EECS 575: Advanced Cryptography
Fall 2021
Lecture 9

Paul Grubbs

paulgrub@umich.edu

Beyster 4709

Agenda for this lecture

- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

Agenda for this lecture

- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

Announcements

- Homework 3 is due 10/4.
  - Take-home exam 1 will be released that day!

Exam will cover everything in lectures 1-9

Agenda for this lecture

- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

Recap from last time

PRF family is a distr. over "uniform-looking" f'ns

PRF Family: $F_S : \{0,1\}^{\ell_1(n)} \to \{0,1\}^{\ell_2(n)}$ s.t.

PRF family if:

* $\exists F$, deterministic poly-time, $F(s,x) = F_s(x)$ $\forall s, x$
* Pseudorandom $S \leftarrow \{0,1\}^n$

$$\left| \Pr_{s \in \{0,1\}^n}[D^{f_s}(1^n) = 1] - \Pr_{f \leftarrow V_{\ell_1(n) \to \ell_2(n)}}[D^{f}(1^n) = 1] \right| = \text{negl}(n)$$

Agenda for this lecture

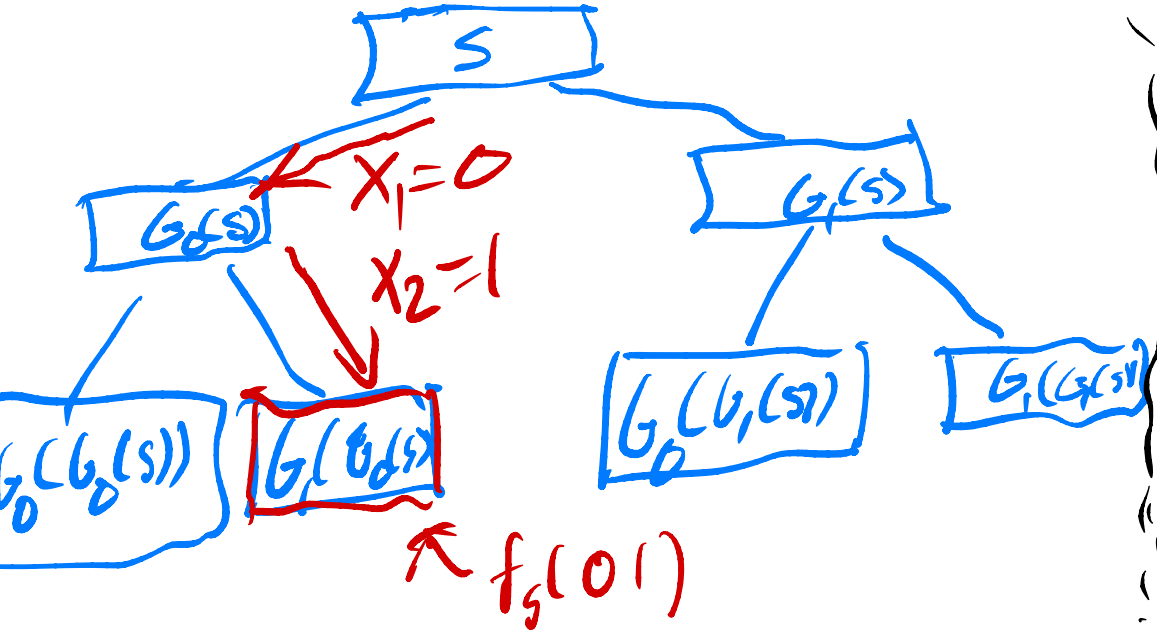
- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

PRF from a length-doubling PRG [GGM84]

Theorem: Let G be a length-doubling PRG, then F is a PRF Family.

Recall $G(s) = G_0(s) \| G_1(s)$! $\ell_1(n) = \ell_2(n) = n$

Example: $F_s(01)$



$F_s(x_1 \cdots x_n) =$

$$G_{x_n}(\cdots G_{x_2}(G_{x_1}(s)))$$

Path in the tree

Analysis of GGM

Theorem: Let G be a length-doubling PRG, then F is a PRF Family.

Proof idea: Hybrid argument. Each hybrid replaces a level of tree with random strings.

* H_0: F_S
* H_i: Have 2^i seeds $S = \{S_y\}_{y \in \{0,1\}^i} \leftarrow U_n$

$$F_S(X_1 \cdots X_n) = G_{X_n}(\cdots G_{X_{i+1}}(S_{X_i \cdots X_1}) \cdots)$$

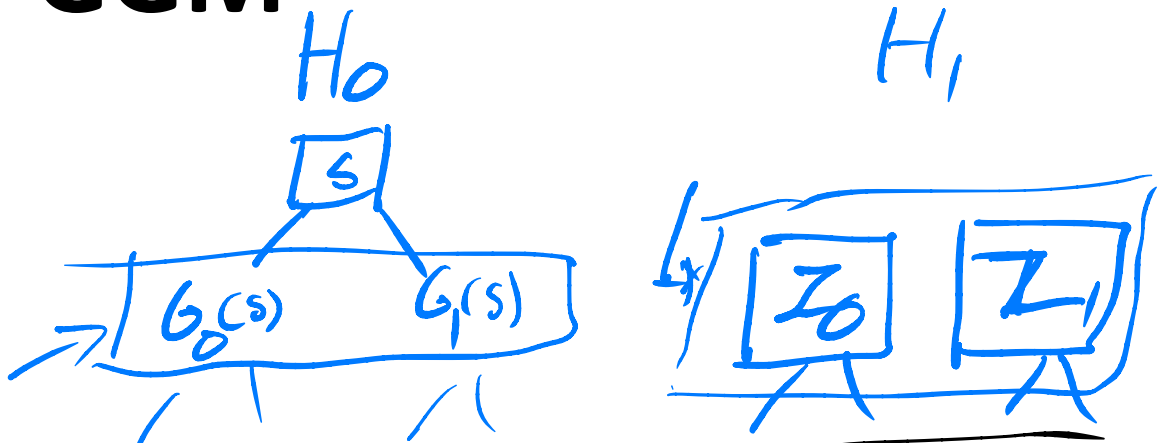
Observe H_n is a random function

2^n seeds

each is output of function

Analysis of GGM

Lemma: $H_0 \approx_c H_1$



Proof: Simulator S

$S(z_0, z_1)$:

On input $x_1 \cdots x_n$,

output $G_{x_n}(\cdots G_{x_2}(z_{x_1})\cdots)$

$x_n \cdots x_2$: tree path; z_{x_1}: input

$$\left| \Pr_{z \leftarrow G(U_n)}[D^{S(z_0,z_1)}(1^n) = 1] - \Pr_{z_0,z_1 \leftarrow U_n}[D^{S(z_0,z_1)}(1^n) = 1] \right|$$

If S' input is $G(U_n)$, S simulating H_0; if it's U_{2n}, simulating H_1.

Because S is efficient, $H_0 \approx_c H_1$ because G is PRG

Analysis of GGM

Problem: Hybrid can only take poly(n) strings

Fix: Each hybrid gets only poly-many random strings.

works because # oracle queries D asks is polynomial in n.

Analysis of GGM

$S_i((z_0^1, z_1^1) \cdots (z_0^q, z_1^q))$: ← input: $q = \text{poly}(n)$ n-bit strings

$j \leftarrow 1$; $T \leftarrow [\,]$

to answer query $x_1 \cdots x_n$: ← D's inputs to its oracle calls

If $x_1 \cdots x_i$ has not been seen

$T[x_1 \cdots x_i] := (z_0^j, z_1^j)$ ← ensures consistency across oracle calls

$j = j + 1$

$z_0, z_1 \leftarrow T[x_1 \cdots x_i]$

Return $G_{x_n}(\cdots G_{x_{i+1}}(z_{x_i}) \cdots)$

Claim: If S_i's inputs are PRG outputs, S_i sims H_{i-1}; S_i's inputs random, S_i sims H_i.

Analysis of GGM

S_i gets access to either poly-many PRG outputs, or poly-many random strings

Need to show, $\text{Poly}(n) = q$,

$$(G(s_1), \ldots, G(s_q)) \approx_c (U_{2n}, \ldots, U_{2n})$$

Exercise [Use hybrid argument]

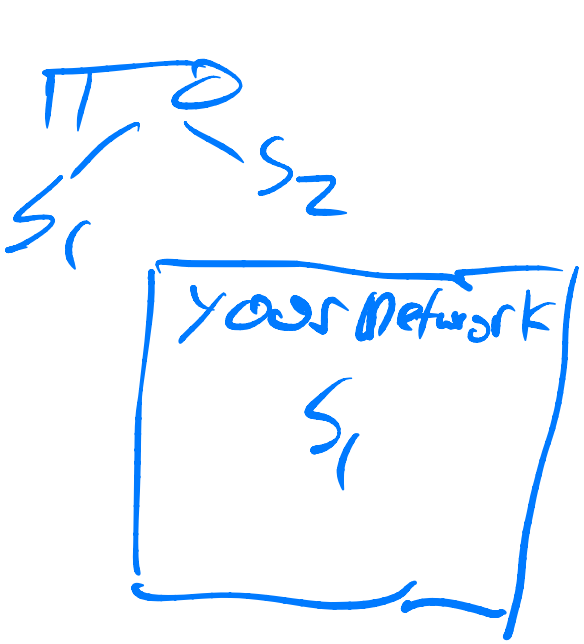
Analysis of GGM

Agenda for this lecture

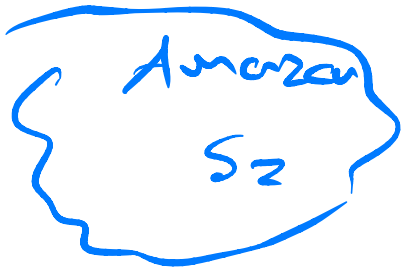
- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

GGM is "puncturable"

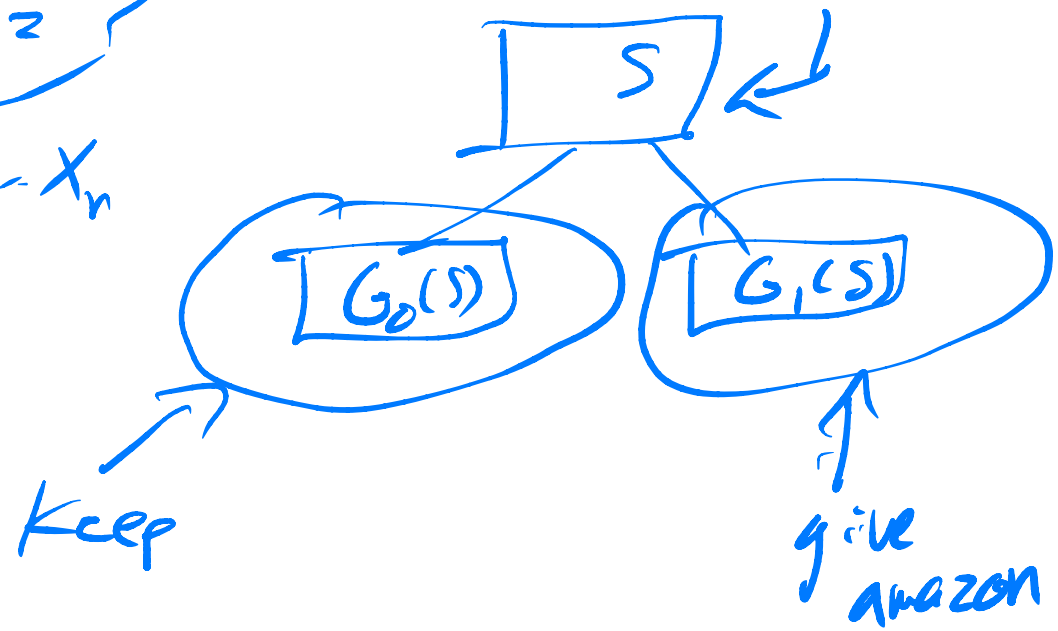
Need to split PRF key into pieces



evaluates $0x_2 \cdots x_n$



$1x_2 \cdots x_n$



To amazon, output of PRF on $0x_2 \cdots x_n$ is still pseudorandom!!

Agenda for this lecture

- Announcements
- Recap from last time
- PRF from a length-doubling PRG (“GGM”)
  - Analysis of GGM
- GGM is ”puncturable”
- Pseudorandom permutations (PRPs)

Pseudorandom Permutations (PRPs)