

# 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

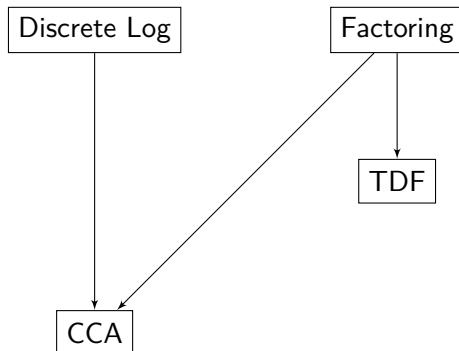
Brent Waters†  
SRI International

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

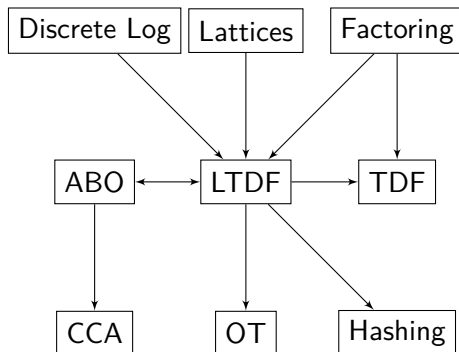
# Paper Results

- ▶ 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.*  $S(1^\lambda) \rightarrow (s, t)$  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:

$S(1^\lambda)$	$F(s, x)$
Select primes $p, q, N := pq$	<b>return</b> $x^e \bmod N$
Select $e$ s.t. $\gcd(e, \phi(n)) = 1$	$F^{-1}(t, y)$
$d := e^{-1} \bmod \phi(n)$	<b>return</b> $y^c \bmod N$
$s := (N, e)$	
$t := (s, d)$	
<b>return</b> $(s, t)$	

Correctness follows since  $x^{ed} = x^1 = x$  and hardness to invert is almost exactly the RSA assumption.

Similar scheme from Pailler cryptosystem.

# 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.*  $S_{inj}(1^\lambda) \rightarrow (s, t)$  s.t. that  $F(s, -)$ ,  $F^{-1}(t, -)$  are functionally as in the trapdoor case
- ▶ *Outputs of  $S_{lossy}$  are easy to compute.*  $S_{lossy}(1^\lambda) \rightarrow (s, \perp)$  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:

- ▶  $S(1^\lambda, b^*) \rightarrow (s, t)$  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 LTDF.
- ▶ Formally, let  $\mathcal{A}$  be an inverter. We build  $\mathcal{D}$

$$\begin{array}{l} \mathcal{D}^{\mathcal{A}}(s) \\ \hline x \leftarrow \$ \{0, 1\}^n \\ y = F(s, x) \\ x' = \mathcal{A}(s, y) \\ \mathbf{return} \ x = x' \end{array}$$

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

# 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 show 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, for  $m \in \{0, 1\}$ ,  $r \in \mathbb{Z}_p$  as randomness.

$S(1^\lambda)$	$E_h(m; r)$
$\mathbb{G} \leftarrow \mathcal{G}(1^\lambda)$	<b>return</b> $(g^r, h^r g^m)$
$z \leftarrow \$\mathbb{Z}_p$	$D_h((c_1, c_2); r)$
$h := g^z$	<b>return</b> $\log_g \left( \frac{c_2}{c_1^z} \right)$
$pk := h$	
$sk := z$	
<b>return</b> $(pk, sk)$	

This scheme is semantically secure and it is additively<sup>1</sup> homomorphic i.e.

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

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

---

<sup>1</sup>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  $z_j$ s as decryption keys. We can represent  $\mathbf{C}$  as the following two matrices:

$$\mathbf{C}_1 = \begin{bmatrix} g^{r_1} \\ \vdots \\ g^{r_n} \end{bmatrix}, \quad \mathbf{C}_2 = \begin{bmatrix} h_1^{r_1} g^{m_{1,1}} & \dots & h_n^{r_1} g^{m_{1,n}} \\ \vdots & \ddots & \vdots \\ h_1^{r_n} g^{m_{n,1}} & \dots & h_n^{r_n} g^{m_{n,n}} \end{bmatrix}$$

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

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

# DDH $\implies$ LTDF

## Subleties

- ▶  $\mathcal{G}$  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.  $\mathcal{G}$ .
- ▶  $\mathbf{x}\mathbf{C}$  is computed by the homomorphic property. In fact, if  $\mathbf{C} = \text{ME}(\mathbf{z}, \mathbf{M})$  with randomness  $\mathbf{r}$  and  $h_j = g^{z_j}$

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

for  $R$  randomness that depends only on  $\mathbf{r}$  and  $\mathbf{x}$ .

- ▶ 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  $S_{\text{lossy}}$  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  $k = n - r \geq n - \lg p$

# LTDF $\implies$ CCA

## Requirements

We will require some primitives<sup>2</sup>. We note that our cryptosystem will have message space  $\{0, 1\}^\ell$ .

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

---

<sup>2</sup>All of these reduce to LTDFs

# LTDF $\implies$ CCA

## Encryption Scheme

$\mathcal{G}(1^\lambda)$	$\mathcal{E}(pk, m)$	$\mathcal{D}(sk, c)$
$(s, t) \leftarrow S_{inj}(1^\lambda)$	$(vk, sk_\sigma) = \text{Gen}(1^\lambda)$	<b>if</b> $\neg \text{Vfy}(vk, (c_i)_{i=1}^3, \sigma)$
$(s', t') \leftarrow S_{abo}(1^\lambda, 0^v)$	$x \leftarrow \$\{0, 1\}^n$	<b>return</b> $\perp$
$h \leftarrow \$\mathcal{H}$	$c_1 = F_{ltdf}(s, x)$	<b>fi</b>
$pk := (s, s', h)$	$c_2 = G_{abo}(s', vk, x)$	$x = F^{-1}(t, c_1)$
$sk := (t, t', pk)$	$c_3 = m \oplus h(x)$	<b>if</b> $c_1 \neq F_{ltdf}(s, x) \vee$
<b>return</b> $(pk, sk)$	$\omega \leftarrow \text{Sign}(sk_\sigma, (c_i)_{i=1}^3)$	$c_2 \neq G_{abo}(s', vk, x)$
	<b>return</b> $(vk, c_1, c_2, c_3, \sigma)$	<b>return</b> $\perp$
		<b>fi</b>
		<b>return</b> $c_3 \oplus h(x)$

# LTDF $\implies$ CCA

## CCA Game

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  $b' = b$ .

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

# 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  $vk^*$ .

$G_1(\lambda)$ : This is the original CCA Security Game

$G_2(\lambda)$ : In DecO if  $vk = vk^*$  return  $\perp$

$G_3(\lambda)$ : In Setup choose the lossy branch of  $G$  to be  $vk^*$

$G_4(\lambda)$ : In DecO find  $x$  using  $G$ 's trapdoor rather than  $F$ 's

$G_5(\lambda)$ : 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.

# Things which I did not have time to show

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

## Related Work

- ▶ More Constructions of LTDFs
- ▶ All-But-N LTDFs
- ▶ All-But-Many LTDFs
- ▶ Identity Based LTDFs

*Thank You!*



# Notation and Entropy

- ▶  $\lambda$  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  $\tilde{H}_\infty(X|Y)$  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:

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

# LTDF $\Rightarrow$ 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  $2^{-\tilde{H}_\infty(x|s, F(s, x))}$ . But note that  $F(s, -)$  takes at most  $2^r$  values and so by the previous lemma  $\tilde{H}_\infty(x|s, F(s, x)) \geq H_\infty(x|s) - r = n - (n - k) = k$ . 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  $\ell r$ . The idea is, for lossy branch  $b^* \in \{0, 1\}^\ell$ , generate  $\ell$  ABOs each with the  $i$ -th having lossy branch  $b_i^*$ .

# Pailler Cryptosystem

## 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  $g^{m_1}m_2^N \bmod N^2$
- ▶  $F^{-1}(t, c)$  decrypts using the factorization to compute Carmichael function