the d -th $\mathcal{F}_{\text{OLE}}$.
- Output:** Party B outputs $y_d + \alpha s_{d-1}^B$.

Fig. 18: OPE Protocol

E.2 OPA Protocol

Protocol: Π_{OPA}

Let $\mathcal{F}_{\text{OLE}}$ be a ideal OLE functionality. The protocol runs between the sender A with input $\mathbf{a}(X) \in \mathbb{F}[X]$ with $\deg(\mathbf{a}) \leq 2d$ and the party B with input $\mathbf{b}(X) \in \mathbb{F}[X]$ with $\deg(\mathbf{b}) \leq d$ with the distinct public points $\alpha_1, \dots, \alpha_{2d+1} \in \mathbb{F}$.

Party A: Sample a random polynomial $\mathbf{r}(X) \xleftarrow{\$} \mathbb{F}^{\leq d}[X]$. For each $i \in [2d+1]$, compute: $a_i := \mathbf{a}(\alpha_i)$, $r_i := \mathbf{r}(\alpha_i)$. For each $i \in [2d+1]$, invoke $\mathcal{F}_{\text{OLE}}$ with sender input (r_i, a_i) .

Party B: For each $i \in [2d+1]$, compute $b_i := \mathbf{b}(\alpha_i)$. For each $i \in [2d+1]$, invoke $\mathcal{F}_{\text{OLE}}$ with receiver input b_i , and receive $c_i = r_i b_i + a_i$. Interpolate the unique polynomial $\mathbf{c}(X) \in \mathbb{F}[X]$ of degree at most $2d$ satisfying $\mathbf{c}(\alpha_i) = c_i$ for all $i \in [2d+1]$.

Output: B outputs $\mathbf{c}(X)$.

Fig. 19: OPA Protocol Π_{OPA}

F Missing Proofs in section 5

F.1 Proof of Lemma 13

Lemma 13. *The protocol Π_{OPE} is correct.*

Proof. We prove correctness using the invariant after the i -th Horner step where $i \leq d-1$, A and B hold shares s_i^A and s_i^B such that $s_i^A + s_i^B = a_d \alpha^i + a_{d-1} \alpha^{i-1} + \dots + a_{d-i}$.

Recall that Alice initializes $s_0^A \leftarrow a_d$, and Bob initializes $s_0^B \leftarrow 0$. Therefore, $s_0^A + s_0^B = a_d$. Hence, the invariant holds for $i = 0$.

Now suppose that after the $(i-1)$ -th Horner step, for some $1 \leq i \leq d-1$, the parties hold additive shares satisfying $s_{i-1}^A + s_{i-1}^B = a_d \alpha^{i-1} + a_{d-1} \alpha^{i-2} + \dots + a_{d-i+1}$. In the i -th OLE call, Alice samples $\rho_i \xleftarrow{\$} \mathbb{F}$ and sets $A_i \leftarrow s_{i-1}^A, B_i \leftarrow a_{d-i} - \rho_i$. Bob inputs α to $\mathcal{F}_{\text{OLE}}$ and receives $y_i = A_i \alpha + B_i = s_{i-1}^A \alpha + a_{d-i} - \rho_i$. Bob then computes $s_i^B \leftarrow y_i + \alpha s_{i-1}^B$. Since Alice sets $s_i^A \leftarrow \rho_i$, we have

$$\begin{aligned} s_i^A + s_i^B &= \rho_i + (y_i + \alpha s_{i-1}^B) \\ &= \rho_i + (s_{i-1}^A \alpha + a_{d-i} - \rho_i) + \alpha s_{i-1}^B \\ &= \alpha (s_{i-1}^A + s_{i-1}^B) + a_{d-i}. \end{aligned}$$

By the induction hypothesis $s_{i-1}^A + s_{i-1}^B = a_d \alpha^{i-1} + a_{d-1} \alpha^{i-2} + \dots + a_{d-i+1}$. Therefore,

$$\begin{aligned} s_i^A + s_i^B &= \alpha (a_d \alpha^{i-1} + a_{d-1} \alpha^{i-2} + \dots + a_{d-i+1}) + a_{d-i} \\ &= a_d \alpha^i + a_{d-1} \alpha^{i-1} + \dots + a_{d-i+1} \alpha + a_{d-i}. \end{aligned}$$

Thus, the invariant holds for every $i = 0, 1, \dots, d-1$. In particular, after the $(d-1)$ -th Horner step $s_{d-1}^A + s_{d-1}^B = a_d \alpha^{d-1} + a_{d-1} \alpha^{d-2} + \dots + a_1$.

For the final d -th OLE call, Alice inputs $A_d = s_{d-1}^A$ and $B_d = a_0$, while Bob inputs α . Hence Bob receives $y_d = A_d \alpha + B_d = s_{d-1}^A \alpha + a_0$ and Bob outputs $y_d + \alpha s_{d-1}^B$. Substituting the value of y_d , we get

$$\begin{aligned} y_d + \alpha s_{d-1}^B &= s_{d-1}^A \alpha + a_0 + \alpha s_{d-1}^B \\ &= \alpha (s_{d-1}^A + s_{d-1}^B) + a_0. \end{aligned}$$

Using the invariant for $i = d-1$,

$$\begin{aligned} \alpha (s_{d-1}^A + s_{d-1}^B) + a_0 &= \alpha (a_d \alpha^{d-1} + a_{d-1} \alpha^{d-2} + \dots + a_1) + a_0 \\ &= a_d \alpha^d + a_{d-1} \alpha^{d-1} + \dots + a_1 \alpha + a_0 \\ &= \sum_{j=0}^d a_j \alpha^j \\ &= \mathbf{p}(\alpha). \end{aligned}$$

Therefore, Bob outputs the correct value $\mathbf{p}(\alpha)$. Alice receives no output except $\perp$ by the specification of the protocol. This proves correctness. $\square$

F.2 Proof of Theorem 6

Proof (Sketch of Theorem 6). The protocol Π_{OPE} invokes the ideal functionality $\mathcal{F}_{\text{OLE}}$ as a subroutine in each of its d Horner steps; all other operations performed by Alice and Bob are local computations on their respective inputs and on the values returned by $\mathcal{F}_{\text{OLE}}$. By Theorem 4, the underlying OLE protocol $\Pi_{\text{OLE}}^{(2)}$ UC-realizes $\mathcal{F}_{\text{OLE}}$ against semi-honest corruptions. By the UC composition

theorem [Can00], it therefore suffices to exhibit a simulator in the $\mathcal{F}_{\text{OLE}}$ -hybrid model.

Corrupt Alice. The simulator $\mathcal{S}_A$ reads $(a_0, \dots, a_d)$ from $\mathcal{A}_A$'s input tape, forwards the induced polynomial $\mathbf{p}$ to $\mathcal{F}_{\text{OPE}}$, and emulates each $\mathcal{F}_{\text{OLE}}$ instance by recording $\mathcal{A}_A$'s sender inputs (A_i, B_i) and replying with the receipt token. Since Alice receives no protocol messages and outputs $\perp$, the joint view of $\mathcal{A}_A$ and $\mathcal{Z}$ is identically distributed in the real and ideal executions.

Corrupt Bob. The simulator $\mathcal{S}_B$ reads α from $\mathcal{A}_B$'s input tape, submits (sid, α) to $\mathcal{F}_{\text{OPE}}$, and obtains $\mathbf{p}(\alpha)$. For the first $d-1$ OLE calls, $\mathcal{S}_B$ samples $\tilde{y}_i \xleftarrow{\$} \mathbb{F}$ uniformly and returns it to $\mathcal{A}_B$ as the i -th $\mathcal{F}_{\text{OLE}}$ output. For the final call, $\mathcal{S}_B$ returns

$$\tilde{y}_d := \mathbf{p}(\alpha) - \alpha \cdot s_{d-1}^B,$$

where $s_{d-1}^B = \sum_{i=1}^{d-1} \alpha^{d-1-i} \tilde{y}_i$ is computed exactly as Bob would. In the real execution, each $y_i = s_{i-1}^A \alpha + a_{d-i} - \rho_i$ is uniformly distributed in $\mathbb{F}$ by the perfect masking of Alice's fresh $\rho_i \xleftarrow{\$} \mathbb{F}$, independently of all other transcripts; hence the distribution of $(y_1, \dots, y_{d-1})$ is identical to $(\tilde{y}_1, \dots, \tilde{y}_{d-1})$. By the correctness invariant of Lemma 13, the honest Bob's final output equals $\mathbf{p}(\alpha)$, which matches $\tilde{y}_d + \alpha s_{d-1}^B$ by construction. Hence the joint view of $\mathcal{A}_B$ and $\mathcal{Z}$ is identically distributed.

Combining both cases and invoking the UC composition theorem to replace each $\mathcal{F}_{\text{OLE}}$ call by Π_{OLE} yields the claim. $\square$

F.3 Proof of Theorem 7

Proof. Let $\mathcal{A}$ be the dummy adversary mounting at most one specious corruption and let $\mathcal{Z}$ be any PPT environment. We must build a simulator $\mathcal{S}$ such that,

$$\text{Exec}_{\Pi_{\text{OPE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OLE}}), \mathcal{S}, \mathcal{Z}}. \quad (5)$$

We use a sequence of $d+1$ hybrid experiments $H_0, H_1, \dots, H_d$. In hybrid H_i , the first i invocations of $\widehat{\Pi}_{\text{OLE}}$ inside Π_{OPE} are replaced by invocations of the wrapped ideal functionality $\text{Wrap}(\mathcal{F}_{\text{OLE}})$, while the remaining $d-i$ invocations are still executed as $\widehat{\Pi}_{\text{OLE}}$. In particular:

- H_0 is the real-world execution $\text{Exec}_{\Pi_{\text{OPE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}}$: all d OLE invocations are run as $\widehat{\Pi}_{\text{OLE}}$ instances over $\mathcal{F}_{\text{SAT}}$.
- For $i \in [d]$, H_i differs from H_{i-1} in exactly one place: the i -th $\widehat{\Pi}_{\text{OLE}}$ invocation is replaced by an ideal call to $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ that takes sender input (A_i, B_i) and receiver input α and returns $A_i \alpha + B_i$ to Bob.

Lemma 14. $H_0 \stackrel{c}{\approx} H_d$.

Proof. For each $i \in [d]$, the only difference between H_{i-1} and H_i is the implementation of the i -th OLE invocation: in H_{i-1} it is realized by $\widehat{\Pi}_{\text{OLE}}$, whereas in

H_i it is replaced by $\text{Wrap}(\mathcal{F}_{\text{OLE}})$. By [Theorem 5](#) (in the 2-round case), the protocol $\widehat{\Pi}_{\text{OLE}}$ wsrUC-realizes $\mathcal{F}_{\text{OLE}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model; equivalently, $\widehat{\Pi}_{\text{OLE}}$ UC-realizes $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model under specious environments and adversaries. Hence there exists a PPT simulator $\mathcal{S}_i$ such that, for every specious PPT environment,

$$H_{i-1} \stackrel{c}{\approx} H_i.$$

Chaining these d steps yields $H_0 \stackrel{c}{\approx} H_d$. $\square$

The hybrid H_d is, by construction, an execution of Π_{OPE} in which each OLE call is to $\text{Wrap}(\mathcal{F}_{\text{OLE}})$. Because $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ behaves on the input–output interface exactly as $\mathcal{F}_{\text{OLE}}$ does (under the standard corruption translation table [Table 1](#)), H_d is statistically equivalent to an execution of Π_{OPE} in the $\mathcal{F}_{\text{OLE}}$ -hybrid model. By [Theorem 6](#), Π_{OPE} UC-realizes $\mathcal{F}_{\text{OPE}}$ in the $\mathcal{F}_{\text{OLE}}$ -hybrid model; instantiating that simulator yields a simulator $\mathcal{S}$ such that

$$H_d \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OPE}}), \mathcal{S}, \mathcal{Z}}.$$

Combining with [Lemma 14](#),

$$\text{Exec}_{\widehat{\Pi}_{\text{OLE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} = H_0 \stackrel{c}{\approx} H_d \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OPE}}), \mathcal{S}, \mathcal{Z}},$$

which establishes [\(5\)](#).

We now show that both composed firewalls $F_{\text{A}}^{\text{OPE}}$ and $F_{\text{B}}^{\text{OPE}}$ are strongly sanitizing to conclude the proof.

Lemma 15. *The composed Alice firewall $F_{\text{A}}^{\text{OPE}}$ is strongly sanitizing.*

Proof. Let $\mathcal{Z}$ be any PPT environment that does not corrupt $F_{\text{A}}^{\text{OPE}}$ but is allowed to mount a specious corruption ($\text{SPECIOUS}, \widetilde{\mathcal{C}}_{\text{A}}$) on Alice’s OPE core. We must show

$$\text{Exec}_{\widehat{\Pi}_{\text{OLE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} \stackrel{c}{\approx} \text{Exec}_{\widehat{\Pi}'_{\text{OLE}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}},$$

where $\widehat{\Pi}'_{\text{OLE}}$ runs with the incorruptible Alice core $\widehat{\mathcal{C}}_{\text{A}}$ (which ignores all SPECIOUS commands and keeps running the honest code of $\mathcal{C}_{\text{A}}$) in place of $\mathcal{C}_{\text{A}}$, and $\mathcal{A}$ is the dummy adversary, so $\mathcal{Z}$ plays Bob’s side ($\mathcal{C}_{\text{B}}, F_{\text{B}}^{\text{OPE}}$).

By construction, Alice’s OPE core $\mathcal{C}_{\text{A}}$ feeds, into the i -th OLE sub-protocol, only the sender input (A_i, B_i) computed from its local state. Equivalently, $\mathcal{C}_{\text{A}}$ decomposes into d parallel sub-cores $\mathcal{C}_{\text{A}}^1, \dots, \mathcal{C}_{\text{A}}^d$, where $\mathcal{C}_{\text{A}}^i$ is exactly the OLE-sender core supplying the i -th OLE instance with (A_i, B_i) . Since $\widetilde{\mathcal{C}}_{\text{A}}$ is specious for $\mathcal{C}_{\text{A}}$, it follows that the marginal $\widetilde{\mathcal{C}}_{\text{A}}^i$ induced on the i -th OLE sub-core is specious for $\mathcal{C}_{\text{A}}^i$.

By [Lemma 10](#) (in the 2-round case), the per-instance sender firewall F_{A}^i is strongly sanitizing for $\mathcal{C}_{\text{A}}^i$. Consequently, for every $i \in [d]$, the marginal view of $\mathcal{Z}$ on the i -th OLE sub-protocol is computationally indistinguishable in the two experiments. Because the firewalls $F_{\text{A}}^1, \dots, F_{\text{A}}^d$ run in parallel and share no

internal state, the joint view of $\mathcal{Z}$ across all d OLE sub-protocols is computationally indistinguishable in the two experiments, by a standard d -step hybrid argument over the OLE instances.

It remains to observe that the only further messages produced by F_A^{OPE} , beyond those produced by the underlying F_A^i 's, are inputs and outputs on the $\mathcal{F}_{\text{SAT}}$ -interfaces of the OPE layer; these are determined by the OLE sub-transcripts already sanitized above (the Horner masks ρ_i and shares s_i^A are computed internally by C_A and never leave Alice's party). Hence F_A^{OPE} is strongly sanitizing. $\square$

Lemma 16. *The composed Bob firewall F_B^{OPE} is strongly sanitizing.*

Proof. Symmetric to the proof of Lemma 15, using Lemma 11 (in the 2-round case) for the strong sanitation of each per-instance receiver firewall F_B^i . Bob's OPE core C_B likewise decomposes, in the Π_{OPE} protocol, into d parallel sub-cores $C_B^1, \dots, C_B^d$, each feeding the corresponding OLE sub-protocol with the receiver input α . A specious $\tilde{C}_B$ projects to specious sub-cores $\tilde{C}_B^i$ on each instance, and the firewall-composition shares no state, so the joint view of $\mathcal{Z}$ across all d OLE sub-instances is computationally indistinguishable in the two experiments by a standard d -step hybrid argument.

The remaining messages produced by F_B^{OPE} beyond the OLE sub-transcripts are the relayed Horner-step outputs $y_1, \dots, y_d$ delivered to C_B on $\mathcal{F}_{\text{SAT}}$. The local post-processing $s_i^B \leftarrow y_i + \alpha s_{i-1}^B$ and the final output $y_d + \alpha s_{d-1}^B$ are deterministic functions of these outputs and of α , both of which lie inside C_B . Conditioning on the per-instance OLE views being computationally indistinguishable across the two experiments, Bob's final output is also computationally indistinguishable. Hence F_B^{OPE} is strongly sanitizing. $\square$

Combining the hybrid argument with the strong sanitation guarantees of Lemma 15 and Lemma 16, we conclude that $\hat{\Pi}_{\text{OPE}}$ wsrUC-realizes $\mathcal{F}_{\text{OPE}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model. This completes the proof of Theorem 7. $\square$

F.4 Proof of Lemma 17

Lemma 17. *The protocol Π_{OPA} is correct.*

Proof. Let $\alpha_1, \dots, \alpha_{2d+1} \in \mathbb{F}$ be distinct public points. The sender computes, for each $i \in [2d+1]$, $a_i = \mathbf{a}(\alpha_i)$, $r_i = \mathbf{r}(\alpha_i)$. The receiver computes $b_i = \mathbf{b}(\alpha_i)$ for every $i \in [2d+1]$. For each i , the parties invoke one instance of $\mathcal{F}_{\text{OLE}}$, where the sender inputs (r_i, a_i) and the receiver inputs b_i . By correctness of $\mathcal{F}_{\text{OLE}}$, the receiver obtains $c_i = r_i b_i + a_i$. Substituting the definitions of a_i, r_i, b_i , we obtain $c_i = \mathbf{r}(\alpha_i) \mathbf{b}(\alpha_i) + \mathbf{a}(\alpha_i)$. Define $\mathbf{c}(X) = \mathbf{r}(X) \mathbf{b}(X) + \mathbf{a}(X)$. Since $\deg(\mathbf{r}) \leq d$ and $\deg(\mathbf{b}) \leq d$, we have $\deg(\mathbf{r} \cdot \mathbf{b}) \leq 2d$. Also, $\deg(\mathbf{a}) \leq 2d$. Therefore, $\deg(\mathbf{c}) \leq 2d$. Moreover, for every $i \in [2d+1]$, $\mathbf{c}(\alpha_i) = \mathbf{r}(\alpha_i) \mathbf{b}(\alpha_i) + \mathbf{a}(\alpha_i) = c_i$. The receiver interpolates the unique polynomial of degree at most $2d$ passing through the $2d+1$ points $(\alpha_i, c_i)_{i=1}^{2d+1}$. Since a polynomial of degree at most $2d$ is uniquely determined by its values on $2d+1$ distinct points, the interpolated polynomial is exactly $\mathbf{c}(X) = \mathbf{r}(X) \mathbf{b}(X) + \mathbf{a}(X)$. Hence Π_{OPA} correctly computes the OPA output. $\square$

F.5 Proof of Theorem 8

Proof (Sketch). The proof follows [GN19, Lemma 9], which establishes the analogous statement in the $\mathcal{F}_{\text{OLE}+}$ -hybrid model against *malicious* adversaries. Two simplifications apply here. (i) Semi-honest parties execute the prescribed code, so every $\mathcal{F}_{\text{OLE}}$ input is a faithful evaluation of a well-formed polynomial of the prescribed degree; no zero-polynomial or degree-violating input arises. Plain $\mathcal{F}_{\text{OLE}}$ thus suffices in place of $\mathcal{F}_{\text{OLE}+}$, and the random-challenge consistency checks of [GN19, Fig. 7, Steps 3–6] become vacuous and are omitted in our OPA protocol. (ii) The simulator reads $(\mathbf{a}, \mathbf{r})$ (resp. $\mathbf{b}$) directly from the corrupted party's input tape and forwards it to $\mathcal{F}_{\text{OPA}}$. Indistinguishability follows from the perfect masking by the sender's fresh $\mathbf{r} \xleftarrow{\$} \mathbb{F}^{\leq d}[X]$, exactly as in [GN19]. $\square$

F.6 Proof of Theorem 9

Proof. Let $\mathcal{A}$ be a dummy adversary corrupting at most one party (specious corruption) and let $\mathcal{Z}$ be any PPT environment. We build a simulator $\mathcal{S}$ such that

$$\text{Exec}_{\widehat{\Pi}_{\text{OPA}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OPA}}), \mathcal{S}, \mathcal{Z}}. \quad (6)$$

We use a sequence of $2d + 2$ hybrid experiments $H_0, H_1, \dots, H_{2d+1}$. In hybrid H_i , the first i invocations of the sanitizing OLE protocol $\widehat{\Pi}_{\text{OLE}}$ inside $\widehat{\Pi}_{\text{OPA}}$ are replaced by invocations of the wrapped ideal functionality $\text{Wrap}(\mathcal{F}_{\text{OLE}})$, while the remaining $2d+1-i$ invocations are still executed as the protocol $\widehat{\Pi}_{\text{OLE}}$. In particular:

- H_0 is the real-world execution $\text{Exec}_{\widehat{\Pi}_{\text{OPA}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}}$: all $2d + 1$ OLE invocations are run as $\widehat{\Pi}_{\text{OLE}}$ instances over $\mathcal{F}_{\text{SAT}}$.
- For $i \in [2d + 1]$, H_i differs from H_{i-1} in exactly one place: the i -th $\widehat{\Pi}_{\text{OLE}}$ invocation is replaced by an ideal call to $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ that takes sender input $(\mathbf{r}(\alpha_i), \mathbf{a}(\alpha_i))$ and receiver input $\mathbf{b}(\alpha_i)$ and returns $\mathbf{a}(\alpha_i) + \mathbf{r}(\alpha_i) \cdot \mathbf{b}(\alpha_i)$.

Lemma 18. $H_0 \stackrel{c}{\approx} H_{2d+1}$.

Proof. For each $i \in [2d+1]$, the only difference between H_{i-1} and H_i is the implementation of the i -th OLE invocation: in H_{i-1} it is realized by $\widehat{\Pi}_{\text{OLE}}$, whereas in H_i it is replaced by $\text{Wrap}(\mathcal{F}_{\text{OLE}})$. By Theorem 5 (in the 2-round case) or Theorem 3 (in the 3-round case), the protocol $\widehat{\Pi}_{\text{OLE}}$ wsrUC-realizes $\mathcal{F}_{\text{OLE}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model; equivalently, $\widehat{\Pi}_{\text{OLE}}$ UC-realizes $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model under specious environments and adversaries. Hence there exists a PPT simulator $\mathcal{S}_i$ such that for every specious PPT environment,

$$H_{i-1} \stackrel{c}{\approx} H_i.$$

Chaining these $2d+1$ steps, $H_0 \stackrel{c}{\approx} H_{2d+1}$. $\square$

The hybrid H_{2d+1} is by construction an execution of Π_{OPA} in which each OLE call is to $\text{Wrap}(\mathcal{F}_{\text{OLE}})$. Because $\text{Wrap}(\mathcal{F}_{\text{OLE}})$ behaves on the input-output interface exactly as $\mathcal{F}_{\text{OLE}}$ does (under the standard corruption translation table [Table 1](#)), H_{2d+1} is statistically equivalent to an execution of Π_{OPA} in the $\mathcal{F}_{\text{OLE}}$ -hybrid model. By [Theorem 8](#), Π_{OPA} UC-realizes $\mathcal{F}_{\text{OPA}}$ in the $\mathcal{F}_{\text{OLE}}$ -hybrid model; instantiating that simulator yields a simulator $\mathcal{S}$ such that

$$H_{2d+1} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OPA}}), \mathcal{S}, \mathcal{Z}}.$$

Combining with the previous chain,

$$\text{Exec}_{\hat{\Pi}_{\text{OPA}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} = H_0 \stackrel{c}{\approx} H_{2d+1} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{OPA}}), \mathcal{S}, \mathcal{Z}},$$

Now, we show that both composed firewalls F_S^{OPA} and F_R^{OPA} are strongly sanitizing to conclude the proof.

Lemma 19. *The composed sender firewall F_S^{OPA} is strongly sanitizing.*

Proof. Let $\mathcal{Z}$ be any PPT environment that does not corrupt F_S^{OPA} but is allowed to mount a specious corruption ($\text{SPECIOUS}, \tilde{C}_S$) on the OPA sender core. We must show

$$\text{Exec}_{\hat{\Pi}_{\text{OPA}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}} \stackrel{c}{\approx} \text{Exec}_{\hat{\Pi}'_{\text{OPA}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}}},$$

where $\hat{\Pi}'_{\text{OPA}}$ runs with the incorruptible OPA sender core $\hat{C}_S$ (which ignores all SPECIOUS commands and keeps running the honest code of C_S) in place of C_S , and where $\mathcal{A}$ is the dummy adversary.

By construction, the OPA sender core C_S feeds, into the i -th OLE sub-protocol, only the values $a_i := \mathbf{a}(\alpha_i)$ and $r_i := \mathbf{r}(\alpha_i)$. Equivalently, C_S decomposes into $2d+1$ *parallel* sub-cores $C_S^1, \dots, C_S^{2d+1}$, where C_S^i is exactly the OLE-sender core supplying the i -th OLE instance with its inputs. Since $\tilde{C}_S$ is specious for C_S , it follows that the marginal $\tilde{C}_S^i$ induced on the i -th OLE sub-core is specious for C_S^i .

By [Lemma 10](#) (in the 2-round case) the per-instance sender firewall F_S^i is strongly sanitizing for C_S^i . Consequently, for every $i \in [2d+1]$, the marginal view of $\mathcal{Z}$ on the i -th OLE sub-protocol is computationally indistinguishable in the two experiments. Because the firewalls $F_S^1, \dots, F_S^{2d+1}$ run in parallel and share no internal state (this is the very content of the $\diamond$ -composition), the joint view of $\mathcal{Z}$ across all $2d+1$ OLE sub-protocols is computationally indistinguishable in the two experiments, by a standard $(2d+1)$ -step hybrid argument over the OLE instances.

It remains to observe that the only further messages produced by F_S^{OPA} , beyond those produced by the underlying F_S^i 's, are inputs and outputs on the $\mathcal{F}_{\text{SAT}}$ -interfaces of the OPA layer; these are determined by the OLE sub-transcripts already sanitized above. Hence F_S^{OPA} is strongly sanitizing. $\square$

Lemma 20. *The composed receiver firewall F_R^{OPA} is strongly sanitizing.*

Proof. Symmetric to the proof of Lemma 19, using Lemma 11 (in the 2-round case) for the strong sanitation of each per-instance receiver firewall F_R^i . The OPA receiver core C_R likewise decomposes, in the Π_{OPA} protocol, into $2d+1$ parallel sub-cores $C_R^1, \dots, C_R^{2d+1}$, each feeding the corresponding OLE sub-protocol with the value $b_i := \mathbf{b}(\alpha_i)$. $\square$

Thus, we conclude that $\hat{\Pi}_{\text{OPA}}$ wsrUC-realizes $\mathcal{F}_{\text{OPA}}$ in $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model. This completes the proof of Theorem 9. $\square$

G PSI from OPE

We begin with describing our protocol.

Protocol: Π_{PSI}

Let $\mathcal{F}_{\text{OPE}}$ be the ideal OPE functionality. The protocol Π_{PSI} runs between the sender A and the receiver B. The sender A has input $S_A = \{\gamma_1, \dots, \gamma_m\} \subseteq \mathbb{F}$, and the receiver B has input $S_B = \{\beta_1, \dots, \beta_n\} \subseteq \mathbb{F}$. The receiver B obtains the intersection $S_A \cap S_B$, while the sender A obtains $\perp$.

Sender A (Input S_A): Encode the set S_A as the polynomial $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$. Participate as the sender in one invocation of $\mathcal{F}_{\text{OPE}}$ with input $r_j \cdot \mathbf{p}_A(X)$, where $r_j \xleftarrow{\$} \mathbb{F} \setminus \{0\}$. In every invocation, A receives $\perp$.

Receiver B (Input S_B): Initialize $S_\cap \leftarrow \emptyset$. For each $\beta_j \in S_B$: (i) Invoke $\mathcal{F}_{\text{OPE}}$ with receiver input β_j , (ii) Receive the value $z_j \leftarrow r_j \cdot \mathbf{p}_A(\beta_j)$ from $\mathcal{F}_{\text{OPE}}$. (iii) If $z_j = 0$, then add β_j to the intersection set: $S_\cap \leftarrow S_\cap \cup \{\beta_j\}$. Finally, output: $S_\cap$.

Output: A outputs $\perp$. B outputs $S_\cap = S_A \cap S_B$.

Fig. 20: One-Sided PSI Protocol from $\mathcal{F}_{\text{OPE}}$

G.1 Correctness of the PSI protocol Π_{PSI}

Lemma 21. *The protocol Π_{PSI} correctly realizes $\mathcal{F}_{\text{PSI}}$.*

Proof. Party A encodes its set S_A as the polynomial $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$. Therefore, for any element $x \in \mathbb{F}$ and $r \xleftarrow{\$} \mathbb{F} \setminus \{0\}$, we have $r \cdot \mathbf{p}_A(x) = 0 \iff r \cdot \prod_{\gamma \in S_A} (x - \gamma) = 0$.

Since $\mathbb{F}$ is a field, a product is zero if and only if one of its factors is zero. Hence, $r \cdot \mathbf{p}_A(x) = 0 \iff \exists \gamma \in S_A$ such that $x - \gamma = 0$. Equivalently, $r \cdot \mathbf{p}_A(x) = 0 \iff x \in S_A$.

Now consider any element $\beta_j \in S_B$. In the j -th invocation of $\mathcal{F}_{\text{OPE}}$, party A inputs the polynomial $\mathbf{p}_A(X)$, and party B inputs the point β_j . By correctness of $\mathcal{F}_{\text{OPE}}$, party B receives $z_j = \mathbf{p}_A(\beta_j)$. The protocol adds β_j to $S_\cap$ if and only if $z_j = 0 \iff \mathbf{p}_A(\beta_j) = 0 \iff \beta_j \in S_A$. Since β_j is already chosen from S_B , we get $z_j = 0 \iff \beta_j \in S_A \cap S_B$. Thus, the protocol includes exactly those elements of S_B that also belong to S_A . Therefore, at the end of the protocol, $S_\cap = \{\beta_j \in S_B : \mathbf{p}_A(\beta_j) = 0\} = S_A \cap S_B$.

Hence, party B outputs the correct intersection. Party A receives no output in each invocation of $\mathcal{F}_{\text{OPE}}$ and outputs $\perp$ by the specification of the protocol. $\square$

G.2 Proof of Theorem 10

Proof. Let $\mathcal{A}$ be any real-world adversary attacking Π_{PSI} in the $\mathcal{F}_{\text{OPE}}$ -hybrid model. We construct an ideal-world simulator $\mathcal{S}$ such that for every environment $\mathcal{Z}$,

$$\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{OPE}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{PSI}}, \mathcal{S}, \mathcal{Z}}.$$

We treat the two corruption cases separately.

Case 1: Corrupt Sender (A). The corrupted party $\mathcal{A}_A$ holds input S_A and participates as the sender in all n invocations of $\mathcal{F}_{\text{OPE}}$, one for each element $\beta_j \in S_B$. In each invocation, A inputs the polynomial $r_j \cdot \mathbf{p}_A(X) = r_j \cdot \prod_{\gamma \in S_A} (X - \gamma)$, $r_j \xleftarrow{\$} \mathbb{F} \setminus \{0\}$ and receives no output. The simulator Sim_A proceeds as follows.

Simulator Sim_A :

1. Read S_A from $\mathcal{A}_A$'s input tape. Encode S_A as the polynomial $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$.
2. Submit S_A to $\mathcal{F}_{\text{PSI}}$ on A's input interface and receive nothing.
3. For each of the n $\mathcal{F}_{\text{OPE}}$ invocations, record $\mathcal{A}_A$'s sender input $r_j \cdot \mathbf{p}_A(X)$ and reply with the receipt token $\perp$ on A's output interface. A receives no output from any $\mathcal{F}_{\text{OPE}}$ call by the specification of $\mathcal{F}_{\text{OPE}}$.
4. Output whatever $\mathcal{A}_A$ outputs, which is $\perp$.

Indistinguishability. A receives no protocol messages beyond the $\mathcal{F}_{\text{OPE}}$ receipts and outputs $\perp$ in both the real and ideal executions. The $\mathcal{F}_{\text{OPE}}$ calls are each realized by the ideal functionality $\mathcal{F}_{\text{OPE}}$, so A's View consists only of her own input tape and the $\perp$ tokens. The joint view of $\mathcal{A}_A$ and $\mathcal{Z}$ is therefore identically distributed in the real and ideal executions.

Case 2: Corrupt Receiver (B). The corrupted party $\mathcal{A}_B$ holds input $S_B = \{\beta_1, \dots, \beta_n\}$ and participates as the receiver in all n invocations of $\mathcal{F}_{\text{OPE}}$. In the j -th invocation, B inputs β_j and receives $z_j = \mathbf{p}_A(\beta_j)$. B then locally computes $S_\cap = \{\beta_j \in S_B : z_j = 0\}$ and outputs it. The simulator Sim_B must generate a simulated view for $\mathcal{A}_B$ without knowing A's polynomial $\mathbf{p}_A$, using only the intersection $S_\cap$ returned by $\mathcal{F}_{\text{PSI}}$.

Simulator Sim_B :

1. Read $S_B = \{\beta_1, \dots, \beta_n\}$ from $\mathcal{A}_B$'s input tape.
2. Submit S_B to $\mathcal{F}_{\text{PSI}}$ on B's input interface and receive the intersection $S_\cap = S_A \cap S_B$.
3. For each $j \in [n]$, simulate the output of the j -th $\mathcal{F}_{\text{OPE}}$ invocation as follows:
 - If $\beta_j \in S_\cap$, set $\tilde{z}_j \leftarrow 0$.
 - If $\beta_j \notin S_\cap$, sample $\tilde{z}_j \xleftarrow{\$} \mathbb{F}^*$.

4. Deliver $\tilde{z}_j$ to $\mathcal{A}_B$ as the output of the j -th $\mathcal{F}_{\text{OPE}}$ call.

Indistinguishability. We argue via the following hybrid sequence over the transcript $(\tilde{z}_1, \dots, \tilde{z}_n)$ seen by $\mathcal{A}_B$.

H₀ (Real world): For each $j \in [n]$, B receives $z_j = r_j \cdot \mathbf{p}_A(\beta_j)$ from $\mathcal{F}_{\text{OPE}}$. By correctness of Π_{PSI} , $z_j = 0$ if and only if $\beta_j \in S_A$, i.e., $\beta_j \in S_\cap$.

H₁ (Ideal world): For each $j \in [n]$, B receives $\tilde{z}_j$ as defined by Sim_B above.

We claim $H_0 \equiv H_1$. Consider any fixed $j \in [n]$:

- *Case $\beta_j \in S_\cap$:* In H_0 , $z_j = r_j \cdot \mathbf{p}_A(\beta_j) = 0$ since $\beta_j \in S_A$; in H_1 , $\tilde{z}_j = 0$ by construction. The distributions are identical.
- *Case $\beta_j \notin S_\cap$:* In H_0 , $z_j = r_j \cdot \mathbf{p}_A(\beta_j) \neq 0$ since $\beta_j \notin S_A$, and $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$ is a nonzero polynomial of degree $|S_A|$ evaluated at a point outside its root set. Since $\mathcal{A}_B$ is semi-honest and does not know S_A or the coefficients of $\mathbf{p}_A$ beyond what is revealed by the $\mathcal{F}_{\text{OPE}}$ outputs, the security of $\mathcal{F}_{\text{OPE}}$ guarantees that B learns nothing about $\mathbf{p}_A$ beyond the single evaluation $\mathbf{p}_A(\beta_j)$. For $\beta_j \notin S_A$, this evaluation is a nonzero element whose distribution, from B 's View, is uniform over $\mathbb{F}^*$. In H_1 , $\tilde{z}_j \xleftarrow{\$} \mathbb{F}^*$ by construction. The two distributions are therefore statistically identical conditioned on $\beta_j \notin S_\cap$.

Since the n $\mathcal{F}_{\text{OPE}}$ invocations are independent and the argument applies identically for each $j \in [n]$, the joint distribution $(\tilde{z}_1, \dots, \tilde{z}_n)$ is identically distributed in H_0 and H_1 . Furthermore, B 's final output

$$S_\cap = \{ \beta_j \in S_B : \tilde{z}_j = 0 \}$$

is computed deterministically from $(\tilde{z}_1, \dots, \tilde{z}_n)$ and S_B , and matches the output delivered by $\mathcal{F}_{\text{PSI}}$ in the ideal execution. Hence, the joint view of $\mathcal{A}_B$ and $\mathcal{Z}$ is identically distributed in the real and ideal executions.

Combining both corruption cases, for every PPT semi-honest adversary $\mathcal{A}$ corrupting at most one party, there exists a PPT simulator $\mathcal{S}$ such that

$$\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{OPE}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{PSI}}, \mathcal{S}, \mathcal{Z}}.$$

for every PPT environment $\mathcal{Z}$. Hence Π_{PSI} UC-realizes $\mathcal{F}_{\text{PSI}}$ in the $\mathcal{F}_{\text{OPE}}$ -hybrid model. $\square$

H PSI from OPA

We begin with describing our OPA based PSI protocol in [Figure 21](#).

Protocol: $\Pi_{\text{PSI}}^{\text{opa}}$

Let $\mathcal{F}_{\text{OPA}}$ be the ideal OPA functionality. The protocol $\Pi_{\text{PSI}}^{\text{opa}}$ is run between two parties A and B with respective input sets S_A and S_B .

Inputs. Party A holds S_A , and party B holds S_B each of size n .

Setup. Party A computes a polynomial $\mathbf{p}_A$ of degree n such that $\mathbf{p}_A(\alpha) = 0$ for every $\alpha \in S_A$. It samples two random polynomials $\mathbf{r}_A, \mathbf{r}'_A$ of degree n , and forms the degree- $2n$ polynomial $\mathbf{q}_A = \mathbf{p}_A \cdot \mathbf{r}'_A$. Party B computes a polynomial $\mathbf{p}_B$ of degree n such that $\mathbf{p}_B(\beta) = 0$ for every $\beta \in S_B$.

OPA Invocation. The parties invoke a single instance of $\mathcal{F}_{\text{OPA}}$ as follows: In $\mathcal{F}_{\text{OPA}}$, party A acts as the *sender* and inputs the polynomial $\mathbf{q}_A$ (of degree $2n$) and $\mathbf{r}'_A$ (of degree n). Party B acts as the *receiver* and inputs $\mathbf{p}_B$. Party B receives the output $\mathbf{p}_\cap = \mathbf{q}_A + \mathbf{r}_A \cdot \mathbf{p}_B$.

Output. Party B recovers the intersection by evaluating $\mathbf{p}_\cap$ on its own input elements. Party B initializes $S_\cap = \emptyset$. For every $\beta \in S_B$, it computes $\mathbf{p}_\cap(\beta)$, if $\mathbf{p}_\cap(\beta) = 0$, party B adds β to $S_\cap$. Party B outputs $S_\cap$.

Fig. 21: PSI Protocol from OPA $\Pi_{\text{PSI}}^{\text{opa}}$

H.1 Proof of Lemma 22

Lemma 22. *The protocol $\Pi_{\text{PSI}}^{\text{opa}}$ is correct.*

Proof. We show that $S_\cap = S_A \cap S_B$ holds with overwhelming probability. For any $\beta \in S_B$, since $\mathbf{p}_B(\beta) = 0$, the intersection polynomial evaluates to

$$\mathbf{p}_\cap(\beta) = \mathbf{q}_A(\beta) + \mathbf{r}_A(\beta) \cdot \underbrace{\mathbf{p}_B(\beta)}_{=0} = \mathbf{q}_A(\beta).$$

If $\beta \in S_A$, then $\mathbf{p}_A(\beta) = 0$, so $\mathbf{p}_\cap(\beta) = 0$ and β is correctly added to $S_\cap$. If $\beta \notin S_A$, then $\mathbf{p}_A(\beta) \neq 0$, and $\mathbf{p}_\cap(\beta) = 0$ only if $\mathbf{r}'_A(\beta) = 0$, which by the Schwartz-Zippel lemma occurs with probability at most $n/|\mathbb{F}|$. By a union bound over all $\beta \in S_B$, the probability of any false positive is at most $n^2/|\mathbb{F}|$, which is negligible for $|\mathbb{F}| \gg n^2$. Hence $\Pi_{\text{PSI}}^{\text{opa}}$ is correct. $\square$

H.2 Proof of Theorem 12

Proof. Let $\mathcal{A}$ be any real-world adversary attacking $\Pi_{\text{PSI}}^{\text{opa}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model. We construct an ideal-world simulator $\mathcal{S}$ such that, for every environment $\mathcal{Z}$,

$$\text{Exec}_{\Pi_{\text{PSI}}^{\text{opa}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{OPA}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{PSI}}, \mathcal{S}, \mathcal{Z}}.$$

We consider the two corruption cases separately.

Case 1: Corrupt Party A. The corrupted party $\mathcal{A}_A$ holds input S_A and acts as the sender in the single invocation of $\mathcal{F}_{\text{OPA}}$. It inputs the pair of polynomials $(\mathbf{q}_A, \mathbf{r}_A)$ and receives no output. The simulator Sim_A proceeds as follows.

Simulator Sim_A :

1. Read S_A from $\mathcal{A}_A$'s input tape.
2. Submit S_A to $\mathcal{F}_{PSI}$ on A's input interface.
3. Upon the invocation of $\mathcal{F}_{OPA}$, record the sender inputs $(\mathbf{q}_A, \mathbf{r}_A)$ provided by $\mathcal{A}_A$ and return the receipt token $\perp$, exactly as specified by $\mathcal{F}_{OPA}$.
4. Output whatever $\mathcal{A}_A$ outputs, namely $\perp$.

Indistinguishability. Since A receives no protocol messages beyond the receipt from $\mathcal{F}_{OPA}$, its view in the real execution consists only of its own input and the local randomness used to generate $(\mathbf{q}_A, \mathbf{r}_A)$. The simulator forwards exactly the same interaction to $\mathcal{A}_A$. Hence the joint view of $\mathcal{A}_A$ and $\mathcal{Z}$ is identically distributed in the real and ideal executions.

Case 2: Corrupt Party B. The corrupted party $\mathcal{A}_B$ holds input set $S_B = \{\beta_1, \dots, \beta_n\}$ and acts as the receiver in the single invocation of $\mathcal{F}_{OPA}$. It inputs the polynomial $\mathbf{p}_B$ and receives the polynomial

$$\mathbf{p}_\cap(X) = \mathbf{q}_A(X) + \mathbf{r}_A(X) \cdot \mathbf{p}_B(X).$$

Party B then evaluates $\mathbf{p}_\cap$ on each $\beta_i \in S_B$ and outputs $S_\cap = \{\beta_i \in S_B : \mathbf{p}_\cap(\beta_i) = 0\}$. The simulator Sim_B must generate a polynomial whose evaluations on S_B reveal exactly the same intersection pattern.

Simulator Sim_B :

1. Read $S_B = \{\beta_1, \dots, \beta_n\}$ from the input tape of $\mathcal{A}_B$.
2. Submit S_B to $\mathcal{F}_{PSI}$ on B's input interface and receive the intersection $S_\cap = S_A \cap S_B$.
3. Construct a degree- $2n$ polynomial $\tilde{\mathbf{p}}_\cap(X)$:
 - sample a random polynomial $\tilde{\mathbf{r}}(X)$ of degree $2n - |S_\cap|$.
 - Then compute $\tilde{\mathbf{p}}_\cap(X) = \prod_{\beta \in S_\cap} (X - \beta) \cdot \tilde{\mathbf{r}}(X)$;
4. Deliver $\tilde{\mathbf{p}}_\cap$ to $\mathcal{A}_B$ as the output of the $\mathcal{F}_{OPA}$ invocation.

Indistinguishability. We compare the following hybrids.

- H_0 : This is the real world. B receives $\mathbf{p}_\cap = \mathbf{q}_A + \mathbf{r}_A \cdot \mathbf{p}_B$ from $\mathcal{F}_{OPA}$.
 H_1 : This is the ideal world. B receives the simulated polynomial $\tilde{\mathbf{p}}_\cap$ generated by Sim_B .

For every $\beta_i \in S_B$, since $\mathbf{p}_B(\beta_i) = 0$, in the real execution we have

$$\mathbf{p}_\cap(\beta_i) = \mathbf{q}_A(\beta_i) = \mathbf{p}_A(\beta_i) \cdot \mathbf{r}'_A(\beta_i).$$

Now:

- If $\beta_i \in S_\cap$, then $\beta_i \in S_A$, hence $\mathbf{p}_A(\beta_i) = 0$, and therefore $\mathbf{p}_\cap(\beta_i) = 0$.
- If $\beta_i \notin S_\cap$, then $\beta_i \notin S_A$, so $\mathbf{p}_A(\beta_i) \neq 0$. Since $\mathbf{r}'_A$ is a uniformly random degree- n polynomial, $\mathbf{r}'_A(\beta_i)$ is uniformly distributed over $\mathbb{F}$, and thus $\mathbf{p}_\cap(\beta_i)$ is uniformly distributed over $\mathbb{F}^*$ except with negligible probability that $\mathbf{r}'_A(\beta_i) = 0$.

Thus, on each point $\beta_i \in S_B$, the evaluation distribution of $\tilde{\mathbf{p}}_\cap$ matches that of the real polynomial: zero exactly on the intersection and uniformly random otherwise. Since a degree- $2n$ polynomial is fully determined by its evaluations on $2n+1$ points, and the simulator samples uniformly from the set of all polynomials consistent with these constraints, the full distribution of $\tilde{\mathbf{p}}_\cap$ is statistically indistinguishable from the real $\mathbf{p}_\cap$.

Finally, B computes its output only by evaluating the received polynomial on elements of S_B , and hence outputs exactly $S_\cap$ in both the real and ideal executions. Therefore, the joint view of $\mathcal{A}_B$ and $\mathcal{Z}$ is statistically indistinguishable in H_0 and H_1 . Combining both corruption cases, for every PPT semi-honest adversary $\mathcal{A}$ corrupting at most one party, there exists a PPT simulator $\mathcal{S}$ such that

$$\text{Exec}_{\Pi_{\text{PSI}}^{\text{opa}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{OPA}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{\text{PSI}}, \mathcal{S}, \mathcal{Z}}$$

for every PPT environment $\mathcal{Z}$. Hence $\Pi_{\text{PSI}}^{\text{opa}}$ UC-realizes $\mathcal{F}_{\text{PSI}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model. $\square$

I Proof of Theorem 11

Proof. Let $\mathcal{A}$ be a dummy adversary mounting at most one specious corruption, and let $\mathcal{Z}$ be any PPT environment. By definition of wsrUC security, we must construct a simulator $\mathcal{S}$ such that

$$\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{PSI}}), \mathcal{S}, \mathcal{Z}}. \quad (7)$$

We use a sequence of $n+1$ hybrid experiments $H_0, H_1, \dots, H_n$. In hybrid H_i , the first i invocations of the sanitizing OPE protocol $\hat{\Pi}_{\text{OPE}}$ inside Π_{PSI} are replaced by invocations of the wrapped ideal functionality $\text{Wrap}(\mathcal{F}_{\text{OPE}})$, while the remaining $n-i$ invocations are still executed as $\hat{\Pi}_{\text{OPE}}$. In particular:

- H_0 is the real-world execution $\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}}$: all n OPE invocations are run as $\hat{\Pi}_{\text{OPE}}$ instances over $\mathcal{F}_{\text{SAT}}$.
- For $j \in [n]$, H_j differs from H_{j-1} in exactly one place: the j -th $\hat{\Pi}_{\text{OPE}}$ invocation is replaced by an ideal call to $\text{Wrap}(\mathcal{F}_{\text{OPE}})$ that takes sender input $\mathbf{p}_A$ and receiver input β_j , and returns $z_j = \mathbf{p}_A(\beta_j)$ to Bob.

Lemma 23. $H_0 \stackrel{c}{\approx} H_n$.

Proof. For each $j \in [n]$, the only difference between H_{j-1} and H_j is the implementation of the j -th OPE invocation: in H_{j-1} it is realized by $\hat{\Pi}_{\text{OPE}}$, whereas in H_j it is replaced by $\text{Wrap}(\mathcal{F}_{\text{OPE}})$. By Theorem 7, the protocol $\hat{\Pi}_{\text{OPE}}$ wsrUC-realizes $\mathcal{F}_{\text{OPE}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model; equivalently, $\hat{\Pi}_{\text{OPE}}$ UC-realizes $\text{Wrap}(\mathcal{F}_{\text{OPE}})$ in that hybrid model under specious environments and adversaries. Hence there exists a PPT simulator $\mathcal{S}_j$ such that, for every specious PPT environment,

$$H_{j-1} \stackrel{c}{\approx} H_j.$$

Chaining these n steps yields $H_0 \stackrel{c}{\approx} H_n$. $\square$

The hybrid H_n is, by construction, an execution of Π_{PSI} in which each OPE call is to $\text{Wrap}(\mathcal{F}_{\text{OPE}})$. Because $\text{Wrap}(\mathcal{F}_{\text{OPE}})$ behaves on the input–output interface exactly as $\mathcal{F}_{\text{OPE}}$ does (under the standard corruption translation table [Table 1](#)), H_n is statistically equivalent to an execution of Π_{PSI} in the $\mathcal{F}_{\text{OPE}}$ -hybrid model. By [Theorem 10](#), Π_{PSI} UC-realizes $\mathcal{F}_{\text{PSI}}$ in the $\mathcal{F}_{\text{OPE}}$ -hybrid model; instantiating that simulator yields a simulator $\mathcal{S}$ such that

$$H_n \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{PSI}}), \mathcal{S}, \mathcal{Z}}.$$

Combining with [Lemma 23](#),

$$\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}} = H_0 \stackrel{c}{\approx} H_n \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{PSI}}), \mathcal{S}, \mathcal{Z}},$$

which establishes [\(7\)](#).

We now show that both PSI firewalls are strongly sanitizing.

Lemma 24. *The composed firewall F_A^{PSI} is strongly sanitizing.*

Proof. Fix a PPT environment $\mathcal{Z}$ that does not corrupt F_A^{PSI} but mounts a specious corruption $(\text{SPECIOUS}, \tilde{C}_A)$ on Alice’s core. We must show

$$\text{Exec}_{\Pi_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}} \stackrel{c}{\approx} \text{Exec}_{\hat{\Pi}_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}},$$

where $\hat{\Pi}_{\text{PSI}}$ runs with the incorruptible Alice core $\hat{C}_A$ that ignores all SPECIOUS commands and keeps executing the honest code of C_A , $\mathcal{A}$ is the dummy adversary, so $\mathcal{Z}$ plays party $B = (C_B, F_B^{\text{PSI}})$ in addition to mounting the specious corruption on A.

By construction, Alice’s PSI core C_A feeds the OPE layer with the same sender input $\mathbf{p}_A(X) = \prod_{\gamma \in S_A} (X - \gamma)$ into all n OPE sub-protocols. Equivalently, C_A decomposes into n parallel sub-cores $C_{S,A}^{(1)}, \dots, C_{S,A}^{(n)}$, where $C_{S,A}^{(j)}$ is exactly the OPE-sender sub-core supplying the j -th OPE instance with input $\mathbf{p}_A$. Since $\tilde{C}_A$ is specious for C_A , it follows that the marginal $\tilde{C}_A^{(j)}$ induced on the j -th OPE sub-core is specious for $C_{S,A}^{(j)}$.

By [Lemma 15](#), the per-instance OPE-sender firewall $F_S^{(j)}$ is strongly sanitizing for $C_{S,A}^{(j)}$. Consequently, for every $j \in [n]$, the marginal view of $\mathcal{Z}$ on the j -th OPE sub-protocol is computationally indistinguishable in the two experiments. Because the firewalls $F_S^{(1)}, \dots, F_S^{(n)}$ run in parallel and share no internal state (this is the very content of the $\diamond$ -composition), the joint view of $\mathcal{Z}$ across all n OPE sub-protocols is computationally indistinguishable in the two experiments, by a standard n -step hybrid argument over the OPE instances.

It remains to observe that, in the one-sided OPE-based PSI, the only messages produced by F_A^{PSI} beyond those produced by the underlying $F_S^{(j)}$ ’s are the inputs and outputs on the $\mathcal{F}_{\text{SAT}}$ -interfaces of the PSI layer; Alice’s core receives no protocol message outside the OPE sub-transcripts and outputs $\perp$. In particular, no additional algebraic value crosses the network at the PSI layer. Hence the

joint view of $\mathcal{Z}$ in the two executions is determined entirely by the n OPE sub-transcripts already shown to be computationally indistinguishable. We conclude that F_A^{PSI} is strongly sanitizing. $\square$

Lemma 25. *The composed firewall F_B^{PSI} is strongly sanitizing.*

Proof. Symmetric to the proof of Lemma 24. Fix a PPT environment $\mathcal{Z}$ that does not corrupt F_B^{PSI} but mounts a specious corruption ($\text{SPECIOUS}, \tilde{C}_B$) on Bob's core. We must show

$$\text{Exec}_{\hat{\Pi}_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}} \stackrel{c}{\approx} \text{Exec}_{\hat{\Pi}_{\text{PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}}},$$

where $\hat{\Pi}_{\text{PSI}}$ runs with the incorruptible Bob core $\hat{C}_B$ in place of C_B , $\mathcal{A}$ is the dummy adversary, and $\mathcal{Z}$ plays Alice's side $A = (C_A, F_A^{\text{PSI}})$ in addition to mounting the specious corruption on B.

By Lemma 16, the per-instance OPE-receiver firewall $F_R^{(j)}$ is strongly sanitizing for $C_{R,B}^{(j)}$. Consequently, for every $j \in [n]$, the marginal view of $\mathcal{Z}$ on the j -th OPE sub-protocol is computationally indistinguishable in the two experiments. Because the firewalls $F_R^{(1)}, \dots, F_R^{(n)}$ run in parallel and share no internal state, the joint view of $\mathcal{Z}$ across all n OPE sub-instances is computationally indistinguishable in the two experiments, by a standard n -step hybrid argument over the OPE instances. $\square$

Combining the hybrid argument of Lemma 23 with the strong sanitation guarantees of Lemma 24 and Lemma 25, we conclude that $\hat{\Pi}_{\text{PSI}}$ wsrUC-realizes $\mathcal{F}_{\text{PSI}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPE}})$ -hybrid model. This completes the proof of Theorem 11. $\square$

J Sanitizing PSI from OPA

We lift the one-sided PSI protocol $\Pi_{\text{PSI}}^{\text{opa}}$ of Figure 21 to the subversion-resilient setting. Recall that the sender A encodes its set S_A as the degree- n polynomial $\mathbf{p}_A$ vanishing on S_A , samples random $\mathbf{r}_A, \mathbf{r}'_A$ of degree n , and forms $\mathbf{q}_A = \mathbf{p}_A \cdot \mathbf{r}'_A$; the receiver B encodes S_B as $\mathbf{p}_B$. The parties invoke a single $\mathcal{F}_{\text{OPA}}$ call: A as sender with input $(\mathbf{q}_A, \mathbf{r}'_A)$ and B as receiver with input $\mathbf{p}_B$, obtaining $\mathbf{p}_\cap = \mathbf{q}_A + \mathbf{r}_A \cdot \mathbf{p}_B$. B then evaluates $\mathbf{p}_\cap$ on S_B and outputs $S_\cap = \{\beta \in S_B : \mathbf{p}_\cap(\beta) = 0\}$.

The key structural observation simplifying the lift is that *the only message that crosses the network between the parties at the PSI layer is the OPA sub-transcript*: the post-OPA computation by B is local, A produces no message outside the OPA call, and the polynomial encodings are formed inside the respective cores. Sanitizing $\Pi_{\text{PSI}}^{\text{opa}}$ therefore amounts *exactly* to replacing the $\mathcal{F}_{\text{OPA}}$ call with an invocation of $\hat{\Pi}_{\text{OPA}}$ from subsection 5.2

Construction. The sanitizing protocol $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ is obtained from $\Pi_{\text{PSI}}^{\text{opa}}$ by replacing the single $\mathcal{F}_{\text{OPA}}$ call with an invocation of $\hat{\Pi}_{\text{OPA}}$, where A plays the OPA sender with input $(\mathbf{q}_A, \mathbf{r}'_A)$ and B plays the OPA receiver with input $\mathbf{p}_B$. Since the OPA

call is the only ideal-functionality invocation mediating communication, the PSI firewalls collapse to the OPA firewalls:

$$F_A^{\text{PSI}} := F_S^{\text{OPA}}, \quad F_B^{\text{PSI}} := F_R^{\text{OPA}},$$

where $F_S^{\text{OPA}}, F_R^{\text{OPA}}$ are themselves the parallel composition ($\diamond$) of the $2d+1$ per-OLE-instance firewalls. No further composition is needed at the PSI layer; every property of the underlying OPA firewalls – in particular, strong sanitation – is inherited verbatim by the PSI firewalls. Correctness of $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ follows from the correctness of $\hat{\Pi}_{\text{OPA}}$ and of $\Pi_{\text{PSI}}^{\text{opa}}$

Lemma 26. *The protocol $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ is correct.*

Proof. When both parties run honestly $\hat{\Pi}_{\text{OPA}}$ delivers to B exactly $\mathbf{p}_\cap(X) = \mathbf{q}_A(X) + \mathbf{r}_A(X) \cdot \mathbf{p}_B(X)$, the same polynomial an ideal $\mathcal{F}_{\text{OPA}}$ would deliver. Substituting this back into $\Pi_{\text{PSI}}^{\text{opa}}$ makes $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ semantically equivalent to $\Pi_{\text{PSI}}^{\text{opa}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model; by correctness of $\Pi_{\text{PSI}}^{\text{opa}}$, B outputs $S_\cap = S_A \cap S_B$ with overwhelming probability (via the Schwartz–Zippel argument), and A outputs $\perp$. $\square$

J.1 Proof of Theorem 15

Theorem 15 (Security of Sanitizing PSI from OPA). *The protocol $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ $\text{wsrUC-realizes } \mathcal{F}_{\text{PSI}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}})$ -hybrid model.*

Proof. Let $\mathcal{A}$ be the dummy adversary mounting at most one specious corruption and $\mathcal{Z}$ any PPT SPECIOUS environment. We must build $\mathcal{S}$ with $\text{Exec}_{\hat{\Pi}_{\text{PSI}}^{\text{opa}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{PSI}}), \mathcal{S}, \mathcal{Z}}$. Define the hybrid H by replacing the single $\hat{\Pi}_{\text{OPA}}$ invocation inside $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ with a call to $\text{Wrap}(\mathcal{F}_{\text{OPA}})$. By Theorem 9, $\hat{\Pi}_{\text{OPA}}$ UC-realizes $\text{Wrap}(\mathcal{F}_{\text{OPA}})$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OLE}})$ -hybrid model under specious environments, so there exists $\mathcal{S}_1$ with $\text{Exec}_{\hat{\Pi}_{\text{PSI}}^{\text{opa}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}}} \stackrel{c}{\approx} \text{H}$. By the corruption translation Table 1, $\text{Wrap}(\mathcal{F}_{\text{OPA}})$ behaves on the input–output interface exactly as $\mathcal{F}_{\text{OPA}}$; thus H is statistically equivalent to $\Pi_{\text{PSI}}^{\text{opa}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model. By Theorem 12, $\Pi_{\text{PSI}}^{\text{opa}}$ UC-realizes $\mathcal{F}_{\text{PSI}}$ there, yielding $\mathcal{S}_2$ with $\text{H} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{\text{PSI}}), \mathcal{S}, \mathcal{Z}}$ for $\mathcal{S}$ the standard composition of $\mathcal{S}_1, \mathcal{S}_2$. Chaining gives the claim.

Strong sanitation. As noted, $F_A^{\text{PSI}} = F_S^{\text{OPA}}$ and $F_B^{\text{PSI}} = F_R^{\text{OPA}}$ by definition, since $\hat{\Pi}_{\text{PSI}}^{\text{opa}}$ introduces no algebraic message at the PSI layer beyond the OPA sub-transcript and all post-OPA processing is local to the cores. Lemma 19 and Lemma 20 give strong sanitation of $F_S^{\text{OPA}}, F_R^{\text{OPA}}$; hence F_A^{PSI} and F_B^{PSI} are strongly sanitizing. No additional hybrid is needed at the PSI layer. $\square$

K Sanitizing Two Sided PSI

K.1 Correctness and UC Security of $\Pi_{2\text{-PSI}}^{\text{opa}}$

Lemma 27 (Correctness of $\Pi_{2\text{-PSI}}^{\text{opa}}$). *If the AHE is correct, $\Pi_{2\text{-PSI}}^{\text{opa}}$ correctly realizes $\mathcal{F}_{2\text{-PSI}}$.*

Proof. By Correctness of $\Pi_{\text{PSI}}^{\text{opa}}$, B obtains $S_{\cap} = S_A \cap S_B$ except with probability $n^2/|\mathbb{F}| = \text{negl}(\lambda)$. Then A runs KeyGen honestly and sends pk to B; for every $\beta \in S_{\cap}$, B sends $\text{Enc}(\text{pk}, \beta)$, and by correctness of AHE, $\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, \beta)) = \beta$ except with $\text{negl}(\lambda)$ probability. Hence A recovers $S_{\cap}$, and both parties output $S_{\cap}$. $\square$

Theorem 16 (UC Security of $\Pi_{2\text{-PSI}}^{\text{opa}}$). *If the AHE is IND-CPA secure, $\Pi_{2\text{-PSI}}^{\text{opa}}$ UC-realizes $\mathcal{F}_{2\text{-PSI}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model against static SEMIHONEST adversaries.*

Proof. Let $\mathcal{A}$ be a semi-honest PPT adversary corrupting at most one party. We build $\mathcal{S}$ with $\text{Exec}_{\Pi_{2\text{-PSI}}^{\text{opa}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{OPA}}} \stackrel{c}{\approx} \text{Exec}_{\mathcal{F}_{2\text{-PSI}}, \mathcal{S}, \mathcal{Z}}$.

Case 1: Corrupt A. The simulator $\mathcal{S}_A$:

1. reads S_A from $\mathcal{A}_A$'s input tape, submits it to $\mathcal{F}_{2\text{-PSI}}$, and receives $S_{\cap}$;
2. simulates the $\mathcal{F}_{\text{OPA}}$ invocation exactly as in Case 1 of [Theorem 12](#) (records $\mathcal{A}_A$'s sender input (q_A, r'_A) , returns the receipt $\perp$);
3. receives pk from $\mathcal{A}_A$;
4. delivers the batch $T := \{\text{Enc}(\text{pk}, \beta) : \beta \in S_{\cap}\}$ with fresh independent randomness per ciphertext, as the simulated message from B.

The view of $\mathcal{A}_A$ consists of S_A and its randomness, the $\mathcal{F}_{\text{OPA}}$ receipt, and T . The first three are produced identically in real and ideal ([Theorem 12](#), Case 1); the batch T is identically distributed since in the real execution B encrypts the same set $S_{\cap}$ under the same pk with fresh independent randomness.

Case 2: Corrupt B. The simulator $\mathcal{S}_B$:

1. reads S_B from $\mathcal{A}_B$'s input tape, submits it to $\mathcal{F}_{2\text{-PSI}}$, and receives $S_{\cap}$;
2. simulates the $\mathcal{F}_{\text{OPA}}$ output as in Case 2 of [Theorem 12](#) (returns a polynomial $\tilde{p}_{\cap}$ vanishing on $S_{\cap}$ and uniform elsewhere);
3. samples $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$ and delivers pk to $\mathcal{A}_B$ as the message from A;
4. receives and discards $\mathcal{A}_B$'s outbound batch (the output for A is delivered by $\mathcal{F}_{2\text{-PSI}}$ in the ideal world).

The simulated $\mathcal{F}_{\text{OPA}}$ output is computationally indistinguishable from the real one; the simulated pk is identically distributed (honest KeyGen); and $\mathcal{A}_B$, being SEMIHONEST, produces T deterministically from $S_{\cap}$, pk and its tape. $\square$

K.2 Message Transmission

The regular ideal functionality $\mathcal{F}_{\text{MT}}$ is given in [Figure 22](#). The textbook realization Π_{MT} is the PKE channel: the receiver samples $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and sends pk to the sender; the sender returns $\langle m \rangle \leftarrow \text{Enc}(\text{pk}, m)$; the receiver outputs $m = \text{Dec}(\text{sk}, \langle m \rangle)$. The sanitizing variant $\hat{\Pi}_{\text{MT}}$ is given in [Figure 23](#).

Theorem 17. *If $(\text{KeyGen}, \text{Enc}, \text{Dec})$ is IND-CPA secure, Π_{MT} UC-realizes $\mathcal{F}_{\text{MT}}$ against static semi-honest adversaries.*

$\mathcal{F}_{\text{MT}}$

Parameters. A key-malleable AHE scheme $(\text{KeyGen}, \text{Enc}, \text{Dec})$ with message space $\mathcal{M}$, and a session identifier sid .

Parties. Sender S , receiver R .

- On (sid, pk) from R : if no record exists, store (sid, pk) and leak it to S . Ignore further pk -messages.
- On (sid, m) from S , $m \in \mathcal{M}$: if (sid, pk) is stored, compute $c \leftarrow \text{Enc}(\text{pk}, m)$, output (sid, c) to R , $(\text{sid}, \text{receipt})$ to S , leak (sid, c) to S , halt.

Fig. 22: The Message Transmission Functionality.

Proof. For a corrupted sender, $\mathcal{S}_S$ reads m from the sender's tape, submits (sid, m) to $\mathcal{F}_{\text{MT}}$, and forwards the leaked pk to the sender. The resulting $\text{Enc}(\text{pk}, m)$ is identically distributed to the $\mathcal{F}_{\text{MT}}$ -leaked ciphertext. For a corrupted receiver, $\mathcal{S}_R$ forwards the received pk to $\mathcal{F}_{\text{MT}}$ and relays the leaked ciphertext to the receiver. Joint views are identically distributed; IND-CPA hides m from $\mathcal{Z}$. $\square$

Protocol: $\hat{\Pi}_{\text{MT}}$

Let AHE be a key-malleable AHE with message space $\mathcal{M}$. Parties $S = (\mathcal{C}_S, \mathcal{F}_S)$ and $R = (\mathcal{C}_R, \mathcal{F}_R)$. S with input m .

- **Round 1:** $\mathcal{C}_R$ samples $(\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and sends pk to $\mathcal{F}_R$. $\mathcal{F}_R$ computes $(\widetilde{\text{pk}}, \rho) \leftarrow \text{MaulPK}(\text{pk})$ and sends $\widetilde{\text{pk}}$ to $\mathcal{F}_S$. $\mathcal{F}_S$ computes $(\text{pk}', \tau) \leftarrow \text{MaulPK}(\widetilde{\text{pk}})$ and hands pk' to $\mathcal{C}_S$.
- **Round 2:** $\mathcal{C}_S$ computes $\langle m \rangle' \leftarrow \text{Enc}(\text{pk}', m)$ and sends it to $\mathcal{F}_S$. $\mathcal{F}_S$ sets $\widetilde{\langle m \rangle} = \text{Rerand}(\text{AlignEnc}(\langle m \rangle', \widetilde{\text{pk}}, \tau))$ and forwards it to $\mathcal{F}_R$. $\mathcal{F}_R$ sets $\langle m \rangle = \text{Rerand}(\text{AlignEnc}(\widetilde{\langle m \rangle}, \text{pk}, \rho))$ and delivers it to $\mathcal{C}_R$.

Output: R outputs $m = \text{Dec}(\text{sk}, \langle m \rangle)$.

Fig. 23: 2-Round Sanitizing MT Protocol $\hat{\Pi}_{\text{MT}}$

Theorem 18. *If AHE is a key-malleable AHE defined in Definition 4 that is IND-CPA secure, then $\hat{\Pi}_{\text{MT}}$ wsrUC-realizes $\mathcal{F}_{\text{MT}}$ in the $\mathcal{F}_{\text{SAT}}$ -hybrid model, and both firewalls $\mathcal{F}_S^{\text{MT}}, \mathcal{F}_R^{\text{MT}}$ are strongly sanitizing.*

Proof. The wsrUC argument is the restriction of Theorem 5 to a single homomorphic operation: by (KM3) both $\widetilde{\text{pk}}$ and pk' are statistically close to fresh KeyGen outputs, and by (KM2) followed by Rerand each aligned ciphertext is statistically close to a fresh encryption of the same plaintext under the target key; switching the encrypted plaintext between m and 0 reduces to IND-CPA. Strong sanitation of $\mathcal{F}_S^{\text{MT}}$ and $\mathcal{F}_R^{\text{MT}}$ follows by the same template as Lemma 10

and [Lemma 11](#) for the 2-round OLE firewalls, thus preserves indistinguishability. $\square$

The two-sided variant $\Pi_{2\text{-PSI}}^{\text{opa}}$ extends $\Pi_{\text{PSI}}^{\text{opa}}$ with a reverse channel that conveys the intersection from B back to A:

1. The parties run $\Pi_{\text{PSI}}^{\text{opa}}$. After this phase, B holds $S_{\cap} = S_A \cap S_B$ (with overwhelming probability) and A holds $\perp$.
2. A runs $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$ and sends pk to B.
3. B encrypts each $\beta \in S_{\cap}$ as $\langle \beta \rangle \leftarrow \text{Enc}(\text{pk}, \beta)$ and sends the (unordered) batch $T = \{\langle \beta \rangle : \beta \in S_{\cap}\}$ to A.
4. A outputs $S_{\cap} := \{\text{Dec}(\text{sk}, c) : c \in T\}$.

The corresponding ideal functionality $\mathcal{F}_{2\text{-PSI}}$ is the natural two-output extension of $\mathcal{F}_{\text{PSI}}$ delivering $S_{\cap}$ to *both* parties.

K.3 Sanitizing Two-Sided PSI from OPA

$\Pi_{2\text{-PSI}}^{\text{opa}}$ is the composition of (i) the one-sided PSI phase, whose only cross-party message is the $\mathcal{F}_{\text{OPA}}$ call; and (ii) the reverse channel transmitting pk from A to B and the batch T from B to A. We sanitize (i) by replacing Π_{PSI} with $\widehat{\Pi}_{\text{PSI}}$ from the previous section, and (ii) by a *single* invocation of the sanitizing message-transmission protocol $\widehat{\Pi}_{\text{MT}}$ of [Figure 23](#), with A playing the MT receiver (it provides pk and gets the encrypted batch) and B the MT sender (it inputs the plaintexts to be encrypted).

A *single batched MT instance* $\widehat{\Pi}_{\text{MT}}^{\text{batch}}$. A naive sanitation would invoke $\widehat{\Pi}_{\text{MT}}$ once per ciphertext in T , yielding $|S_{\cap}|$ parallel MT instances. We instead use a single MT instance handling the entire batch: one MaulPK chain on the forward trip, followed by per-ciphertext AlignEnc + Rerand on the return trip with fresh randomness per ciphertext.

Round 1 (public-key forward, A $\rightarrow$ B):

- C_A runs $(\text{sk}, \text{pk}) \leftarrow \text{KeyGen}(1^\lambda)$ and sends pk to its firewall.
- F_A^{MT} computes $(\text{pk}', \rho) \leftarrow \text{MaulPK}(\text{pk})$ and forwards pk' to F_B^{MT} .
- F_B^{MT} computes $(\widetilde{\text{pk}}, \tau) \leftarrow \text{MaulPK}(\text{pk}')$ and hands $\widetilde{\text{pk}}$ to C_B .

Round 2 (ciphertext batch return, B $\rightarrow$ A):

- For each $\beta_i \in S_{\cap}$, C_B computes $\langle \beta_i \rangle' \leftarrow \text{Enc}(\widetilde{\text{pk}}, \beta_i)$ and sends the batch $T' = (\langle \beta_1 \rangle', \dots, \langle \beta_k \rangle')$ to F_B^{MT} , where $k = |S_{\cap}|$.
- F_B^{MT} computes, independently for each i , $\widetilde{\langle \beta_i \rangle} \leftarrow \text{Rerand}(\text{AlignEnc}(\langle \beta_i \rangle', \text{pk}', \tau))$ with fresh Rerand randomness, and forwards T'' to F_A^{MT} .
- F_A^{MT} computes, again with fresh per-ciphertext randomness, $\langle \beta_i \rangle \leftarrow \text{Rerand}(\text{AlignEnc}(\widetilde{\langle \beta_i \rangle}, \text{pk}, \rho))$ and delivers T to C_A .

Output: C_A outputs $\{\text{Dec}(\text{sk}, c) : c \in T\}$.

The firewalls' work is independent of k at the public-key level but applies per ciphertext on the batch with fresh independent randomness for each $\text{AlignEnc} + \text{Rerand}$. The IND-CPA argument extends from one ciphertext to the batch by a standard k -step hybrid (proved below).

Construction of $\widehat{\Pi}_{2\text{-PSI}}$ and composed firewalls. $\widehat{\Pi}_{2\text{-PSI}}$ is obtained from $\Pi_{2\text{-PSI}}^{\text{opa}}$ by replacing the embedded Π_{PSI} with $\widehat{\Pi}_{\text{PSI}}$, and the reverse channel with $\widehat{\Pi}_{\text{MT}}^{\text{batch}}$. Since the two sub-protocols execute on disjoint time windows of the network, the party-level firewalls are the *sequential composition* ($\boxtimes$) of the corresponding sub-firewalls:

$$F_A^{2\text{-PSI}} := F_A^{\text{PSI}} \boxtimes F_A^{\text{MT}}, \quad F_B^{2\text{-PSI}} := F_B^{\text{PSI}} \boxtimes F_B^{\text{MT}},$$

where $F_A^{\text{PSI}} = F_S^{\text{OPA}}$ and $F_B^{\text{PSI}} = F_R^{\text{OPA}}$ as in the previous section. The sub-firewalls share no internal state; F_A^{PSI} is active during $\widehat{\Pi}_{\text{PSI}}$, then F_A^{MT} takes over for the reverse channel.

Lemma 28 (Correctness of $\widehat{\Pi}_{2\text{-PSI}}$). *If the AHE is correct and satisfies the key-malleability properties of Definition 4, then $\widehat{\Pi}_{2\text{-PSI}}$ correctly realizes $\mathcal{F}_{2\text{-PSI}}$.*

Proof. The embedded $\widehat{\Pi}_{\text{PSI}}$ leaves B with $S_\cap = S_A \cap S_B$ with overwhelming probability. For each $\beta \in S_\cap$ in $\widehat{\Pi}_{\text{MT}}^{\text{batch}}$: C_B produces $\langle \beta \rangle' \leftarrow \text{Enc}(\widetilde{\text{pk}}, \beta)$; by (KM4) of Definition 4 followed by Rerand , $\langle \beta \rangle = \text{Rerand}(\text{AlignEnc}(\langle \beta \rangle', \text{pk}', \tau))$ is distributed as a fresh $\text{Enc}(\text{pk}', \beta)$; applying (KM4) and Rerand again inside F_A^{MT} yields $\langle \beta \rangle$ distributed as a fresh $\text{Enc}(\text{pk}, \beta)$. Correctness of Dec gives $\text{Dec}(\text{sk}, \langle \beta \rangle) = \beta$, so C_A recovers $S_\cap$. The argument is generic, relying only on correctness of $\widehat{\Pi}_{\text{PSI}}$, properties (KM3), (KM4), plaintext-preservation of Rerand , and correctness of Dec . $\square$

Theorem 19 (wsrUC Security of $\widehat{\Pi}_{2\text{-PSI}}$). *If the AHE is IND-CPA secure and satisfies the key-malleability properties of Definition 4, then $\widehat{\Pi}_{2\text{-PSI}}$ wsrUC-realizes $\mathcal{F}_{2\text{-PSI}}$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}}, \mathcal{F}_{\text{MT}})$ -hybrid model.*

Proof. Let $\mathcal{A}$ be the dummy adversary mounting at most one specious corruption and $\mathcal{Z}$ any PPT SPECIOUS environment. We must exhibit $\mathcal{S}$ with $\text{Exec}_{\widehat{\Pi}_{2\text{-PSI}}, \mathcal{A}, \mathcal{Z}}^{\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}}, \mathcal{F}_{\text{MT}}} \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{2\text{-PSI}}), \mathcal{S}, \mathcal{Z}}$. Define: H_0 is the real-world execution; H_1 replaces the embedded $\widehat{\Pi}_{\text{PSI}}$ with $\text{Wrap}(\mathcal{F}_{\text{PSI}})$; H_2 further replaces $\widehat{\Pi}_{\text{MT}}^{\text{batch}}$ with $\text{Wrap}(\mathcal{F}_{\text{MT}})$.

Step 1: $H_0 \stackrel{c}{\approx} H_1$. As, $\widehat{\Pi}_{\text{PSI}}$ UC-realizes $\text{Wrap}(\mathcal{F}_{\text{PSI}})$ in the $(\mathcal{F}_{\text{SAT}}, \mathcal{F}_{\text{OPA}})$ -hybrid model under specious environments, yielding $\mathcal{S}_1$.

Step 2: $H_1 \stackrel{c}{\approx} H_2$. We extend Theorem 18 from a single ciphertext to a batch of $k = |S_\cap|$ via a standard hybrid. For $i \in \{0, 1, \dots, k\}$, let $H_{1,i}$ be H_1 with the first i ciphertexts in T' replaced by encryptions of $0_{\mathbb{F}}$ (firewall transformations applied as in the real protocol).

- $H_{1,0} = H_1$ by construction.
- For each $i \in [k]$, the only difference between $H_{1,i-1}$ and $H_{1,i}$ is the plaintext of the i -th ciphertext in T' . A distinguisher with advantage ε yields an IND-CPA adversary against the AHE on key $\widetilde{\text{pk}}$ (statistically close to a fresh KeyGen output by (KM3)) with the same advantage: embed the IND-CPA challenge as the i -th slot, simulate the other $k-1$ encryptions, and apply the firewall code to the whole batch with fresh Rerand randomness. Hence $H_{1,i-1} \stackrel{c}{\approx} H_{1,i}$.
- Chaining the k steps gives $H_{1,0} \stackrel{c}{\approx} H_{1,k}$; in $H_{1,k}$ the batch T' carries no information about $S_\cap$.

By (KM4) followed by fresh Rerand, each $\langle \widetilde{\beta_i} \rangle$ and $\langle \beta_i \rangle$ are statistically close to fresh encryptions of β_i under pk' and pk respectively, independently across i since the Rerand operations use fresh per-ciphertext randomness. The simulator $\mathcal{S}_2$ therefore extracts the batch from $\text{Wrap}(\mathcal{F}_{\text{MT}})$, samples one keypair, and applies the firewall code symbolically to $S_\cap$; the resulting transcript is statistically close to the one in H_1 .

Step 3: $H_2 \stackrel{c}{\approx} \text{Exec}_{\text{Wrap}(\mathcal{F}_{2\text{-PSI}}), \mathcal{S}, \mathcal{Z}}$. By the corruption translation table (Table 1), $\text{Wrap}(\mathcal{F}_{\text{OPA}})$ and $\text{Wrap}(\mathcal{F}_{\text{MT}})$ behave on the input-output interface as their unwrapped counterparts, so H_2 is statistically equivalent to $\Pi_{2\text{-PSI}}^{\text{opa}}$ in the $(\mathcal{F}_{\text{OPA}}, \mathcal{F}_{\text{MT}})$ -hybrid model. By Theorem 16, $\Pi_{2\text{-PSI}}^{\text{opa}}$ UC-realizes $\mathcal{F}_{2\text{-PSI}}$ in the $\mathcal{F}_{\text{OPA}}$ -hybrid model; extending that simulator to additionally extract the batch from $\mathcal{F}_{\text{MT}}$ (resp. submit pk on behalf of a corrupted receiver) yields $\mathcal{S}_3$, and we set $\mathcal{S} := \mathcal{S}_3 \circ \mathcal{S}_2 \circ \mathcal{S}_1$.

Strong sanitation of $\mathcal{F}_A^{2\text{-PSI}}, \mathcal{F}_B^{2\text{-PSI}}$. We argue strong sanitation by leveraging the strong sanitation guarantees already established for the sub-firewalls:

- By Theorem 15, $\mathcal{F}_A^{\text{PSI}}$ and $\mathcal{F}_B^{\text{PSI}}$ are strongly sanitizing.
- By Theorem 15, $\mathcal{F}_A^{\text{MT}}$ and $\mathcal{F}_B^{\text{MT}}$ are strongly sanitizing for single-instance MT; the batched $\widehat{\Pi}_{\text{MT}}^{\text{batch}}$ inherits strong sanitation from the same proof, since firewall transformations are applied per ciphertext with fresh randomness (a per-ciphertext invocation of (KM3), (KM4) and Rerand).

Sequential composition on disjoint time windows with no shared state preserves strong sanitation: a SPECIOUS core decomposes into a PSI-phase sub-core and an MT-phase sub-core; the marginal speciousness on each sub-core is neutralized by the corresponding strongly sanitizing sub-firewall; and a two-step hybrid over the two phases yields the joint indistinguishability of $\mathcal{F}_A^{2\text{-PSI}}$ and $\mathcal{F}_B^{2\text{-PSI}}$. $\square$