Sparse Graph Obfuscation

September 24, 2023

1 Preliminaries

1.1 Bounded Functional Encryption

We will use the notation of static, bounded functional encryption as presented in [GGLW22].

Security

See page 14 of [GGLW22] for now. I'll put in the actual definition later.

2 A sketch for the boys

2.1 Graph Label Randomization

Say that we have a sparse graph $\mathcal{G} = (V, E)$ such that $|V| = n$ and $\forall v \in V, \deg(v) = d$. (TODO: padding).

Then, we want to create a pseudo-randomized label mapping of the graph, $\phi : \{0,1\}^\lambda \times V \to \{0,1\}^{c \cdot \lambda}$ such that ϕ is deterministic and pseudo-random. In particular, we require that for an adversary that does not know a path from $v, u \in \{v_1, ..., v_p\}$ where $v \neq u$, then for $K_1, K_2 \in \{0,1\}^\lambda$,

$$\begin{aligned} &\mathbf{Pr}\left[\mathcal{A}(\phi(K_1, v_1), \ldots \phi(K_1, v_p), v_1, \ldots, v_p, C_\Gamma) = 1\right] \\ &- \mathbf{Pr}\left[\mathcal{A}(\phi(K_2, v_1), \ldots \phi(K_2, v_p), v_1, \ldots, v_p, C_\Gamma) = 1\right] \leq \texttt{negl}(\lambda) \end{aligned} \tag{1}$$

where C_Γ is the neighbor function for the embedded space: i.e. $C_\Gamma = \phi \circ \Gamma \circ \phi^{-1}$.

Algorithm 1 The circuit for the neighbor function, C_Γ.

```
function $\texttt{INNER}_i(\texttt{Dec}(\phi(v)) = v, K)$
    $u_1, \ldots, u_d = \Gamma(v)$
    $u = u_i$
    $r = \texttt{PRF}(K, u)$
    return $\texttt{Enc}(\text{MPK}, (u, K))$ where we encrypt with randomness from $r$.
function $C_\Gamma(\phi(v))$
    for $i \in [d]$ do
        $u_i = \texttt{Dec}(\text{SK}_{\texttt{inner}_i}, \phi(v))$
    return $(u_1, \ldots, u_d)$
```

Claim 2.1. *eq. (1) holds for any PPT adversary, $\mathcal{B}$ when C_Γ is implemented as in algorithm 1.*

Proof of Claim 2.1. We proceed via a hybrid argument and then show that if there exists an adversary $\mathcal{B}$ that can beak eq. (1), then we can build an adversary $\mathcal{A}$ which can distinguish between the hybrids.

- $\texttt{Hyb}_0$: As the LHS of the FE game
- $\texttt{Hyb}_1$: As the RHS of the FE game
- $\texttt{Hyb}_2$: As the above but we replace $\Pi_m = \big((\texttt{inner}_1, \texttt{inner}_1(v, K')), \ldots, (\texttt{inner}_d, \texttt{inner}_d(v, K')\big)$ for all $v \in \{v_1, ..., v_p\}$

Now, note that if $\mathcal{B}$ can distinguish between eq. (1), then we can build adversary $\mathcal{A}$ to distinguish between $\texttt{Hyb}_0$ and $\texttt{Hyb}_2$ by invoking $\mathcal{B}$ to distinguish □

3 Wrong below

We want to "randomize" the labels of the graph via a poly-time embedding function ϕ such that the embeddings are indistinguishable from a truly random embedding, Φ.

We model Φ as function from V to $\{0,1\}^{c \cdot \lambda}$ for some small constant c such that

$$I_{\texttt{min}}(\phi(V) \mid V = v) \geq 2 \cdot \lambda.$$

Indeed, we do not require that the labels are uniformly random, but rather that each label is "random enough", containing at least 2λ bits of min-entropy.

We can now propose a game to characterize the pseudo-random embedding ϕ. For any PPT adversary, $\mathcal{A}$,

$$\left| \mathbf{Pr}\left[\mathcal{A}(\phi(v_1), \dots \phi(v_i)) = 1\right] - \mathbf{Pr}\left[\mathcal{A}(\Phi(v_1), \dots \Phi(v_i)) = 1\right] \right| \leq \mathtt{negl}(\lambda) \tag{2}$$

for some $i \in \text{poly}(\lambda)$.

The above game may prove to be uninteresting as we can simply describe ϕ to be a PRF which takes in the vertex label and outputs a pseudo-random string of length $c \cdot \lambda$.

This brings us to our notion of graph-label randomization obfuscation (GRO).

Definition 3.1 (Graph-label randomization obfuscation (GRO, pronounced grow))**.** Given a circuit C_Γ realizing $\phi \circ \Gamma \circ \phi^{-1} : \{0, 1\}^{c \cdot \lambda} \to \{0, 1\}^{c \cdot d \cdot \lambda}$ (the neighbor function for the embedded space), $w_1, \dots, w_p, v_1, \dots, v_p \in V$, and any polynomial time adversary, $\mathcal{A}$, such that there does not exist a knowledge extractor from the adversary which, can extract a path from u to v for $u, v \in w_1, \dots, w_p, v_1, \dots v_p$,

$$\left| \mathbf{Pr}\left[\mathcal{A}(\phi(v_1), \dots \phi(v_p), C_\Gamma) = 1\right] - \mathbf{Pr}\left[\mathcal{A}(\phi(w_1), \dots \phi(w_p), C_\Gamma) = 1\right] \right| \leq \mathtt{negl}(\lambda). \tag{3}$$

Pseudo Random Construction

We are now going to proceed to give a pseudo random construction for GRO using simulation secure bounded functional encryption in the CRS model. We will adopt the notation from [AV19].

We will define $\phi(v) = \mathtt{Enc}\left(\text{MPK}, (r, \mathtt{pad}(v) \oplus \mathtt{PRF}(K_1, r), K_1, K_2\right))$ where K_1 is a random key for the PRF with output 2λ bits and the randomness used in $\mathtt{Enc}$ is fixed via a PRG expansion of $r = \mathtt{PRF}(K_2, v)$.

Algorithm 2 The circuit for the neighbor function, C_Γ.

```
function $\mathtt{INNER}_i(\mathtt{Dec}(\phi(v)) = r, \mathtt{pad}(v) \oplus \mathtt{PRF}(K_1, r), K_1, K_2)$
    $r' = \mathtt{PRF}(K_1, r)$
    $v = r' \oplus \mathtt{pad}(v) \oplus \mathtt{PRF}(K_1, r)$
    $u_1, \dots, u_d = \Gamma(v)$
    $u = u_i$
    $r = \mathtt{PRF}(K_2, u)$
    return $\mathtt{Enc}(\text{MPK}, (r, \mathtt{pad}(u) \oplus \mathtt{PRF}(K_1, r), K_1, K_2))$ where we encrypt with randomness from a PRG expansion of $r$.
function $C_\Gamma(\phi(v))$
    for $i \in [d]$ do
        $u_i = \mathtt{Dec}(\text{SK}_{\mathtt{inner}_i}, \phi(v))$
    return $(u_1, \dots, u_d)$
```

3.1 Indistinguishably Proof

Let $\mathtt{Hyb}_0$ be the LHS distribution in the static-bounded-collusion simulation security game in [GGLW22].

Then, we have $\mathtt{Hyb}_1$ be the RHS distribution in the above. By [GGLW22] (TODO: cite more), we have that these two hybrids are indistinguishable.

Then,

- $\mathtt{Hyb}_0$ is indistinguishable from $\mathtt{Hyb}_1$ by [GGLW22].
- $\mathtt{Hyb}_2$: As $\mathtt{Hyb}_1$ except that we fix the message, m, outputted by $\mathcal{A}$ to be $(r, v \oplus \mathtt{PRF}(K_1, r), K_1, K_2)$ where $r = \mathtt{PRF}(K_2, v)$ and $v \in V$.
- $\mathtt{Hyb}_3$: As the above but we replace $\Pi_m = \big((\mathtt{inner}_1, \mathtt{inner}_1(m), \ldots, (\mathtt{inner}_d, \mathtt{inner}_d(m)\big)$ with $\mathtt{inner}_i, \mathtt{inner}'_i(m)$ where $\mathtt{inner}'_i$ is the same as $\mathtt{inner}_i$ except that we replace m with MESSAGE TODO m' which encodes a different vertex such that the adversary does not know a path from m to m'.

Now, we show that if one can distinguish between the security game outlined in eq. (1), then one can distinguish between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_3$.

Lemma 3.2. *Assume that adversary $\mathcal{B}$ can distinguish between the game outline in eq. (1) for any fixed sequence of vertices $v_1, \ldots, v_p$. Then, we build an adversary $\mathcal{A}$ which can distinguish between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_3$ for $Q = d$ and $f_i = \mathtt{inner}_i$.*

Proof. Define $\Phi : V \to c \cdot \lambda$ as $\Phi(v) = \mathtt{Enc}(\mathrm{MPK}, m)$ where $m = (r_{v,1}, v \oplus r_{v,2}, r_3, r_4)$ $\mathtt{Enc}$ is encrypted with randomness $r_{v,5}$ and $r_{v,1}, r_{v,2}, r_{v,5} \leftarrow U$ and are fixed for v and $r_3, r_4 \leftarrow U$ are fixed for a graph $\mathcal{G}$.

Then, we have that $I_{\mathtt{min}}(m \mid V = v) \geq 2 \cdot \lambda$. We can then note that $I_{\mathtt{min}}(\mathtt{Enc}(\mathrm{MPK}, m) \mid V = v) \geq I_{\mathtt{min}}(m \mid V = v)$ as $\mathtt{Enc}(\mathrm{MPK}, \cdot)$ is injective on the message space.

We can then note that $\mathtt{inner}'_i$ is realization of $(\Phi \circ \Gamma \circ \Phi^{-1})_i$ and that Π_m in $\mathtt{Hyb}_3$ is equivalent to $\big((\mathtt{inner}_1, (\Phi \circ \Gamma \circ \Phi^{-1})_1), \ldots, (\mathtt{inner}_d, (\Phi \circ \Gamma \circ \Phi^{-1})_d)\big)$.

Now, let $u_\ell = (\Gamma(v_\ell))_1$ for $\ell \in [p]$. Note that as we are only considering non-directed graphs, $v_\ell \in \Gamma(u_\ell)$. WLOG, assume that $v_\ell = \Gamma(u_\ell)_1$. Now, we can then see that if $\mathcal{B}$ can distinguish between eq. (1), then $\mathcal{B}$ can distinguish between $(\phi(v_1), \ldots, \phi(v_\ell))$ and $(\Phi(v_1), \ldots, \Phi(v_\ell))$. Then, we have that $\mathcal{B}$ can distinguish between $(\mathtt{inner}_1 = \phi \circ \Gamma \circ \phi^{-1}, \phi(v_\ell))$ and $(\mathtt{inner}_1, \Phi(v_\ell))$. We can then simply build $\mathcal{A}$ to distinguish between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_3$ by invoking $\mathcal{B}$ to distinguish $\big(\phi(u_1), C_\Gamma = (\mathtt{inner}_1, \ldots, \mathtt{inner}_d)\big)$ and $\big(\Phi(u_1)\big)$. We thus have that $\mathcal{A}$ can distinguish between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_3$. □

4 Hardness of “Guessing” Labels

In this section we aim to show how we show that the labels in a GRO graph equipped with a neighbor function, C_Γ, are “hard” to guess unless the label is the output of the neighbor function.

Claim 4.1. *For any ppt adversary $\mathcal{A}$, there exists a ppt extractor $\mathtt{Extract}$ such that if $\mathcal{A}$ knows $\phi(v)$ where $v \neq v_\ell$ for $\ell \in [p]$, then $\mathtt{Extract}(\mathcal{A}, \phi(v_1), \ldots, \phi(v_p), v_1, \ldots, v_p, v) = u_1, u_2, \ldots, u_k$ where $u_1 = v_\ell$ and $u_k = v$ and $u_1, u_2, \ldots u_k$ is a path from v_ℓ to v where $k \in \mathrm{poly}(\lambda)$.*

Proof. First, note that if an adversary does not know of any labels in the graph to begin with, then they have negligible success probability of guessing a label as the labels are indistinguishable from a sample drawn from a high entropy distribution. So, we will assume that the adversary knows of some labels in the graph, $\phi(v_1), \ldots \phi(v_p)$.

Now, we can simply assert that if the adversary does not know a path from v_ℓ to v for some $\ell \in [p]$ and $v \in V$, then the adversary cannot guess $\phi(v)$ as each labeling $\phi(v)$ is independent from $\phi(u)$ for all $u \in V$ and $v \neq u$. And, as $\phi(v) \approx \Phi(v)$, the probability of guessing $\phi(v)$ is negligible as Φ is drawn from a high min-entropy distribution.

□

5 Building Witness Encryption

6 Open questions

Can we de-randomize specific labels (like the outcome of a branching program) and create obfuscation via giving over evaluation points on the graph along with the circuit?

Can we build something like NISC or private function evaluation??

Can the output with iO to decrypt some things of interest?

September 24, 2023

Abstract

References

[AV19] Prabhanjan Ananth and Vinod Vaikuntanathan. Optimal bounded-collusion secure functional encryption. In *Theory of Cryptography Conference*, pages 174–198. Springer, 2019. 3

[GGLW22] Rachit Garg, Rishab Goyal, George Lu, and Brent Waters. Dynamic collusion bounded functional encryption from identity-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 736–763. Springer, 2022. 1.1, 1.1, 3.1