# Lossy Trapdoor Functions

Giacomo Fenzi

ETH Zurich

22 April 2021

## Motivation

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

## Results

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

## Connections

$$\begin{tikzcd}
\boxed{\text{Discrete Log}} \arrow[dd] & \boxed{\text{Factoring}} \arrow[d] \arrow[ddl] \\
 & \boxed{\text{TDF}} \\
\boxed{\text{CCA}} &
\end{tikzcd}$$

## Connections

$$\begin{tikzcd}
\boxed{\text{Discrete Log}} \arrow[dr] & \boxed{\text{Lattices}} \arrow[d] & \boxed{\text{Factoring}} \arrow[dl] \arrow[d] \\
\boxed{\text{ABO}} \arrow[r, leftrightarrow] \arrow[d] & \boxed{\text{LTDF}} \arrow[r] \arrow[d] \arrow[dr] & \boxed{\text{TDF}} \\
\boxed{\text{CCA}} & \boxed{\text{OT}} & \boxed{\text{HF}}
\end{tikzcd}$$

## Notation and Entropy

- $\lambda$ is the security parameter, and we will abbreviate $n(\lambda) = \mathsf{poly}(\lambda)$ 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:

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

## 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**

A $(n, k)$-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 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 first outputs of $S_{inj}(1^\lambda)$ and $S_{lossy}(1^\lambda)$ are computationally indistinguishable.

## Subleties

- 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: Most branches are trapdoors, except one which is lossy. You cannot tell which one it is.

### 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.

## 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 $b_i^*$.