Algorithm: DH.Process_4
Input: $(t'_{r,s}, r, \overline{\boldsymbol{I}}, \boldsymbol{k}, \xi_{r,s})$
Output: $p'_{r,s}$
Initialize $p'_{r,s}$ as an empty sequence;
for $k = 1 \ldots |Auth(r)|$ **do**
  $\text{shares}_{r,s,w} \leftarrow \eta$ random values;
  if $k \leq |a_{r,s,w}|$ **then**
    $\text{shares}_{r,s,w}[1] \leftarrow \bigoplus_{j \neq 1} \text{shares}_{r,s,w}[j]$;
  $p''_{r,s,w} \leftarrow \text{OT.Forge}(\text{shares}_{r,s,w})$;
  Random $\overline{w}$;
  Append $(\overline{w}, p''_{r,s,w})$ to $p'_{r,s}$;
Randomly reorder $p'_{r,s}$;

depends on $t_{r,s}$. Thanks to the blinding factors $\xi_{r,s}$ and the hybrid argument, the output of $\mathcal{S}_5$ is indistinguishable from the output of $\mathcal{S}_4$.

Algorithm: Reader.Trapdoor_5
Input: $(q_{r,s}, \rho_{priv,r})$
Output: $(\xi_{r,s}, t_{r,s})$
$\xi_{r,s} \xleftarrow{\$} \mathbb{Z}_Q$;
$t_{r,s} \xleftarrow{\$} \mathbb{G}_T$;

Finally it is trivial to build an algorithm that has the same exact output distribution as $\mathcal{S}_5$ while only having $\mathcal{L}(\mathcal{H})$ as input. □

B FULL PROOF OF PRIVACY AGAINST A HONEST-BUT-CURIOUS DH

The proof of privacy against DH is similar to the one against QM yet much simpler. The view of DH $\mathcal{V}'(\mathcal{H})$ consists of the public values, the symmetric key of each reader k_r $\forall r \in R$, the encrypted indexes $\overline{I}_w$ $\forall w \in W$, the transformed trapdoors $t'_{r,s}$ $\forall q_{r,s} \in \boldsymbol{q}$, and the informations from users colluding with DH: $(\rho_{priv,r}$ $\forall r \in R')$, $(\gamma_w$ $\forall w \in W')$, $(I_w$ $\forall w \in W'')$, $(\boldsymbol{q}_r$ $\forall r \in R')$, and $(p_{r,s}$ $\forall r \in R$ $\forall q_{r,s} \in \boldsymbol{q}_r)$.

We could define leakage the same way as against QM, but we can actually prove a leakage even smaller against DH because DH does not learn the result length. The leakage $\mathcal{L}'(\mathcal{H})$ is then defined as $(I_w$ $\forall w \in W'')$, $(\boldsymbol{q}_r$ $\forall r \in R')$, $(|I_w|$ $\forall w \in W)$, $(|\boldsymbol{q}_r|$ $\forall r \in R)$, and $Auth$.

We define three simulators $\mathcal{S}'_0$ to $\mathcal{S}'_2$ which structure is described in algorithm $\mathcal{S}'_i$. Again, $\mathcal{S}'_0$ outputs the real view $\mathcal{V}'(\mathcal{H})$ of DH and thus calls the real algorithms, meaning that $\text{Reader.Trapdoor}'_0 = \text{Reader.Trapdoor}$ etc.

In $\mathcal{S}'_1$, $\text{Reader.Trapdoor}'_1$ outputs random values. The only affected values in the view are the $t'_{r,s}$ values because DH only receive the $p_{r,s}$ values corresponding to corrupted readers. The indistinguishability of the output of $\mathcal{S}'_1$ and the one of $\mathcal{S}'_0$ follows simply from the sender privacy of OT and the hybrid argument.

In $\mathcal{S}'_2$, $\text{Writer.Encrypt}'_2$ outputs random values. Recall that the function $\tilde{h}$ used in Writer.Encrypt is modeled as a random oracle; because each index is encrypted using a different independent

Algorithm: $\mathcal{S}'_i$
Input: $(\boldsymbol{I}, \boldsymbol{q}, Auth, R', W')$
Output: The view of QM
Create all keys, all deltas;
for $w \in W$ **do**
  $\overline{\boldsymbol{I}}[w] \leftarrow \text{Writer.Encrypt}'_i(\boldsymbol{I}[w], \gamma_w)$;
for $q_{r,s}$ *in* $\boldsymbol{q}$ **do**
  $(t_{r,s}, \xi_{r,s}) \leftarrow \text{Reader.Trapdoor}'_i(q_{r,s}, \rho_{priv,r})$;
  $t'_{r,s} \leftarrow \text{QM.Transform}'_i(t_{r,s}, r, \boldsymbol{\Delta})$;
  $p'_{r,s} \leftarrow \text{DH.Process}'_i(t'_{r,s}, r, \overline{\boldsymbol{I}}, \boldsymbol{k}, \xi_{r,s})$;
  $p_{r,s} \leftarrow \text{QM.Filter}'_i(p'_{r,s})$;

key, we can use a different random oracle for each index. The indistinguishability of the output of $\mathcal{S}'_2$ and the one of $\mathcal{S}'_1$ is then straightforward.

Finally it is trivial to build a simulator which on input the leakage $\mathcal{L}'(\mathcal{H})$ has the same output distribution as $\mathcal{S}'_2$. □

C SECURITY OF ZERO-SUM GARBLED BLOOM FILTERS

We show that the proof of Theorem 4 of [12] on the security of GBF applies to ZGBFs as well. We first reproduce Theorem 4 of [12] with our notation:

Theorem 1. *Let C and S be two sets, and GBF-BF intersection be as defined in Section 4.2, then:*

$$\text{GBF}_S \cap \text{BF}_C \overset{c}{\equiv} \text{GBF}_{S \cap C}$$

In our protocol C will always be some singleton $\{c\}$. We show that the proof of this theorem given by Dong et al. in [12] applies to ZGBF as well. In their proof, Dong et al. consider two cases: The first case is when some element of $S - \{c\}$ had none of its shares overwritten during the GBF-BF intersection operation. This corresponds to

$$\exists x \in S - \{c\},\ \text{GBF.Map}(x) = \text{BF.Map}(c)$$

Dong et al. remark that this event correspond to a BF false positive, which has a negligible probability due to how the parameters of GBF are picked; This remains true for ZGBFs. The second case is when each element of $S - \{c\}$ had at least one of its shares overwritten by a random value. Dong et al. show that thanks to the distribution of XOR secret shares, the distribution of $\text{GBF}_S \cap \text{BF}_C$ is the same as the distribution of $\text{GBF}_{S \cap C}$; this ends their proof. Because shares have the same distribution in a ZGBF than in a GBF (XOR random shares of some determined value), this step applies as well to ZGBF. As a result the proof of Dong et al. applies to ZGBF. □