Sparse Graph Label Randomization

October 18, 2023

1 Preliminaries

1.1 Bounded Functional Encryption

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

Security

We will slightly weaken the security notion such that the adversary does not choose which circuits it can learn the functional secret key for. Indeed, this is a weaker notion of functional encryption which fixes the adversary's output circuit. We will assume that we get circuit $C_1, \ldots, C_d$.

For completeness, we have the original security definition of [GGLW22] below:

$$\left\{ \mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK},\cdot)}(\mathrm{CT}) \quad \begin{array}{l} (1^n, 1^q) \leftarrow \mathcal{A}^{(1)} \\ (\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}\,(1^n, 1^q) \\ m \leftarrow \mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK})}(\mathrm{MPK}) \\ \mathrm{CT} \leftarrow \mathrm{Enc}(\mathrm{MPK}, m) \end{array} \right\}_{\lambda \in \mathbb{N}}$$
$$\overset{c}{\approx}$$
$$\left\{ \mathcal{A}^{\mathrm{Sim}_3^{U_m(\cdot)}}(\mathrm{CT}) \quad \begin{array}{l} (1^n, 1^q) \leftarrow \mathcal{A}(1^\lambda) \\ (\mathrm{MPK}, \mathbf{st}_0) \leftarrow \mathrm{Sim}_0\left(1^\lambda, 1^n, q\right) \\ m \leftarrow \mathcal{A}^{S_1(\mathbf{st}_0)}(\mathrm{MPK}) \\ (\mathrm{CT}, \mathbf{st}_2) \leftarrow \mathrm{Sim}_2(\mathbf{st}_1, \Pi^m) \end{array} \right\}_{\lambda \in \mathbb{N}}$$

whenever the following admissibility constraints and properties are satisfied:

- $\mathrm{Sim}_1, \mathrm{Sim}_3$ are stateful in that after each invocation, they updated their states $\mathbf{st}_1, \mathbf{st}_3$ respectively which is carried over to the next invocation.
- Π^m contains a list of functions f_i queried by $\mathcal{A}$ in the pre-challenge phase along with their output on the challenge message m. That is, if f_i is the i-th function queried by $\mathcal{A}$ to oracle Sim_1 and $q_{[re}$ be the number of queries $\mathcal{A}$ makes before outputting m, then $\Pi^m = \big((f_1, f_1(m)), \ldots, (f_{q_{pre}}, f_{q_{pre}}(m))\big)$.
- $\mathcal{A}$ makes at most q queries combined tote key generation oracle in both games.
- Sim_3 for eac queried function f_i, in the post challenge phase, makes a single query to its message oracle U_m on the same f_i itself.

Our modified security definition is as follows:

$$\begin{array}{c} \left\{ \mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK},\{C_1,\ldots,C_d\})}(\mathrm{CT}) \quad \begin{array}{l} (1^n, 1^q) \leftarrow \mathcal{A}^{(1)} \\ (\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}\,(1^n, 1^q) \\ m \leftarrow \mathcal{A}(\mathrm{MPK}, \mathrm{SK}_{C_1}, \ldots, \mathrm{SK}_{C_d}) \\ \mathrm{CT} \leftarrow \mathrm{Enc}(\mathrm{MPK}, m) \end{array} \right\}_{\lambda \in \mathbb{N}} \\ \overset{c}{\approx} \\ \left\{ \mathcal{A}^{\mathrm{Sim}_3^{U_m(\{C_1,\ldots,C_d\})}}(\mathrm{CT}) \quad \begin{array}{l} (1^n, 1^q) \leftarrow \mathcal{A}(1^\lambda) \\ (\mathrm{MPK}, \mathbf{st}_0) \leftarrow \mathrm{Sim}_0\left(1^\lambda, 1^n, q\right) \\ m \leftarrow \mathcal{A}^{S_1(\mathbf{st}_0)}(\mathrm{MPK}, C_1, \ldots, C_d) \\ (\mathrm{CT}, \mathbf{st}_2) \leftarrow \mathrm{Sim}_2(\mathbf{st}_1, \Pi^m) \end{array} \right\}_{\lambda \in \mathbb{N}} \end{array} \tag{1}$$

where the admissibility constraints remain the same.

1.2 Non-malleable Bounded FE

Here, we introduce the notion of non-malleable bounded functional encryption.

We define non-malleable security of bounded functional encryption in almost the exact notion of [Pas06] for public key encryption. First, let $NM(m_1, \ldots, m_q, \mathcal{A})$ be a game as follows for $q = \mathrm{poly}(\lambda)$:

1. $(\mathrm{MPK}, \mathrm{MSK}) \leftarrow \mathrm{FE.Setup}(1^\lambda)$
2. $\mathrm{CT}_1, \ldots, \mathrm{CT}_q \leftarrow \mathrm{FE.Enc}(\mathrm{MPK}, m_1), \ldots \mathrm{FE.Enc}(\mathrm{MPK}, m_q)$
3. $\mathrm{CT}'_1, \ldots, \mathrm{CT}'_\ell \leftarrow \mathcal{A}(\mathrm{MPK}, \mathrm{CT}_1, \ldots, \mathrm{CT}_q, 1^{|m|})$
4. $m'_i \leftarrow \perp$ is $\mathrm{CT}_i = \mathrm{CT}'_j$ for any $i \in [q]$, $j \in [\ell]$ and $\mathrm{FE.Dec}(\mathrm{SK}_{\mathrm{identity}}, c_i)$ otherwise.

Then, we say that a bounded functional encryption scheme is non-malleable if for all PPT $\mathcal{A}$ and every PPT $\mathcal{D}$, there exists a negligible function $\mathtt{negl}$ such that for all $\{\, m \,\}_0, \{\, m \,\}_1 \in \{0,1\}^{nq}$, we have

$$\left| \mathbf{Pr}[\mathcal{D}(NM(\{m\}_0, \mathcal{A})) = 1] - \mathbf{Pr}[\mathcal{D}(NM(\{m\}_1, \mathcal{A})) = 1] \right| \leq \mathtt{negl}. \tag{2}$$

As outlined in [Pas06], we can equivalently define non-malleability in terms of a PPT recognizable relation R such that

$$\begin{aligned} &\Bigg| \mathbf{Pr}\left[NM\left(m_1, \ldots m_q, \mathcal{A}(z)\right) \in \bigcup_{m \in \{m\}} R(m) \right] - \\ &\mathbf{Pr}\left[c \leftarrow \mathrm{Sim}_{NM}(1^n, z); m' = \mathrm{FE.Dec}(\mathrm{SK}_{\mathrm{identity}}, c); m' \in \bigcup_{m \in \{m\}} R(m) \right] \Bigg| \leq \mathtt{negl}(\lambda). \end{aligned} \tag{3}$$

Note that in the above definition, we do not give the adversary access to any SK_{C_i}. We simply require that the scheme is public key (many message) non-malleable.

2 Using Weak Extractible Obfuscation

2.1 Graph Randomized Traversal

Say that we have a sparse, potentially exponentially sized, graph $\mathcal{G} = (V, E)$ and $\forall v \in V, \deg(v) = d$. We also require that $\mathcal{G}$ is equipped with a neighbor function, Γ, which can be computed in polynomial time. We define a randomized and keyed labelling function $\phi : \{0,1\}^\lambda \times V \to \{0,1\}^{\mathrm{poly}(\lambda)}$ such that given, $\phi(K, v_0)$ for root v_0, an adversary, $\mathcal{A}$, which does not know a path from v_0 to v,

$$\mathbf{Pr}[\mathcal{A}(\mathcal{O}(C_\Gamma), v_0, v, \phi(K, v_0)) = \phi(K, v)] \leq \mathtt{negl}(\lambda) \tag{4}$$

for function C_Γ where $C_\Gamma(\phi(K, u)) = \phi(K, \Gamma(u)_1), \ldots, \phi(K, \Gamma(u)_d)$ if $\Gamma(u) \neq \emptyset$ and otherwise $\Gamma(u)$ returns a $\perp$ string; and, $\mathcal{O}$ represents an indistinguishable obfuscator.

2.2 Instantiation

We define

$$\phi(K, v) = (F(K, v), v).$$

For shorthand, we will write σ_v to connote an attempted "signature" of v where a correct signature is $F(K, v)$.

We can now define C_Γ:

Algorithm 1 The circuit for the neighbor function, C_Γ.

function $C_\Gamma(f(\sigma_v), v)$
  if $f(\sigma_v) \neq f(F(K, v))$ **then**
    return $\perp$
  if $\Gamma(v) = \emptyset$ **then**
    return $\perp$
  $u_1, \ldots u_d = \Gamma(v)$
  return $f(F(K, u_1)), f(F(K, u_2)), \ldots, f(F(K, u_d))$

Algorithm 2 Circuit for the neighbor function, $C_\Gamma^{w^*, \Gamma(w^*)_1, \ldots, \Gamma(w^*)_d}$ with punctured PRF key $K(\{w^*\})$ and constant $z^*, z_1^*, z_2^*, \ldots, z_d^*$

function $C_\Gamma(f(\sigma_v), v)$
  if $v \neq w$ and $f(\sigma_v) \neq f(F(K, v))$ **then**
    return $\perp$
  if $v = w$ and $f(\sigma_v) \neq z^*$ **then**
    return $\perp$
  if $\Gamma(v) = \emptyset$ **then**
    return $\perp$
  if $v = w$ **then**
    return $z_1^*, z_2^*, \ldots, z_d^*$
  $u_1, \ldots u_d = \Gamma(v)$
  return $f(F(K, u_1)), f(F(K, u_2)), \ldots, f(F(K, u_d))$

Proof of eq. (4). We are going to use a series of inductively built indistinguishable hybrids along with algorithm 2 to show that eq. (4) holds.

- $\texttt{Hyb}_0$: In the first hybrid, the following game is played

  1. $K \xleftarrow{\$} \{0,1\}^{\lambda'}$ and $\phi(K, v_0) = (F(K, v_0), v)$
  2. The challenger generates $\mathcal{O}(C_\Gamma)$ and gives the program to $\mathcal{A}$
  3. The challenger chooses a v and gives the adversary v in plaintext.
  4. $\mathcal{A}$ outputs guess g and wins if $g = \phi(K, v)$

- $\texttt{Hyb}_1$: Let $\mathcal{P}$ be the set of all paths from v_0 to v. For each path $P \in \mathcal{P}$ where P is an ordered list of connected vertices, we have that the adversary does not know some part of P. We can note that this implies that $\mathcal{A}$ does not know $\phi(K, p)$ for all $p \in P$ as then $\mathcal{A}$ can recover P. Let u_P be the first vertex in P such that $\mathcal{A}$ does not know *a path* from v_0 to u_P. Define $\mathrm{Suff}'(P)$ to be the path in P from this u_P to v. Then, necessarily, $\mathcal{A}$ does not know $\phi(K, w_P)$ for at least one $w_P \in \mathrm{Suff}'(P)$ as then $\mathcal{A}$ would know a path from v_0 to v. Now, let $\mathrm{Suff}(P)$ be the path which starts at w_P, ends at v.

  We now inductively build up a series of hybrids to show that a hybrid distribution which shows $\phi(K, s)$ for $s \in \mathrm{Suff}(P)$ indistinguishable from random. We perform the following procedure for each $P \in \mathcal{P}$. So, for $P \in \mathcal{P}$,

  - For the base case, let $U = \mathrm{Suff}(P)_1$ where $\mathrm{Suff}(P)_1$ is the first vertex in P such that $\mathcal{A}$ does not know $\phi(K, p)$ for $p \in P$.
    1. Set $w^* = p$. Then, replace C_Γ with $C_\Gamma^{w^*, \Gamma(w^*)_1, \ldots, \Gamma(w^*)_d}$ as defined in algorithm 2. Fix the constant $z^* = f(F(K, p))$ and $z_1^* = f(F(K, \Gamma(w^*)_1)), \ldots, z_d^* = f(F(K, \Gamma(w^*)_d))$.
    2. Set $z^* = f(t), z_1^* = f(t_1), \ldots z_d^* = f(t_d)$ where $t, t_1, \ldots, t_d$ are chosen at random
  - For the ℓ-th inductive step where $1 \le \ell < |\mathrm{Suff}(P)|$, we are going to assume that we are given a hybrid such that $w^* = \mathrm{Suff}(P)_\ell$ and $z^* = f(t), z_1^* = f(t_1), \ldots z_d^* = f(t_d)$ for random $t, \ldots t_d$ in algorithm 2. Now, we change the hybrid in a similar manner as in the base case:
    1. Set $w^* = \mathrm{Suff}(P)_{\ell+1}$. Then, replace C_Γ with $C_\Gamma^{w^*, \Gamma(w^*)_1, \ldots, \Gamma(w^*)_d}$ as defined in algorithm 2. Fix the constant $z^* = f(F(K, p))$ and $z_1^* = f(F(K, \Gamma(w^*)_1)), \ldots, z_d^* = f(F(K, \Gamma(w^*)_d))$.
    2. Set $z^* = f(t), z_1^* = f(t_1), \ldots z_d^* = f(t_d)$ where $t, t_1, \ldots, t_d$ are chosen at random

    to puncture on $\mathrm{Suff}(P)_{\ell+1}$ where we update $z^*, \ldots z_d^*$ with new randomness.

Finally, we can note that if $\texttt{Hyb}_0 \stackrel{c}{\approx} \texttt{Hyb}_3$,

$$\mathbf{Pr}[\mathcal{A}(C_\Gamma, v_0, v, \phi(K, v_0)) \in \mathrm{Image}(\phi(K, v))] \stackrel{c}{\approx} \mathbf{Pr}[\mathcal{A}(C'_\Gamma, v_0, v, \phi(K, v_0)) \in \mathrm{Image}(\phi(K, v))]$$

where C'_Γ is C_Γ except that C'_Γ uses $\texttt{inner}_i^p$ where $p = \max_{P \in \mathcal{P}} |P|$. We can note that C'_Γ returns $\perp$ for any query on $\phi(K, w^v)$ where $w^v \in \Gamma^{-1}(v)$. Using lemma 2.3 and the fact that $C'_\Gamma(u)_i$ returns $\perp$ for all $u \in V$ and $i \in [d]$ where $v = \Gamma(u)_i$, we have that

$$\mathbf{Pr}[\mathcal{A}(C'_\Gamma, v_0, v, \phi(K, v_0)) \in \mathrm{Image}(\phi(K, v))] \le \texttt{negl}(\lambda).$$

□

Lemma 2.1. *$\texttt{Hyb}_0 \stackrel{c}{\approx} \texttt{Hyb}_{2b}$.*

Proof. First we show that $\texttt{Hyb}_0 \overset{c}{\approx} \texttt{Hyb}_1$. Note that if $\mathcal{A}$ can distinguish between $\texttt{Hyb}_0$ and $\texttt{Hyb}_1$ then an adversary can distinguish between an FE scheme and its simulated counterpart where m is fixed to (K, v_0, r). We can see this as $\texttt{Hyb}_1$ is direct simulation of the FE scheme.

Then, if $\mathcal{A}$ can distinguish $\texttt{Hyb}_1$ and $\texttt{Hyb}_{2a}$, then we can break the security of the $\texttt{PRG}$ used in line **??** of algorithm 1. We can create an adversary $\mathcal{B}$ which, for some fixed K, distinguishes between FE.Enc(MPK, (K, u, r_2)) with random coins r_1 where $r_1, r_2 = \texttt{PRG}(r)$ and FE.Enc(MPK, (K, u, r_1^*)) encrypted with random coins r_2^* where r_1^*, r_2^* are truly random.

Then, if $\mathcal{A}$ can distinguish any transformation from $\texttt{Hyb}_{2a}$ to $\texttt{Hyb}_{2b}$, then we can break the security of the FE scheme. We can see this by noting that if we fix $m = (K, w, r)$ for random r and K, then $\mathcal{A}^{\mathrm{Sim}_3^{Um(\cdot)}}(\mathrm{CT})$ is distinguishable and $\mathcal{A}^{\mathrm{Sim}_3^{um(\cdot)}}(\mathrm{CT}')$ where CT is the real cipher-text and CT' is simulated. We can then note that if the above are distinguishable, then $\mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK}, \{\texttt{inner}_1, \ldots \texttt{inner}_d\})}(\mathrm{CT})$ and $\mathcal{A}^{\mathrm{Sim}_3^{um(\cdot)}}(\mathrm{CT}')$ are distinguishable as $\mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK}, \{\texttt{inner}_1, \ldots \texttt{inner}_d\})}$ can simply simulate $\mathcal{A}^{\mathrm{Sim}_3^{Um(\cdot)}}(\mathrm{CT})$.

Then, if $\mathcal{A}$ can distinguish any transformation from $\texttt{Hyb}_{2b}$ to $\texttt{Hyb}_{2a}$, then we can break the security of a PRG in the same manner as distinguishing $\texttt{Hyb}_1$ and $\texttt{Hyb}_{2a}$.

By the chain rule, we get that $\texttt{Hyb}_0$ and $\texttt{Hyb}_{2b}$ are indistinguishable even after a repeated number of sequential invocations of the transformation in $\texttt{Hyb}_{2a}$ and $\texttt{Hyb}_{2b}$. $\square$

Lemma 2.2. *Let $\mathcal{A}$ be a PPT adversary and assume that we have a non-malleable and simulation secure FE scheme. Then, we have that the inductive step of $\texttt{Hyb}_3$ holds.*

Proof. We construct an adversary $\mathcal{B}$ that can break NM security using $\mathcal{A}$ if $\mathcal{A}$ can distinguish between the hybrids in the inductive step. Note that in order to distinguish between the hybrids, $\mathcal{A}$ must have queried $\texttt{inner}_i^\ell$ or $\texttt{inner}_{i+1}^\ell$ on $\phi(K, w^u)$ where $u \in \{\,\mathrm{Suff}(P)_{\ell+1} \mid P \in \mathcal{P}\,\}$ as this is the only difference between the hybrids. Thus, we see that $\mathcal{A}$ is able to produce $\mathrm{CT} \in \phi(K, w^u)$. By definition of $\texttt{inner}_i^\ell$ though, we know that $\texttt{inner}_i^\ell(\phi(k, q)) \neq \phi(K, w^u)$ for any $q \in V$ as we define $\texttt{inner}_i^\ell(K, q) = \perp$ if $\texttt{inner}_i'(K, q) = \phi(K, w^u)$. Thus, the adversary has to be able to produce $\mathrm{CT} \in \phi(K, w^u)$ without calling C_Γ^ℓ where C_Γ^ℓ uses $\texttt{inner}_i^\ell$ instead of $\texttt{inner}_i$.

Thus, if $\mathcal{A}(w^u, v_0, C_\Gamma, \phi(K, v_0))$ can produce $\mathrm{CT} \in \phi(K, w^u)$, we can have $\mathcal{B}(\phi(K, v_0), \phi(K, q_1), \ldots, \phi(K, q_{\mathrm{poly}(\lambda)}))$ produce $\phi(K, w^u)$ where $q_1, \ldots, q_{\mathrm{poly}(\lambda)}$ are all the vertices that $\mathcal{A}$ has queried C_Γ on. $\mathcal{B}$ simply has to invoke Sim_3 to create a simulated function key for $\mathrm{SK}'_{\texttt{inner}_i}$ and thus a simulated C'_Γ. $\mathcal{B}$ then gives $\mathcal{A}$ $(w^u, v_0, C'_\Gamma, \phi(K, v_0))$. $\mathcal{B}$ then breaks eq. (3) (the relational notion of non-malleability) as $\mathcal{A}$ is able to create an encryption of $\phi(K, w^u)$ with non-negligible probability while the simulator in eq. (3) cannot. $\square$

Lemma 2.3. *Define C'_Γ where C'_Γ is defined as in algorithm 1 except that for some set $U \subset V$, $C_\Gamma(w^u)_i = \perp$ for all $w^u \in V$ such that $u = \Gamma(w^u)_i$ for some $u \in U$. In words, the parent of all $u \in U$ do not return $\phi(K, u)$ when queried on C'_Γ. Then, assuming the non-malleability and simulation security of FE, we have that for all PPT $\mathcal{A}$ and all $u \in U$,*

$$\mathbf{Pr}[\mathcal{A}(C'_\Gamma, v_0, u, U, \phi(K, v_0)) \in \mathit{Image}(\phi(K, u))] \leq \mathtt{negl}(\lambda). \tag{5}$$

Proof. Almost identically to lemma 2.2, we construct an adversary $\mathcal{B}$ that can break NM security using $\mathcal{A}$ if $\mathcal{A}$ can produce $\mathrm{CT} \in \phi(K, u)$ for some $u \in U$.

If $\mathcal{A}(w^u, v_0, C'_\Gamma, u, \phi(K, v_0))$ can produce $\mathrm{CT} \in \phi(K, u)$, we can have $\mathcal{B}(\phi(K, v_0), \phi(K, q_1), \ldots, \phi(K, q_{\mathrm{poly}(\lambda)}))$ produce $\phi(K, u)$ where $q_1, \ldots, q_{\mathrm{poly}(\lambda)}$ are all the vertices that $\mathcal{A}$ has queried C'_Γ on. $\mathcal{B}$ simply has to invoke Sim_3 to create a simulated set of function keys for $\texttt{inner}'_i$ for all $i \in [d]$ and can then simulate C'_Γ with these function keys.

We can then have $\mathcal{B}$ invoke Sim_3 to create a simulated function key for $\mathrm{SK}'_{\mathtt{inner}_i}$ and thus a simulated C^*_Γ. $\mathcal{B}$ then gives $\mathcal{A}$ $(w^u, v_0, C^*_\Gamma, \phi(K, v_0))$. If we define the relation R to break in eq. (3) to be $R(K, v_0, r) = \{ (K, v, r*) : \forall r^* \leftarrow \{0, 1\}^\lambda \}$, we can then break eq. (3) (the relational notion of security for non-malleability). We can see this as $\mathcal{A}$ is able to create an encryption of $\phi(K, w^u)$ given encryptions of $\phi(K, q_1), \ldots, \phi(K, q_{\mathrm{poly}(\lambda)})$ with non-negligible probability while the simulator in eq. (3) cannot. □

October 18, 2023

Abstract

References

[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

[Pas06] Rafael Pass. Lecture 16: Non-malleability and public key encryption, October 2006. 1.2, 1.2