

COS433/Math 473: Cryptography

Mark Zhandry

Princeton University

Spring 2017

Last Time

Hardcore Bits

Hardcore Bits

Let \mathbf{F} be a one-way function with domain \mathbf{x} , range \mathbf{y}

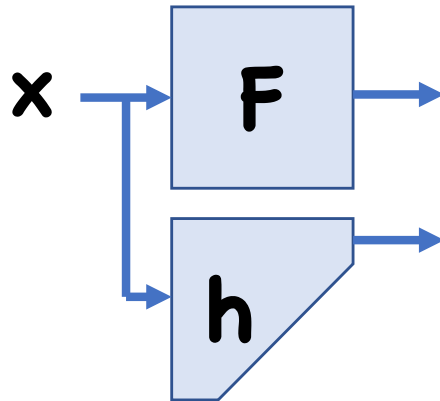
Definition: A function $\mathbf{h}:\mathbf{x}\rightarrow\{0,1\}$ is a “hardcore bit” for \mathbf{F} if the following two distributions are computationally indistinguishable:

- $(\mathbf{F}(\mathbf{x}), \mathbf{h}(\mathbf{x}))$ for a random \mathbf{x}
- $(\mathbf{F}(\mathbf{x}), \mathbf{b})$ for a random \mathbf{x}, \mathbf{b}

In other words, even given $\mathbf{F}(\mathbf{x})$, hard to guess $\mathbf{h}(\mathbf{x})$

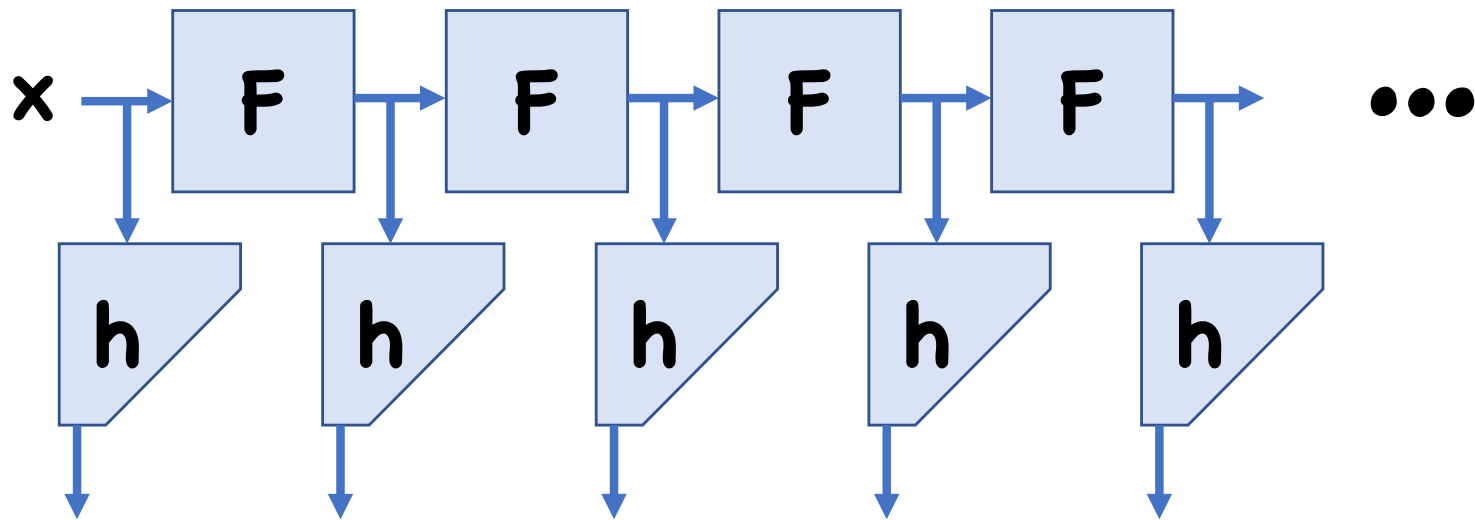
Application: PRGs

Let \mathbf{F} be a one-way permutation with hardcore bit \mathbf{h}



Theorem: If \mathbf{h} is a hc bit for \mathbf{F} and \mathbf{F} is a OWP, then $G(x) = (\mathbf{F}(x), \mathbf{h}(x))$ is a secure PRG

Application: PRGs



Theorem: If h is a hc bit for F and F is a OWP, then $G(x) = (h(x), h(F(x)), h(F(F(x))), \dots)$ is a secure PRG

Examples of Hardcore Bits

Define **lsb(x)** as the least significant bit of **x**

For **x** \in **Z_N**, define **Half(x)** as **1** iff **0** \leq **x** $<$ **N/2**

Theorem: Let p be a prime, and $F: \mathbb{Z}_p^* \rightarrow \mathbb{Z}_p^*$ be $F(x) = g^x \bmod p$, for some generator g

Half is a hardcore bit for F (assume F is one-way)

Theorem: Let N be a product of two large primes p, q , and $F: \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$ be $F(x) = x^e \bmod N$ for some e relatively prime to $(p-1)(q-1)$

Lsb and Half are hardcore bits for F (assuming RSA)

Theorem: Let N be a product of two large primes p, q , and $F: \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$ be $F(x) = x^2 \bmod N$

Lsb and Half are hardcore bits for F (assuming factoring)

Random Self Reduction

Suppose given Dlog instance $y=g^x$

Have adversary that works for random Dlog instances

- May not work for my particular instance

Nonetheless, want to use adversary to solve my instance

Random Self Reduction

Goal: randomize procedure that takes $y \rightarrow y'$

- From solution to y' , can compute solution to y
- y' is uniformly random

Dlog random self reduction:

- Choose random z
- Let $y' \leftarrow y \times g^z$
- Run adversary on y' to get Dlog x'
- $x = x' - z$

Today

Constructing PRPs with hardcore bits

Basing cryptography on on-way functions

Yao's Method

Let \mathbf{F} be a OWF with domain $\{0,1\}^n$

Claim: $\exists i$ such that \forall PPT \mathbf{A}
 $\Pr[\mathbf{A}(\mathbf{F}(\mathbf{x})) = x_i] < 1 - 1/2n$

Proof: otherwise, $\forall i, \exists \mathbf{A}_i$ s.t.
 $\Pr[\mathbf{A}_i(\mathbf{F}(\mathbf{x})) = x_i] \geq 1 - 1/2n$

Adversary $\mathbf{A}(\mathbf{y}) = \mathbf{A}_1(\mathbf{y}) \parallel \mathbf{A}_2(\mathbf{y}) \parallel \dots$
 $\Pr[\mathbf{A}(\mathbf{F}(\mathbf{x})) = \mathbf{x}] \geq 1/2$

Yao's Method

Let \mathbf{F} be a OWF with domain $\{0,1\}^n$

Claim: $\exists i$ such that \forall PPT \mathbf{A}
 $\Pr[\mathbf{A}(\mathbf{F}(\mathbf{x})) = x_i] < 1 - 1/2n$

Let $\mathbf{F}'(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}) = (\mathbf{F}(\mathbf{x}^{(1)}), \dots, \mathbf{F}(\mathbf{x}^{(t)}))$
 $\mathbf{h}(\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}) = x^{(1)}_i \oplus x^{(2)}_i \oplus \dots \oplus x^{(t)}_i$

Yao's XOR lemma $\Rightarrow \mathbf{h}$ is hardcore for \mathbf{F}'

Goldreich Levin

Let \mathbf{F} be a OWF with domain $\{0,1\}^n$ and range Y

Let $\mathbf{F}':\{0,1\}^{2n} \rightarrow \{0,1\}^n \times Y$ be:

$$\mathbf{F}'(r,x) = r, \mathbf{F}(x)$$

Define $\mathbf{h}(r,x) = \langle r,x \rangle = \sum r_i x_i \pmod 2$


Theorem (Goldreich-Levin): If \mathbf{F} is one-way, then \mathbf{h} is a hc bit for \mathbf{F}'

Theorem (Goldreich-Levin): If F is one-way, then h is a hc bit for F'


Proof Sketch:


First attempt: suppose  predicts $\langle x, r \rangle$ given $r, F(x)$ with certainty

Let $e_i = 0^{i-1}10^{n-i}$

Algorithm: $x_i \leftarrow$  $(e_i, F(x))$


Theorem (Goldreich-Levin): If F is one-way, then h is a hc bit for F'

Second attempt: suppose  predicts $\langle x, r \rangle$ given $r, F(x)$ with prob $3/4 + \epsilon$

Claim: For an $\epsilon/2$ fraction of x ,  predicts $\langle x, r \rangle$ given $r, F(x)$ for a random r with prob $3/4 + \epsilon/2$

Call such x “good”

For rest of proof, assume we are given a “good” x

For "good" \mathbf{x} ,  predicts $\langle \mathbf{x}, \mathbf{r} \rangle$ given $\mathbf{r}, F(\mathbf{x})$ for a random \mathbf{r} with prob $3/4 + \epsilon/2$


Want to perform $\mathbf{x}_i \leftarrow \text{robot}(\mathbf{e}_i, F(\mathbf{x}))$ attack like before

- Problem:  might not work on \mathbf{e}_i

Solution: Random Self Reduction

- Choose random \mathbf{r}
- $\mathbf{b}_0 \leftarrow \text{robot}(\mathbf{r}, F(\mathbf{x}))$
- $\mathbf{b}_1 \leftarrow \text{robot}(\mathbf{r} \oplus \mathbf{e}_i, F(\mathbf{x}))$
- $\Pr[\mathbf{x} = \mathbf{b}_0 \oplus \mathbf{b}_1] = 1/2 + \epsilon$
- Can increase accuracy by repeating multiple times

Theorem (Goldreich-Levin): If \mathbf{F} is one-way, then \mathbf{h} is a hc bit for \mathbf{F}'

Second attempt: suppose  predicts $\langle \mathbf{x}, \mathbf{r} \rangle$ given $\mathbf{r}, \mathbf{F}(\mathbf{x})$ with prob $1/2 + \epsilon$

Can similarly define “good” \mathbf{x}

Additional ideas required to get inverter

Summary

A hc bit for any OWF

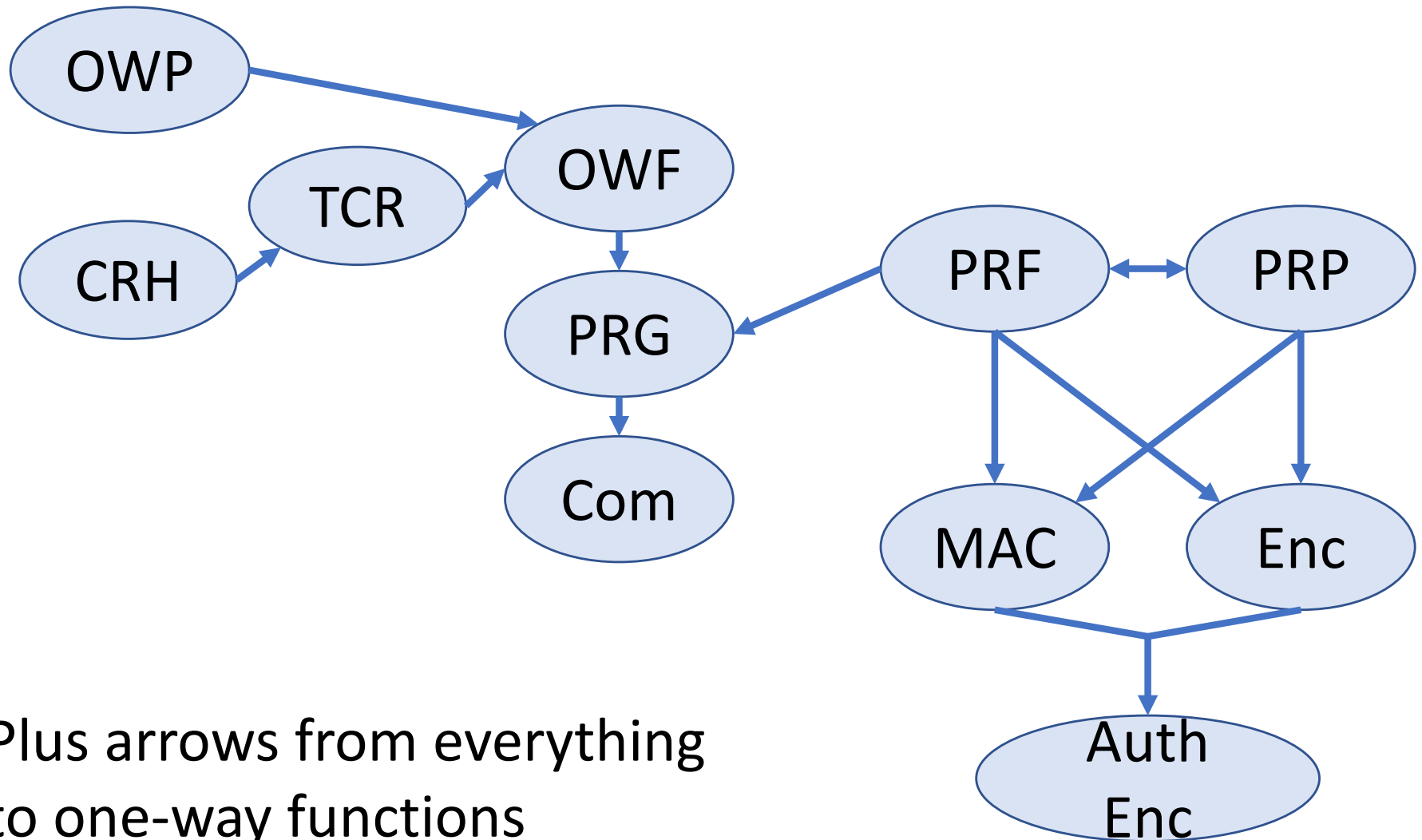
Implies PRG from any OWP

- PRG from Dlog (Blum-Micali)
- PRG from Factoring
- PRG from RSA

Actually, can construct PRG from any OWF

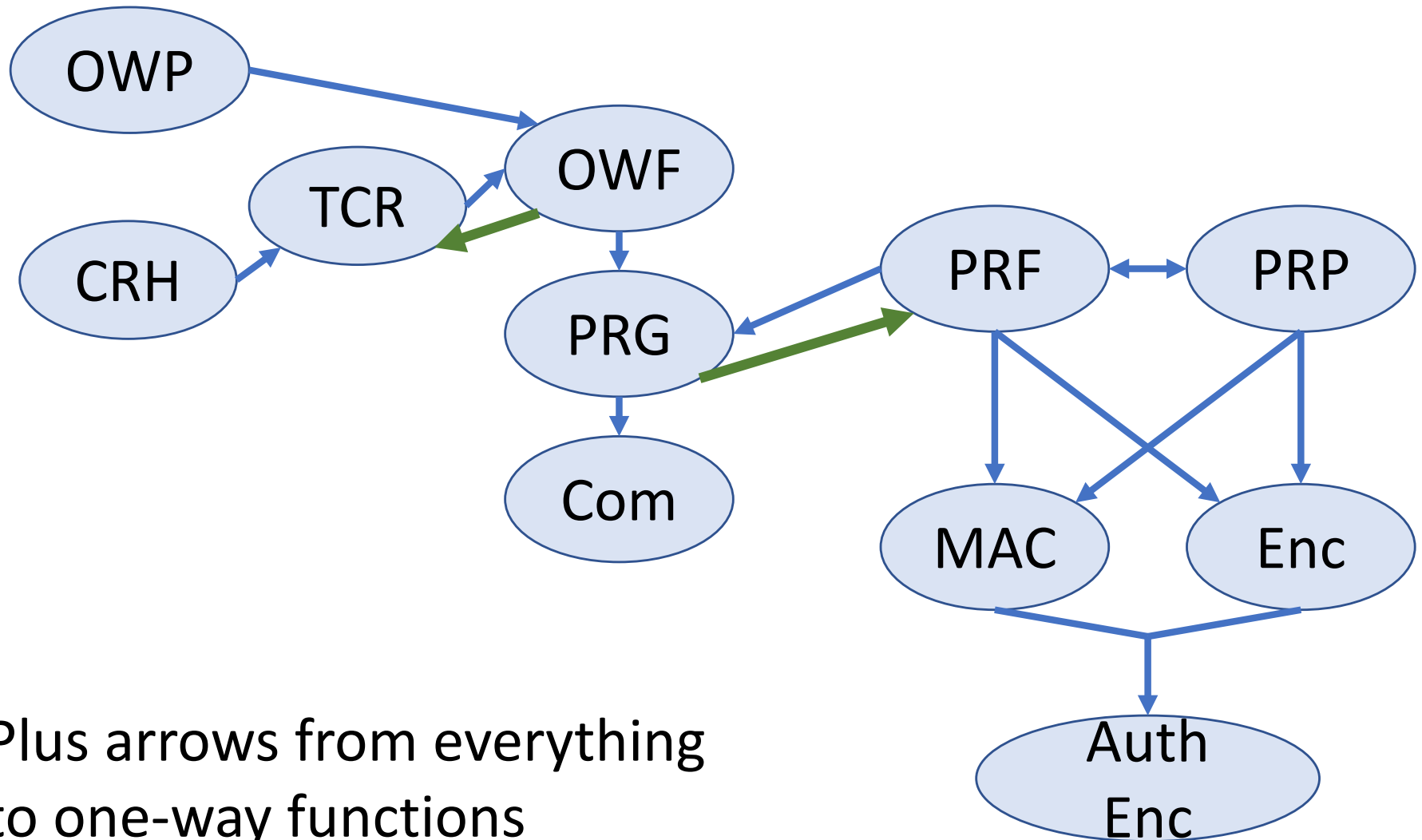
- Proof beyond scope of course

So Far



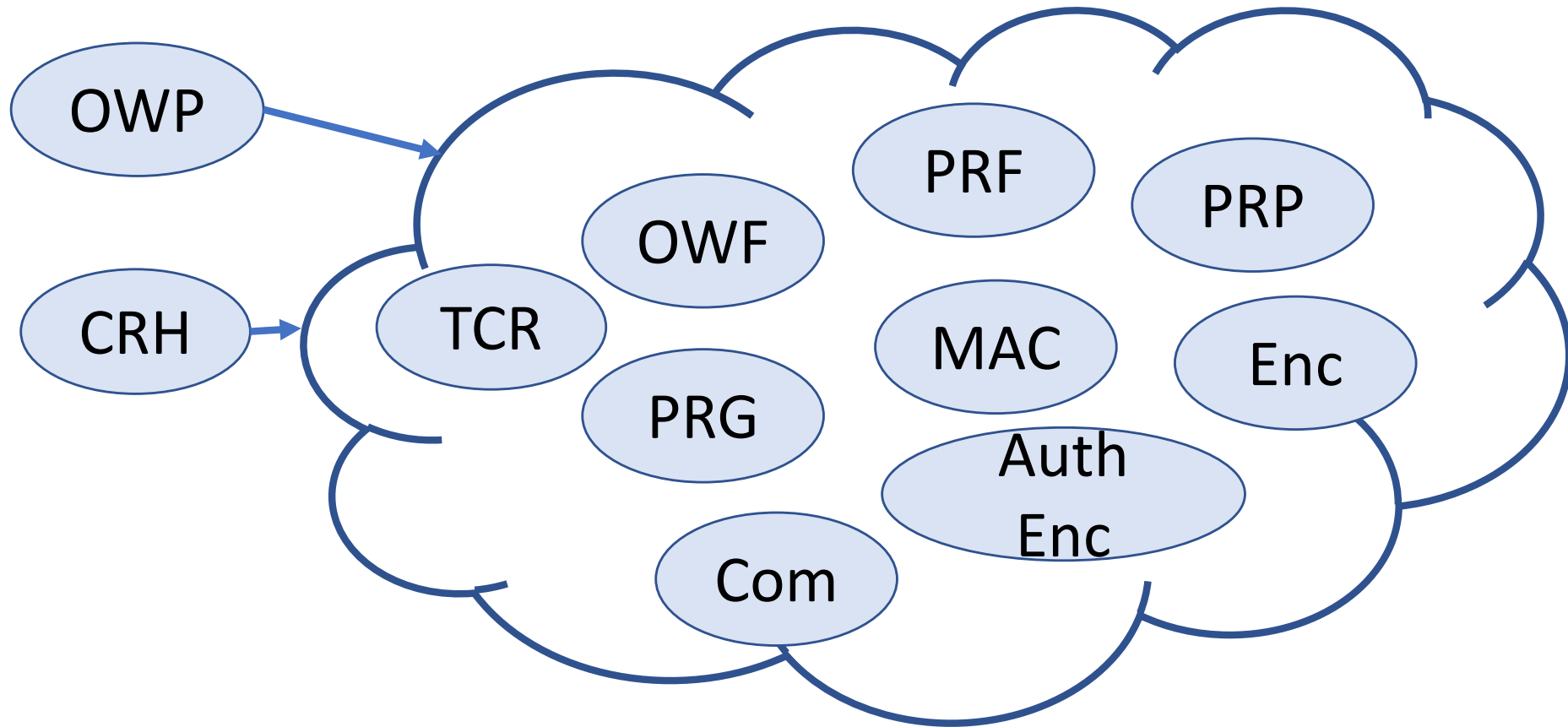
Plus arrows from everything to one-way functions

What's Known



Plus arrows from everything to one-way functions

What's Known

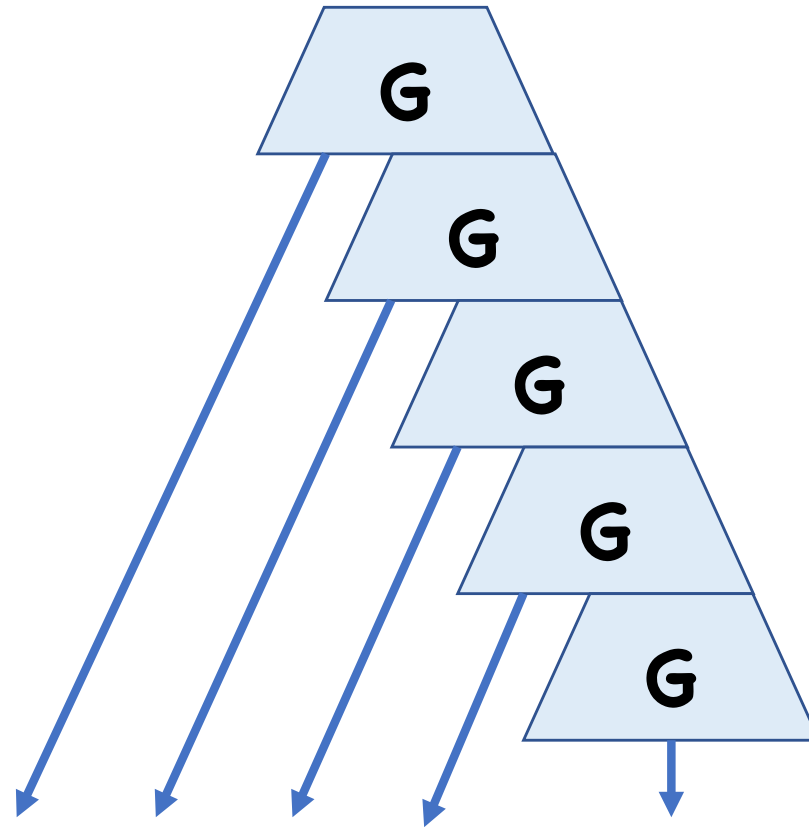


PRGs \rightarrow PRFs

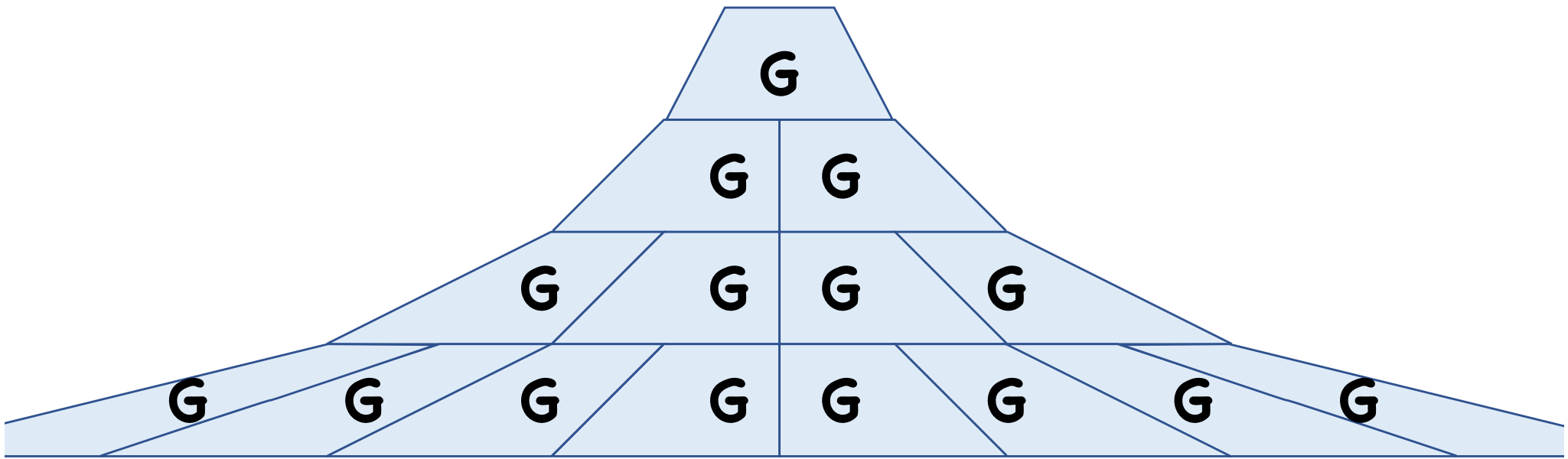
Today, we will show how to construct PRFs from PRGs

(Target collision resistance from one-way functions beyond scope of course)

First: Expanding Length of PRGs



A Different Approach



Advantage of Tree-based Approach

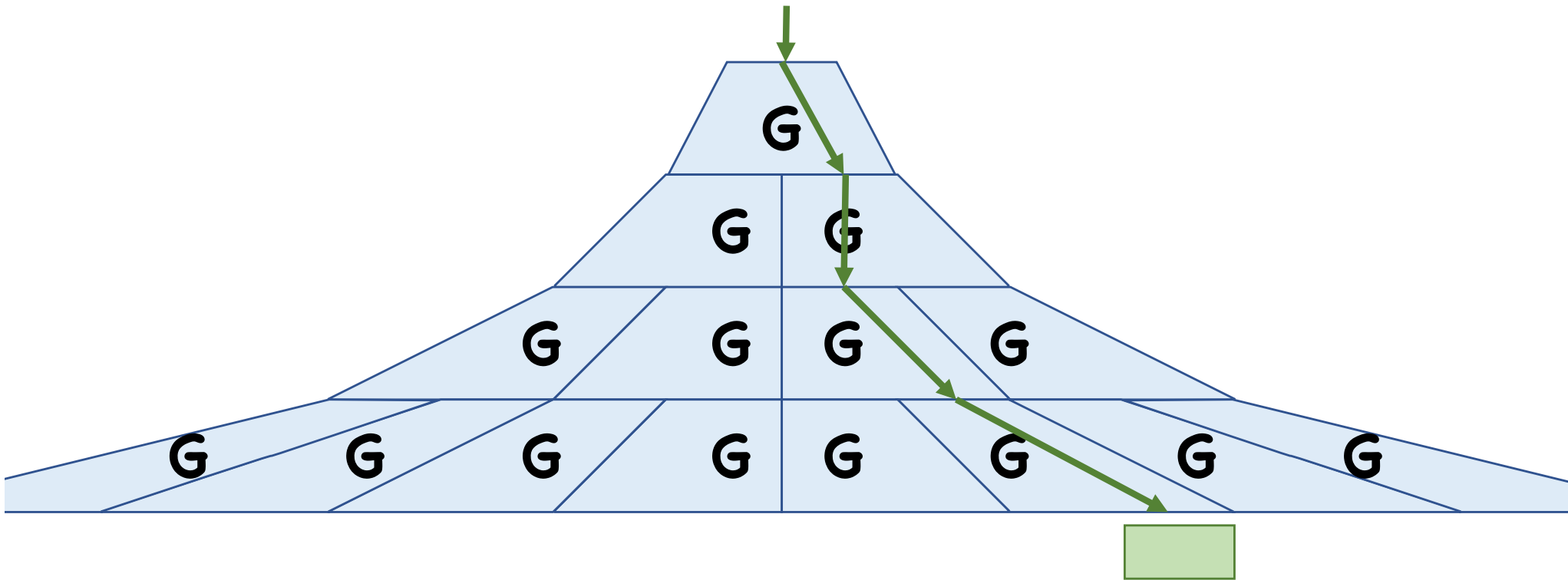
To expand λ bits into $2^h \lambda$ bits, need h levels

Can compute output locally:

- To compute i th chunk of λ bits, only need h PRG evaluations

In other words, can locally compute in logarithmic time

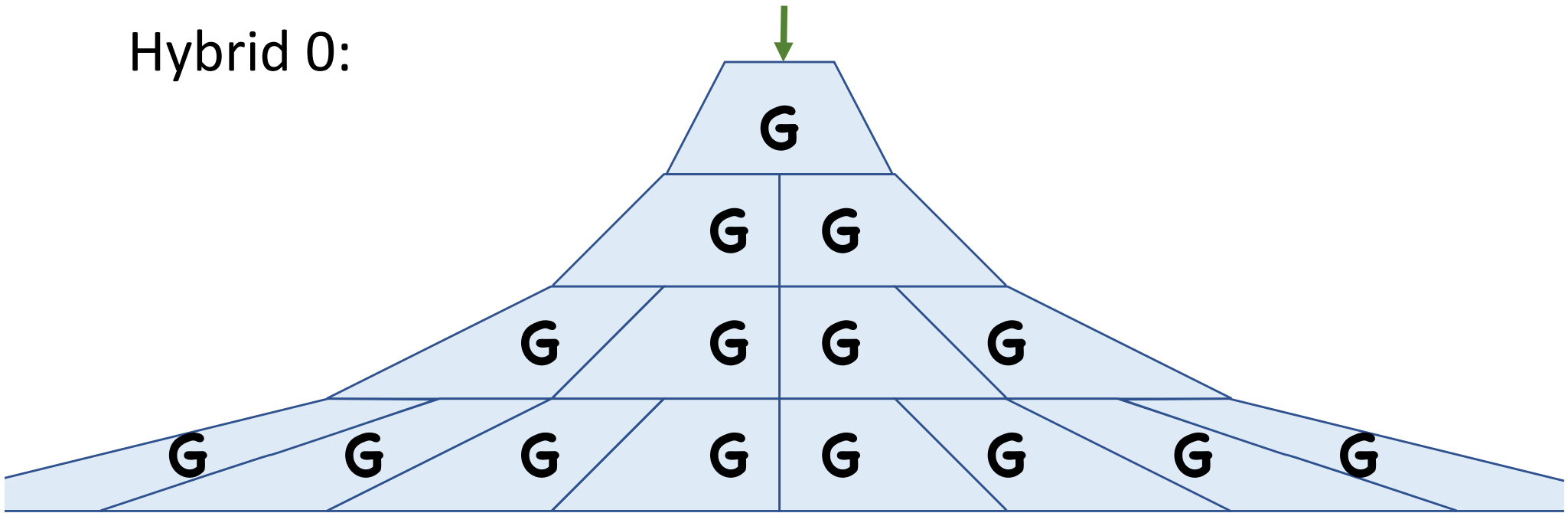
Advantage of Tree-based Approach



Theorem: For any logarithmic h , if \mathbf{G} is a secure PRG, then so is the tree-based PRG

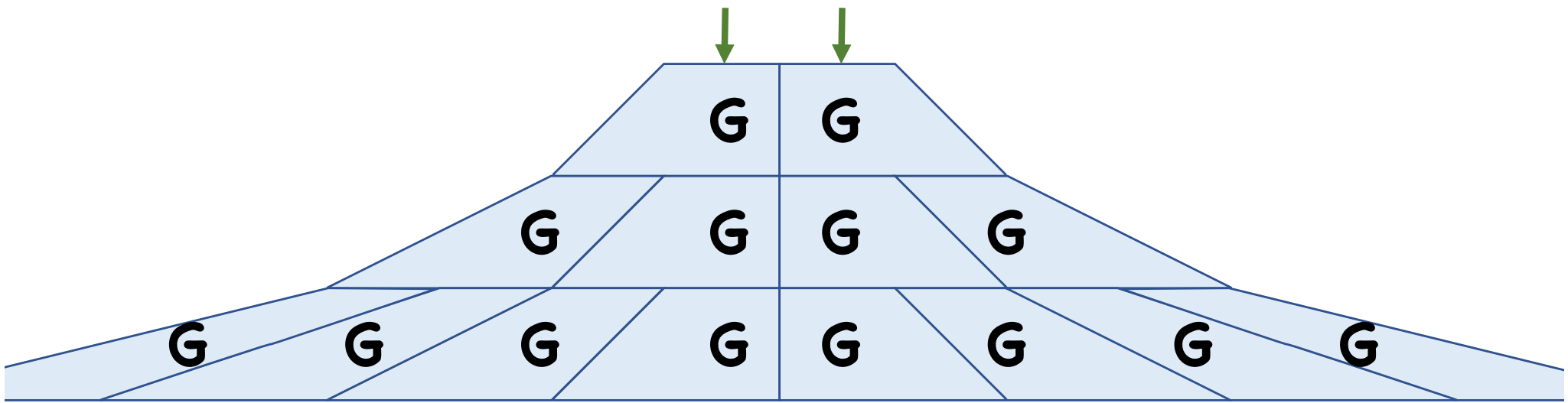
Proof

Hybrid 0:



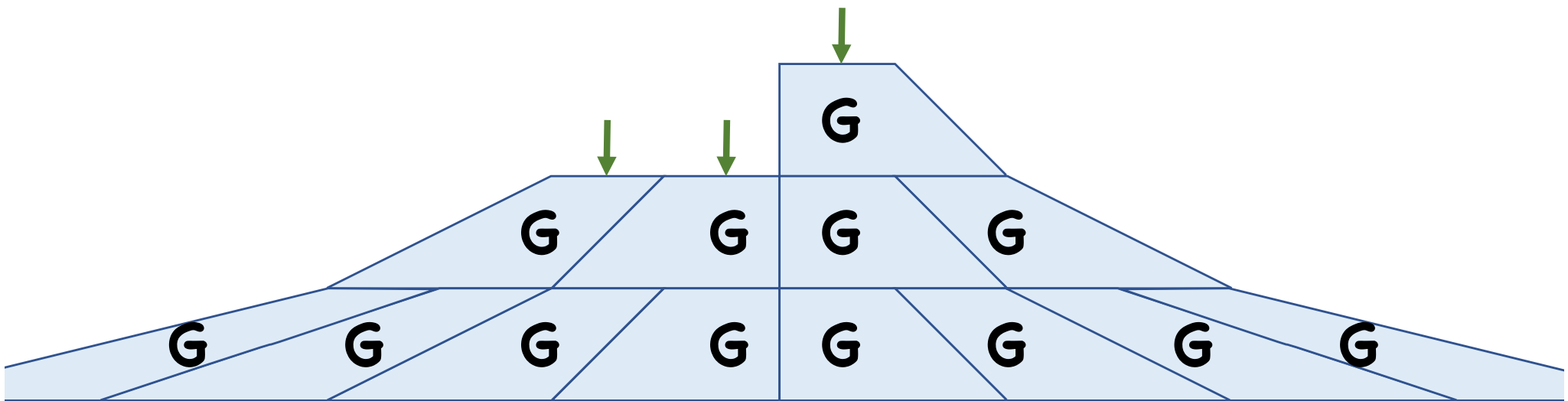
Proof

Hybrid 1:



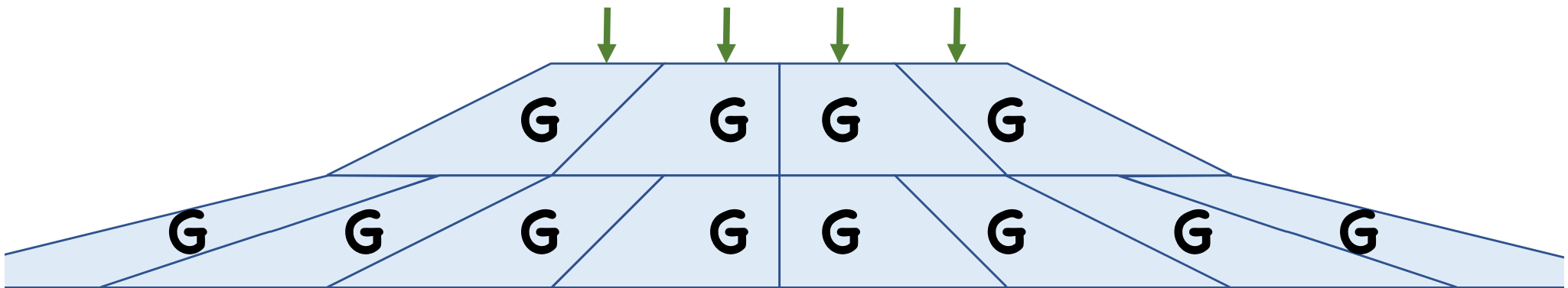
Proof

Hybrid 2:



Proof

Hybrid 3:



Proof

Hybrid **t**:



Proof

What is t in terms of h ?

PRG adversary distinguishes Hybrid 0 from Hybrid t with advantage ϵ

- $\exists i$ such that adversary distinguishes Hybrid $i-1$ from Hybrid i with advantage ϵ/t
- Can use to construct adversary for G with advantage ϵ/t

A PRF

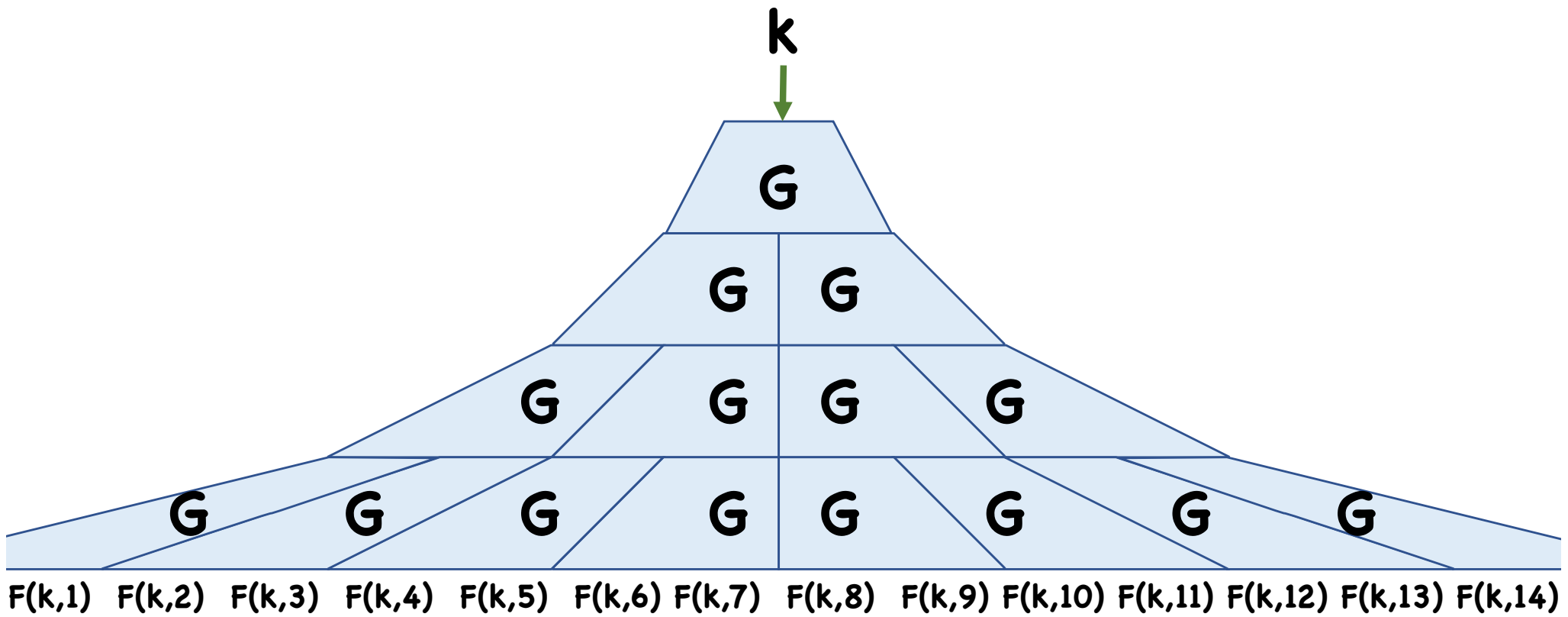
Domain $\{0,1\}^n$

Set $h = n$

$F(k, x)$ is the x th block of λ bits

- Computation involves h evals of G , so efficient

A PRF

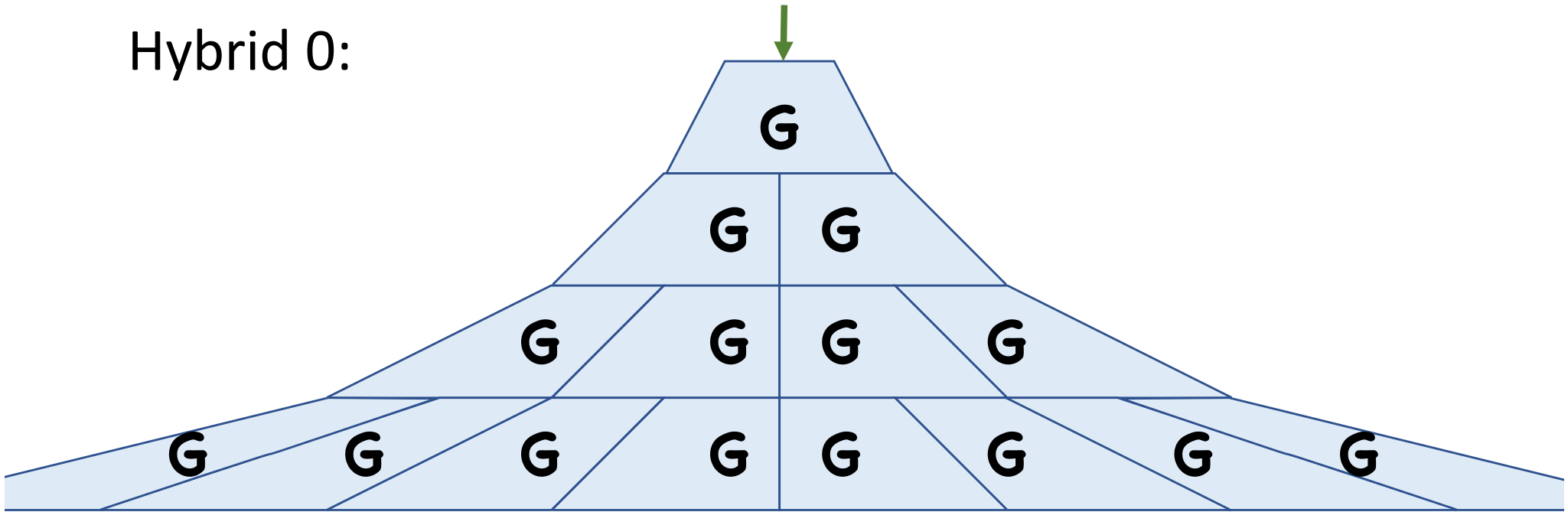


Problem with Security Proof

Suppose I have a PRF adversary with advantage ϵ . In the proof, what is the advantage of the derived PRG adversary?

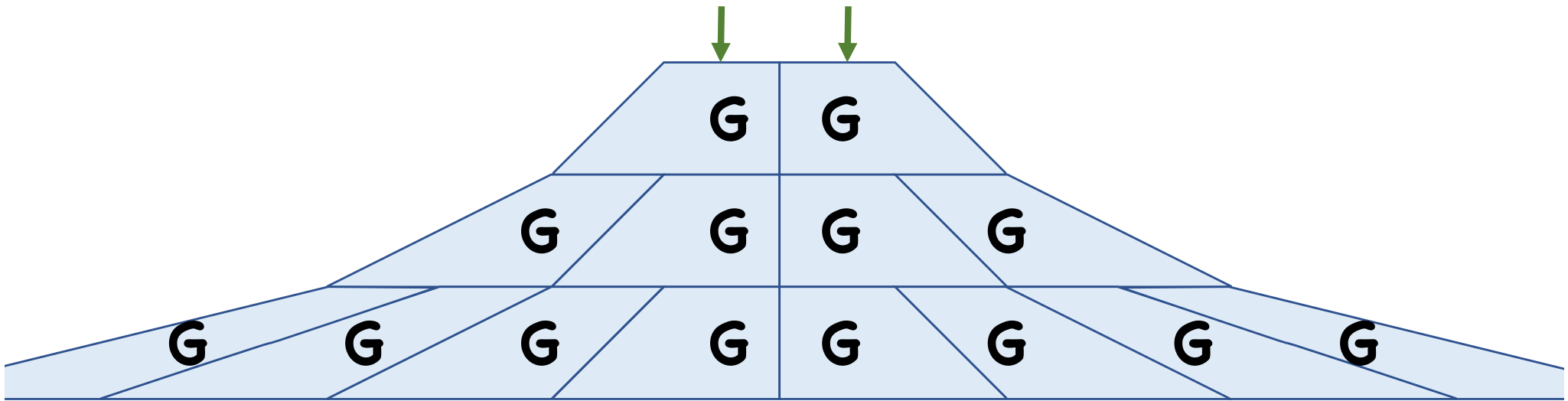
A Better Proof

Hybrid 0:



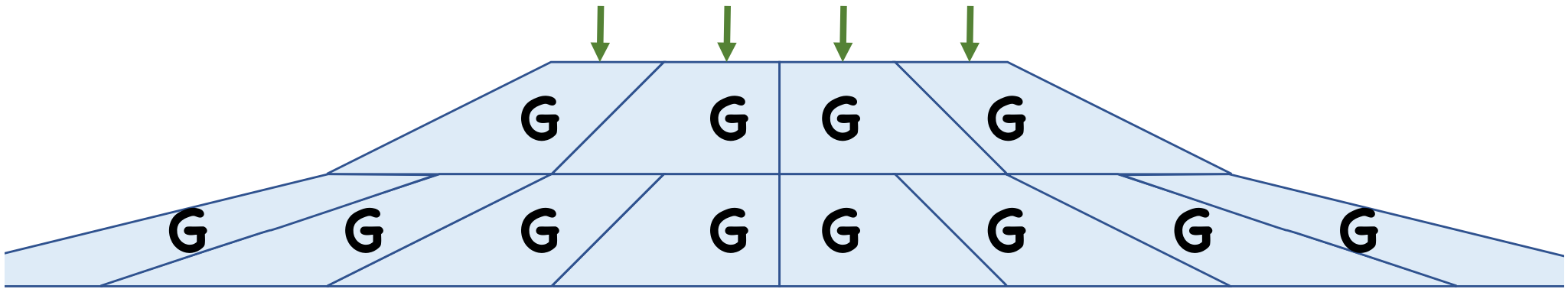
A Better Proof

Hybrid 1:



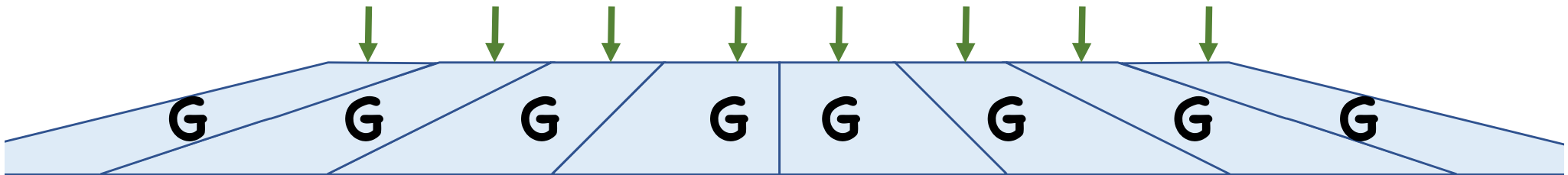
A Better Proof

Hybrid 2:



A Better Proof

Hybrid 3:



A Better Proof

Hybrid **$h=n$** :



A Better Proof

Now if PRF adversary distinguishes Hybrid 0 from Hybrid $h=n$ with advantage ϵ , $\exists i$ such that adversary distinguishes Hybrid $i-1$ from Hybrid i with advantage ϵ/n

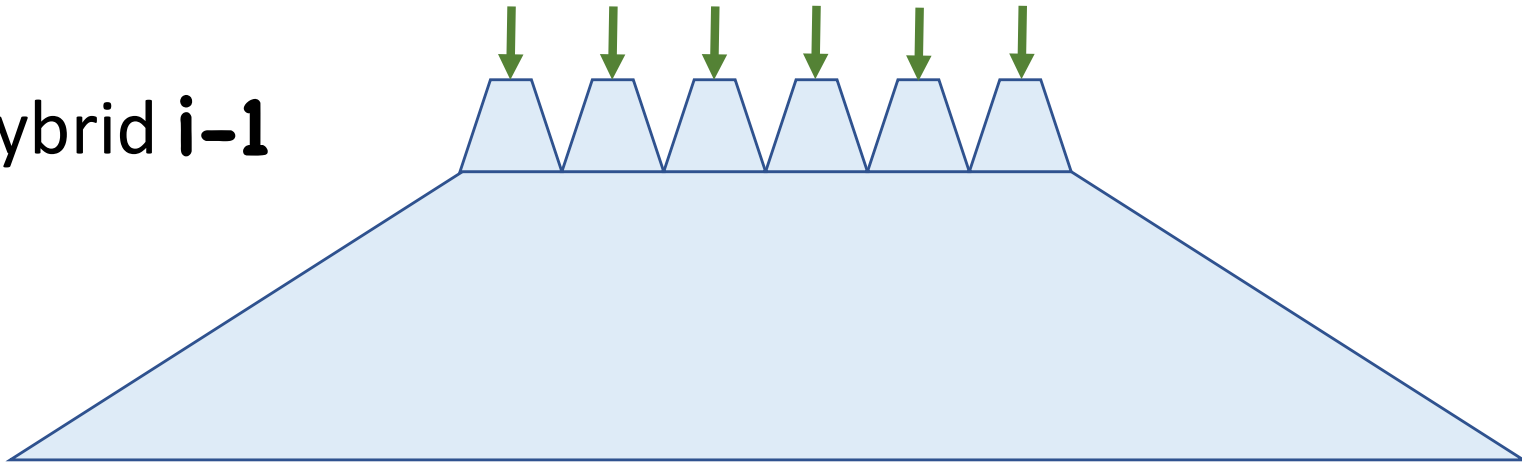
- Non-negligible advantage

Not quite done: Distinguishing Hybrid $i-1$ from Hybrid i does not immediately give a PRG distinguisher

- Exponentially many PRG values changed!

A Better Proof

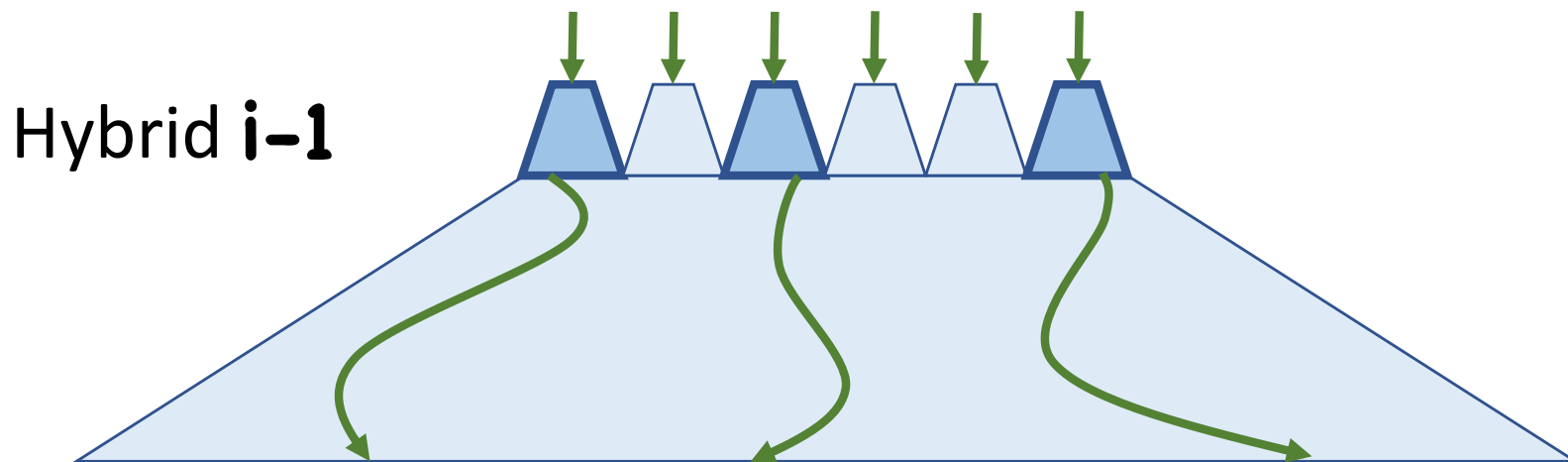
Hybrid $i-1$



Hybrid i



Key Observation:



Adversary only queries polynomially many outputs
 \Rightarrow Only need to worry about polynomially many PRG instances in level i

A Better Proof

More Formally:

Given distinguisher **A** for Hybrid **i-1** and Hybrid **i**, can construct distinguisher **B** for the following two oracles from $\{0,1\}^{i-1} \rightarrow \{0,1\}^{2\lambda}$

- **H₀**: each output is a fresh random PRG sample
- **H₁**: each output is uniformly random

If **A** makes **q** queries, **B** makes at most **q** queries

A Better Proof

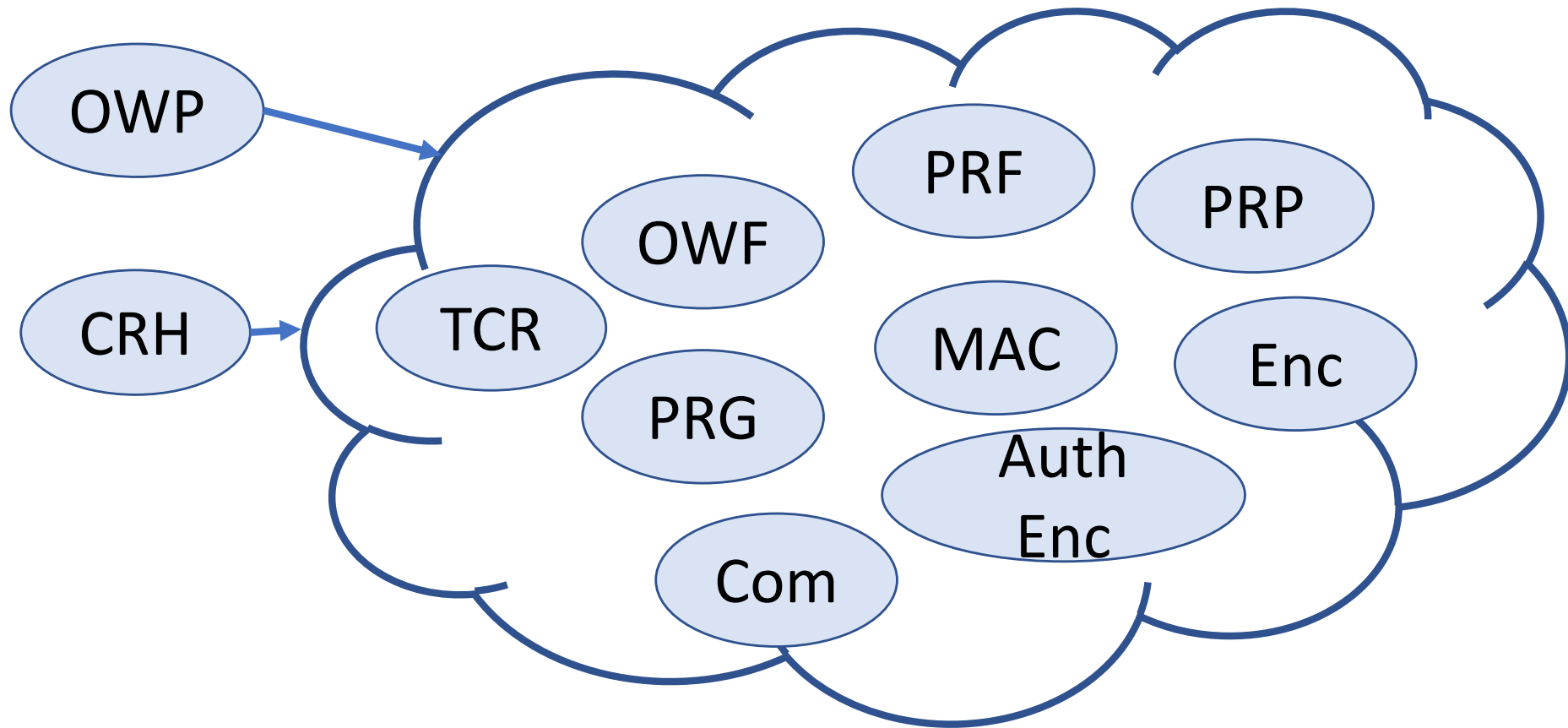
Now we have a distinguisher B with advantage ϵ/n that sees at most q values, where either

- Each value is a random output of the PRG, or
- Each value is uniformly random

By introducing q hybrids, can construct a PRG distinguisher with advantage ϵ/qn

\Rightarrow non-negligible

What's Known



What about OWP,CRH?

Generally Believed That...

Cannot construct OWP from OWF

Cannot construct CRH from OWF

Cannot construct CRH from OWP

Cannot construct OWP from CRH

Black Box Separations

How do we argue that you cannot build collision resistance from one-way functions?

- We generally believe both exist!

Observation: most natural constructions treat underlying objects as black boxes (don't look at code, just input/output)

Maybe we can rule out such natural constructions

Black Box Separations

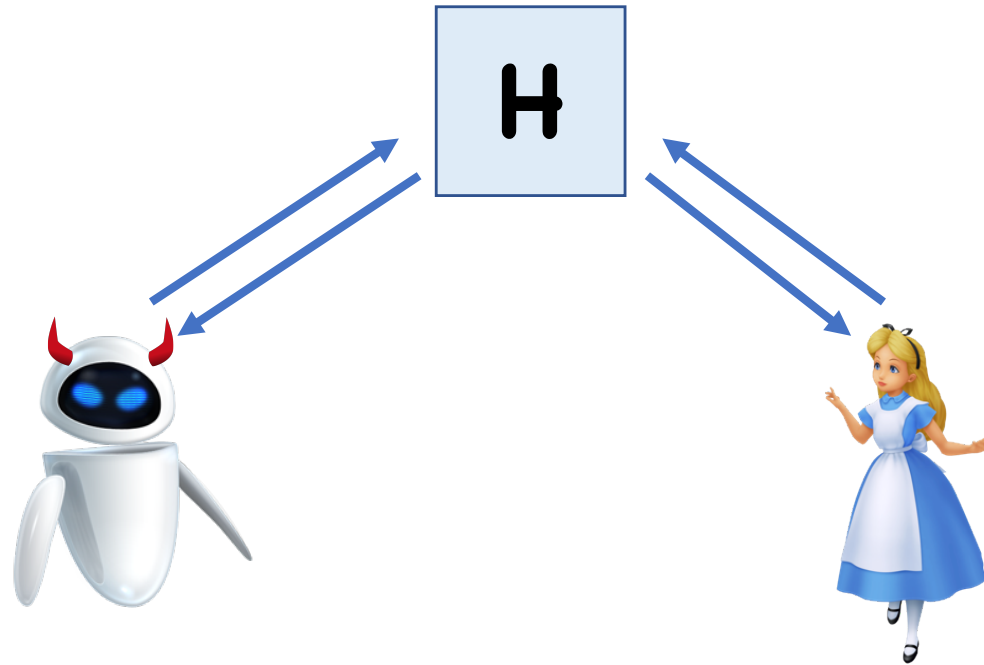
Present a world where one-way functions exist, but collision resistance does not

Hopefully, natural (black box) constructions make sense in this world

- Can construct PRGs, PRFs, PRPs, Auth-Enc, etc

Separating CRH from OWF

Starting point: random oracle model



Computation power is unlimited, but number of calls to random oracle is polynomial

Separating CRH from OWF

In ROM, despite unlimited computational power, one-way functions exist

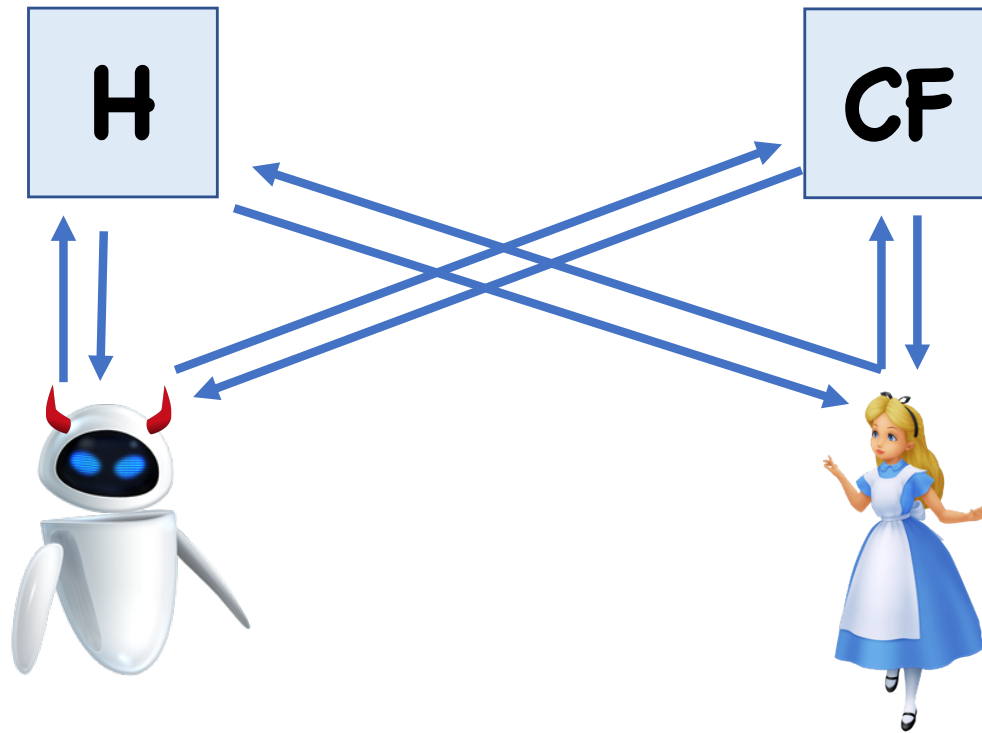
- **$F(x) = H(x)$**
- Can only invert oracle by making exponentially-many calls

Unfortunately, collision resistant hashing exists too!

- **$F(x) = H(x)$**

To fix, also add collision finding oracle

Separating CRH from OWF



Separating CRH from OWF

What does **CF** do?

- Takes as input a circuit **C**
- Circuit may have “oracle gates” that make calls to **H** or **CF**
- Outputs a collision for **C**

Impossibility of Collision Resistance?

- Consider BB construction of CRHF from OWF
- Replace calls to OWF with **H** queries
- Feed circuit computing CRHF to **CF** to find collision

Separating CRH from OWF

So we have a world in which collision resistance does not exist

However, maybe **CF** can be used to invert **H**

- So maybe one-way functions don't exist either

Must be careful in defining **CF**

- Random pair of colliding inputs will allow for inverting **H**

Separating CRH from OWF

Correct **CF**:

- Choose random input **x** to circuit
- Choose random input **y** that collides with **x**

Note that **x** will sometimes equal **y**. However, if circuit shrinks input, then with probability at least $\frac{1}{2}$ **x** \neq **y**

Careful analysis shows that **H** is still one-way

Next Time

Begin public key cryptography

Key agreement: how to exchange keys without ever meeting