Sparse Graph Obfuscation

October 5, 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:

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

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 $q = \text{poly}(\lambda)$:

1. $(\text{MPK}, \text{MSK}) \leftarrow \text{FE.Setup}(1^\lambda)$
2. $\text{CT}_1, \ldots, \text{CT}_q \leftarrow \text{FE.Enc}(\text{MPK}, m_1), \ldots \text{FE.Enc}(\text{MPK}, m_q)$
3. $\text{CT}'_1, \ldots, \text{CT}'_\ell \leftarrow \mathcal{A}(\text{MPK}, \text{CT}_1, \ldots, \text{CT}_q, 1^{|m|})$
4. $m'_i \leftarrow \perp$ is $\text{CT}_i = \text{CT}'_j$ for any $i \in [q]$, $j \in [\ell]$ and $\text{FE.Dec}(\text{SK}_{\text{identity}}, c_i)$ 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 $\texttt{negl}$ such that for all $\{\, m\,\}_0\,, \{\, m\,\}_1 \in \{0,1\}^{nq}$, we have

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

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

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

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 Randomized DAG Traversal Sketch

2.1 DAG 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,

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

for function C_Γ where $C_\Gamma(\phi(K, u)) \in \phi(K, \Gamma(u)_1), \ldots, \phi(K, \Gamma(u)_d)$ if $\Gamma(u) \neq \emptyset$ and otherwise $\Gamma(u)$ returns a $\perp$ string padded to length $d|\phi(K, \cdot)|$.

2.2 Instantiation

We define $\phi(K, v)$ to be as follows:

1. Let $r_1, r_2 \xleftarrow{\$} \{0,1\}^\lambda$ or r_1, r_2 is drawn from a pseudorandom distribution.
2. Return FE.Enc(MPK, (K, v, r_2)) where encryption is done with randomness from r_1.

We can now define, C_Γ.

Algorithm 1 The circuit for the neighbor function, C_Γ.

function $\mathtt{INNER}_i(K, v, r)$
  if $\Gamma(v) = \emptyset$ **then**
    return $\perp$
  $u_1, \ldots, u_d = \Gamma(v)$
  $u = u_i$
  $r_1, r_2 = \mathtt{PRG}(r)$
  return FE.Enc(MPK, (u, K, r_2)) where we encrypt with randomness from r_1.
function $C_\Gamma(\phi(K, v))$
  for $i \in [d]$ **do**
    $u_i = \mathtt{Dec}(\mathrm{SK}_{\mathtt{inner}_i}, \phi(K, v))$
  return $(u_1, \ldots, u_d)$

Proof of eq. (4). We are going to use a series of indistinguishable hybrids and then use non-malleability of FE along with the last hybrid to show that eq. (4) holds.

- $\mathtt{Hyb}_0$: In the first hybrid, the following game is played
  1. $K \xleftarrow{\$} \{0,1\}^{\lambda'}$ and MPK, SK $\leftarrow$ FE.Setup$(1^{\lambda'})$.
  2. The challenger generates $\mathrm{SK}_{\mathtt{inner}_i} \leftarrow$ FE.Keygen(MSK, $\mathtt{inner}_i$) for $i \in [d]$ and gives these keys to $\mathcal{A}$
  3. The challenger chooses a v and gives the adversary v in plaintext.
  4. The challenger picks random $r_1, r_2 \xleftarrow{\$} \{0,1\}^{\lambda'}$ and generates $\phi(K, v_0) =$ FE.Enc(MPK, (K, v_0, r_2)) using r_1 as the random coins and gives $\phi(K, v_0)$ to $\mathcal{A}$.

5. $\mathcal{A}$ outputs guess g and wins if $g \in \phi(K, v)$

- $\texttt{Hyb}_1$: We replace MPK with its simulated counterpart using Sim_0, $\phi(K, v_0)$ with a simulated cipher-text using the simulator Sim_2, and $\text{SK}_{\texttt{inner}_i}$ with its simulated counterpart using Sim_3 as defined in eq. (1).

- $\texttt{Hyb}_{2a}$: For any input into Sim_2 via $\Pi^{K,w,r}$ where $w \in V$ and r is random, we replace the output of $\texttt{inner}_i$ with $\texttt{inner}'_i$ which uses true randomness r_1^*, r_2^* instead of r_1, r_2. For any call to $C_\Gamma(\phi(K, w))$ by $\mathcal{A}$ for $w \in V$, we replace the output of $\texttt{inner}_i$ with $\texttt{inner}'_i$ which uses true randomness r_1^*, r_2^* instead of r_1, r_2. This is equivalent to changing Π^m to $\Pi^{m\prime}$ in eq. (1) where $\Pi^{m\prime}$ is the list $\left(\texttt{inner}_1, \texttt{inner}'_1(\cdot), \ldots, \texttt{inner}_d, \texttt{inner}'_d(\cdot)\right)$. Note that this gives us that $\texttt{inner}'_i(K, w, r) = \phi(K, u) = \text{FE.Enc}(\text{MPK}, (K, u, r_2^*))$ where $u = \Gamma(w)_i$.

- $\texttt{Hyb}_{2b}$: For any call by $\mathcal{A}$ to $\texttt{inner}'_i(K, w, r) = \text{CT}$, we replace CT with CT' where CT' is the output of Sim_2 with input $\Pi^{(K,u,r)'}$ where $u = \Gamma(w)_i$.

  Note that the replacement of $\texttt{Hyb}_{2a}$ and $\texttt{Hyb}_{2b}$ are repeated multiple times. Specifically, these replacements are repeated at most α times where α is the number of unique times $\mathcal{A}$ runs $\text{FE.Dec}(\text{SK}_{\texttt{inner}_i}, \phi(K, w))$.

- $\texttt{Hyb}_3$: 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 $\mathcal{A}$ never queries $\texttt{inner}_i(w^u)$ where $u = \Gamma(w^u)_i$ for some $u \in P$ and the adversary knows a path from v_0 to w. We can see this because if there is no $u \in P$ such that $\mathcal{A}$ never queries $\texttt{inner}_i(w^u)$, then the adversary knows and queries a path from v_0 to v. Define $\text{Suff}(P)$ to be the path which starts at u, ends at v, and is a suffix of P. We now inductively build up a series of hybrids to show that a hybrid distribution which "erases" $\phi(K, v)$ from $\texttt{inner}_i$ is indistinguishable from the above hybrid.

  - For the base case, let $U = \bigcup_{P \in \mathcal{P}} \{\, \text{Suff}(P)_1 \,\}$ where $\text{Suff}(P)_1$ is the first vertex in P such that $\mathcal{A}$ never queries $\texttt{inner}_i(w^u)$ as defined above. Then, replace $\texttt{inner}'_i(\cdot)$ with $\texttt{inner}^1_i(\cdot)$ in Π_m such that $\texttt{inner}^1_i(w) = \texttt{inner}'_i(w)$ if $w \neq w^u$ for $u \in U$ and otherwise $\texttt{inner}^1_i(w^u) = \perp$. We can note that this hybrid is indistinguishable as $\texttt{inner}'_i$ only changes for input ciphertexts which the adversary never queries.
  - For the ℓ-th inductive step where $1 \leq \ell < \max_{P \in \mathcal{P}}(|\text{Suff}(P)|)$, we are going to assume that we are given a hybrid such that $\texttt{inner}^\ell_i$ where $\texttt{inner}^\ell_i(w^u) = \perp$ for $u \in U^\ell$ where $U^\ell = \bigcup_{P \in \mathcal{P}} \text{Suff}(P)_1, \ldots, \text{Suff}(P)_\ell$ and otherwise $\texttt{inner}^\ell_i(\cdot) = \texttt{inner}'_i(\cdot)$. We now show that if $\mathcal{A}$ can distinguish between a hybrid with $\texttt{inner}^\ell_i(\cdot)$ and $\texttt{inner}^{\ell+1}_i(\cdot)$, then the adversary can break the non-malleability of the FE scheme. We defer this proof to lemma 2.2.

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

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

where C'_Γ is C_Γ except that C'_Γ uses $\texttt{inner}^p_i$ 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

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

$\square$

Lemma 2.1. *$\mathtt{Hyb}_0 \overset{c}{\approx} \mathtt{Hyb}_{2b}$.*

Proof. First we show that $\mathtt{Hyb}_0 \overset{c}{\approx} \mathtt{Hyb}_1$. Note that if $\mathcal{A}$ can distinguish between $\mathtt{Hyb}_0$ and $\mathtt{Hyb}_1$ 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 $\mathtt{Hyb}_1$ is direct simulation of the FE scheme.

Then, if $\mathcal{A}$ can distinguish $\mathtt{Hyb}_1$ and $\mathtt{Hyb}_{2a}$, then we can break the security of the $\mathtt{PRG}$ used in line 6 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 $r_1, r_2 = \mathtt{PRG}(r)$ 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 $\mathtt{Hyb}_{2a}$ to $\mathtt{Hyb}_{2b}$, 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 $\mathcal{A}^{\mathrm{Sim}_3^{U_m(\cdot)}}(\mathrm{CT})$ is distinguishable and $\mathcal{A}^{\mathrm{Sim}_3^{u_m(\cdot)}}(\mathrm{CT}')$ where CT is the real cipher-text and CT' is simulated. We can then note that if the above are distinguishable, then $\mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK}, \{\mathtt{inner}_1, \ldots \mathtt{inner}_d\})}(\mathrm{CT})$ and $\mathcal{A}^{\mathrm{Sim}_3^{u_m(\cdot)}}(\mathrm{CT}')$ are distinguishable as $\mathcal{A}^{\mathrm{KeyGen}(\mathrm{MSK}, \{\mathtt{inner}_1, \ldots \mathtt{inner}_d\})}$ can simply simulate $\mathcal{A}^{\mathrm{Sim}_3^{U_m(\cdot)}}(\mathrm{CT})$.

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

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

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 $\mathtt{Hyb}_3$ 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 $\mathtt{inner}_i^\ell$ or $\mathtt{inner}_{i+1}^\ell$ 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 $\mathtt{inner}_i^\ell$ though, we know that $\mathtt{inner}_i^\ell(\phi(k, q)) \neq \phi(K, w^u)$ for any $q \in V$ as we define $\mathtt{inner}_i^\ell(K, q) = \perp$ if $\mathtt{inner}_i'(K, q) = \phi(K, w^u)$. 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 $\mathtt{inner}_i^\ell$ instead of $\mathtt{inner}_i$.

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 $\mathrm{SK}'_{\mathtt{inner}_i}$ and thus a simulated C'_Γ. $\mathcal{B}$ then gives $\mathcal{A}$ $(w^u, v_0, C'_\Gamma, \phi(K, v_0))$. $\mathcal{B}$ 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. ☐

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 $\mathtt{inner}'_i$ 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. $\square$

October 5, 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