Pseudorandomness

Tuesday, September 24, 2019 11:00 AM

Founding Crypto on OWF

- we need pseudorandomness for practical cryptography

One time pad

$m \in \{0,1\}^n$

$k \xleftarrow{\$} \{0,1\}^\lambda$

ciphertext

$y \; \underline{c} \leftarrow Enc(k, m)$: $c = m \oplus k$ (xor)

$m' \leftarrow Dec(c, k)$

$$\forall m_1, m_2 \quad \left\{ \begin{matrix} k \xleftarrow{\$} \{0,1\}^\lambda \\ c \leftarrow Enc(k, m_1) \end{matrix} \right\} = \left\{ \begin{matrix} k \leftarrow \{0,1\}^\lambda \\ c \leftarrow Enc(k, m_2) \end{matrix} \right\}$$

w/ pseudorandomness, small key k, but send lots of enc. message

- We want cryptography from minimal assumptions

Weakest possible: $\underline{P} \neq \underline{NP}$

find solutions — check solutions

$\longrightarrow$ existence of OWF

Stronger assumption: DLOG is specifically not in P

$$OWF \longrightarrow PRG \longrightarrow PRF$$

one way functions — pseudorandom generators — pseudorandom functions

- One way function:

$f_\lambda : D_\lambda \longrightarrow C_\lambda$ is a (family) of OWF

iff $\forall \mathcal{A}, \Pr\left[\begin{matrix} x \xleftarrow{\$} D_\lambda \\ y = f(x) \end{matrix} \; x' \leftarrow \mathcal{A}(1^n, y) : f(x') = y \right] = negl(\lambda)$

ex. $f(x) = g^x$ in DLOG group

- Pseudorandom Generator

$f_\lambda : D_\lambda \longrightarrow C_\lambda$ is a PRG

$$\left\{ \begin{matrix} x \xleftarrow{\$} D_\lambda \\ y = f(x) \end{matrix} : y \right\} \underset{N_c}{\sim} \left\{ y \xleftarrow{\$} C_\lambda : y \right\}$$

computationally distinguishable

"gives apparently random outputs, given random inputs"

$$\forall \mathcal{A}, \left| \Pr\left[\begin{matrix} x \xleftarrow{\$} D_\lambda \\ b \leftarrow \mathcal{A}(1^n, f(x)) \end{matrix} : b = 1 \right] - \Pr\left[\begin{matrix} y \xleftarrow{\$} C_\lambda \\ b \leftarrow \mathcal{A}(1^n, y) \end{matrix} : b = 1 \right] \right| = negl(\lambda)$$

- efficiently computable
- expansion $|C_\lambda| > |D_\lambda|$

Diff. between OWF and PRG

OWF $\longleftarrow$ hard invert

PRG $\longleftarrow$ hard to even learn partially or to distinguish from

random

Counter example:

suppose $f(x): \{0,1\}^{2\lambda} \to \{0,1\}^{2\lambda}$ is a OWF

Define $g(x) = \underline{f(x)} \,\|\, \underline{0^\lambda} \quad : \{0,1\}^{2\lambda} \to \{0,1\}^{3\lambda}$

Another example:

$\mathrm{Ser}(x\,G)$

Pseudorandom Functions

$f_\lambda : D_\lambda \times I_\lambda \longrightarrow C_\lambda$ is a PRF iff

key space — input space

$k \xleftarrow{\$} D_\lambda$

$\mathcal{A} \xrightarrow{q_1} y_1 = f_\lambda(k, q_1)$

$\xleftarrow{x}$

$\xrightarrow{q_2} y_2 = f_\lambda(k, q_2)$

$\xleftarrow{\ }$

$$\forall \mathcal{A}, \left| \Pr\left[k \xleftarrow{\$} D_\lambda, b \leftarrow \mathcal{A}^{f_\lambda(k,\cdot)}(1^\lambda) : b = 1 \right] - \Pr\left[b \leftarrow \mathcal{A}^{RO(\cdot)}(1^\lambda) : b = 1 \right] \right| = \mathrm{negl}(\lambda)$$

$$\Downarrow$$

$$= \Pr\left[f' \xleftarrow{\$} S_{I_\lambda \to C_\lambda} : \mathcal{A}^{f'(\cdot)}(1^\lambda) \right]$$

space of all functions

$|C_\lambda|^{|I_\lambda|}$

How PRFs are used for encryption

$K = \{0,1\}^\lambda$

$f: K \times \{0,1\}^n \to \{0,1\}^\lambda$

$k \xleftarrow{\$} K$ key space

k — k

Alice — Bob

$(i, m_i \oplus f(K, i))$

i'th ciphertext, for i'th m_i

"wiretap" adversary

eavesdrop adversary

$\mathcal{A}$ passive

$\mathrm{Dec}(c, k) \to$ parse c as (i, c')

$m_i' = c' \oplus f(k, i) = (m_i \oplus f(k,i)) \oplus f(k,i) = m$

$\mathcal{A}$ sees: $(0, c_0')$, $(1, c_1')$, $\approx_c$ $\quad 0, q_1 \xleftarrow{\$} \{0,1\}^\lambda$

$\vdots$

$(2, c'_2)$ $\qquad$ $a_k \xleftarrow{\$} \{0,1\}^\lambda$

OWF $\to$ PRG

Hardcore predicate or hardcore bit

Def'n: A h.c.p. for a OWF $f: D_\lambda \to C_\lambda$,

"is a bit hard to predict even after seeing $f(x)$"

$h: D_\lambda \to \{0,1\}$

$$\Pr\left[x \xleftarrow{\$} D_\lambda,\ b \leftarrow \mathcal{A}(f(x)) : b = h(x)\right] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

gets to see $f(x)$

$$\underbrace{f(x)}_{\text{OWF}} \quad \underbrace{h(x)}_{} \quad \mathcal{A}$$

$$x$$

can't predict next bit

Universal Hardcore Predicate $\qquad$ Goldreich-Levin construction

Let $F_\lambda : \{0,1\}^\lambda \to C_\lambda$ be a OWF

Define $f' : (\{0,1\}^\lambda \times \{0,1\}^\lambda) \to (\{0,1\}^\lambda \times \{0,1\}^\lambda)$

$$f'(x, r) = (f(x), r)$$

Claim: f' is still a OWF.

Define $h(x, r) = \bigoplus_{i<\lambda} x_i \wedge r_i$ $\qquad$ $\left(\sum_i (x_i \wedge r_i)\right) \bmod 2$

$$= \langle x, r \rangle$$

Proof:

By reduction. for random

Suppose we have $\mathcal{A}$ where $\mathcal{A}$ can guess $\langle x, r \rangle$, given $f'(x, r) = f(x), r$

we can choose y, r, $b \leftarrow \mathcal{A}(y, r)$

we then need to construct $\mathcal{A}'$ which finds x given y, s.t. $f(x) = y$

Simplified case:

assume $\mathcal{A}$ is <u>always</u> a <u>correct guess</u> for every r

$$e_1 = 1\underbrace{000\ldots0}_{\lambda-1\ 0's} \leftarrow e_i \text{ selects } i\text{'th bit}$$

$$e_2 = 0100\ldots0$$

$$b_0 \leftarrow \mathcal{A}(y, e_0)$$

---

next harder case: queries

$\rightarrow r \xleftarrow{\$} \{0,1\}^\lambda$, (y, r) and $(y, r + e_i)$

$$b \leftarrow \mathcal{A}(y, r) = \langle x, r\rangle$$

$$\mathcal{A}(y, r \oplus e_i) = \langle x, r \oplus e_i\rangle$$

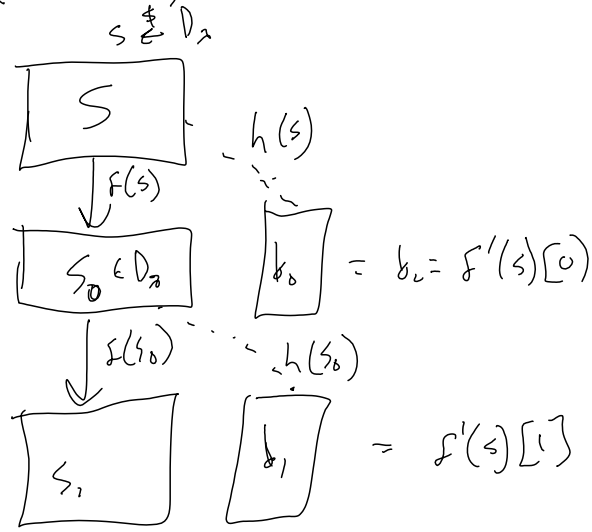
$$\langle x, r\rangle \oplus \langle x, r \oplus e_i\rangle = x_i$$

Composing hardcore predicates to get a <u>PRG</u>

Suppose we have a <u>OWF</u> $f: D_\lambda \to D_\lambda$

that is a permutation (f is one-to-one)

and $h: D_\lambda \to \{0,1\}$ is a hcp for f:



Claim: given f, h as above,

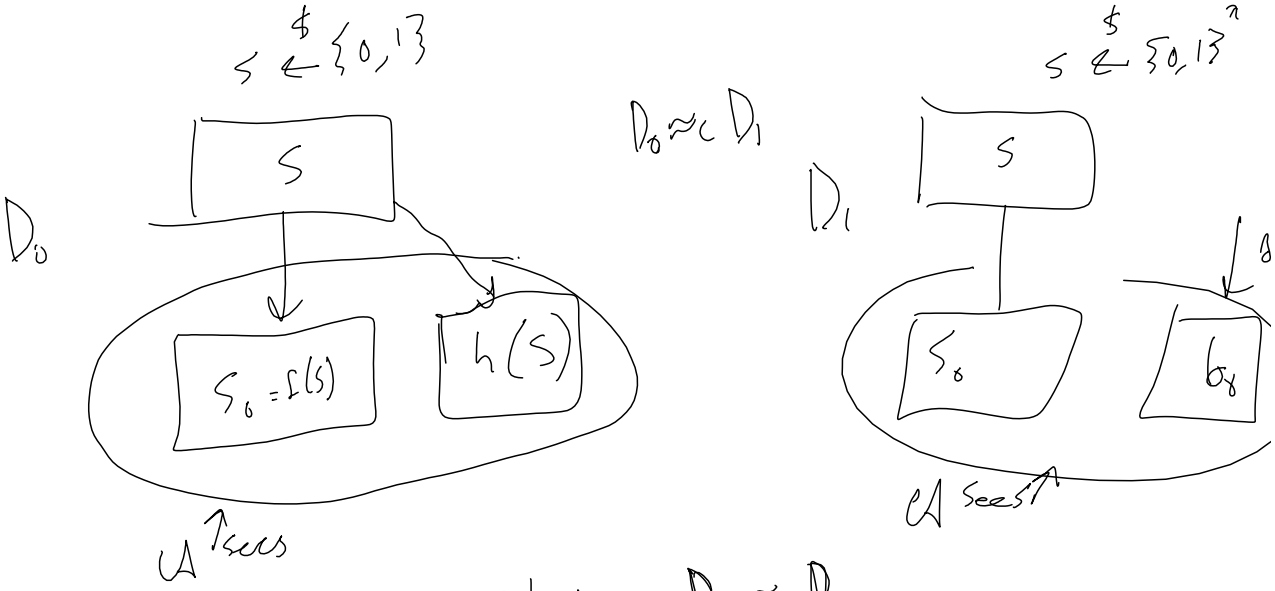
$f'(s): \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ is a PRG

where $f'(s)[i]$ $h(f^{(i-1)}(s))$

Proof: Assume we have a distinguisher $\mathcal{A}$,

$$P_r\left[\begin{array}{l} s \xleftarrow{\$} \{0,1\}^n \\ y = f'(s), \end{array} : \mathcal{A}(f'(s)) = 1\right] - P_r\left[y \xleftarrow{\$} \{0,1\}^{2n} : \mathcal{A}(y) = 1\right] > \frac{1}{2^n}$$

We need to construct $\mathcal{A}'$ that predicts $h(s)$ given $f(s)$



Claim: $D_0 \approx_c D_1$

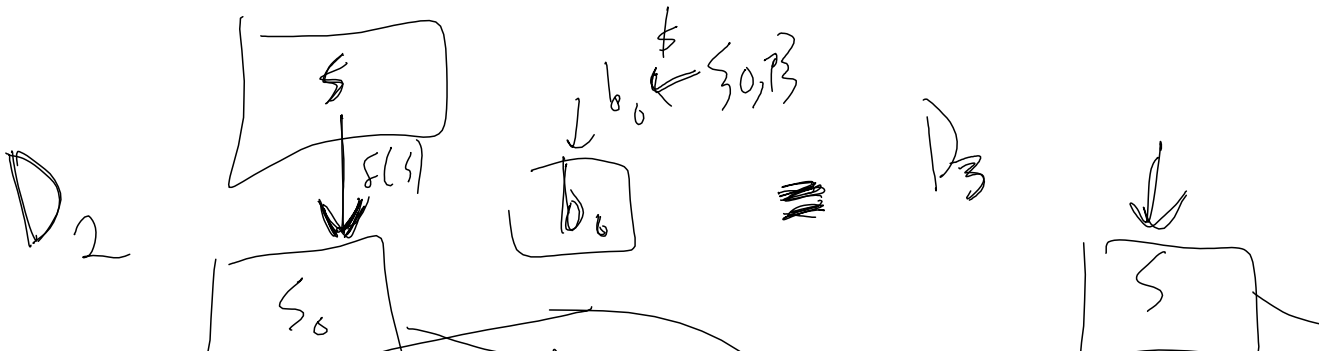
Proof: Suppose $\mathcal{A}(y, b_0)$ distingui
$\mathcal{A}(y, h(s))$

Construct $\mathcal{A}'$ where $\mathcal{A}'(y)$ gives
with prob better than $1/2$

$\mathcal{A}'(y)$ runs $\mathcal{A}(y, 0)$
$\mathcal{A}(y, 1)$

First run $\mathcal{A}$ on many inputs to determine its bias.

Note: finish more carefully



Hybrid Lemma: $\{a\}_\lambda \approx_c \{b\}_\lambda$ $\{b\}_\lambda$

$\Longrightarrow$ $\{a\}_\lambda \approx_c \{c\}$

From PRG to PRF:

PRG: $f: D_\lambda \to C_\lambda$ apparently random

PRF: $f: D_\lambda \times I_\lambda \to C_\lambda$ apparently random

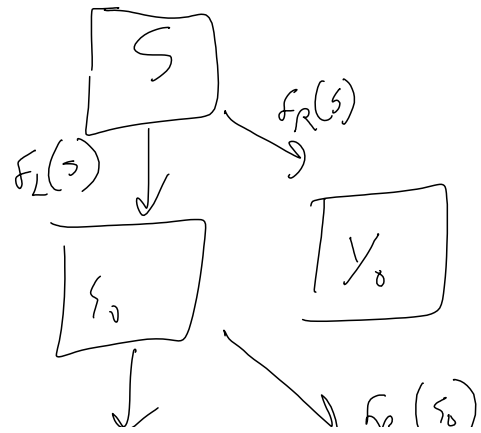
random seed

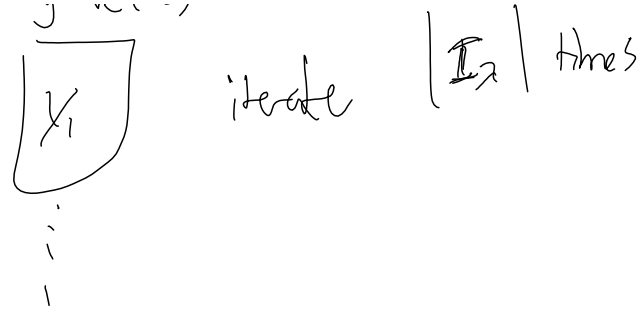
adversarially chosen

Assume we have a length-doubling PRG

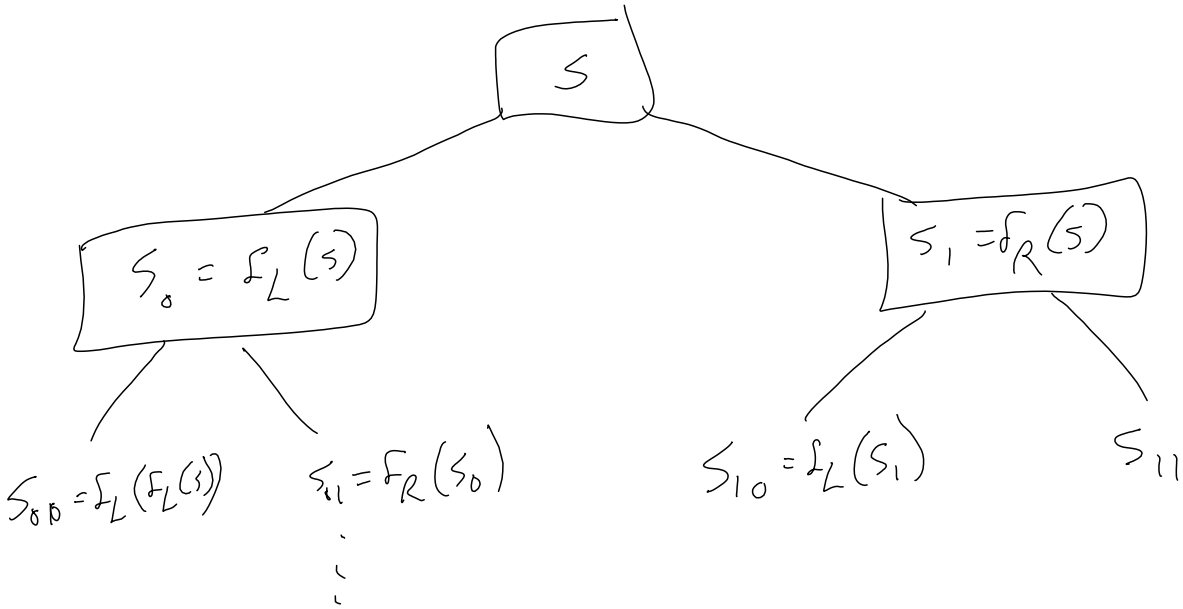
Naive approach: $f: \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$

$f'(s, x) =$





GGM - prf construction
tree-based construction



$s_{101000\ldots} = \ldots f_L f_L f_R f_L f_R(s)$

Proof goes layer by layer, applying hybrid lemma at each step.

Practically, we use as a prg or prf, an encryption scheme like AES

or index → AES → pseudorandom string

really efficient

structured algebraic crypto g^x dlog x^e mod n RSA factoring proven by reductions	usually more expensive	product $g^{ab} = A^b$ more functionality, zero knowledge proofs, relations between ciphertexts
symmetric crypto unstructured P SHA-1, SHA-2 AES	usually efficient on processors	only good for prfs and symmetric