Lossy Trapdoor Functions
by Chris Peikert, Brent Waters

Giacomo Fenzi

ETH Zurich

22 April 2021

Motivation

Lossy Trapdoor Functions and Their Applications

Chris Peikert[*] — SRI International

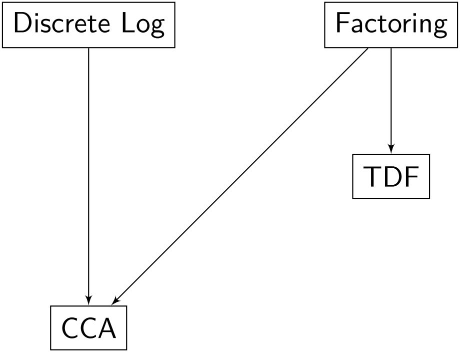
Brent Waters[†] — SRI International

- Trapdoor Functions are a basic primitive, but hard to instantiate
- IND-CCA Security for PKE from factoring and discrete log but not lattices

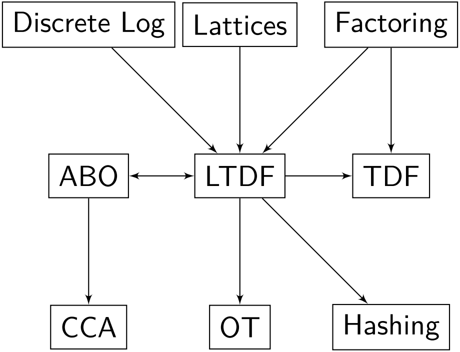
Paper Results

- Introduce Lossy Trapdoor Functions (LTDFs)
- Realize LTDFs from factoring, discrete log *and* lattices
- Show LTDFs imply TDFs
- Black box construction of CCA-secure (witness recovering) cryptosystems, collision-resistant hash functions and oblivious transfer protocols.

Connections



Connections



Trapdoor Functions

Informally, a trapdoor function is family of functions that are hard to invert without access to some additional information called a trapdoor

Definition

A trapdoor function consists of three PPT algorithms (S, F, F^{-1}) such that:

- *Easy to sample and invert with trapdoor.* $S(1^\lambda) \to (s, t)$ such that $F(s, -)$ is an injective function on $\{0,1\}^n$ and $F^{-1}(t, -)$ is its inverse
- *Hard to invert without.* For *any* PPT inverter $\mathcal{A}$ we have that $\mathcal{A}(1^\lambda, s, F(s, x))$ outputs x with negligible probability.

Example of Trapdoor

RSA Encryption! In trapdoor form:

- $S(1^\lambda)$ generates N, e, d as in RSA, set $s := (N, e)$ and $t := (d)$ and returns (s, t)
- $F(s, x)$ computes $x^e \mod N$
- $F^{-1}(t, c)$ computes $c^d \mod N$

Composite Residuosity

- $S(1^\lambda)$ generates $N = pq$ as a product of large primes, select g suitably, $s := (N, g)$, $t := (p, q)$
- $F(s, x)$ splits $x = m_1 + Nm_2$ and returns $g^{m_1} m_2^N \mod N^2$
- $F^{-1}(t, c)$ decrypts using the factorization to compute Carmichael function

Lossy Trapdoors

Informally, you either get an injective trapdoor or a 'lossy' function, and *cannot tell which is which*

Definition

An (n, k)-lossy trapdoor function consists of three PPT algorithms (S, F, F^{-1}). We denote $S_{inj}(-) \triangleq S(-, 0)$ and $S_{lossy}(-) \triangleq S(-, 1)$.

- *Outputs of S_{inj} are easy to compute and easy to invert with trapdoor.* $S_{inj}(1^\lambda) \to (s, t)$ s.t. that $F(s, -)$, $F^{-1}(t, -)$ are functionally as in the trapdoor case
- *Outputs of S_{lossy} are easy to compute.* $S_{lossy}(1^\lambda) \to (s, \perp)$ s.t. $F(s, -)$ is a function on $\{0, 1\}^n$ with image size at most 2^{n-k}.
- The description output of $S_{inj}(1^\lambda)$ and $S_{lossy}(1^\lambda)$ are computationally indistinguishable.

Subtleties

- The definition really relates to a collection of lossy trapdoor functions.
- $k \triangleq k(\lambda) = \text{poly}(\lambda) \leq n$ is a parameter that represents how 'lossy' the collection is.
- We also write $r \triangleq n - k = \text{poly}(\lambda)$ as the *residual leakage*.
- No hardness requirement on inverting outputs of S_{inj}
- Requirements are too strict in lattices, leads to *almost-always* lossy functions.

All-But-One TDFs

Intuition: You have a family of functions, most of them are trapdoors, one is not. It is very hard to tell them apart.

Definition

An (n, k)-ABO TDF is a triple of PPT algorithms S, F, F^{-1} such that:

- $S(1^\lambda, b^*) \to (s, t)$ as before
- For any $b \neq b^*$, $F(s, b, -)$, $F^{-1}(t, b, -)$ are as in the previous definition.
- $F(s, b^*, -)$ is a lossy function as before
- For any b, b' the first outputs of $S(1^\lambda, b)$, $S(1^\lambda, b')$ are computationally indistinguishable.

LTDF $\implies$ TDF

- Completeness: Use the injective functions generated by S_{inj}.
- Soundness: We cannot (information theoretically) invert the lossy branch, so if we could invert the injective trapdoors we could distinguish outputs of S_{inj}, S_{lossy}, contradicting LDTF.
- Formally, let $\mathcal{A}$ be an inverter. We build $\mathcal{D}$

$$\underline{\mathcal{D}^{\mathcal{A}}(s)}$$

$$x \leftarrow\$ \{0,1\}^n$$

$$y = F(s, x)$$

$$x' = \mathcal{A}(s, y)$$

$$\textbf{return } x = x'$$

If $s \leftarrow S_{inj}(1^\lambda)$ then it succeeds nonneglibly, while otherwise it will fail

LTDF $\implies$ CCA

Requirements

We will require some primitives[1]. We note that our cryptosystem will have message space $\{0,1\}^\ell$.

- We have $\Sigma = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vfy})$ a strongly unforgeable one-time signature scheme. We require that the public keys are in $\{0,1\}^v$.
- $F = (S_{ltdf}, F_{ltdf}, F^{-1}_{ltdf})$ is a (n,k)-lossy trapdoor function.
- $G = (S_{abo}, F_{abo}, F^{-1}_{abo})$ is a (n,k')-ABO trapdoor function with branch space $\{0,1\}^v$.
- $\mathcal{H}$ is a collection of pairwise independent hash functions $\{0,1\}^n \to \{0,1\}^\ell$.

[1] All of these reduce to LTDFs

LTDF $\implies$ CCA

Encryption Scheme

$$\begin{array}{l}
\underline{\mathcal{G}(1^\lambda)} \\
(s,t) \leftarrow S_{inj}(1^\lambda) \\
(s',t') \leftarrow S_{abo}(1^\lambda, 0^v) \\
h \leftarrow\$\, \mathcal{H} \\
pk := (s, s', h) \\
sk := (t, t', pk) \\
\textbf{return } (pk, sk)
\end{array}$$

$$\begin{array}{l}
\underline{\mathcal{E}(pk, m)} \\
(vk, sk_\sigma) = \mathrm{Gen}(1^\lambda) \\
x \leftarrow\$\, \{0,1\}^n \\
c_1 = F_{ltdf}(s, x) \\
c_2 = G_{abo}(s', vk, x) \\
c_3 = m \oplus h(x) \\
\omega \leftarrow \mathrm{Sign}(sk_\sigma, (c_i)_{i=1}^3) \\
\textbf{return } (vk, c_1, c_2, c_3, \sigma)
\end{array}$$

$$\begin{array}{l}
\underline{\mathcal{D}(sk, c)} \\
\textbf{if } \neg\mathrm{Vfy}(vk, (c_i)_{i=1}^3, \sigma) \\
\quad \textbf{return } \perp \\
\textbf{fi} \\
x = F^{-1}(t, c_1) \\
\textbf{if } c_1 \neq F_{ltdf}(s, x) \vee \\
\quad c_2 \neq G_{abo}(s', vk, x) \\
\quad \textbf{return } \perp \\
\textbf{fi} \\
\textbf{return } c_3 \oplus h(x)
\end{array}$$

Setup is to be called once at the beginning of the game, and the attacker is allowed a single query to EncO and oracle access to DecO. The attacker wins if it outputs $b' = b$.

$\mathrm{Setup}(\lambda)$	$\mathrm{EncO}(m_0, m_1)$	$\mathrm{DecO}(c^*)$
$b \leftarrow\!\$\ \{0,1\}$	$c \leftarrow \mathcal{E}(pk, m_b)$	**if** $c^* \in \mathcal{T}_{enc}$
$\mathcal{T}_{enc} = \emptyset$	$\mathcal{T}_{enc} \coloneqq \mathcal{T}_{enc} \cup \{c\}$	**return** $\perp$
$pk, sk \leftarrow \mathcal{G}(\lambda)$	**return** c	**fi**
return pk		**return** $\mathcal{D}(sk, c^*)$

LTDF $\implies$ CCA

Game Hops

We proceed by a sequence of games. We note that, since a single query is made to EncO we move the signature scheme generation in Setup and denote that verification key as vk^*.

- $\mathsf{G}_1(\lambda)$: This is the original CCA Security Game
- $\mathsf{G}_2(\lambda)$: In DecO if $vk = vk^*$ return $\perp$
- $\mathsf{G}_3(\lambda)$: In Setup choose the lossy branch of G to be vk^*
- $\mathsf{G}_4(\lambda)$: In DecO find x using G's trapdoor rather than F's
- $\mathsf{G}_5(\lambda)$: In Setup replace S_{inj} with S_{lossy}

The hops are as follows:

$$G_1 \approx_{\Sigma} G_2 \approx_{abo} G_3 \equiv G_4 \approx_{ltdf} G_5$$

Finally, an argument as in the TDF case shows that in G_5 even an unbounded attacker has only negligible success probability.

Realizations

We can realize LTDF from *any* encryption scheme that is:

- Additively Homomorphic. This allows to encrypt matrices such as $\mathbf{I}_n$ or $\mathbf{0}_n$ indistinguishably and to evaluate matrix vector products with an encrypted matrix.
- Secure to Reuse Randomness, so that we can use the same randomness with different keys securely.
- Isolated Randomness, so that it is only dependent on the input randomness and not on keys/messages.

We shows next a realization from the DDH assumption, but a similar technique can also be employed with lattices (based on LWE) with some difficulties.

DDH $\implies$ LTDF

Consider the following variant of the ElGamal cryptosystem, with public key $h = g^z$ and secret key z. The encryption function is $E_h(m; r) = (g^r, h^r g^m)$ for randomness r. To decrypt (c_1, c_2) we output $\log_g(c_2/c_1^z)$ which is easy to compute if $m \in \{0, 1\}$. This scheme is semantically secure and it is additively[2] homomorphic i.e.

$$E_h(m; r) \odot E_h(m'; r') = E_h(m + m', r + r')$$

$$E_h(m; r)^x = E_h(mx; rx)$$

[2] All operations done component wise

DDH $\implies$ LTDF

We show how to use the previous scheme to encrypt a matrix $\mathbf{M} = (m_{i,j}) \in \mathbb{Z}_p^{n \times n}$. Select n pk/sk pairs $h_i = g^{z_i}$ and n pieces of randomness r_i. Then the encryption is the matrix $\mathbf{C} = (c_{i,j}) = (E_{h_j}(m_{i,j}; r_i))$ with the z_js as decryption keys. We can represent $\mathbf{C}$ as the following two matrices:

$$\mathbf{C_1} = \begin{bmatrix} g^{r_1} \\ \vdots \\ g^{r_n} \end{bmatrix}, \; \mathbf{C_2} = \begin{bmatrix} h_1^{r_1} g^{m_{1,1}} & \dots & h_n^{r_1} g^{m_{1,n}} \\ \vdots & \ddots & \vdots \\ h_1^{r_n} g^{m_{n,1}} & \dots & h_n^{r_n} g^{m_{n,n}} \end{bmatrix}$$

Via n^2 hybrid games we can show that this encryption produces indistinguishable ciphertext under DDH. We denote this operation as $\mathrm{ME}(\mathbf{z}, \mathbf{M})$ for $\mathbf{z}$ the vector of private keys.

DDH $\implies$ LTDF

We build a LTDF from the previous scheme as it follows. Note that in the injective case we encrypt the identity matrix, while in the lossy case the all zero matrix.

$S_{inj}(1^\lambda)$	$S_{lossy}(1^\lambda)$	$F_{ltdf}(\mathbf{C}, \mathbf{x})$
$\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$	$\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$	**return** $\mathbf{y} = \mathbf{x} \cdot \mathbf{C}$
$\mathbf{z} \leftarrow\$ \mathbb{Z}_p^n$	$\mathbf{z} \leftarrow\$ \mathbb{Z}_p^n$	
$\mathbf{C} \leftarrow \mathrm{ME}(\mathbf{z}, \mathbf{I}_n)$	$\mathbf{C} \leftarrow \mathrm{ME}(\mathbf{z}, \mathbf{0}_n)$	$F_{ltdf}^{-1}(\mathbf{z}, \mathbf{y})$
return $(\mathbf{C}, \mathbf{z})$	**return** $(\mathbf{C}, \perp)$	$x_i = D_{z_i}(y_i)$
		return $\mathbf{x}$

- $\mathcal{G}$ is the group generation algorithms, it returns (G, p, g) where G is a cyclic group of prime order p with generator g. We assume DDH hardness w.r.t. $\mathcal{G}$.
- $\mathbf{xC}$ is computed by the homomorphic property. In fact, if $\mathbf{C} = \mathrm{ME}(\mathbf{z}, \mathbf{M})$ with randomness $\mathbf{r}$ and $h_j = g^{z_j}$

$$y_j = \bigodot_{i=1}^{n} c_{i,j}^{x_i} = E_{h_j}((\mathbf{xM})_j; R \triangleq \langle \mathbf{r}, \mathbf{x} \rangle)$$

- Note that if $\mathbf{M} = \mathbf{I}_n$ then $y_j = E_{h_j}(x_j; R)$
- If instead $\mathbf{M} = \mathbf{0}_n$ then $y_j = E_{h_j}(0; R)$

DDH $\implies$ LTDF

Final Checks

Now, we just have to check that the LTDF conditions are satisfied. In particular, the above construction is $(n, n - \lg p)$-lossy.

- The three algorithms are clearly PPT
- A quick thought shows that the injective conditions are met
- Indistinguishability follows from the indistinguishability of ME.
- Finally, for outputs generated by S_{lossy} we have that $y_i = E_{h_i}(0; R)$ for some $R \in \mathbb{Z}_p$ that depends on x. R can take at most p values, the residual leakage is at most $\lg p$ and so the loss is $k = n - r \geq n - \lg p$

Things which I did not have time to show

- ABO $\equiv$ LTDF (see extra)
- More efficient ABO construction from DDH
- LDTFs from LWE
- CPA from LTDFs
- SUF one time signatures from LDTFs
- UOWHFs, CRHFs from LDTFs
- OT from LDFTs

Related Work

- All-But-N LTDFs
- All-But-Many LTDFs
- Identity Based LTDFs

Thank You!

Notation and Entropy

- λ is the security parameter, and we will abbreviate $n(\lambda) = \text{poly}(\lambda)$ as simply n
- $f(-)$ denotes the function taking $x \mapsto f(x)$
- Write $H_\infty(X)$ for the min-entropy of X. This corresponds to the optimal probability of guessing X.
- We let $\widetilde{H}_\infty(X|Y)$ be the average min-entropy of X conditioned on Y. This corresponds to the optimal probability of guessing X knowing Y.
- We use the following lemma, if Y takes at most 2^r values then:

$$\widetilde{H}_\infty(X|Y) \geq H_\infty(X) - r$$

LTDF $\implies$ TDF

Note that if s is generated by S_{inj} then with some non negligible probability we have that $\mathcal{A}$ succeeds and $\mathcal{D}$ succeeds whenever $\mathcal{A}$ does.
Instead, if s is generated by S_{lossy} even an unbounded adversary would have best possible probability given by $2^{-\widetilde{H}_\infty(x|s,F(s,x))}$. But note that $F(s,-)$ takes at most 2^r values and so by the previous lemma $\widetilde{H}_\infty(x|s,F(s,x)) \geq H_\infty(x|s) - r = n-(n-k) = k$. So the probability is bounded by 2^{-k} and as such is negligible.
From the above it follows that $\mathcal{D}$ will win the distinguishing game with non negligible probability.

ABO $\equiv$ LTDF

- ABOs and LTDFs are equivalent.
- ABO $\implies$ LTDF. Take ABO on $\{0,1\}$ and evaluate always on one of the branches, but switch lossy branch on generation.
- LTDF $\implies$ ABO. Generate an ABO on $\{0,1\}$ by having $s = (s_0, s_1)$ where one of the two is lossy, and evaluation by using s_b
- Finally, we can extend ABOs on $\{0,1\}$ to ABOs on $\{0,1\}^\ell$ at the cost of having residual leakage ℓr. The idea is, for lossy branch $b^* \in \{0,1\}^\ell$, generate ℓ ABOs each with the i-th having lossy branch b_i^*.