The Many Entropies of One-Way Functions

Omer Reingold

Microsoft Research

Joint With

Iftach Haitner

Thomas Holenstein

Salil Vadhan

Hoeteck Wee

Cryptography

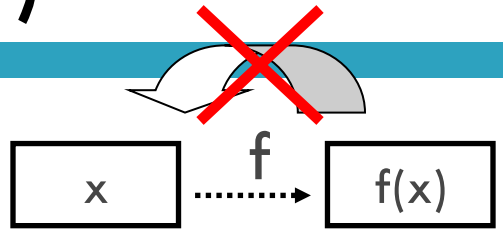
- Rich array of applications and powerful implementations.
- In some cases (e.g Zero-Knowledge), more than we would have dared to ask for.

Cryptography

- Proofs of security very important
- BUT, almost entirely based on computational hardness assumptions (factoring is hard, cannot find collisions in SHA-1, …)

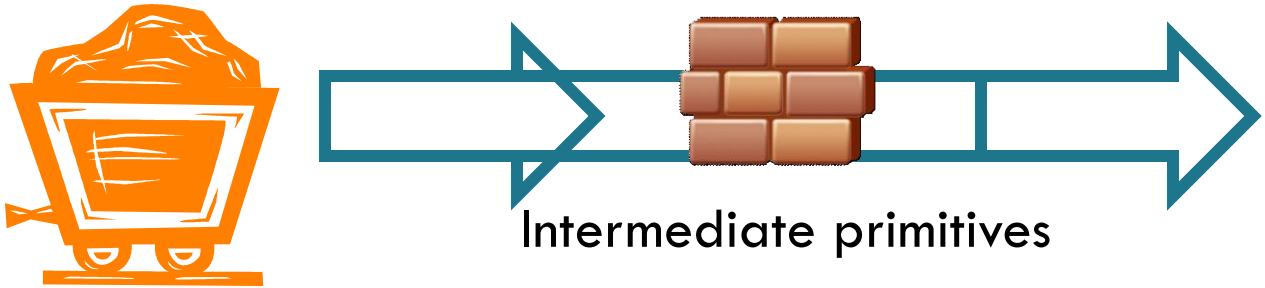
One Way Functions (OWF)

- Easy to compute
- Hard to invert (even on the average)



The most basic, unstructured form of cryptographic hardness [Impagliazzo-Luby '95]

Major endeavor: base as much of Crypto on existence of OWFs – Great success (even if incomplete)



Primitives Hierarchy

One-way functions

Pseudorandom generators

[Håstad-Impagliazzo-Levin-Luby '91]

Private Key Encryption | Message Authentication | Pseudorandom functions | …




Applications

Pseudorandomness naturally corresponds to secrecy. But Crypto is also about Unforgeability (preventing cheating)

Primitives Hierarchy

One-way functions

Pseudor… generators

Universal One-way Hash

Statistically hiding commitment

Digit… signatu…

Very involved and seems far from optimal

[Haitner- Nguyen-Ong-R-Vadhan '07]

[Rompel '90]

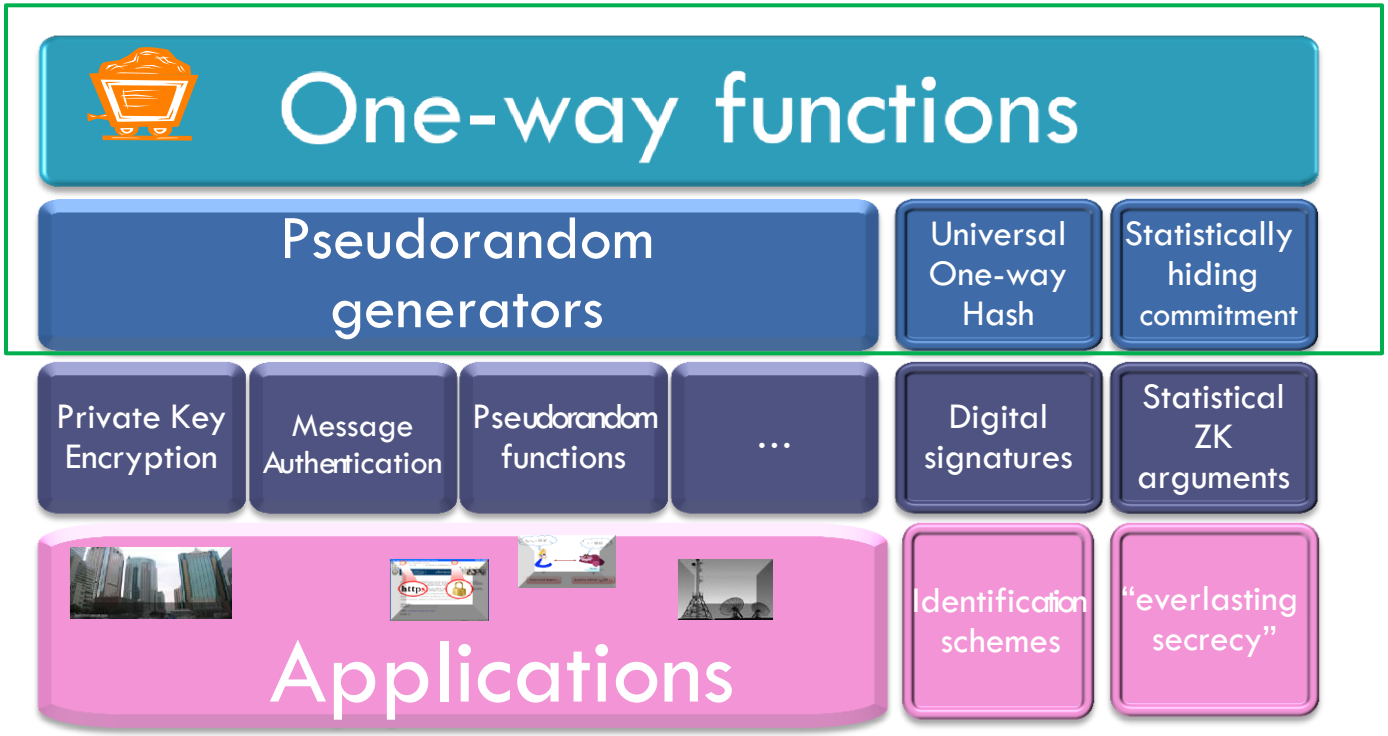
[HRV '10, VZ '12]

[Haitner-R-Vadhan-Wee '09]

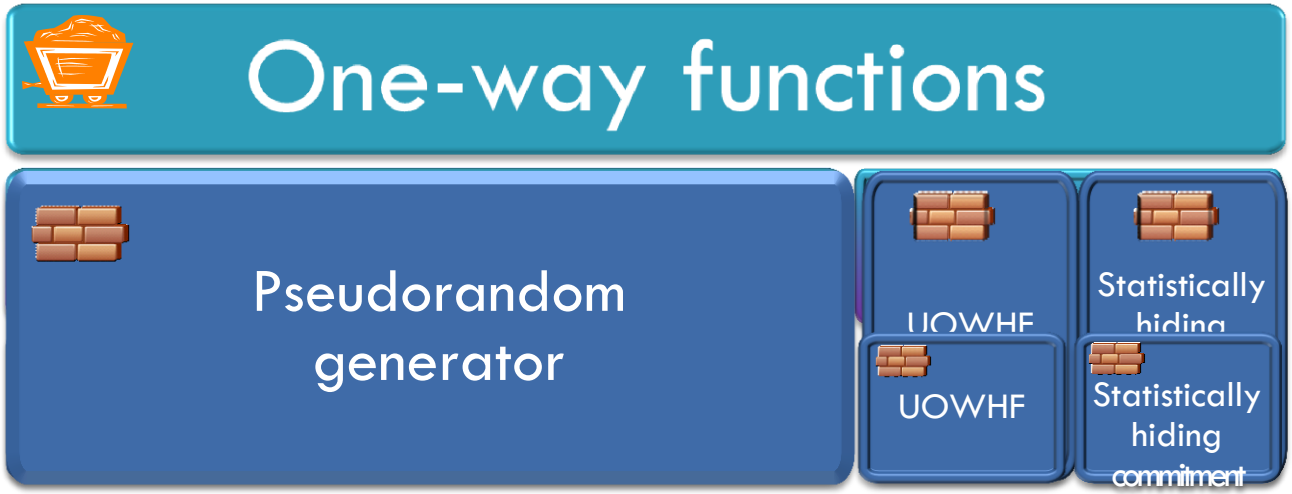
[Haitner-Holenstein-R-Vadhan-Wee '10]

- Simplifications and improvements in efficiency
- Based on new notions of computational entropy

Primitives Hierarchy



Building the First Layer



Entropy and Pseudoentropy

- For a random variable X denote by $H(X)$ the entropy of X. Intuitively: how many bits of randomness in X.
- Various measures of entropy: Shannon entropy ($H(X) = E_{x \leftarrow X}[\log(1/\Pr[X=x])]$), min-entropy, max-entropy, …
- For this talk, enough to imagine X that is uniform over 2^k elements. For such X, $H(X)=k$.
- X has pseudoentropy $\geq k$ if $\exists$ Y with $H(Y) \geq k$ such that X and Y are computationally indistinguishable [HILL]

Pseudorandom Generators [BM,Yao]

Efficiently computable function $G:\{0,1\}^s \to \{0,1\}^m$

- Stretching ($m > s$)
- Output is computationally indistinguishable from uniform (i.e., has pseudoentropy m).

False Entropy Generator

- Loosely, the most basic object in HILL is:

$$G_{fe}(x,g,i)=f(x),g,g(x)_{1..i}$$

(*think* of g as matrix multiplication).

Lemma Let $k=\log|f^{-1}(f(x))|$, then when $i=k+\log n$ then $g,g(x)_{1..i}$ is pseudorandom (even given $f(x)$).

- Intuition: first $k-c\cdot\log n$ bits are statistically close to uniform (Leftover Hash Lemma) and next $(c+1)\log n$ bits are pseudorandom (GL Hard-Core Function).

False Entropy Generator (II)

$G_{fe}(x,g,i)=f(x),g,g(x)_{1..i}$

Lemma: For the variable $G_{fe}(x,g,i)$ (with random inputs)

$$\Delta = \text{pseudoentropy} - \text{real entropy} > (\log n)/n$$

Reason: w.p $1/n$ over choice of i (when $i=k+\log n$) the output $G_{fe}(x,g,i)$ is indistinguishable from distribution with entropy $|x|+|g|+\log n$ (whereas real entropy $\leq |x|+|g|$)

- Disadvantages: Δ rather small, value of real entropy unknown, pseudoentropy < entropy of input

Building Block of [HRV '10]

- Simply do not truncate:

$$G_{nb}(x,g)=f(x),g,\ g(x)$$

- Nonsense: $G_{nb}(x,g)$ is invertible and therefore has no pseudoentropy!
- Well yes, but: $G_{nb}(x,g)$ does have pseudoentropy from the point of view of an online distinguisher (getting one bit at a time).

$G_{nb}(x,g)=f(x),g,g(x)_{1,\ldots n}$

Next-Bit Pseudoentropy

- X has pseudoentropy $\geq k$ if $\exists$ Y with $H(Y) \geq k$ such that X and Y are computationally indistinguishable
- $X=(X_1 \ldots X_n)$ has next-bit pseudoentropy $\geq k$ if $\exists$ $\{Y_i\}_{i\in[n]}$ with
  - $\Sigma_i H(Y_i | X_1 \ldots X_{i-1}) \geq k$ such that
  - X_i and Y_i are computationally indistinguishable given $X_1,\ldots,X_{i-1}$
- Remarks:
  - X and $\{Y_i\}$ are jointly distributed
  - The two notions are identical for $k=n$ [BM, Yao, GGM]
  - Generalizes to blocks (rather than bits)
  - Next-bit pseudoentropy is weaker than pseudoentropy

Our Next-Block Pseudoentropy Generator

- $G_{nb}(x,g)=f(x),g,g(x)$
- Next-block pseudoentropy $> |x|+|g|+\log n$
  - $X=G(x,g)$ and $\{Y_i\}$ obtained from X by replacing first $k+\log n$ bits of $g(x)$ with uniform bits, where $k=\log|f^{-1}(f(x))|$
- Advantages:
  - Δ = (next-block pseudoentropy – real entropy) $> \log n$
  - Entropy bounds known (on total entropy)
  - "No bit left behind"

Simple form of PRGs in OWFs

In conclusion: for OWF $f:\{0,1\}^n \to \{0,1\}^n$ & (appropriate) pair-wise independent hash function $g:\{0,1\}^n \to \{0,1\}^n$

- Has pseudoentropy in the eyes of an online distinguisher (i.e., next-block pseudoentropy)
- [Vadhan-Zheng '12] Don't need g at all + additional efficiency improvement.

Pseudoentropy vs. Inaccessible Entropy

[HILL '91]: A distribution X has ps
indistinguishable from X' such th

Secrecy

- X looks like it has more entropy than it really does

[HRVW '09] X has in inaccess
efficient algorithm A, if A '
ort" of X then $H(A(\cdot)) < H(X)$

Unforgeability

- X has entropy but some of it is inaccessible

Universal One-Way Hash Functions [NY]

$G=\{g\}$ a family of efficiently computable hash functions such that

- (2^{nd} pre-image) Given random g and x, hard to find x' such that $g(x)=g(x')$.
- Compare with collision resistance: Given g, hard to find x and x' such that $g(x)=g(x')$.

Simple form of UOWHFs in OWFs

OWF $f:\{0,1\}^n \to \{0,1\}^n$

- Define $F(x,i)$= first i bits of $f(x)$
- Given random x,i may be possible to find x' such that $F(x,i)= F(x',i) \Rightarrow F$ may be broken as a UOWHF
- But it is infeasible to sample such x' with full entropy $\Rightarrow$ F is "a bit like" a UOWHF

Simple form of UOWHFs in OWFs

Proof idea: Assume that given x,i algorithm A samples x' with full entropy such that $F(x,i)= F(x',i)$.

In other words, x' is uniform such that first i bits of $f(x)$ equal first i bits of $f(x')$

Given y find $x=f^{-1}(y)$ (breaking f) as follows:

- Let x_i be such that $f(x_i)$ and y agree on first i bits.
- To get x_{i+1} from x_i use A on input (x_i,i) until it samples x' such that $f(x')$ and y agree on first $i+1$ bits (set $x_{i+1} = x'$).
- Output $x=x_n$.

Inaccessible Entropy Generator

OWF $f:\{0,1\}^n \to \{0,1\}^n$

- Inaccessible entropy generator:

  Define $G_{ie}(x)=f(x)_1,f(x)_2,\ldots,f(x)_n,x$

- Informal thm: There is no algorithm that produces each one of the $n+1$ blocks (in an online fashion) with full entropy (hence an inaccessible entropy generator).
- Useful in construction of statistically hiding commitment schemes and inspiring in construction of UOWHFS (slightly different analysis).

Connection to Statistical Commitment

- Inaccessible entropy generator:

  Define $G_{ie}(s)=Z_1,Z_2,\ldots,Z_n$

- Assume Z_i is a uniform bit (from the point of view of an observer) but is completely fixed conditioned on the internal state of any algorithm generating it.
- Use Z_i to mask a bit σ (output $Z_1,Z_2,\ldots,Z_{i-1}, Z_i \oplus \sigma$).
- Then σ is statistically hidden (for outside observer) but the committer can only open a single value.

Two Computational Entropy Generators

$f:\{0,1\}^n \to \{0,1\}^n$ OWF.

- Next block pseudoentropy generator:

  $G_{nb}(x)=f(x),x_1,x_2,\ldots,x_n$

  - Looks (on-line) like it has entropy $|x|+\log n$.
- Inaccessible entropy generator:

  $G_{ie}(x)=f(x)_1,f(x)_2,\ldots,f(x)_n,x$

  - Can generate (on-line) at most $|x| - \log n$ bits of entropy.

Summary

- When viewed correctly, one-way functions rather directly imply simple forms of the “first layer” of cryptographic primitives.
- This view relies on setting the “right” computational notions of entropy.
- Open problems: Beyond the world of OWFs, Use for lower bounds, Further Unifications, Better constructions,

Widescreen Test Pattern (16:9)