One-Way Functions

CS 276: Introduction to Cryptography

Sanjam Garg

January 22, 2026

Overview

Why One-Way Functions?

- Cryptographers base results on **computational assumptions**
- Security is only as good as the assumptions

  Cryptographers seldom sleep well.
- Goal: Base cryptography on **minimal** necessary assumptions
- Use **abstract primitives** rather than specific number-theoretic problems
  - Existence can be based on multiple computational problems
  - More flexible and future-proof

Motivating Example

Password Hashing

Consider password hashing - we want a function that's:

- **Easy to compute**: Hash the password quickly
- **Hard to invert**: Recover the password from the hash

This is exactly what one-way functions formalize!

- $f(\text{password}) = \text{hash}$
- Easy: Computing hash from password
- Hard: Finding password from hash

One-Way Functions: The Weakest Primitive

Key Insight

One-way functions are the **weakest** abstract primitive cryptographers consider.

- **Virtually every** cryptographic goal implies one-way functions
- Most cryptographic tasks would be **impossible** without OWFs
- Realizing tasks from **just** one-way functions would be ideal
- Existence of OWFs would imply $P \neq NP$

Connection to Complexity Theory

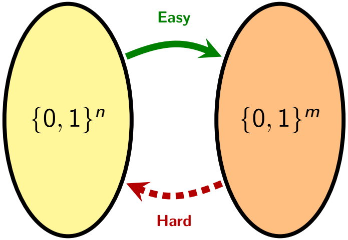
This connection highlights the fundamental nature of one-way functions - they represent the boundary between what is efficiently computable and what is not.

Intuitive Notion

One-Way Function

A function that is:

- **Easy to compute**: Given x, can compute $f(x)$ efficiently
- **Hard to invert**: Given $f(x)$, hard to find any x' with $f(x') = f(x)$



Formal Definition

One-Way Function

A function $f : \{0,1\}^* \to \{0,1\}^*$ is **one-way** if:

1. **Easy to Compute:** $\exists$ polynomial-time machine M such that $\forall x$, $M(x) = f(x)$
2. **Hard to Invert:** $\forall$ non-uniform PPT adversary $\mathcal{A}$:

$$\mu_{\mathcal{A},f}(n) = \Pr_{x \leftarrow \{0,1\}^n}[\mathcal{A}(1^n, f(x)) \in f^{-1}(f(x))]$$

   is **negligible**

- $f^{-1}(f(x)) = \{x' \mid f(x') = f(x)\}$ (not necessarily unique)
- Adversary gets 1^n to know input length (important!)
- Function is **not necessarily one-to-one**

Non-One-to-One Functions

Important Point

The function is not necessarily one-to-one. It is possible that $f(x) = f(x')$ for $x \neq x'$ – and the adversary is allowed to output any such x'.

Example 1

If $f(x) = x \bmod 2$, then:

- $f(0) = f(2) = 0$
- $f(1) = f(3) = 1$
- Adversary succeeds if it outputs **either** 0 or 2 (for input 0)

Why 1^n in the Definition?

Problem

What if we drop 1^n from the adversary's input?

Example 2

Consider $f(x) = |x|$ (length of x).

- Given $y = |x|$, adversary gets $m = \log_2 n$ bits
- Adversary runs in time $\text{poly}(m)$
- But $n = 2^m$ is exponential in m!
- Adversary can't even write down the answer
- Flawed definition would call this "one-way"

Solution

Providing 1^n allows adversary to run in time polynomial in both m and n.

Why Not Perfect Security?

Question

What if we require probability $= 0$ instead of negligible?

Answer: Too Strong!

- Requiring perfect security (probability 0) is too strong
- An adversary that outputs an arbitrarily fixed value x_0 succeeds with probability at least $1/2^n$
- Even a trivial adversary that always outputs a fixed value succeeds with non-zero probability
- This condition is false for every function f

Candidate One-Way Functions

- **Not known** whether one-way functions exist
- Existence would imply $P \neq NP$
- But we have **candidates** based on:
  - Factoring: $f(p, q) = p \cdot q$
  - Discrete Logarithm: $f(x) = g^x$ in group $\mathbb{G}$

From Discrete-Log

If discrete-log assumption holds for group ensemble $\mathcal{G}$, then:

$$f_n(x) = g^x \text{ (where } g \text{ is generator of } \mathbb{G}_n\text{)}$$

is a one-way function family.

Robustness: Can We Modify OWFs?

Question

Given one-way function f, can we fix specific values?

$$g(x) = \begin{cases} y_0 & \text{if } x = x_0 \\ f(x) & \text{otherwise} \end{cases}$$

Answer: Yes!

- Adversary learns how to invert y_0 (with probability $1/2^n$)
- This is negligible, so g is still one-way
- Can fix **exponential** number of values

Formal Argument

More formally, if an adversary could break g with non-negligible probability, we could use it to break f by handling the negligible case where $x = x_0$ separately.

The Apparent Paradox

Paradox

We could keep fixing values to 0, eventually getting a function that outputs 0 for all inputs. How could this still be one-way?

Resolution

- One-wayness only required in the **limit** as $n \to \infty$
- No matter how many values we fix, we're only fixing a **finite** number
- For larger n, we need larger n_0 in the proof
- This illustrates why cryptographic definitions are asymptotic

Key Insight

Asymptotic definitions allow for 'bad' behavior on finitely many inputs as long as security holds in the limit.

Brittleness: Composition Doesn't Always Work

Question

If f is one-way, is $f^2(x) = f(f(x))$ also one-way?

Intuitive (Wrong) Reduction

One might try to invert f by:

1. First inverting f^2 to get some x' with $f^2(x') = f^2(x)$
2. Then inverting f on $f(x')$

Why This Fails

$f^2(x') = f^2(x)$ doesn't guarantee $f(x') = f(x)$! We might have $f(x') \neq f(x)$ but $f(f(x')) = f(f(x))$.

Counterexample: Composition

Construction

Let $g : \{0,1\}^n \to \{0,1\}^n$ be one-way. Define:

$$f(x_1, x_2) = 0^n \| g(x_1)$$

Two Observations

1. $f^2(x_1, x_2) = 0^{2n}$ (constant function, easily invertible!)
2. f is one-way (reduces to g)

Proof: f is One-Way

Reduction

If adversary $\mathcal{A}$ breaks f with non-negligible probability, we construct $\mathcal{B}$ that breaks g:

- $\mathcal{B}$ on input y outputs lower n bits of $\mathcal{A}(1^{2n}, 0^n \| y)$
- If $\mathcal{A}$ inverts $f(x_1, x_2) = 0^n \| g(x_1)$, then $\mathcal{B}$ inverts $g(x_1)$
- Success probability: $\mu_{\mathcal{B},g}(n) = \mu_{\mathcal{A},f}(2n)$

Key Insight

- f is one-way (reduces to g)
- f^2 is constant (trivially invertible)
- **Composition is not guaranteed to preserve one-wayness**

Another Brittleness Example: Dropping Bits

Claim

Given one-way function g, let $g'(x)$ be $g(x)$ with first bit dropped. Then g' is **not necessarily** one-way.

Proof Strategy

We must:

1. Construct a contrived one-way function g from h
2. Show g is one-way (reduction to h)
3. Show g' is not one-way (adversary can invert with probability 1)

Step 1: Construct g from h

Construction

Assume there exists a one-way function $h : \{0,1\}^n \to \{0,1\}^n$. Define $g : \{0,1\}^{2n} \to \{0,1\}^{2n}$ as:

$$g(x\|y) = \begin{cases} 0^n\|y & \text{if } x = 0^n \\ 1\|0^{n-1}\|h(y) & \text{otherwise} \end{cases}$$

where $|x| = |y| = n$.

Intuition

- If $x = 0^n$: output is $0^n\|y$ (easy case)
- If $x \neq 0^n$: output starts with 1 followed by $h(y)$
- The first bit distinguishes the two cases

Step 2: Prove g is One-Way

Goal

Show: If h is one-way, then g is one-way.

Proof by Contradiction

Assume adversary $\mathcal{A}$ breaks g with non-negligible probability $\mu(n)$:

$$\Pr_{x,y}[\mathcal{A}(1^{2n}, g(x\|y)) \in g^{-1}(g(x\|y))] = \mu(n)$$

Construct $\mathcal{B}$ to Break h

$\mathcal{B}$ on input $(1^n, h(y))$ for random y:

1. Sample $x \leftarrow \{0,1\}^n$ uniformly
2. If $x = 0^n$: output random $y' \leftarrow \{0,1\}^n$
3. Otherwise: run $\mathcal{A}(1^{2n}, 1\|0^{n-1}\|h(y))$ to get $x'\|y'$, output y'

Step 2: Analysis of $\mathcal{B}$

Running Time

- Steps 1-2: polynomial time
- Step 3: runs $\mathcal{A}$ which is polynomial time
- Total: polynomial time

Success Probability

$$
\begin{aligned}
&\Pr[\mathcal{B}(1^n, h(y)) \in h^{-1}(h(y))] \\
&\geq \Pr[x = 0^n] \cdot \frac{1}{2^n} + \Pr[x \neq 0^n] \cdot \mu(n) \\
&= \frac{1}{2^{2n}} + \left(1 - \frac{1}{2^n}\right) \mu(n) \\
&\geq \mu(n) - \left(\frac{1}{2^n} - \frac{1}{2^{2n}}\right)
\end{aligned}
$$

Step 3: Prove g' is Not One-Way

Definition of g'

Drop the first bit of g:

$$g'(x\|y) = \begin{cases} 0^{n-1}\|y & \text{if } x = 0^n \\ 0^{n-1}\|h(y) & \text{otherwise} \end{cases}$$

Key Observation

Notice that $g'(0^n\|y) = 0^{n-1}\|y$ for all y!

Step 3: Adversary $\mathcal{C}$ Breaks g'

Construction of $\mathcal{C}$

$\mathcal{C}$ on input $(1^{2n}, g'(x\|y))$:

1. Parse $g'(x\|y)$ as $0^{n-1}\|\overline{y}$
2. Output $0^n\|\overline{y}$

Why This Works

- If $x = 0^n$: $g'(0^n\|y) = 0^{n-1}\|y$, so $\mathcal{C}$ outputs $0^n\|y$ (correct!)
- If $x \neq 0^n$: $g'(x\|y) = 0^{n-1}\|h(y)$, so $\mathcal{C}$ outputs $0^n\|h(y)$
  - But $g'(0^n\|h(y)) = 0^{n-1}\|h(y)$
  - So $0^n\|h(y) \in (g')^{-1}(g'(x\|y))$ (also correct!)

Success Probability

$$\Pr[\mathcal{C}(1^{2n}, g'(x\|y)) \in (g')^{-1}(g'(x\|y))] = 1$$

Therefore, g' is **not** one-way!

Key Lesson

Cryptographic Primitives are Delicate

Small modifications can break security!

- Composition doesn't always work
- Dropping bits can break one-wayness
- Need to be very careful with transformations