ARIEL UNIVERSITY

אוניברסיטת אריאל בשומרון

Foundations of Modern Cryptography

Edward A. Hirsch

https://edwardahirsch.github.io/edwardahirsch

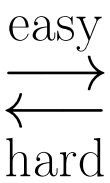
Neapolis University Pafos

Lecture 2: October 9, 2024

Lecture Plan

- One-Way Functions
  - Warm-up: toy example, worse-case one-way functions
  - One-way functions and families
  - A universal one-way function
- Trapdoor functions (families)

Worst-case "one-way functions"

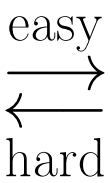


«Naïve approach»

no randomness

- «easy»: $\in$ **fP** (computable in deterministic polynomial time),
- «hard»: $\notin$ **fP** (or **FNP**) (not computable in ...).

Worst-case "one-way functions"



«Naïve approach»

no randomness

- «easy»: $\in$ **fP** (computable in deterministic polynomial time),
- «hard»: $\notin$ **fP** (or **FNP**) (not computable in ...).

Theorem

Let $\mathbf{P} \neq \mathbf{NP}$. Then there is $f \in \mathbf{fP}$ such that $f^{-1} \notin \mathbf{fP}$.

Worst-case "one-way functions": Proof

Theorem

Let $\mathbf{P} \neq \mathbf{NP}$. Then there is $f \in \boldsymbol{f}\mathbf{P}$ such that $f^{-1} \notin \boldsymbol{f}\mathbf{P}$.

Proof.

Let Φ be a Boolean formula,
let $A \in \{0,1\}^n$ be a Boolean assignment to its n variables.
Define

$$f(\Phi, A) = \begin{cases} (\Phi, 1^n), & \text{if } \Phi[A] = \texttt{True}, \\ (\Phi, 0^n), & \text{otherwise.} \end{cases}$$

$1^n = \overbrace{11\ldots1}^{n}$

Worst-case "one-way functions": Proof

Theorem

Let $\mathbf{P} \neq \mathbf{NP}$. Then there is $f \in \boldsymbol{f}\mathbf{P}$ such that $f^{-1} \notin \boldsymbol{f}\mathbf{P}$.

Proof.

Let Φ be a Boolean formula,
let $A \in \{0,1\}^n$ be a Boolean assignment to its n variables.
Define

$$f(\Phi, A) = \begin{cases} (\Phi, 1^n), & \text{if } \Phi[A] = \texttt{True}, \\ (\Phi, 0^n), & \text{otherwise.} \end{cases}$$

$1^n = \overbrace{11\ldots1}^{n}$

If we could invert f, we could solve SAT by computing

$$f^{-1}(\Phi, 1^n).$$

□

Worst-case "one-way functions": Proof

Theorem

Let $\mathbf{P} \neq \mathbf{NP}$. Then there is $f \in \boldsymbol{f}\mathbf{P}$ such that $f^{-1} \notin \boldsymbol{f}\mathbf{P}$.

Proof.

Let Φ be a Boolean formula,
let $A \in \{0,1\}^n$ be a Boolean assignment to its n variables.
Define

$$f(\Phi, A) = \begin{cases} (\Phi, 1^n), & \text{if } \Phi[A] = \texttt{True}, \\ (\Phi, 0^n), & \text{otherwise.} \end{cases}$$

$1^n = \overbrace{11\ldots1}^{n}$

If we could invert f, we could solve SAT by computing

$$f^{-1}(\Phi, 1^n).$$

□

Exercise

Prove the converse to our "worst-case theorem": If $\exists f\ (f \in \boldsymbol{f}\mathbf{P} \wedge f^{-1} \notin \mathbf{FNP})$, then $\mathbf{P} \neq \mathbf{NP}$.

Are there many one-way functions?

- Are there hard functions?

Are there many one-way functions?

- Are there hard functions?

  Yes, a lot of them: there are 2^{2^n} functions $\{0,1\}^n \to \{0,1\}$ and $2^{O\left(n \cdot 2^{\sqrt{n}}\right)}$ circuits of size $\leqslant 2^{\sqrt{n}}$.

- Are there one-way functions?

Are there many one-way functions?

- Are there hard functions?

  Yes, a lot of them: there are 2^{2^n} functions $\{0,1\}^n \to \{0,1\}$ and $2^{O\left(n \cdot 2^{\sqrt{n}}\right)}$ circuits of size $\leqslant 2^{\sqrt{n}}$.

- Are there one-way functions?

  Hmm... at least, not many: a randomly generated function has a great chance to be hard in both directions, so not one-way.

Exercise

Almost all length-preserving functions have "similar" circuit complexity in both directions.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

A "sliced" one-way function:

- Consider families $f_i\colon \{0,1\}^i \to \{0,1\}^i$.
- ... or for some other polynomial length, $f_i\colon \{0,1\}^{n(i)} \to \{0,1\}^{m(i)}$.
- Especially reasonable when our adversary has a devoted algorithm for each i.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

A "sliced" one-way function:

- Consider families $f_i\colon \{0,1\}^i \to \{0,1\}^i$.
- ... or for some other polynomial length, $f_i\colon \{0,1\}^{n(i)} \to \{0,1\}^{m(i)}$.
- Especially reasonable when our adversary has a devoted algorithm for each i.

The most useful definition: $\{f_n\}_{n\in\mathbb{N}}$ s.t.

- $f_n\colon \{0,1\}^n \to \{0,1\}^n$ is computed in time $\mathrm{poly}(n)$.
- For every polynomial-time randomized adversary A,

$$\mathbf{Pr}\{A(f_n(x)) \text{ inverts } f_n\} \text{ is negligible}$$

- Here $\mathbf{Pr}$ is over $x \leftarrow U_n$ and A's randomness.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

A "sliced" one-way function:

- Consider families $f_i\colon \{0,1\}^i \to \{0,1\}^i$.
- ... or for some other polynomial length, $f_i\colon \{0,1\}^{n(i)} \to \{0,1\}^{m(i)}$.
- Especially reasonable when our adversary has a devoted algorithm for each i.

The most useful definition: $\{f_n\}_{n\in\mathbb{N}}$ s.t.

A poly-time algorithm (uniform) or a family of poly-size circuits (non-uniform)

- $f_n\colon \{0,1\}^n \to \{0,1\}^n$ is computed in time $\mathrm{poly}(n)$.
- For every polynomial-time randomized adversary A,

$$\mathbf{Pr}\{A(f_n(x)) \text{ inverts } f_n\} \text{ is less than any } 1/\mathrm{poly}(n)$$

- Here $\mathbf{Pr}$ is over $x \leftarrow U_n$ and A's randomness.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

A "sliced" one-way function:

- Consider families $f_i\colon \{0,1\}^i \to \{0,1\}^i$.
- ... or for some other polynomial length, $f_i\colon \{0,1\}^{n(i)} \to \{0,1\}^{m(i)}$.
- Especially reasonable when our adversary has a devoted algorithm for each i.

The most useful definition: $\{f_i\}_{i\in\mathbb{N}}$ s.t.

A poly-time algorithm (uniform) or a family of poly-size circuits (non-uniform)

- $f_n\colon \{0,1\}^n \to \{0,1\}^n$ is computed in time $\mathrm{poly}(n)$.
- For every polynomial-time randomized adversary A, for every k,

$$\mathbf{Pr}\{A(f_n(x)) \text{ inverts } f_n\} < \frac{1}{n^k}$$

for n big enough.

- Here **Pr** is over $x \leftarrow U_n$ and A's randomness.

Families of one-way functions

A single one-way function for all input lengths:

- One could consider $f\colon \{0,1\}^* \to \{0,1\}^*$.
- Technically involved, not convenient, equivalent to simpler approach.

A "sliced" one-way function:

- Consider families $f_i\colon \{0,1\}^i \to \{0,1\}^i$.
- ... or for some other polynomial length, $f_i\colon \{0,1\}^{n(i)} \to \{0,1\}^{m(i)}$.
- Especially reasonable when our adversary has a devoted algorithm for each i.

The most useful definition: $\{f_n\}_{n\in\mathbb{N}}$ s.t.

- $f_n\colon \{0,1\}^n \to \{0,1\}^n$ is computed in time $\mathrm{poly}(n)$.
- For every polynomial-time randomized adversary A,

$$\mathbf{Pr}\{A(f_n(x)) \text{ inverts } f_n\} \text{ is negligible}$$

- Here $\mathbf{Pr}$ is over $x \leftarrow U_n$ and A's randomness.
- For a non-uniform adversary, randomness does not matter.

Notation: negligible

Usually applied to probabilities

From now on:

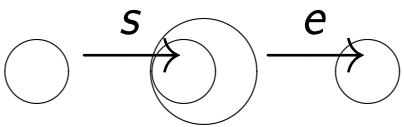
A sequence $t_1, t_2, \ldots$ is negligible
if for every k and every n large enough,

$$t_n < \frac{1}{n^k}.$$

It is non-negligible (inverse polynomial)
if there is k such that for every n large enough,

$$t_n \geqslant \frac{1}{n^k}.$$

Families of one-way functions (more generality)

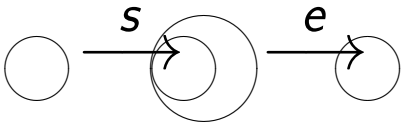


One-way function family (owff) is a deterministic polynomial-time algorithm $G\colon (1^n, r_g) \mapsto (s_n, e_n)$ that generates two Boolean circuits

- $e_n\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ (the function itself),
- $s_n\colon \{0,1\}^{\sigma(n)} \to \{0,1\}^n$ (sampler)

such that

Families of one-way functions (more generality)



One-way function family (owff) is a deterministic polynomial-time algorithm $G\colon (1^n, r_g) \mapsto (s_n, e_n)$ that generates two Boolean circuits

- $e_n\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ (the function itself),
- $s_n\colon \{0,1\}^{\sigma(n)} \to \{0,1\}^n$ (sampler)

such that

$\forall\, A$ (adversary, randomized polynomial-time)
$\forall k \in \mathbb{N}\ \ \exists N\ \ \forall n > N$

$$\mathbf{Pr}\{A(e_n(s_n(r_s)), 1^n, e_n, s_n) \in e_n^{-1}(e_n(s_n(r_s)))\} < \frac{1}{n^k},$$

where **Pr** is over A's randomness and uniformly distributed r_g, r_s.

Such an ugly and boring notation even at this level.
We will try to avoid it ☺.

Assumptions regarding lengths in one-way functions

Our owf's will be nice:

- length-regular: $|x| = |y| \Rightarrow |f(x)| = |f(y)|$ (typically),

Assumptions regarding lengths in one-way functions

Our owf's will be nice:

- length-regular: $|x| = |y| \Rightarrow |f(x)| = |f(y)|$ (typically),
- length-preserving: $|f(x)| = |x|$ (sometimes),

Assumptions regarding lengths in one-way functions

Our owf's will be nice:

- length-regular: $|x| = |y| \Rightarrow |f(x)| = |f(y)|$ (typically),
- length-preserving: $|f(x)| = |x|$ (sometimes),
- length-poly-bounded: $|x|^{\Omega(1)} \leqslant |f(x)| \leqslant |x|^{O(1)}$ (always),

Assumptions regarding lengths in one-way functions

Our owf's will be nice:

- length-regular: $|x| = |y| \Rightarrow |f(x)| = |f(y)|$ (typically),
- length-preserving: $|f(x)| = |x|$ (sometimes),
- length-poly-bounded: $|x|^{\Omega(1)} \leqslant |f(x)| \leqslant |x|^{O(1)}$ (always),
- not necessarily defined on each length,

Assumptions regarding lengths in one-way functions

Our owf's will be nice:

- length-regular: $|x| = |y| \Rightarrow |f(x)| = |f(y)|$ (typically),
- length-preserving: $|f(x)| = |x|$ (sometimes),
- length-poly-bounded: $|x|^{\Omega(1)} \leqslant |f(x)| \leqslant |x|^{O(1)}$ (always),
- not necessarily defined on each length,
- can be sliced.

Exercise

The existence of

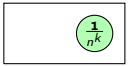
- one-way functions,
- owf with these nice "properties",
- families of one-way functions,

what is equivalent, what is not, what is unclear?

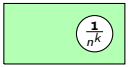
Note that a family has more than one pair (s_n, e_n) for each n.

"Weak" one-way functions

- "Strong" one-way functions: $\forall k$ an adversary cannot reach success probability $1/n^k$.

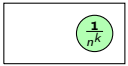


- "Weak" one-way functions: $\exists k$ an adversary cannot reach success probability $1 - 1/n^k$.

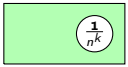


"Weak" one-way functions

- "Strong" one-way functions: $\forall k$ an adversary cannot reach success probability $1/n^k$.



- "Weak" one-way functions: $\exists k$ an adversary cannot reach success probability $1 - 1/n^k$.



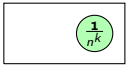
Weak $\mapsto$ Strong: *invert many copies*

$$f_s(x_1, x_2, \ldots, x_m) = (f_w(x_1), f_w(x_2), \ldots, f_w(x_m)),$$

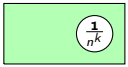
for a polynomial $m = m(n) = \Omega(n)$.

"Weak" one-way functions

- "Strong" one-way functions: $\forall k$ an adversary cannot reach success probability $1/n^k$.



- "Weak" one-way functions: $\exists k$ an adversary cannot reach success probability $1 - 1/n^k$.



Weak $\mapsto$ Strong: *invert many copies*

$$f_s(x_1, x_2, \ldots, x_m) = (f_w(x_1), f_w(x_2), \ldots, f_w(x_m)),$$

for a polynomial $m = m(n) = \Omega(n)$.

Proof by reduction:

Assume that A_s inverts f_s with prob. $\geqslant 1/n^{k_s}$

$\Downarrow$

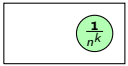
Construct A_w that breaks f_w.

$A'_w(y)$:

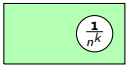
- `for each` j `in` $1..m$,
  - $x_1, \ldots, x_m \leftarrow$ random;
  - $x = A_s(f_w(x_1), \ldots, f_w(x_{j-1}), y, f_w(x_{j+1}), \ldots, f_w(x_m))_j$.
  - `if` $f_w(x) = y$, `then return` x.

"Weak" one-way functions

- "Strong" one-way functions: $\forall k$ an adversary cannot reach success probability $1/n^k$.



- "Weak" one-way functions: $\exists k$ an adversary cannot reach success probability $1 - 1/n^k$.



Weak $\mapsto$ Strong: *invert many copies*

$$f_s(x_1, x_2, \ldots, x_m) = (f_w(x_1), f_w(x_2), \ldots, f_w(x_m)),$$

for a polynomial $m = m(n) = \Omega(n)$.

Proof by reduction:

Assume that A_s inverts f_s with prob. $\geqslant 1/n^{k_s}$

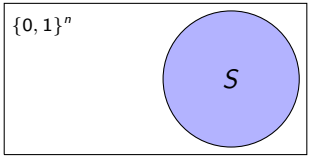
$\Downarrow$

Construct A_w that breaks f_w.

$A'_w(y)$: *// Final A_w repeats it $a(n)$ times.*

- `for each` j `in` $1..m$,
  - $x_1, \ldots, x_m \leftarrow$ random;
  - $x = A_s(f_w(x_1), \ldots, f_w(x_{j-1}), y, f_w(x_{j+1}), \ldots, f_w(x_m))_j$.
  - `if` $f_w(x) = y$, `then return` x.

Weak-to-strong owf: Proof

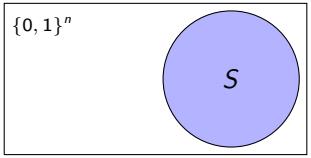


S is a "success set" for A'_w:

$$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$$

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
for each j in $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ if $f_w(x) = y$, then return x

Weak-to-strong owf: Proof

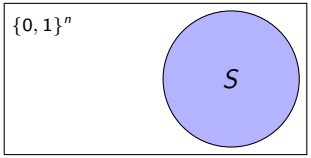


S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
`for each` j `in` $1..m$
$\triangleright$ $x_1, \ldots, x_m \leftarrow$ random
$\triangleright$ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
$\triangleright$ `if` $f_w(x) = y$, `then return` x

Weak-to-strong owf: Proof



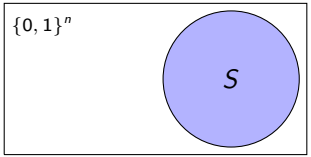
S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(1 - \frac{1}{a(n)/n}\right)^{a(n)}$$

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
`for each` j `in` $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ `if` $f_w(x) = y$, `then return` x

Weak-to-strong owf: Proof



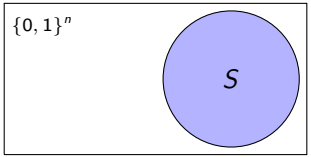
S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
`for each` j `in` $1..m$
$\triangleright$ $x_1, \ldots, x_m \leftarrow$ random
$\triangleright$ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
$\triangleright$ `if` $f_w(x) = y$, `then return` x

Weak-to-strong owf: Proof



S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

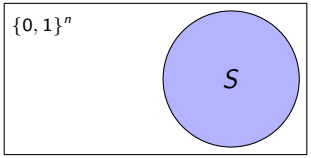
Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

Claim: $\mathbf{Pr}\{x \in S\} \geqslant 1 - \frac{n}{m}$. Enough to break f_w, as we choose m.

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
for each j in $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ if $f_w(x) = y$, then return x

Weak-to-strong owf: Proof



S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

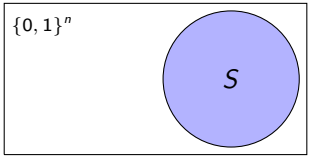
$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

Claim: $\mathbf{Pr}\{x \in S\} \geqslant 1 - \frac{n}{m}$. Enough to break f_w, as we choose m.
Assume: $\mathbf{Pr}\{x \in S\} < 1 - \frac{n}{m}$.

$\mathbf{Pr}\{A_s \text{ succeeds }\} \leqslant \mathbf{Pr}\{\text{success if all } x_i \in S\} + \mathbf{Pr}\{\text{success if not all}\}$

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
for each j in $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ if $f_w(x) = y$, then return x

Weak-to-strong owf: Proof



S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

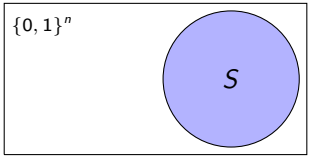
Claim: $\mathbf{Pr}\{x \in S\} \geqslant 1 - \frac{n}{m}$. Enough to break f_w, as we choose m.
Assume: $\mathbf{Pr}\{x \in S\} < 1 - \frac{n}{m}$.

$\mathbf{Pr}\{A_s \text{ succeeds }\} \leqslant \mathbf{Pr}\{\text{success if all } x_i \in S\} + \mathbf{Pr}\{\text{success if not all}\}$

All: $\leqslant \left(\left(1 - \frac{1}{m/n} \right)^{m/n} \right)^n \sim \frac{1}{e^n}$.

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
`for each` j `in` $1..m$
$\triangleright$ $x_1, \ldots, x_m \leftarrow$ random
$\triangleright$ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
$\triangleright$ `if` $f_w(x) = y$, `then return` x

Weak-to-strong owf: Proof



S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

Claim: $\mathbf{Pr}\{x \in S\} \geqslant 1 - \frac{n}{m}$. Enough to break f_w, as we choose m.
Assume: $\mathbf{Pr}\{x \in S\} < 1 - \frac{n}{m}$.

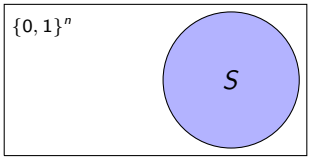
$\mathbf{Pr}\{A_s \text{ succeeds }\} \leqslant \mathbf{Pr}\{\text{success if all } x_i \in S\} + \mathbf{Pr}\{\text{success if not all}\}$

All: $\leqslant \left(\left(1 - \frac{1}{m/n} \right)^{m/n} \right)^n \sim \frac{1}{e^n}$.

Not all: $\leqslant \frac{n}{a(n)} \times m$.

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
for each j in $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ if $f_w(x) = y$, then return x

Weak-to-strong owf: Proof



S is a "success set" for A'_w:
$S = \left\{ y \in \{0,1\}^n \mid \mathbf{Pr}\{A'_w \text{ inverts } f_w \text{ in } y\} > \frac{n}{a(n)} \right\}$

Overall success of A_w is $\geq \mathbf{Pr}\{y \in S\} \times \underbrace{\mathbf{Pr}\{A_w \text{ inv. } f_w \text{ in } y \mid y \in S\}}$.

$$1 - \left(\left(1 - \frac{1}{a(n)/n} \right)^{a(n)/n} \right)^n \sim 1 - \frac{1}{e^n}$$

Claim: $\mathbf{Pr}\{x \in S\} \geqslant 1 - \frac{n}{m}$. Enough to break f_w, as we choose m.
Assume: $\mathbf{Pr}\{x \in S\} < 1 - \frac{n}{m}$.

$\mathbf{Pr}\{A_s \text{ succeeds }\} \leqslant \mathbf{Pr}\{\text{success if all } x_i \in S\} + \mathbf{Pr}\{\text{success if not all}\}$

All: $\leqslant \left(\left(1 - \frac{1}{m/n} \right)^{m/n} \right)^n \sim \frac{1}{e^n}$.

Not all: $\leqslant \frac{n}{a(n)} \times m$.

Let $a(n) >> m$, then A_s does not break f_s.

$A_w(y)$: repeat $a(n)$ times $A'_w(y)$:
for each j in $1..m$
▷ $x_1, \ldots, x_m \leftarrow$ random
▷ $x = A_s(f_w(x_1)..f_w(x_{j-1}), y..f_w(x_m))_j$
▷ if $f_w(x) = y$, then return x

Candidate one-way functions

Example (Not really)

$f(a, b) = a \cdot b$.

Candidate one-way functions

Example (Not really)

$f(a, b) = a \cdot b$.

Example

$f(p, q) = p \cdot q$, where $p, q \in \mathbb{P}$, and $p < q$, and $\log p \approx \log q$.

Candidate one-way functions

Example (Not really)

$f(a, b) = a \cdot b$.

Example

$f(p, q) = p \cdot q$, where $p, q \in \mathbb{P}$, and $p < q$, and $\log p \approx \log q$.

Example

$f(x_1, x_2, \ldots, x_n, I) = (x_1, x_2, \ldots, x_n, \sum_{i \in I} x_i)$, where $I \subseteq \{1, \ldots, n\}$
(This is SUBSET SUM, it is also **NP**-complete).

Candidate one-way functions

Example (Not really)

$f(a, b) = a \cdot b$.

Example

$f(p, q) = p \cdot q$, where $p, q \in \mathbb{P}$, and $p < q$, and $\log p \approx \log q$.

Example

$f(x_1, x_2, \ldots, x_n, I) = (x_1, x_2, \ldots, x_n, \sum_{i \in I} x_i)$, where $I \subseteq \{1, \ldots, n\}$
(This is SUBSET SUM, it is also **NP**-complete).

What do we mean by commas here?

Candidate one-way functions

Example (Not really)

$f(a, b) = a \cdot b$.

Example

$f(p, q) = p \cdot q$, where $p, q \in \mathbb{P}$, and $p < q$, and $\log p \approx \log q$.

Example

$f(x_1, x_2, \ldots, x_n, I) = (x_1, x_2, \ldots, x_n, \sum_{i \in I} x_i)$, where $I \subseteq \{1, \ldots, n\}$
(This is SUBSET SUM, it is also **NP**-complete).

Example (Discrete logarithm)

Take $p \in \mathbb{P}$. Take g a generator in $\mathbb{F}_p$. Consider $x \in \mathbb{Z}_{p-1}$.

$f(x) = g^x \mod p$

Why "logarithm"? To invert, find " $\log_g(g^x)$ ".

Universal ("complete") one-way function

no complexity claim

$f \to g$, if $\exists$ (randomized) polynomial-time $T^\bullet$ $\forall k_f$ $\exists k_g$ $\forall$ adversary A

$$A \text{ inverts } g \text{ with prob. } 1 - \frac{1}{n^{k_g}} \implies T^A \text{ inverts } f \text{ with prob. } 1 - \frac{1}{n^{k_f}}.$$

Universal ("complete") one-way function

no complexity claim

$f \to g$, if $\exists$ (randomized) polynomial-time $T^\bullet$ $\forall k_f$ $\exists k_g$ $\forall$ adversary A

$$A \text{ inverts } g \text{ with prob. } 1 - \frac{1}{n^{k_g}} \implies T^A \text{ inverts } f \text{ with prob. } 1 - \frac{1}{n^{k_f}}.$$

Theorem (the hardest-to-invert function)

There is a universal weak one-way function u, i.e., s.t. $\forall f \in \boldsymbol{f}\mathbf{P}$ $f \to u$.

algorithm, infinite storage

$u(M, x) = (M, M(x))$, where $x \in \{0,1\}^*$, M is the description of a Turing machine, $M(x)$ is the output of M on x *within* $|x|^2$ *steps (or just* $|x|$*).*

Universal ("complete") one-way function

no complexity claim

$f \to g$, if $\exists$ (randomized) polynomial-time $T^\bullet$ $\forall k_f$ $\exists k_g$ $\forall$ adversary A

$$A \text{ inverts } g \text{ with prob. } 1 - \frac{1}{n^{k_g}} \implies T^A \text{ inverts } f \text{ with prob. } 1 - \frac{1}{n^{k_f}}.$$

Theorem (the hardest-to-invert function)

There is a universal weak one-way function u, i.e., s.t. $\forall f \in \boldsymbol{f}\mathbf{P}$ $f \to u$.

algorithm, infinite storage

$u(M, x) = (M, M(x))$, where $x \in \{0,1\}^*$, M is the description of a Turing machine, $M(x)$ is the output of M on x *within* $|x|^2$ *steps (or just* $|x|$*).*

Lemma

W.l.o.g. a weak owf is computable in $|x|^2$ steps.

$\tilde{f}(x_1x_2) = f(x_1)x_2$, where $|x_1| = n$, $|x_2| = m = m(n)$. Choose $m(n)$.

To break f:

- on input $y = f(x_1)$ concatenate a random x_2 to it;
- invert $y \cdot x_2$ using $\tilde{f}$;
- drop the suffix.

Proof of universality

$u(M, x) = (M, M(x))$

Invert f^* computed by M^* using an inverter for u.

Proof of universality

$u(M, x) = (M, M(x))$

Invert f^* computed by M^* using an inverter for u.

$u(x)$ computes $M^*(x)$ for $\mu = \frac{1}{2^{|M^* \cdot \mathrm{const}|}} = \mathrm{const}$ (sufficiently long) inputs.

Proof of universality

$u(M, x) = (M, M(x))$

Invert f^* computed by M^* using an inverter for u.

$u(x)$ computes $M^*(x)$ for $\mu = \frac{1}{2^{|M^* \cdot \mathrm{const}|}} = \mathrm{const}$ (sufficiently long) inputs.

If we invert all but fraction $\frac{1}{n^k}$ of u's inputs, then we also invert many enough inputs of the form $u(M^*, x)$ (namely, fraction $\mu - \frac{1}{n^k}$, which is $\nu := 1 - \frac{1}{\mu n^k}$ in terms of M^* on length $n - |M^*|$).

Choose k and n so that ν is large enough.

Proof of universality

$u(M, x) = (M, M(x))$

Invert f^* computed by M^* using an inverter for u.

$u(x)$ computes $M^*(x)$ for $\mu = \frac{1}{2^{|M^* \cdot \text{const}|}} = \text{const}$ (sufficiently long) inputs.

If we invert all but fraction $\frac{1}{n^k}$ of u's inputs, then we also invert many enough inputs of the form $u(M^*, x)$ (namely, fraction $\mu - \frac{1}{n^k}$, which is $\nu := 1 - \frac{1}{\mu n^k}$ in terms of M^* on length $n - |M^*|$).

Choose k and n so that ν is large enough.

Exercise

What about...

- strong owf?
- deterministic or non-uniform adversaries?

Administrivia

Grading scheme (not a cryptographic scheme)

Administrivia

Grading scheme

Suspense!

Administrivia

Grading scheme

- Homework exercises: 20%

Administrivia

Grading scheme

- Homework exercises: 20%
- Midterm (oral, open book, before everyone): 40%

Administrivia

Grading scheme

- Homework exercises: 20%
- Midterm (oral, open book, before everyone): 40%
- Final exam (oral, open book, before Edward): 40%

Administrivia

Grading scheme

- Homework exercises: 20%
- Midterm (oral, open book, before everyone): 40%
- Final exam (oral, open book, before Edward): 40%
- Lecture notes: up to additional 10% (depending on the lecture)

Trapdoor functions

. . . come with their families

(first attempt)

Trapdoor function family (tdff) is a collection of polynomial-time computable functions e_n, d_n such that $d_n(e_n(x)) = x$ and for every adversary A,

$$\mathbf{Pr}\{A \text{ inverts } e_n \text{ in } x\} \text{ is negligible.}$$

Trapdoor functions

. . . come with their families

(first attempt)

Trapdoor function family (tdff) is a collection of polynomial-time computable functions e_n, d_n such that $d_n(e_n(x)) = x$ and for every adversary A,

$$\mathbf{Pr}\{A \text{ inverts } e_n \text{ in } x\} \text{ is negligible.}$$

Not quite accurate: there is no uniform algorithm $D(n, x)$ for d_n (and for e_n as well).

Trapdoor functions

. . . come with their families

(first attempt)

Trapdoor function family (tdff) is a collection of polynomial-time computable functions e_n, d_n such that $d_n(e_n(x)) = x$ and for every adversary A,

$$\mathbf{Pr}\{A \text{ inverts } e_n \text{ in } x\} \text{ is negligible.}$$

Not quite accurate: there is no uniform algorithm $D(n, x)$ for d_n (and for e_n as well).

Trapdoor function family (tdff) is a deterministic polynomial-time $G\colon (1^n, r_g) \mapsto (e, \quad s)$ (Boolean circuits) s.t.

- $e\colon \{0,1\}^n \to \{0,1\}^{\epsilon(n)}$ (encryptor)
- $s\colon \{0,1\}^{\sigma(n)} \to \{0,1\}^n$ (sampler),

such that
$\forall$ adversary A $\quad \forall k \in \mathbb{N} \;\; \exists N \;\; \forall n > N$

$$\mathbf{Pr}\{A(e(s(r_s)), 1^n, e, s) \in e^{-1}(e(s(r_s)))\} < \frac{1}{n^k},$$

where $G(1^n, r_g) = (e, d, s)$ and $\mathbf{Pr}$ is taken over A's randomness and over uniformly distriuted r_g, r_s.

Trapdoor functions

. . . come with their families

(first attempt)

Trapdoor function family (tdff) is a collection of polynomial-time computable functions e_n, d_n such that $d_n(e_n(x)) = x$ and for every adversary A,

$$\mathbf{Pr}\{A \text{ inverts } e_n \text{ in } x\} \text{ is negligible.}$$

Not quite accurate: there is no uniform algorithm $D(n, x)$ for d_n (and for e_n as well).

Trapdoor function family (tdff) is a deterministic polynomial-time $G\colon (1^n, r_g) \mapsto (e, d, s)$ (Boolean circuits) s.t.

- $e\colon \{0,1\}^n \to \{0,1\}^{\epsilon(n)}$ (encryptor)
- $d\colon \{0,1\}^{\epsilon(n)} \to \{0,1\}^n$ (decryptor)
- $s\colon \{0,1\}^{\sigma(n)} \to \{0,1\}^n$ (sampler),

such that $\forall x \in \mathrm{Im}(s)\ d(e(x)) = x$ и
$\forall$ adversary A $\quad \forall k \in \mathbb{N}\ \ \exists N\ \ \forall n > N$

$$\mathbf{Pr}\{A(e(s(r_s)), 1^n, e, s) \in e^{-1}(e(s(r_s)))\} < \frac{1}{n^k},$$

where $G(1^n, r_g) = (e, d, s)$ and $\mathbf{Pr}$ is taken over A's randomness and over uniformly distriuted r_g, r_s.

Trapdoor function: example

Example (RSA, Rivest–Shamir–Adleman)

Take $p, q \in \mathbb{P}$. Let $N = p \cdot q$.

Pick ε, δ from $\mathbb{Z}_N$ s.t. $\varepsilon \cdot \delta \equiv 1 \pmod{(p-1)(q-1)}$.

Let

$$s(x) = x$$
$$e(x) = x^{\varepsilon} \mod N$$
$$d(x) = x^{\delta} \mod N$$

Discussion questions:

- why it works?
- give me p, q, ϵ, δ
- what if we know how to find factors efficiently

Takeout

- Weak and strong one-way functions
- Universal owf
- Trapdoor functions

Coming next: Godreich–Levin hardcore predicate, public-key encryption schemes