Appendix D. Proof of Framework Security

In this appendix, we prove security for the framework of Fig. 2, under the assumption that the constituent functionalities $\mathcal{F}_{\mathsf{EM}}, \mathcal{F}_{\mathsf{TC}}, \mathcal{F}_{\mathsf{IC}}$ are secure. (The proofs of security for these functionalities can be found in Sec. E.) We reduce the security of our framework to typical properties of types of function families, which we recount below.

Definition 5 (Weak preimage resistance)**.** *Let $g : \mathcal{K} \times \mathcal{D} \to \mathcal{R}$ be a family of functions, and let $\mathtt{A}$ be an algorithm that takes an oracle $\mathcal{D} \to \mathcal{R}$ and returns an element of $\mathcal{D}$. Let*

```
Experiment $\mathbf{Expt}_g^{wpr}(\mathtt{A})$
    $k \overset{\$}{\leftarrow} \mathcal{K}$
    $w \overset{\$}{\leftarrow} \mathcal{D}$
    $\gamma \leftarrow g_k(w)$
    $w' \leftarrow \mathtt{A}^{g_k(\cdot)}(\gamma)$
    if $g_k(w') = \gamma$
        then return 1
        else return 0
```

Then,

$$\mathsf{Adv}_g^{wpr}(\mathtt{A}) = \mathbb{P}\Big(\mathbf{Expt}_{\mathsf{PRF}}^{wpr}(\mathtt{A}) = 1\Big)$$
$$\mathsf{Adv}_g^{wpr}(t, q) = \max_{\mathtt{A}} \mathsf{Adv}_g^{wpr}(\mathtt{A})$$

where the maximum is taken over all algorithms $\mathtt{A}$ that run in time at most t and make at most q oracle queries.

Definition 6 (One-time pairwise unpredictability)**.** *Let $g : \mathcal{K} \times \mathcal{D} \to \mathcal{D}$ be a family of permutations, and let $\mathtt{A}$ be an algorithm that returns a pair of elements from $\mathcal{D}$. Let*

```
Experiment $\mathbf{Expt}_{\mathsf{P}}^{opu} g(\mathtt{A})$
    $k \overset{\$}{\leftarrow} \mathcal{K}$
    $w \leftarrow \mathtt{A}()$
    $x \leftarrow g_k(w)$
    $(w', y) \leftarrow \mathtt{A}(x)$
    if $g_k(w') = y$
        then return 1
        else return 0
```

Then,

$$\mathsf{Adv}_g^{opu}(\mathtt{A}) = \mathbb{P}\Big(\mathbf{Expt}_{\mathsf{P}}^{opu}(\mathtt{A}) = 1\Big)$$
$$\mathsf{Adv}_g^{opu}(t) = \max_{\mathtt{A}} \mathsf{Adv}_g^{opu}(\mathtt{A})$$

where the maximum is taken over all algorithms $\mathtt{A}$ that run in time at most t.

D.1. Security against C

D.1.1. In threat model T1

Assuming $\mathcal{F}_{\mathsf{EM}}$, $\mathcal{F}_{\mathsf{TC}}$, and $\mathcal{F}_{\mathsf{IC}}$ are secure, then no transcripts are leaked to C and the only information C learns are the outputs. Since $\mathcal{L}$ is not involved in the computation of any outputs, it is trivial that no information about $\mathcal{U}_{\mathsf{blk}}$ will be leaked to the client than the decision ($w \in \mathcal{U}_{\mathsf{blk}}$ or not) implies in one execution of the explicit check phase.

D.1.2. In threat model T2

In this threat model, C is malicious and aims to have a commitment stored at S on some $w \in \mathcal{U}_{\mathsf{blk}}$. As such, the goal of such a C is to cause the following experiment to return 1, where C is denoted as a triple of algorithms $(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3)$. For each call to the functionalities, each input (output) pairs will be denoted in parenthesis, with the C input (output) as the first item and the S input (output) as the second item.

```
Experiment $\mathbf{Expt}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3)$
    $\langle \phi_1, w \rangle \leftarrow \mathtt{A}_1^{\mathsf{H}(\cdot)}()$
    $k1_{\mathsf{P}} \overset{\$}{\leftarrow} \mathcal{K}_{\mathsf{P}},\ k2_{\mathsf{P}} \overset{\$}{\leftarrow} \mathcal{K}_{\mathsf{P}},$
    $k_{\mathsf{PRF}} \overset{\$}{\leftarrow} \mathcal{K}_{\mathsf{PRF}},\ k_f \overset{\$}{\leftarrow} \mathcal{K}_f$
    $((p_1, p_2), \cdot) \leftarrow \mathcal{F}_{\mathsf{EM}}^{\mathsf{H}(\cdot)}(w, (k1_{\mathsf{P}}, k2_{\mathsf{P}}, k_f))$
    $\langle \phi_2, p_1', p_2' \rangle \leftarrow \mathtt{A}_2^{\mathsf{H}(\cdot)}(\phi_1, p_1, p_2)$
    $(\gamma, \vec{y}) \leftarrow \mathcal{F}_{\mathsf{TC}}^{\mathsf{H}(\cdot)}((p_1', p_2'), (k1_{\mathsf{P}}, k2_{\mathsf{P}}, k_{\mathsf{PRF}}))$
    if $\gamma = \perp$
        then return 0
    $\vec{y}' \leftarrow \mathtt{A}_3^{\mathsf{H}(\cdot), \mathcal{F}_{\mathsf{IC}}^{\mathsf{H}(\cdot)}(\cdot, (\vec{y}, k_{\mathsf{PRF}}))}(\phi_2, \gamma, k_f)$
    if $w \in \mathcal{U}_{\mathsf{blk}} \wedge \vec{y}' = \vec{y}$
        then return 1
        else return 0
```

Then,

$$\mathsf{Adv}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3) = \mathbb{P}\Big(\mathbf{Expt}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3) = 1\Big)$$
$$\mathsf{Adv}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}(t, q) = \max_{\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3} \mathsf{Adv}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3)$$

where the maximum is taken over all adversaries $(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3)$ running in total time t and making at most q oracle queries to $\mathcal{F}_{\mathsf{IC}}(\cdot, (\vec{y}, k_{\mathsf{PRF}}))$ from $\mathtt{A}_3$.

The key distinctions between the prescribed algorithm in Fig. 2 and $\mathbf{Expt}_{\mathsf{PRF,P,H}}^{\mathsf{CC}}$ are that (i) the (malicious) client is permitted to *compute* $\vec{y}'$ (in $\mathtt{A}_3$) and p_1', p_2' (in $\mathtt{A}_2$), whereas a correct client simply sets $\vec{y}' \leftarrow f_{k_f}(w) + \mathsf{H}(w, f_{k_f}(w), k_f)$, $p_1' \leftarrow p_1$, and $p_2' \leftarrow p_2$; and (ii) the (malicious) client is given oracle access to $\mathcal{F}_{\mathsf{IC}}(\cdot, (\gamma, k_{\mathsf{PRF}}))$, to model repeated attempts at implicit check.

Proposition D.1. *If $\mathcal{F}_{\mathsf{EM}}, \mathcal{F}_{\mathsf{TC}}, \mathcal{F}_{\mathsf{IC}}$ are secure and output correctly, then*

$$\mathsf{Adv}_{\mathsf{PRF,P,H}}^{CC}(t, q) \le \mathsf{Adv}_{\mathsf{PRF}}^{wpr}(t', 1) + \mathsf{FAR}_{\mathcal{L},T}^{\mathcal{U}_{\mathsf{blk}}} + \frac{2}{|\mathcal{K}_{\mathsf{P}}|}$$

for $t' = t + \mathsf{O}(1)$.

Proof. Let $(\mathtt{A}_1, \mathtt{A}_2, \mathtt{A}_3)$ be a B-OPRF adversary that runs in time t and makes at most q queries to $\mathcal{F}_{\mathsf{IC}}$ oracle. First note that

$$
\begin{aligned}
&\mathsf{Adv}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3)\\
&= \mathbb{P}\Big(\mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1\Big)\\
&= \mathbb{P}\left(\mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1 \,\middle|\, \begin{array}{r}\neg blocked^{\mathcal{L},T}(f_{k_f}(w))\\ \wedge\ w \in \mathcal{U}_{\mathsf{blk}}\end{array}\right)\\
&\quad\times \mathbb{P}(\neg blocked^{\mathcal{L},T}(f_{k_f}(w)) \mid w \in \mathcal{U}_{\mathsf{blk}}) \times \mathbb{P}(w \in \mathcal{U}_{\mathsf{blk}})\\
&\quad+ \mathbb{P}\left(\mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1 \,\middle|\, \begin{array}{r}blocked^{\mathcal{L},T}(f_{k_f}(w))\\ \wedge\ w \in \mathcal{U}_{\mathsf{blk}}\end{array}\right)\\
&\quad\times \mathbb{P}(blocked^{\mathcal{L},T}(f_{k_f}(w)) \mid w \in \mathcal{U}_{\mathsf{blk}}) \times \mathbb{P}(w \in \mathcal{U}_{\mathsf{blk}})\\
&\le \mathbb{P}(\neg blocked^{\mathcal{L},T}(f_{k_f}(w)) \mid w \in \mathcal{U}_{\mathsf{blk}})\\
&\quad+ \mathbb{P}\left(\mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1 \,\middle|\, \begin{array}{r}blocked^{\mathcal{L},T}(f_{k_f}(w))\\ \wedge\ w \in \mathcal{U}_{\mathsf{blk}}\end{array}\right)
\end{aligned}
$$

The first term is just $\mathsf{FAR}^{\mathcal{U}_{\mathsf{blk}}}_{\mathcal{L},T}$, and so for the rest of the proof, we focus on bounding the second term. Recall that $\mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1$ implies $\neg blocked^{\mathcal{L},T}(\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1))$ and $\vec{y} = \vec{y}'$. This implies $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1) \neq f_{k_f}(w)$ because $blocked^{\mathcal{L},T}(f_{k_f}(w))$. Since $\vec{y} = \mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)+\mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p'_2)$ and $\vec{y} \in \mathbb{F}^{\theta}$. The mapping from $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$ to $\vec{y}$ is injective. $\mathtt{A}_3$ needs to produce $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$ in order to produce $\vec{y}' = \vec{y}$. Note that $\mathcal{F}_{\mathsf{IC}}$ produces an output only if invoked with $\vec{y}$, otherwise it returns $\perp$. Thus, having oracle access to $\mathcal{F}_{\mathsf{IC}}$ provides no advantage to $\mathtt{A}_3$ since it receives an output only if it already knows $\vec{y}$.
Therefore, $\mathtt{A}_3$ can produce $\vec{y}$ only in two ways.

Case (i): $\mathtt{A}_3$ produces $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$ from p'_1 using p_1, p_2. First, note that $\mathtt{A}_1$ gets one invocation to $\mathsf{P}_{k1_{\mathsf{P}}}(\cdot)$ and one invocation to $\mathsf{P}_{k2_{\mathsf{P}}}(\cdot)$ (both via $\mathcal{F}_{\mathsf{EM}}$) to produce p_1 and p_2, respectively. Though $\mathtt{A}_2$ generates p'_1 and p'_2 that are passed to $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(\cdot)$ and $\mathsf{P}^{-1}_{k2_{\mathsf{P}}}(\cdot)$, $\mathtt{A}_3$ never receives the results (except as γ, which is covered in the next case). Thus, $\mathtt{A}_3$'s advantage in producing $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$ from p_1, p_2 is bounded by $\mathsf{Adv}^{\mathsf{opu}}_{\mathsf{P}}(t') = \frac{2}{|\mathcal{K}_{\mathsf{P}}|}$ since P is an unpredictable function. Note that the keys $k1_{\mathsf{P}}$ and $k2_{\mathsf{P}}$ are chosen independently, and if $k2_{\mathsf{P}} \neq k1_{\mathsf{P}}$, p_2 does not provide any information about $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$. Due to the random key selection, the probability that $k2_{\mathsf{P}} = k1_{\mathsf{P}}$ is $\frac{1}{|\mathcal{K}_{\mathsf{P}}|}$.

Case (ii): $\mathtt{A}_3$ obtains $\vec{y}' = \mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1) + \mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p'_2)$ from γ. If $\mathtt{A}_3$ does not know $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1)$ from case (i) then $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1) \oplus \mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p'_2)$ is unknown. Thus, $\mathtt{A}_2$ has not made an oracle query to $\mathsf{PRF}(\cdot)$ for $\mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1) + \mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p'_2)$. In this case, the advantage of $\mathtt{A}_3$ in obtaining $\vec{y}'$ is bounded by $\mathsf{Adv}^{\mathsf{wpr}}_{\mathsf{PRF}}(t', 1)$, the one oracle query being that by $\mathcal{F}_{\mathsf{TC}}$ to make γ.
From the above, we have

$$
\begin{aligned}
&\mathbb{P}\left(\begin{array}{r} y' = \mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p'_1) + \mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p'_2)\\ \wedge\ \mathbf{Expt}^{\mathsf{CC}}_{\mathsf{PRF,P,H}}(\mathtt{A}_1,\mathtt{A}_2,\mathtt{A}_3) = 1\end{array} \,\middle|\, \begin{array}{r}blocked^{\mathcal{L},T}(f_{k_f}(w))\\ \wedge\ w \in \mathcal{U}_{\mathsf{blk}}\end{array}\right)\\
&\le \frac{2}{|\mathcal{K}_{\mathsf{P}}|} + \mathsf{Adv}^{\mathsf{wpr}}_{\mathsf{PRF}}(t', 1) \qquad (9)
\end{aligned}
$$

Summing up the two cases above, gives the result.

□

D.2. Security against S

We prove security for threat model **T3** in this section. Assuming $\mathcal{F}_{\mathsf{EM}}$, $\mathcal{F}_{\mathsf{TC}}$, and $\mathcal{F}_{\mathsf{IC}}$ are secure, the method available to S to attack the framework in Fig. 2 is to guess the input w of C . As such, the goal of such a S is to cause the following experiment to return 1, where S is denoted as algorithm $(\mathsf{S}_1, \mathsf{S}_2)$. In this experiment, we model H as a random oracle [5]. Moreover, we stress that in the experiment below, the statement "$w \xleftarrow{\$} \mathcal{U}$" selects w from $\mathcal{U}$ according to an application-specific distribution that need not be uniform.

Experiment $\mathbf{Expt}^{\mathsf{S}}_{\mathsf{H,PRF}}(\mathsf{S}_1, \mathsf{S}_2)$

```
    $w \xleftarrow{\$} \mathcal{U}$
    $k1_{\mathsf{P}} \xleftarrow{\$} \mathcal{K}_{\mathsf{P}}$, $k2_{\mathsf{P}} \xleftarrow{\$} \mathcal{K}_{\mathsf{P}}$
    $k_{\mathsf{PRF}} \xleftarrow{\$} \mathcal{K}_{\mathsf{PRF}}$, $k_f \xleftarrow{\$} \mathcal{K}_f$
    $((p_1, p_2), \cdot) \leftarrow \mathcal{F}^{\mathsf{H}(\cdot)}_{\mathsf{EM}}(w, (k1_{\mathsf{P}}, k2_{\mathsf{P}}, k_f))$
    $(\gamma, \vec{y}) \leftarrow \mathcal{F}^{\mathsf{H}(\cdot)}_{\mathsf{TC}}((p_1, p_2), (k1_{\mathsf{P}}, k2_{\mathsf{P}}, k_{\mathsf{PRF}}))$
    $\langle \phi_1, k'_f \rangle \leftarrow \mathsf{S}^{\mathsf{H}(\cdot)}_1(\vec{y}, k1_{\mathsf{P}}, k2_{\mathsf{P}}, k_{\mathsf{PRF}}, k_f)$
    $\vec{y}' \leftarrow f_{k'_f}(w) + \mathsf{H}(w, f_{k'_f}(w), k'_f)$
    $\hat{w} \leftarrow \mathsf{S}_2^{\mathsf{H}(\cdot), \mathcal{F}^{\mathsf{H}(\cdot)}_{\mathsf{IC}}(\vec{y}', \cdot)}(k'_f, \phi_1)$
    if $\hat{w} = w$
      then return 1
      else return 0
```

Then,

$$
\begin{aligned}
\mathsf{Adv}^{\mathsf{S}}_{\mathsf{H,PRF}}(\mathsf{S}_1, \mathsf{S}_2) &= \mathbb{P}\Big(\mathbf{Expt}^{\mathsf{S}}_{\mathsf{H,PRF}}(\mathsf{S}_1, \mathsf{S}_2) = 1\Big)\\
\mathsf{Adv}^{\mathsf{S}}_{\mathsf{H,PRF}}(t, q, q_{\mathsf{H}}) &= \max_{\mathsf{S}_1,\mathsf{S}_2} \mathsf{Adv}^{\mathsf{S}}_{\mathsf{H,PRF}}(\mathsf{S}_1, \mathsf{S}_2)
\end{aligned}
$$

where the maximum is taken over all adversaries $(\mathsf{S}_1, \mathsf{S}_2)$ running in total time t, with at most q_{H} queries to H, and at most q oracle queries to $\mathcal{F}^{\mathsf{H}(\cdot)}_{\mathsf{IC}}(y', \cdot)$ from S_2.

To prove a result against such a server adversary, consider the posterior distribution on w, after observing $\neg blocked^{\mathcal{L},T}(f_{k_f}(w))$ for a known k_f. Then, let $\pi(q_{\mathsf{H}})$ denote the probability of guessing w within q_{H} guesses, when guessing values in order of nonincreasing probability according to that posterior distribution. No matter the properties of H, the $(\mathsf{S}_1, \mathsf{S}_2)$ adversary can win the above experiment with probability at least $\pi(q_{\mathsf{H}})$, simply by guessing elements w' of $\mathcal{U}$ in that order, and checking if $\vec{y} = f_{k_f}(w') + \mathsf{H}(w', f_{k_f}(w'), k_f)$. Prop. D.2 says that if H is a random oracle, then this is the best that $(\mathsf{S}_1, \mathsf{S}_2)$ can do.

Proposition D.2. *If* H *is a random oracle and* $\mathcal{F}_{\mathsf{EM}}, \mathcal{F}_{\mathsf{TC}}, \mathcal{F}_{\mathsf{IC}}$ *are secure and output correctly, then*

$$\mathsf{Adv}^{\mathsf{S}}_{\mathsf{H,PRF}}(t, q, q_{\mathsf{H}}) \le \pi(q_{\mathsf{H}})$$

for $t' = t + \mathsf{O}(1)$.

Proof. To win the experiment $\mathbf{Expt}^{\mathsf{S}}_{\mathsf{H,PRF}}$, S_2 needs to successfully guess w. First recall that $(\mathsf{S}_1, \mathsf{S}_2)$ receives no output from $\mathcal{F}_{\mathsf{EM}}$ and so does not learn p_1 or p_2 individually. It does learn $\vec{y} = \mathsf{P}^{-1}_{k1_{\mathsf{P}}}(p_1) + \mathsf{P}^{-1}_{k2_{\mathsf{P}}}(p_2) = f_{k_f}(w) + \mathsf{H}(w, f_{k_f}(w), k_f)$. H is a random oracle that maps $\{0, 1\}^*$ into $\mathbb{F}^\theta$, where $\mathbb{F}$ is a finite field. $\vec{y} \in \mathbb{F}^\theta$ is a vector of field elements, so the addition is a linear combination in the finite field, which is a secure one-time-pad with perfect secrecy. $\vec{y}$ carries no information about w or $f_{k_f}(w)$. S_1 or S_2 therefore has to invoke $\mathsf{H}(w, f_{k_f}(w), k_f)$. As such, the probability the adversary succeeds is as stated in the proposition. $\square$

Appendix E. Full Proofs of Sec. 5

Theorem E.1 ([15]). *Let $\mathbb{F}$ be a finite field of order q. Fix polynomials $p_1(x)$ and $p_2(x)$ of degree δ, and an arbitrary set of points $\mathcal{X} \leftarrow \{x_1, \dots, x_\theta\}$. Let $\overline{r}(x)$ and $\overline{r'}(x)$ be random polynomials of degree $\geq \delta$. Also, let $p'(x) \leftarrow \prod_{x_i \xleftarrow{\$} \mathbb{F}} (x - x_i)$.*

Then, when $\mathsf{Deg}(\gcd(p_1(x), p_2(x))) < \delta - \frac{T}{2}$ *and* $\theta = 2\delta - \frac{T}{2} + 2$,

$$\sum_{Y \leftarrow \mathbb{F}^\theta} |\mathbb{P}\Big(\langle \overline{r}(x_k) \cdot p_1(x_k) + \overline{r'}(x_k) \cdot p_2(x_k)\rangle_{k=1}^{\theta} = Y\Big) - \mathbb{P}\Big(\langle \overline{r}(x_k) \cdot p_1(x_k) + \overline{r'}(x_k) \cdot p'(x_k)\rangle_{k=1}^{\theta} = Y\Big)| \leq \frac{1}{|\mathbb{F}|} \quad (10)$$

Theorem. *Assuming that there is a protocol that securely realizes a garbled circuit, Π_{EM} securely realizes $\mathcal{F}_{\mathsf{EM}}$ against a semi-honest server, and a malicious client (threat model **T2** and **T3** in Sec. 3.3).*

Proof. The entire functionality is implemented in a garbled circuit construction that is generated by the semi-honest S. The party C playing the role of the evaluator is malicious, however it does not get outputs from the circuit unless it correctly evaluates it. Assuming that there is a construction realizing $\mathcal{F}_{GC}$ with a semi-honest garbler and a malicious evaluator, Π_{EM} can be proved to be secure in the $\mathcal{F}_{GC}$-hybrid model. □

Theorem. *Assuming that there are protocols to securely realize $\mathcal{F}_{ole}$ and $\mathcal{F}_{npa}$, Π_{TC} securely realized $\mathcal{F}_{\mathsf{TC}}$ against a semi-honest server, and a malicious client (threat models **T2** and **T3** described in Sec. 3.3).*

Proof. We will show that there is a polynomial time simulator that simulates C 's and S's views.

When S is corrupt: The simulator obtains $k1_{\mathsf{P}} \leftarrow \langle \vec{a}, \vec{b} \rangle$ and $k2_{\mathsf{P}} \leftarrow \langle \vec{a'}, \vec{b'} \rangle$, and $r'_1(x), \dots, r'_n(x)$ from S's random tape. Let $\vec{a} \leftarrow \langle a_1, \dots, a_\theta \rangle$ and $\vec{b} \leftarrow \langle b_1, \dots, b_\theta \rangle$. Also, let $\vec{a'} \leftarrow \langle a'_1, \dots, a'_\theta \rangle$ and $\vec{b'} \leftarrow \langle b'_1, \dots, b'_\theta \rangle$.
For $i \in [1, n]$, the simulator obtains $l_i \in \mathcal{L}$ from S's input tape; Recall S is assumed to be semi-honest. Let $l_i = \langle \vec{l}_{i,1}, \dots, \vec{l}_{i,\theta} \rangle$ and let $e_{l_i}(x)$ be the polynomial obtained by interpolating the points $\{(x_1, \vec{l}_{i,1}), \dots, (x_\theta, \vec{l}_{i,\theta})\}$.
First consider the case where S's output tape has $\gcd(e_w(x), e_{l_i}(x))$ for $i \in [1, n]$. There are two cases here

1) *For $j \neq i$:* When S's output tape does not have $\gcd(e_w(x), e_{l_j}(x))$, then the degree of $\gcd(e_w(x), e_{l_j}(x)) < \delta - T/2$. In this case, the simulator sets $\overline{\mathcal{S}_w} \leftarrow \{r_1, \dots, r_\delta\}$, where $r_1, \dots, r_\delta \xleftarrow{\$} \mathbb{F}$, and the polynomial $\overline{e_w(x)} \leftarrow \prod_{s \in \overline{\mathcal{S}_w}} (x - s)$. The simulator simulates $\mathcal{F}_{npa}$ returning evaluations of $\langle a_k \cdot \Big(r'_j(x_k) \cdot \overline{e_w(x_k)} + \overline{r_j(x_k)} \cdot e_{l_j}(x_k)\Big) + r'_j(x_k) \cdot b_k \rangle_{k=1}^{\theta}$ to S, where $\overline{r_j(x)}$ is a random polynomial of degree δ. The simulation is indistinguishable from the real protocol execution: $\overline{r_j(x)}$ is sampled from the same distribution as $r_j(x)$, and from Theorem E.1, $\Big\langle r'_j(x_k) \cdot \overline{e_w(x_k)} + \overline{r_j(x_k)} \cdot e_{l_j}(x_k) \Big\rangle_{k=1}^{\theta}$ is indistinguishable from $\Big\langle r'_j(x_k) \cdot e_w(x_k) + r_j(x_k) \cdot e_{l_j}(x_k) \Big\rangle_{k=1}^{\theta}$.
2) *For $j = i$:* When S's output tape has $\gcd(e_w(x), e_{l_j}(x))$, let $\mathcal{S}_{w \cap l_j}$ be the set of the roots of $\gcd(e_w(x), e_{l_j}(x))$ and $\left|\mathcal{S}_{w \cap l_j}\right| = d$. Then, the simulator computes a set $\overline{\mathcal{S}_w} \leftarrow \{r_1, \dots, r_{\delta-d}\} \cup \mathcal{S}_{w \cap l_j}$ where $r_1, \dots, r_{\delta-d} \notin \mathcal{S}_{w \cap l_j}$. The simulator sets $\overline{e_w(x)} \leftarrow \prod_{s \in \overline{\mathcal{S}_w}} (x - s)$. The simulator simulates $\mathcal{F}_{npa}$ returning evaluations of $\langle a_k \cdot \Big(r'_j(x_k) \cdot \overline{e_w(x_k)} + \overline{r_j(x_k)} \cdot e_{l_j}(x_k)\Big) + r'_j(x_k) \cdot b_k \rangle_{k=1}^{\theta}$ to S, where $\overline{r_j(x)}$ is a random polynomial of degree δ. Let $r'_j(x) \cdot e_w(x) + r_j(x) \cdot e_{l_j}(x) = \gcd(e_w(x), e_{l_j}(x)) \cdot R(x)$ and $r'_j(x) \cdot \overline{e_w(x)} + \overline{r_j(x)} \cdot e_{l_j}(x) = \gcd(e_w(x), e_{l_j}(x)) \cdot \overline{R(x)}$ Since $r'_j(x)$, $r_j(x)$ and $\overline{r_j(x)}$ are all random polynomials, due to Lemma B.2, $R(x)$ and $\overline{R(x)}$ are both random polynomials of the same degree, and are therefore indistinguishable. Thus, $r'_j(x) \cdot e_w(x) + r_j(x) \cdot e_{l_j}(x)$ is indistinguishable from $r'_j(x) \cdot \overline{e_w(x)} + \overline{r_j(x)} \cdot e_{l_j}(x)$, and consequently the simulation is indistinguishable from the real execution.

In the case where there is no l_i where S's output tape has $\gcd(e_w(x), e_{l_i}(x))$, the simulator follows case (1) for all $i \in [1, n]$.
In the commit phase, the simulator implements the random oracle, and sets $\langle \overline{h_1}, \dots, \overline{h_\theta} \rangle \xleftarrow{\$} \mathbb{F}^\theta$. The simulator computes $\langle \overline{p_{2,1}}, \dots, \overline{p_{2,\theta}} \rangle \leftarrow \mathsf{P}_{k2_{\mathsf{P}}}(\langle \overline{h_1}, \dots, \overline{h_\theta} \rangle)$, where the simulator gets $k2_{\mathsf{P}}$ from the random tape of S. The simulator simulates the calls to $\mathcal{F}_{ole}$ in Steps 8–9 with inputs $\overline{r_i(x_k)}$ and $\overline{p_{2,k}}$ respectively. The simulator does not need to simulate an output to S.

Then, for Step 10 the simulator returns $\overline{\vec{v}'} \leftarrow \langle \sum_{i=1}^{n} t_{i,k} + t'_{i,k} - \overline{r_i(x_k)} \cdot e_{l_i}(x_k) + \frac{r_c(x_k)}{a'_k} \cdot \overline{p_{2,k}} \rangle_{k=1}^{\theta}$ to S. $\overline{\vec{v}'}$ is indistinguishable from $\vec{v}'$ as $\overline{r_i(x)}$ is identically distributed to $r_i(x)$, and $\langle \overline{h_1}, \dots, \overline{h_\theta} \rangle$ is indistinguishable from $\langle h_1, \dots, h_\theta \rangle$ assuming that $\mathsf{H}(\cdot)$ is a random oracle.

When C is corrupt: Note that C is malicious, and so we will prove the protocol secure in the $(\mathcal{F}_{ole}, \mathcal{F}_{npa})$-hybrid model. The simulator does the following

1) During Step 4 of Π_{TC}, the simulator obtains $\mathcal{A}$'s inputs $\langle r_i(x_k) \rangle_{k=1}^{\theta}$ and p'_1. The simulator simulates $\mathcal{F}_{npa}$ with C 's inputs. There is no output to $\mathcal{A}$ from this step.
2) In Step 8, the simulator gets $r_i(x_k)^*$ from $\mathcal{A}$'s input to $\mathcal{F}_{ole}$ in for $i \in [1, n], k \in [1, \theta]$. If $r_i(x_k)^* = r_i(x_k)$ (obtained earlier), then the simulator sends $r_i(x_k)$ to $\mathcal{F}_{ole}$, obtains $out_{i,k}$ from $\mathcal{F}_{ole}$, and simulates $\mathcal{F}_{ole}$ returning $out_{i,k}$ to C . Otherwise, the simulator sets $\overline{out_{i,k}} \xleftarrow{\$} \mathbb{F}$ and simulates $\mathcal{F}_{ole}$ returning $\overline{out_{i,k}}$ to C . The simulator outputs whatever $\mathcal{A}$ outputs.
3) In Step 9, the simulator gets $p_{2,1}{}', \dots, p_{2,\theta}{}'$ as inputs to $\mathcal{F}_{ole}$. The simulator simulates returning the output

of $\mathcal{F}_{ole}$ to C , and outputs whatever $\mathcal{A}$ outputs.

4) The simulator sends to $\mathcal{F}_{\mathsf{TC}}$ the inputs p_1' and $p_2' \leftarrow \langle p_{2,1}', \ldots, p_{2,\theta}' \rangle$. The functionality returns γ or $\perp$.
5) The simulator gets $\vec{v}'$ from $\mathcal{A}$ in Step 10. If the output of $\mathcal{F}_{\mathsf{TC}}$ is $\perp$, then the simulator sets $\overline{\gamma} = \perp$, returns $\overline{\gamma}$ to the client, and outputs whatever $\mathcal{A}$ outputs. Otherwise, the simulator checks whether $\vec{v}'$ has been correctly computed as the sum of the values returned to C in the previous steps. If so, the simulator sets $\overline{\gamma} \leftarrow \gamma$. Otherwise, the simulator sets $\overline{\gamma} \overset{\$}{\leftarrow} \mathbb{F}$. The simulator sends $\overline{\gamma}$ to C , and outputs whatever $\mathcal{A}$ outputs.

The simulation is indistinguishable from the real execution of the protocol. In Step 2, $out_{i,k}$ and $\overline{out_{i,k}}$ are indistinguishable. This is because in the real execution of Π_{TC}, $out_{i,k} = t_{i,k}' - r_i(x_k) \cdot e_{l_i}(x_k) \overset{\$}{\leftarrow} \mathbb{F}$, when $t_{i,k}' \overset{\$}{\leftarrow} \mathbb{F}$, which is indistinguishable from $\overline{out_{i,k}} \overset{\$}{\leftarrow} \mathbb{F}$. In Step 3, the simulator simply simulates $\mathcal{F}_{ole}$ using $\mathcal{A}$'s input and thus follows the protocol exactly.

In Step 5, first note that the ideal functionality outputs $\perp$ when $\left\| f_{k_f}(w) - l \right\|_{\mathsf{SUR}} \le T$, which equivalent to checking that the degree of $\gcd(e_w(x), e_l(x)) \ge \delta - \frac{T}{2}$. On the other hand, the simulation outputs $\perp$ when the degree of $c(x) \leftarrow \gcd(e_w(x), r_i'(x) \cdot e_w(x) + r_i(x) \cdot e_l(x)) \ge \delta - \frac{T}{2}$. From Prop. B.1, we have

$$\mathbb{P}\left(\left\| f_{k_f}(w) - l \right\|_{\mathsf{SUR}} \neq 2 \cdot \delta - \mathsf{Deg}(c(x))\right) \le 1/|\mathbb{F}|$$

Thus,

$$\begin{aligned} &\mathbb{P}(\overline{\gamma} \neq \perp \mid \gamma = \perp) \\ &\quad = \mathbb{P}\left(\left\| f_{k_f}(w) - l \right\|_{\mathsf{SUR}} \neq 2 \cdot \delta - \mathsf{Deg}(c(x))\right) \le 1/|\mathbb{F}| \end{aligned} \tag{11}$$

Thus, with overwhelming probability, the simulation output and the real execution output are identical.

Next, note that when $\vec{v}'$ has been correctly computed, the simulator simply returns the output from $\mathcal{F}_{\mathsf{TC}}$. The functionality and in turn the simulator return $\overline{\gamma} \leftarrow \mathsf{PRF}_{k_{\mathsf{PRF}}}(\vec{y})$ where $\vec{y} \leftarrow \mathsf{P}_{k1_{\mathsf{P}}}^{-1}\left(p_1'\right) + \mathsf{P}_{k2_{\mathsf{P}}}^{-1}\left(p_2'\right)$. In the real execution, $\vec{y}$ is identical when the value of $\vec{v}'$ has been correctly computed.

Finally, if $\mathcal{A}$ provides inconsistent value of $\vec{v}'$, either due to the fact that $\vec{v}'$ is not correctly computed as a sum of the results returned before to C , or because $r_i(x_k)^* \neq r_i(x_k)$ for some k, i, then the simulator returns $\overline{\gamma} \overset{\$}{\leftarrow} \mathbb{F}$. Since, in this case, $\vec{y}' \leftarrow \left\langle \frac{1}{r_c(x_k)} \left(\sum_{i=1}^{n} \vec{c}_i + \vec{v}' - \vec{v} \right) \right\rangle_{k=1}^{\theta} \neq \mathsf{P}_{k1_{\mathsf{P}}}^{-1}\left(p_1'\right) + \mathsf{P}_{k2_{\mathsf{P}}}^{-1}\left(p_2'\right)$, there is at least one component of $\vec{y}'$ that differs from $\mathsf{P}_{k1_{\mathsf{P}}}^{-1}\left(p_1'\right) + \mathsf{P}_{k2_{\mathsf{P}}}^{-1}\left(p_2'\right)$. The real protocol execution will return $\mathsf{PRF}_{k_{\mathsf{PRF}}}(\vec{y}')$ while the simulation returns a random element in the range of PRF. This implies in the real execution, $\mathcal{A}$ gets exactly one query to the PRF oracle. Thus, $\mathcal{A}$'s advantage in distinguishing between the simulation and the real protocol execution is exactly equal to $\mathcal{A}$'s advantage in distinguishing between a random function and a PRF with one query to PRF oracle. For a secure PRF construction, this is negligible and thus the simulation is indistinguishable to $\mathcal{A}$.

□

Theorem. *Assuming that there is a protocol to securely realize $\mathcal{F}_{ole}$, Π_{IC} securely realizes $\mathcal{F}_{\mathsf{IC}}$ against a malicious server (threat model* ***T3****).*

Proof. We will show that there is a polynomial time simulator that indistinguishably simulates S's and C 's views of the protocol in the ideal world simulation.

When S is corrupt: S does not obtain any output from the protocol. So it is straightforward to simulate S's view when interacting with $\mathcal{F}_{ole}$. Specifically, the simulator simulates $\mathcal{F}_{ole}$ with S's inputs r_k and ss_k for $k \in [1, \theta]$ (which it gets from S's input tape), and $r_k' \overset{\$}{\leftarrow} \mathbb{F}$ as C 's input. The simulation is indistinguishable from the protocol execution if $\mathcal{F}_{ole}$ can be simulated since C 's input in the real protocol execution $\vec{y} \overset{\$}{\leftarrow} \mathbb{F}$ is identically distributed to the simulated input assuming that $\mathsf{H}(\cdot)$ is a random oracle.

When C is corrupt: The simulator implements the random oracle $\mathsf{H}(.)$ in the simulation. When C queries the random oracle with the input $(w, f_{k_f}(w), k_f)$ in Step 2, the simulator responds with $r \in \mathbb{F}^{\theta}$ and stores this locally. The simulator computes $\vec{y}' \leftarrow f_{k_f}(w') + r$. The simulator sends $\vec{y}'$ to $\mathcal{F}_{\mathsf{IC}}$, which either returns γ or $\perp$. If the output from $\mathcal{F}_{\mathsf{IC}}$ is $\perp$, then the simulator sets $\overline{\gamma} \overset{\$}{\leftarrow} \mathbb{F}$. Otherwise, the simulator sets $\overline{\gamma} \leftarrow \gamma$. The simulator secret shares $\overline{\gamma}$ into θ shares $\overline{ss_1}, \ldots, \overline{ss_\theta}$.

in Step 6, the simulator gets inputs to $\mathcal{F}_{ole}$ from $\mathcal{A}$. For $k \in [1, \theta]$, the simulator simulates returning $\overline{ss_k}$ to C . The simulation is indistinguishable from the real protocol execution. Consider first the case when C 's input $\vec{y}' \neq \vec{y}$. In this case, the real protocol execution returns for $k \in [1, \theta]$, $r_k(y_k - y_k') + ss_k$. Since $\vec{y} \neq \vec{y}'$, there is some $j \in [1, \theta]$ where $y_j \neq y_j'$ and thus $r_j(y_j - y_j') + ss_j \overset{\$}{\leftarrow} \mathbb{F}$. Consequently, in the real execution, C does not get all the correct θ secret shares of γ, i.e., at least one of the secret shares is replaced with a random element. Thus, for an information theoretically secure secret sharing scheme, C obtains a random element upon recombining the shares. In the ideal execution, the simulator secret shares a random element and provides the shares to C . Combining the shares, C obtains a random element. The outputs of C are identically distributed in the real world execution and the ideal world simulation.

For the case when $\vec{y} = \vec{y}'$, in the real execution for all $k \in [1, \theta]$, $r_k(y_k - y_k') + ss_k = ss_k$. In the simulation, the simulator secret shares γ into $\overline{ss_1}, \ldots, \overline{ss_\theta}$ and provides these to C . Thus, C 's output is identical in both cases, and the simulation is indistinguishable from the real protocol execution.

□