CS 594 – Advanced Crypto (Spring 2026)

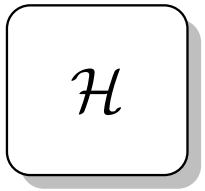
Alex Block

Lecture 4

January 28, 2026

Last Time: Random Oracle Model

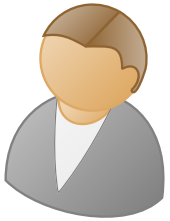
Last Time: Random Oracle Model



Last Time: Random Oracle Model

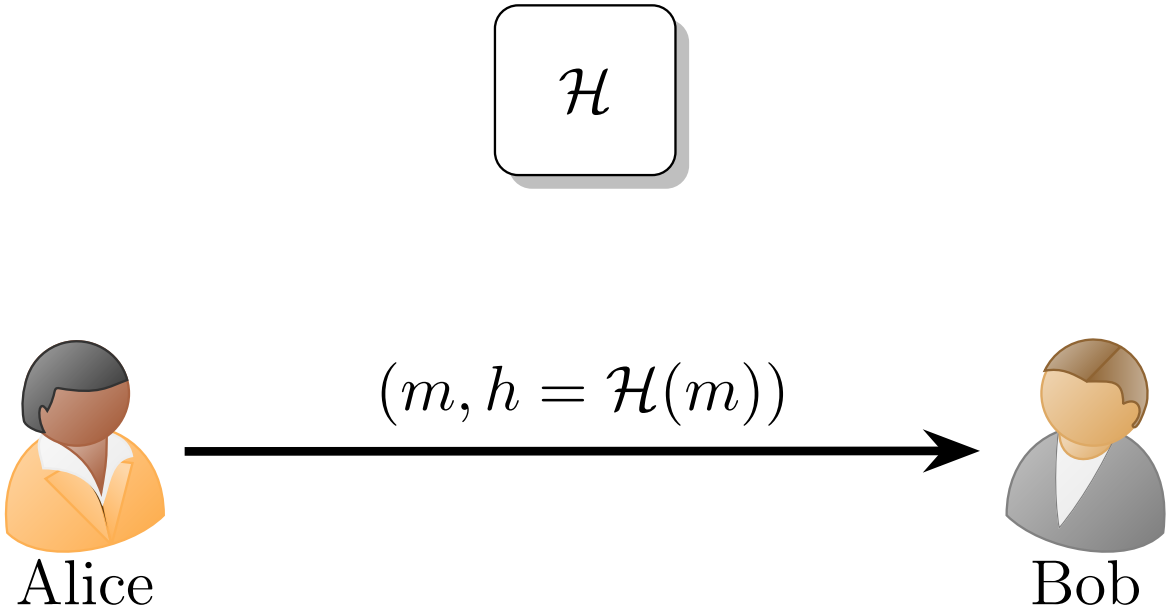
$\mathcal{H}$

Alice

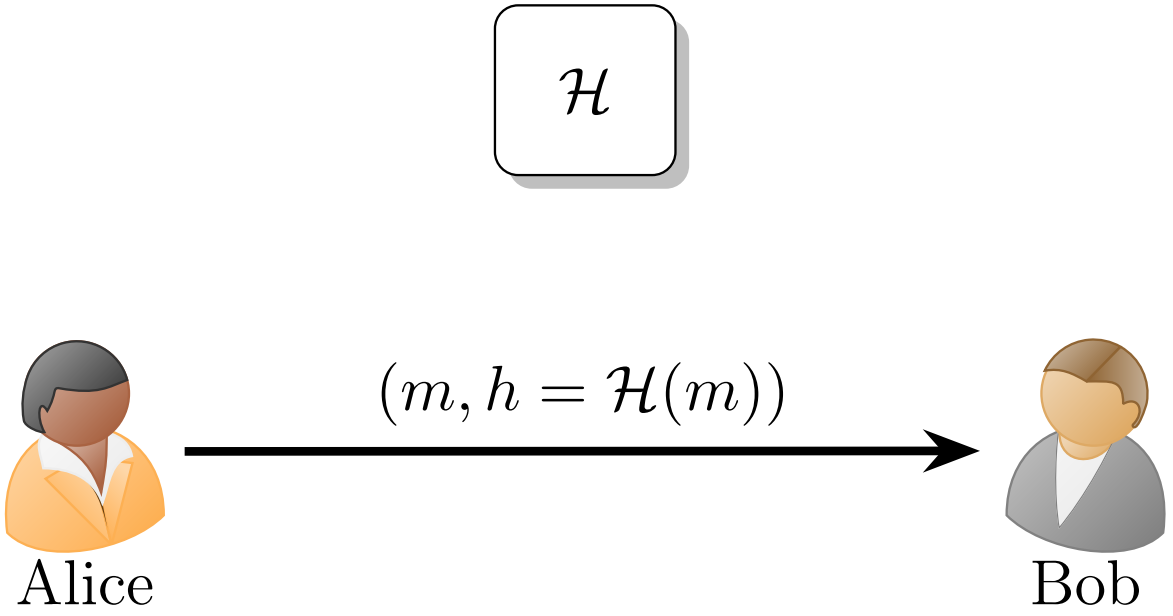


Bob

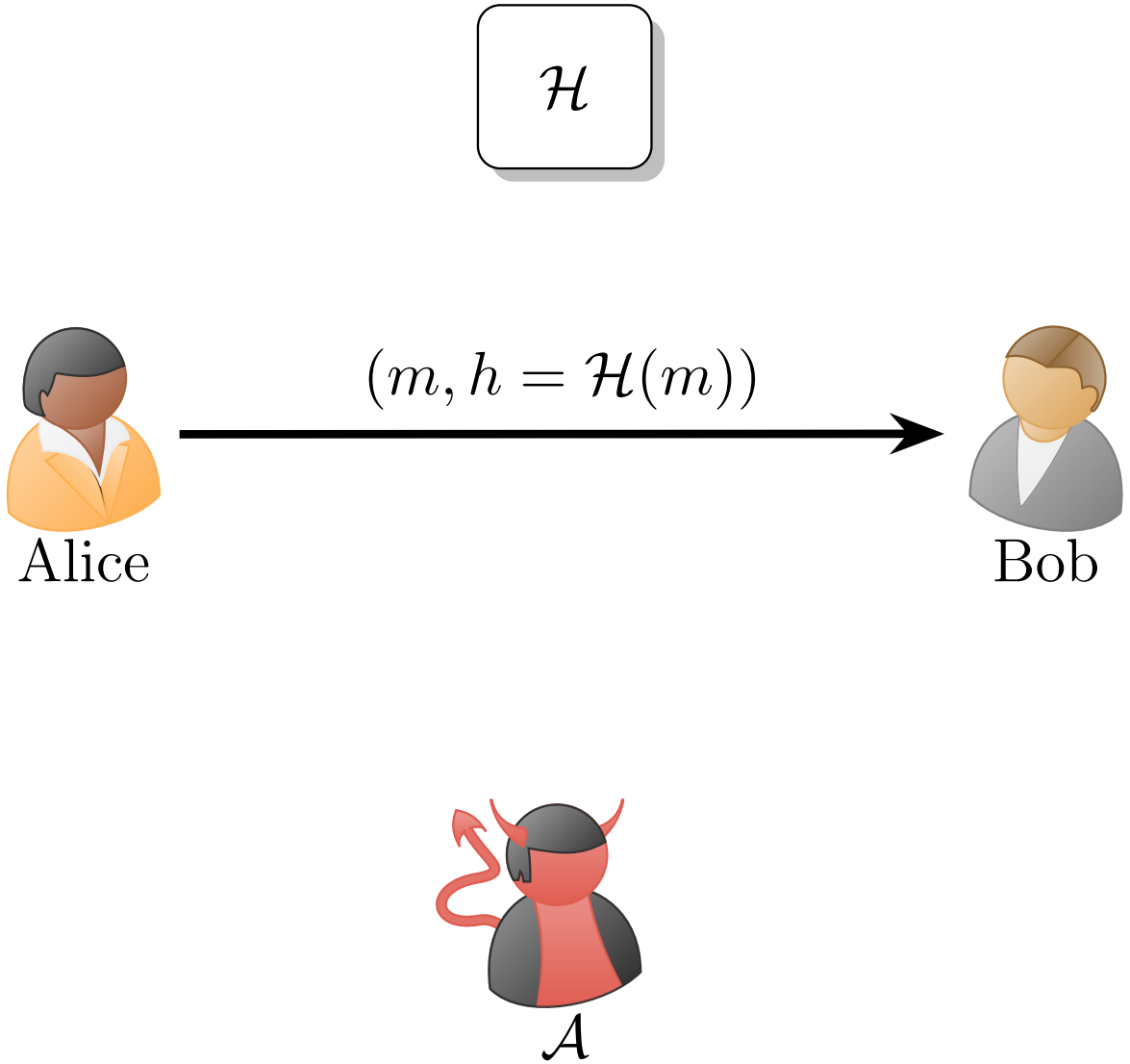
Last Time: Random Oracle Model



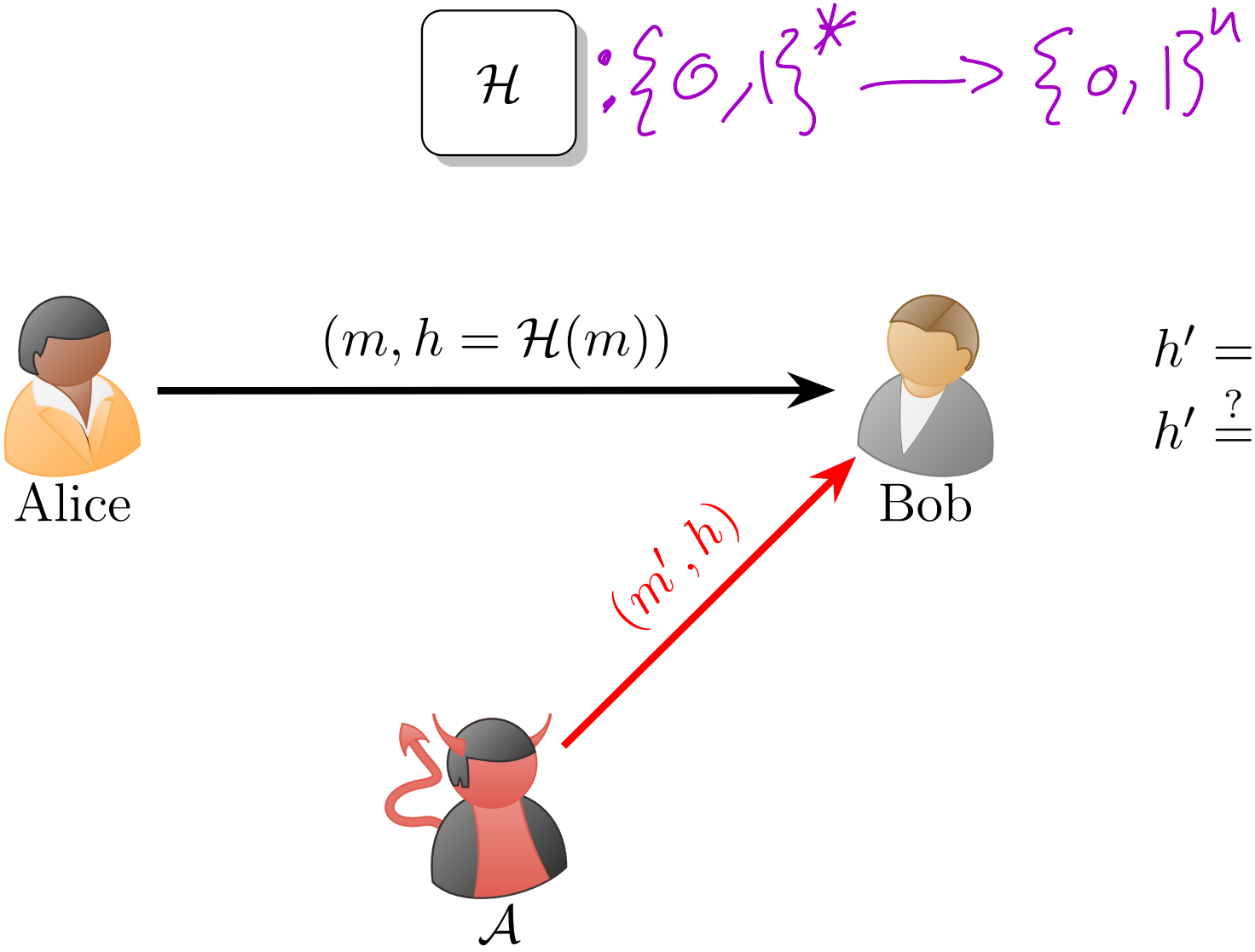
Last Time: Random Oracle Model



Last Time: Random Oracle Model



Last Time: Random Oracle Model



Random Oracle Implementation

We shall always assume that random oracles are a *lazy dictionary*.

Random Oracle Implementation

We shall always assume that random oracles are a *lazy dictionary*.

$\mathcal{H}\colon \{0,1\}^* \to \{0,1\}^n$

Random Oracle Implementation

We shall always assume that random oracles are a *lazy dictionary*.

$\mathcal{H}\colon \{0,1\}^* \to \{0,1\}^n$

$H = \{\}$

Random Oracle Implementation

We shall always assume that random oracles are a *lazy dictionary*.

$\mathcal{H}\colon \{0,1\}^* \to \{0,1\}^n$

$H = \{\}$

$\underline{H(y)\text{:}}$
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^n$
return $H[y]$

Pseudorandom Functions

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers.

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions.

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k \colon \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k \colon \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$,

Pseudorandom Functions

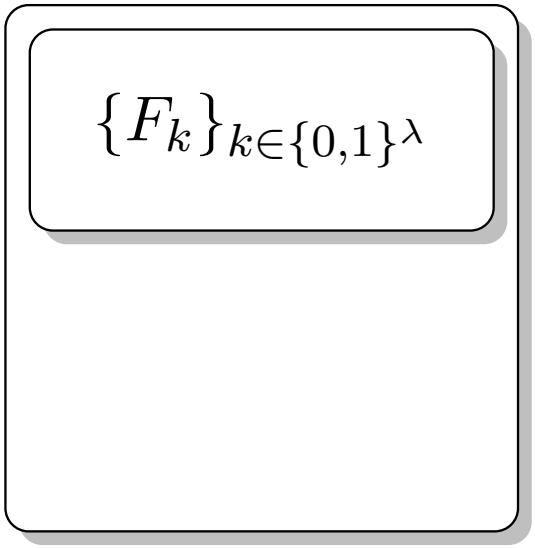
Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f : \{0,1\}^n \to \{0,1\}^m\}$.

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f : \{0,1\}^n \to \{0,1\}^m\}$.



Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f : \{0,1\}^n \to \{0,1\}^m\}$.

$\{F_k\}_{k \in \{0,1\}^\lambda}$

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k \colon \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f \colon \{0,1\}^n \to \{0,1\}^m\}$.

$\{F_k\}_{k \in \{0,1\}^\lambda}$

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

$\mathcal{F}_{n,m} := \{f \colon \{0,1\}^n \to \{0,1\}^m\}$

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k \colon \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f \colon \{0,1\}^n \to \{0,1\}^m\}$.

$\{F_k\}_{k \in \{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f \colon \{0,1\}^n \to \{0,1\}^m\}$

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k \colon \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f \colon \{0,1\}^n \to \{0,1\}^m\}$.

$\{F_k\}_{k \in \{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f \colon \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

Pseudorandom Functions

Definition 1 (Pseudorandom Function Family)

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$ be integers. Let $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ be a set of efficiently computable functions. Then, $\{F_k\}_k$ is a *pseudorandom function family* if for uniformly random $k \xleftarrow{\$} \{0,1\}^\lambda$, F_k is computationally indistinguishable from $\widehat{F} \xleftarrow{\$} \{f : \{0,1\}^n \to \{0,1\}^m\}$.

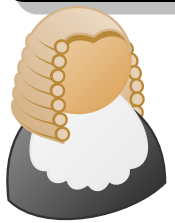
$\{F_k\}_{k \in \{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$



Distinguisher

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$.

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$.

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$,

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$, there exists a negligible function negl such that

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$, there exists a negligible function negl such that

$$\left|\Pr[\mathcal{D}^{D_0}(1^\lambda) = 0] - \Pr[\mathcal{D}^{D_1}(1^\lambda) = 1]\right| \leqslant \mathsf{negl}(\lambda).$$

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$, there exists a negligible function negl such that

$$\left|\Pr[\mathcal{D}^{D_0}(1^\lambda) = 0] - \Pr[\mathcal{D}^{D_1}(1^\lambda) = 1]\right| \leqslant \mathsf{negl}(\lambda).$$

- $\mathcal{D}$ is given *oracle access* to the distributions

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$, there exists a negligible function negl such that

$$\left|\Pr[\mathcal{D}^{D_0}(1^\lambda) = 0] - \Pr[\mathcal{D}^{D_1}(1^\lambda) = 1]\right| \leqslant \mathsf{negl}(\lambda).$$

- $\mathcal{D}$ is given *oracle access* to the distributions
  - D_0 and D_1 might require *exponential length* to write down!

Computational Indistinguishability

- Need to define *computational indistinguishability* formally.

Definition 2 (Computational Indistinguishability)

Let $\lambda \in \mathbb{N}$. Let D_0 and D_1 be two distributions over $\{0,1\}^{n(\lambda)}$. Then, we say that D_0 and D_1 are *computationally indistinguishable* if for every PPT distinguisher $\mathcal{D}$, there exists a negligible function negl such that

$$\left|\Pr[\mathcal{D}^{D_0}(1^\lambda) = 0] - \Pr[\mathcal{D}^{D_1}(1^\lambda) = 1]\right| \leqslant \mathsf{negl}(\lambda).$$

- $\mathcal{D}$ is given *oracle access* to the distributions
  - D_0 and D_1 might require *exponential length* to write down!

- If D_0 and D_1 are computationally indistinguishable, we write $D_0 \approx D_1$.

PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.

PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.

$\{F_k\}_{k \in \{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.

$\{F_k\}_{k\in\{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

Distinguisher $\mathcal{D}$

PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.

$\{F_k\}_{k\in\{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

flipped once

$b \xleftarrow{\$} \{0,1\}$

Distinguisher $\mathcal{D}$

PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.

$\{F_k\}_{k\in\{0,1\}^\lambda}$

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

F_k

$\mathcal{F}_{n,m} := \{f : \{0,1\}^n \to \{0,1\}^m\}$

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

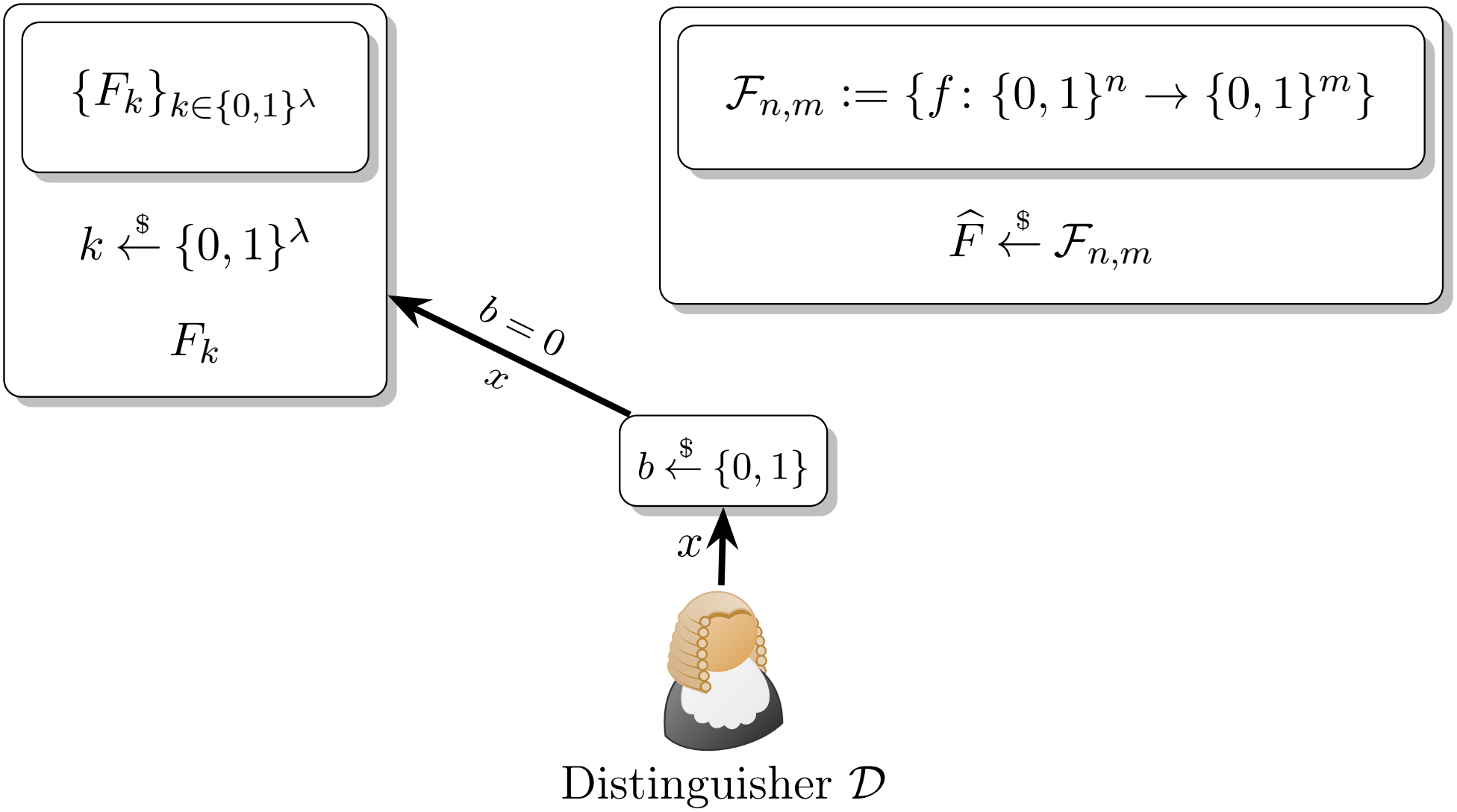
$b \xleftarrow{\$} \{0,1\}$

x

Distinguisher $\mathcal{D}$

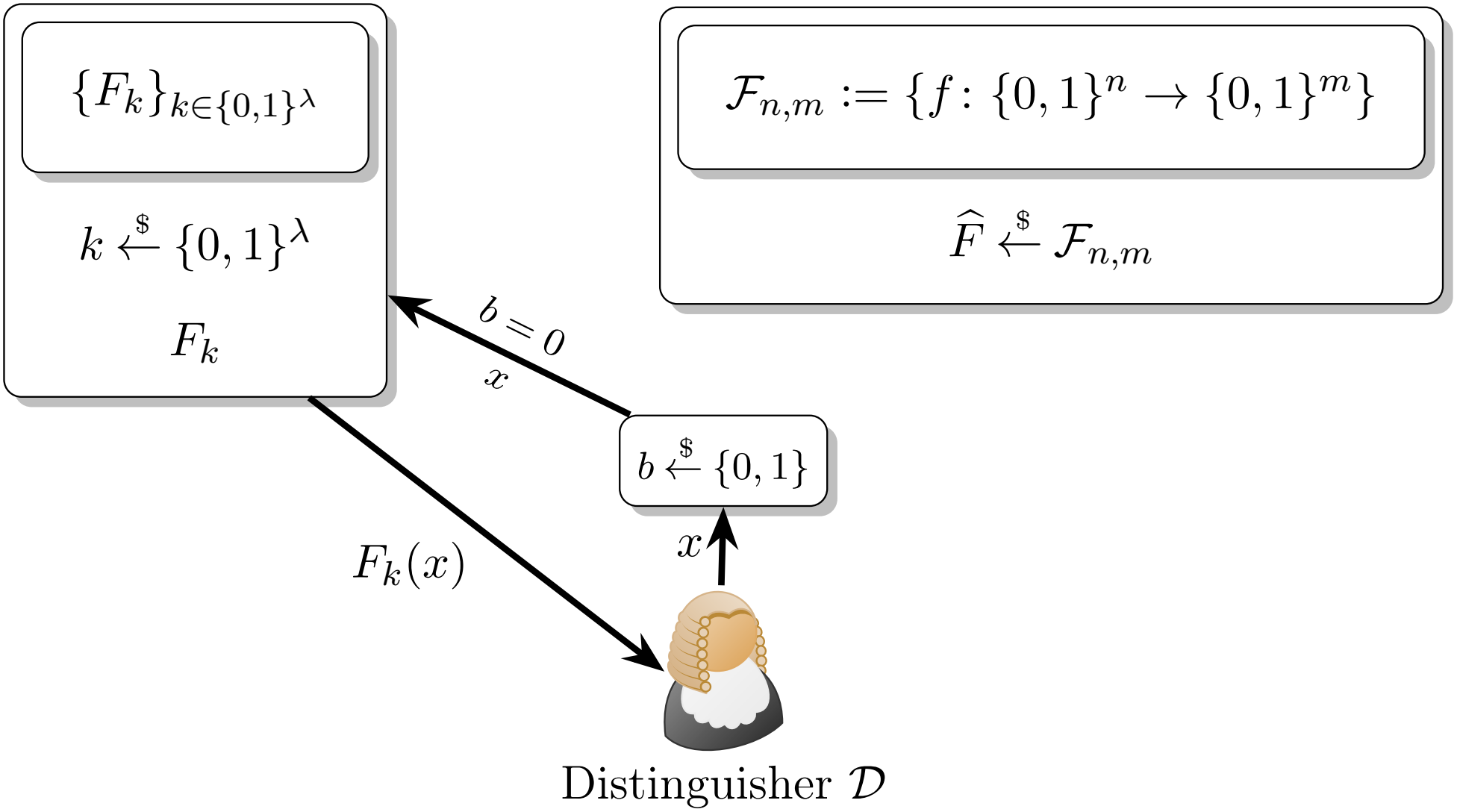
PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.



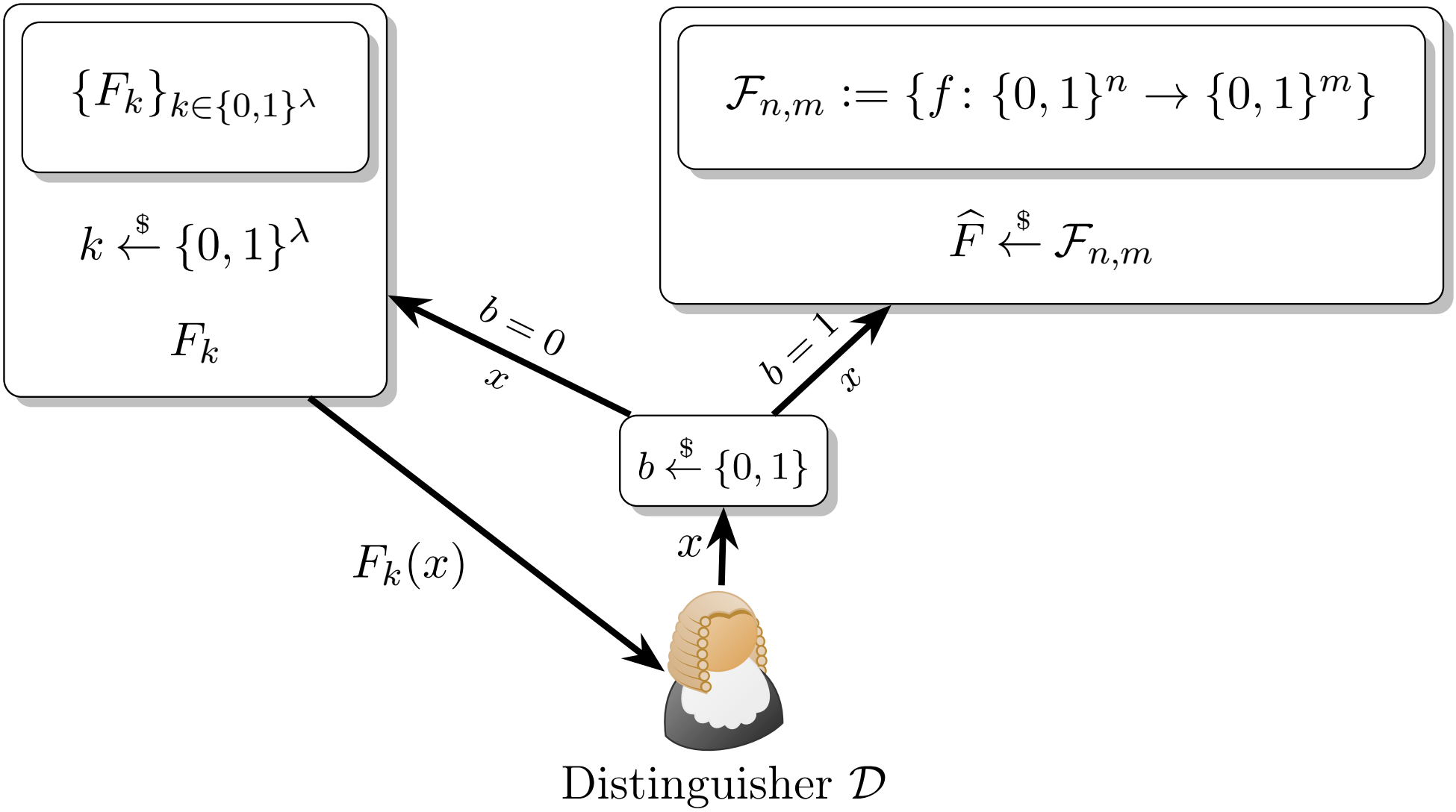
PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.



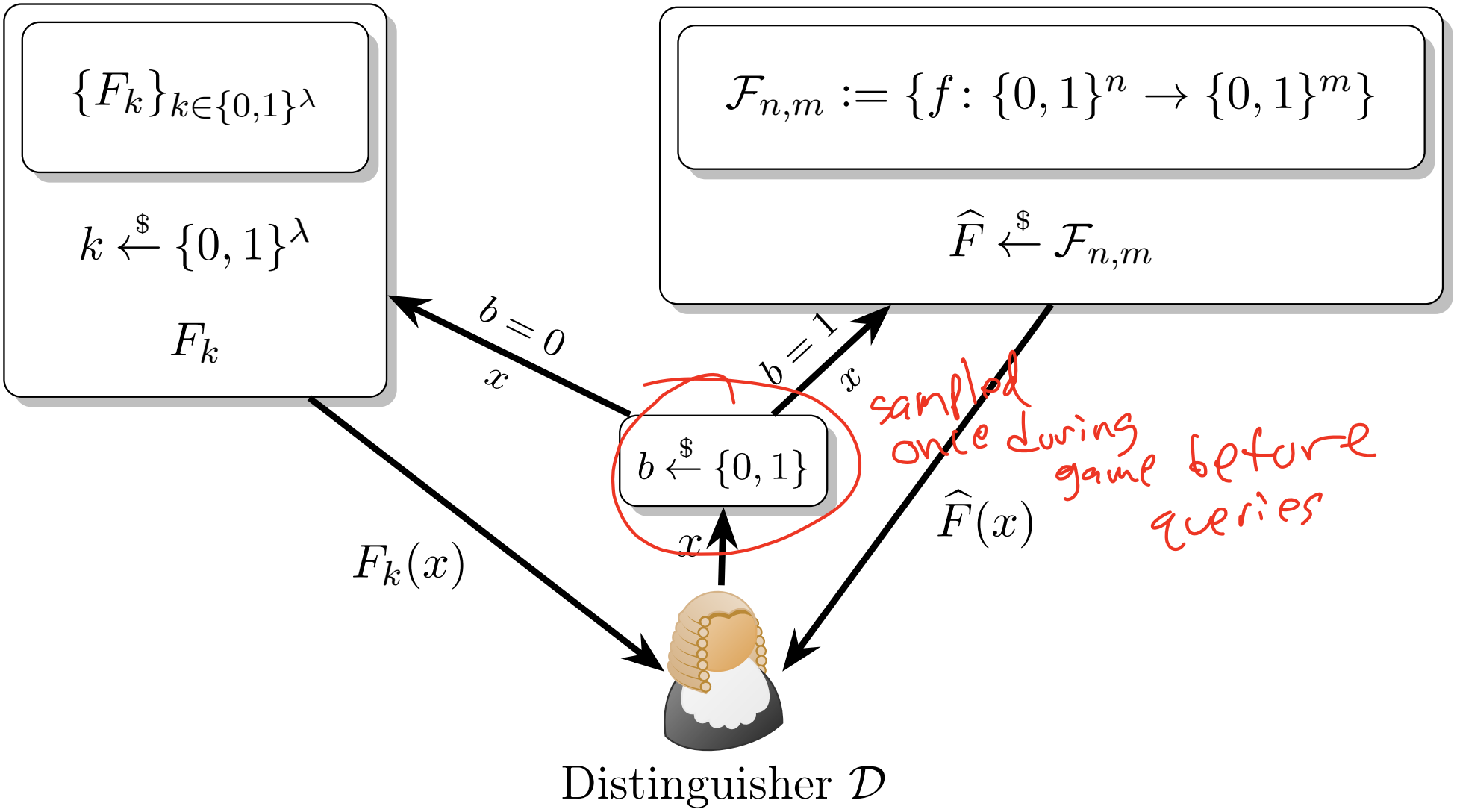
PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.



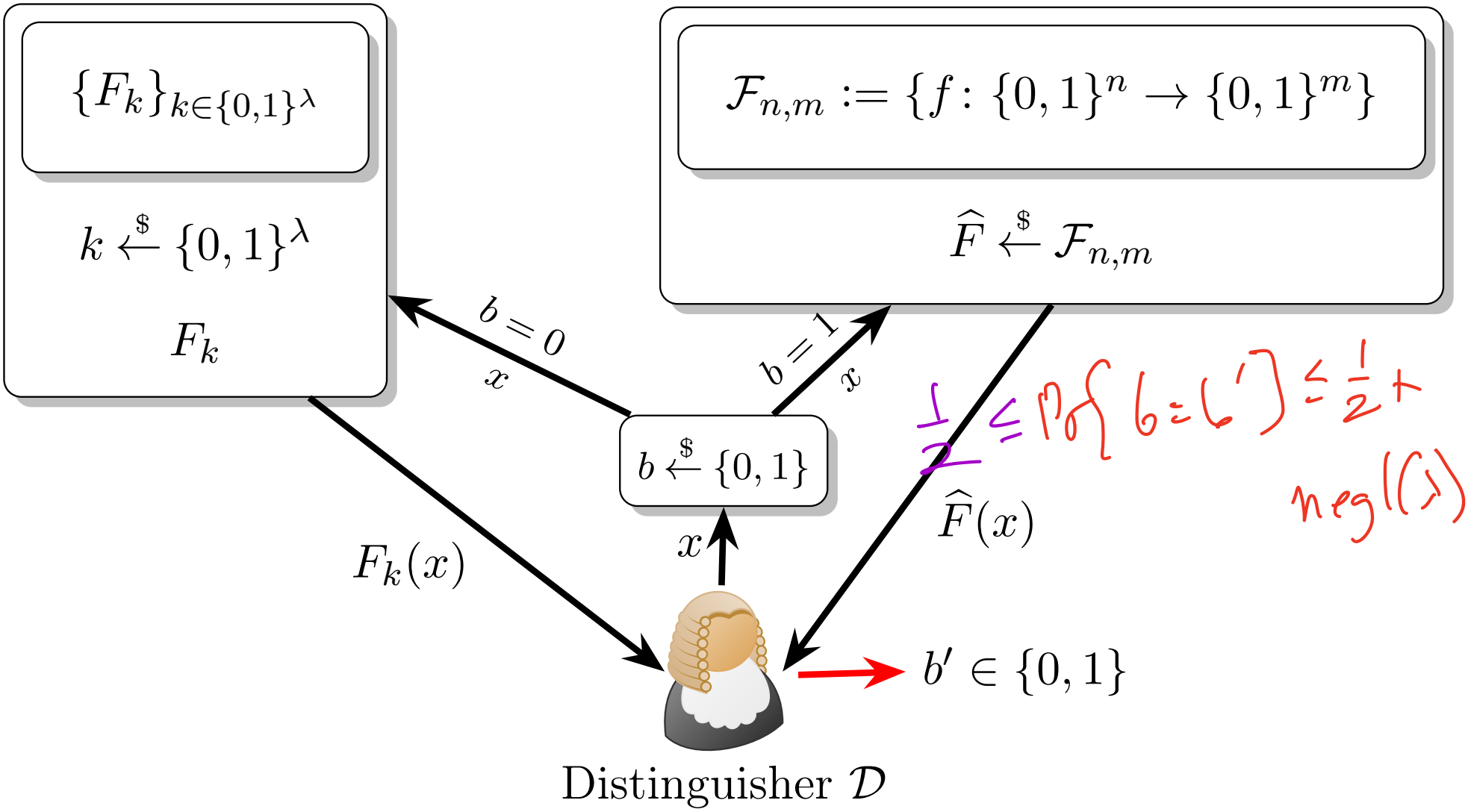
PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.



PRF Security

- Our PRF definition says that the two distributions we saw before are computationally indistinguishable.



PRF Security: Notes

PRF Security: Notes

Oracle access *very* is important!

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k.

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**
  - $\mathcal{D}$ could read the key k!

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**
  - $\mathcal{D}$ could read the key k!
- $\widehat{F}$ may not in general be efficiently computable!

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**
  - $\mathcal{D}$ could read the key k!
- $\widehat{F}$ may not in general be efficiently computable! Every $\widehat{F} \in \mathcal{F}_{n,m}$ might need $m \cdot 2^n$ bits to write down!

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**
  - $\mathcal{D}$ could read the key k!

- $\widehat{F}$ may not in general be efficiently computable! Every $\widehat{F} \in \mathcal{F}_{n,m}$ might need $m \cdot 2^n$ bits to write down! $\mathcal{D}$ couldn't even read $\widehat{F}$, let alone evaluate it.

PRF Security: Notes

Oracle access *very* is important!

- Every F_k is efficiently computable, so $\mathcal{D}$ *could* look at the "code" for F_k. **We do not want this! Why?**
  - $\mathcal{D}$ could read the key k!

- $\widehat{F}$ may not in general be efficiently computable! Every $\widehat{F} \in \mathcal{F}_{n,m}$ might need $m \cdot 2^n$ bits to write down! $\mathcal{D}$ couldn't even read $\widehat{F}$, let alone evaluate it.
  - In light of this, if $\mathcal{D}$ is given a function it can *read*, then it can output $b' = 0$ and win the game with noticeable probability.

PRFs from Random Oracles

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

Lemma 1

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

Lemma 1

Let $\lambda \in \mathbb{N}$ *and* $n := n(\lambda), m := m(\lambda)$.

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

Lemma 1

Let $\lambda \in \mathbb{N}$ *and* $n := n(\lambda), m := m(\lambda)$. *Let* $H \colon \{0,1\}^* \to \{0,1\}^m$ *be a random oracle for arbitrary length inputs.*

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

Lemma 1

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$. Let $H : \{0,1\}^* \to \{0,1\}^m$ be a random oracle for arbitrary length inputs. Then, the following function family is a secure PRF family.

PRFs from Random Oracles

We can easily construct PRFs in the Random Oracle Model.

Lemma 1

Let $\lambda \in \mathbb{N}$ and $n := n(\lambda), m := m(\lambda)$. Let $H : \{0,1\}^* \to \{0,1\}^m$ be a random oracle for arbitrary length inputs. Then, the following function family is a secure PRF family.

$$\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda} \qquad F_k^H(x) := H(k \| x)$$

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

Claim 1

The following two distributions are identical.

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

Claim 1

The following two distributions are identical.

$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$

On input $x \in \{0,1\}^n$
Output $\widehat{F}(x)$

D_0

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

Claim 1

The following two distributions are identical.

$$\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m}$$

On input $x \in \{0,1\}^n$
Output $\widehat{F}(x)$

D_0

$\equiv$

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

Claim 1

The following two distributions are identical.

$$\boxed{\begin{array}{c}\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m} \\ \\ \textit{On input } x \in \{0,1\}^n \\ \textit{Output } \widehat{F}(x)\end{array}} \quad D_0$$

$$\equiv$$

$$\boxed{\begin{array}{c}L = \{\} \\ \\ \textit{On input } x \in \{0,1\}^n \\ \textit{If } L[x] \textit{ undefined} \\ L[x] \xleftarrow{\$} \{0,1\}^m \\ \textit{Output } L[x]\end{array}} \quad D_1$$

PRFs from Random Oracles

Before we prove the lemma, we first switch to an equivalent definition of PRF security.

Claim 1

The following two distributions are identical.

$$\boxed{\begin{array}{c}\widehat{F} \xleftarrow{\$} \mathcal{F}_{n,m} \\ \\ \textit{On input } x \in \{0,1\}^n \\ \textit{Output } \widehat{F}(x)\end{array}} \quad D_0$$

$$\equiv$$

$$\boxed{\begin{array}{c}L = \{\} \\ \\ \textit{On input } x \in \{0,1\}^n \\ \textit{If } L[x] \textit{ undefined} \\ L[x] \xleftarrow{\$} \{0,1\}^m \\ \textit{Output } L[x]\end{array}} \quad D_1$$

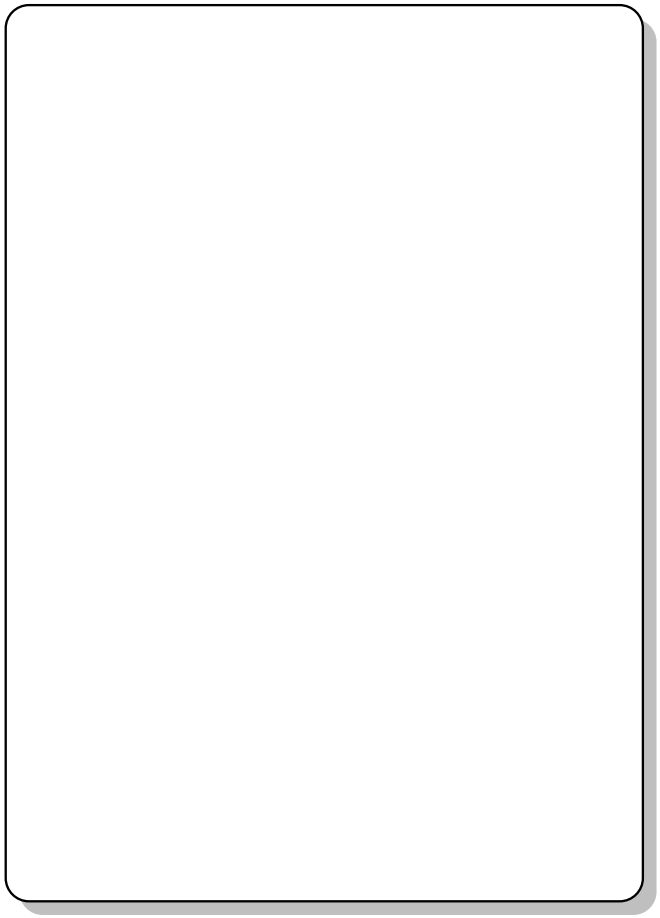
So a function family $\{F_k : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ is a secure PRF family if it is computationally indistinguishable from D_1.

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable

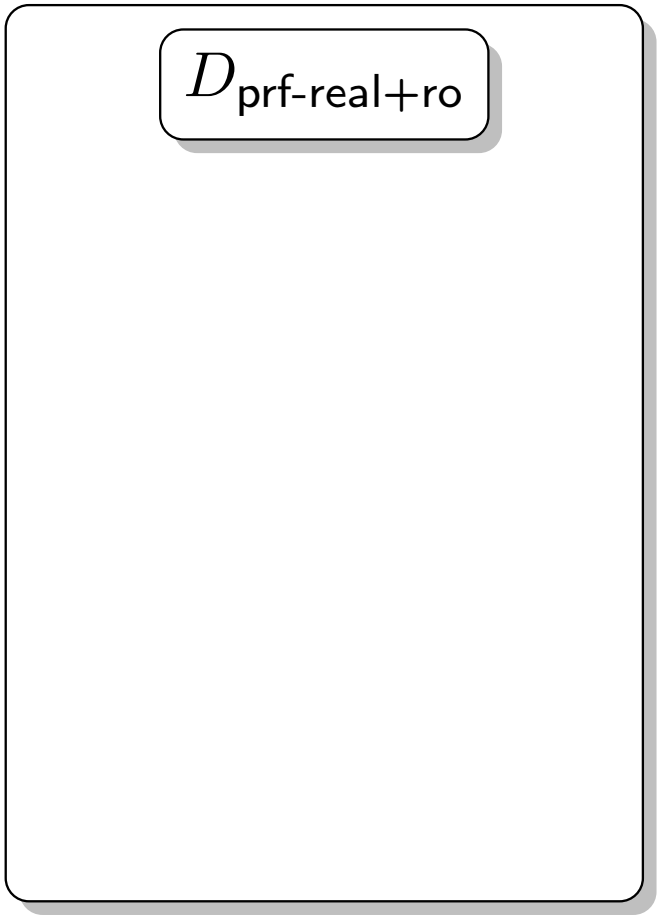
PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



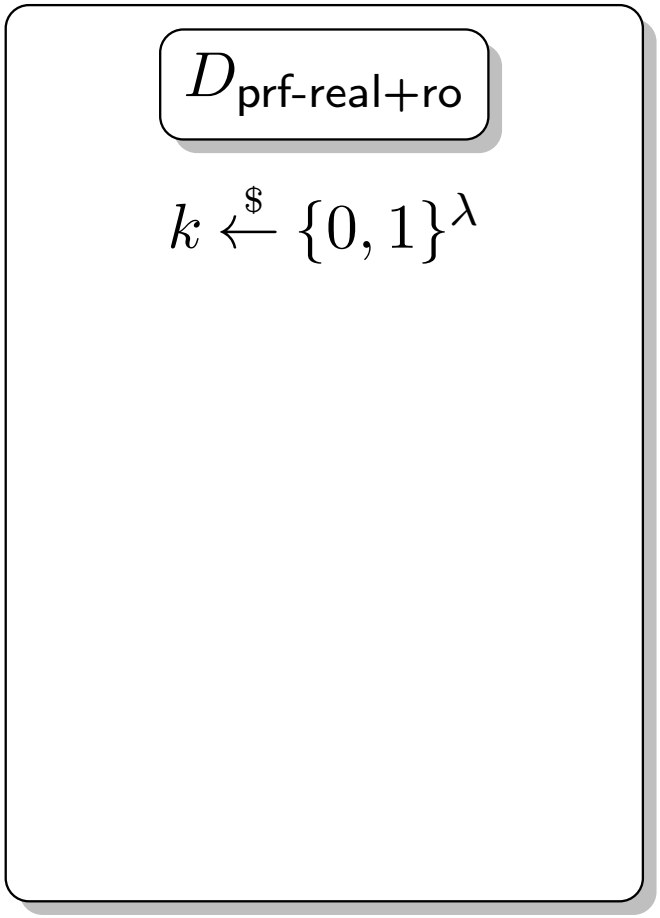
PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



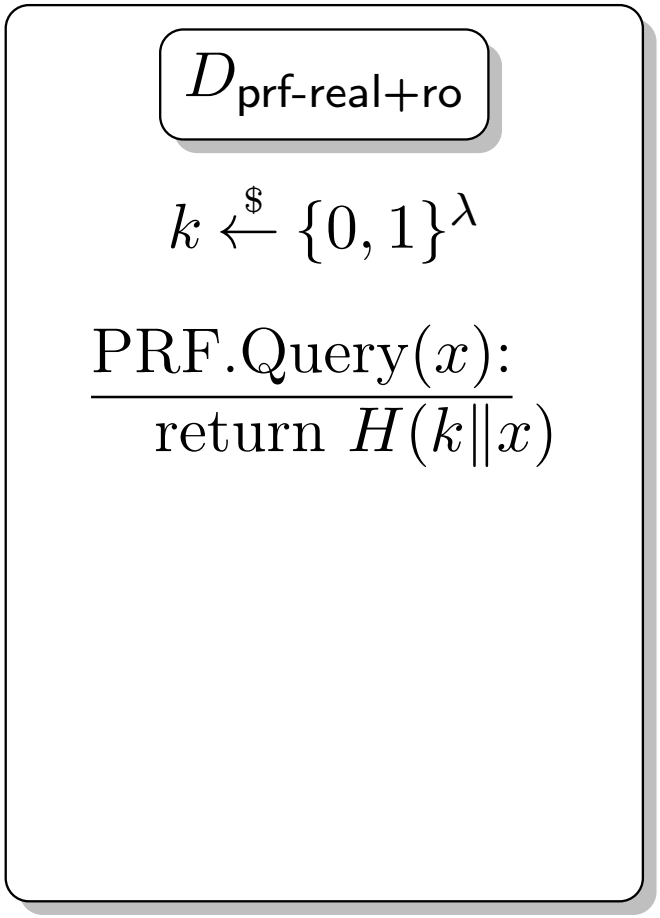
PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



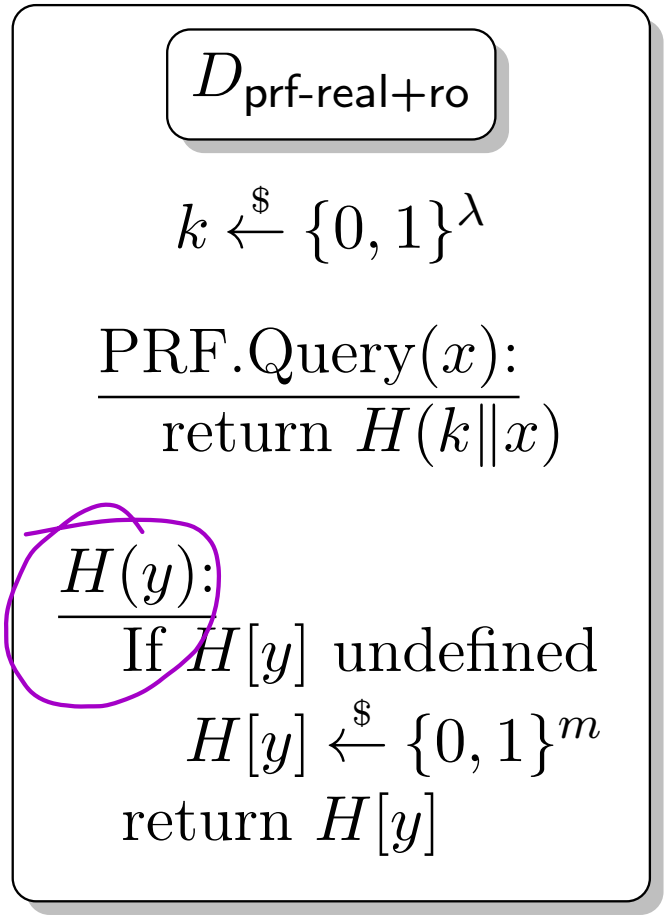
PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



Everyone has access to this

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable

$D_{\mathsf{prf\text{-}real+ro}}$

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

PRF.Query(x):
return $H(k \| x)$

$H(y)$:
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^m$
return $H[y]$

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable

$D_{\mathsf{prf\text{-}real+ro}}$

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

PRF.Query(x):
return $H(k \| x)$

$H(y)$:
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^m$
return $H[y]$

$D_{\mathsf{prf\text{-}rand+ro}}$

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable

$D_{\text{prf-real+ro}}$

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

PRF.Query(x):
return $H(k \| x)$

$H(y)$:
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^m$
return $H[y]$

$D_{\text{prf-rand+ro}}$

$L = \{\}$

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable

$D_{\mathsf{prf\text{-}real+ro}}$

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

PRF.Query(x):
  return $H(k \| x)$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

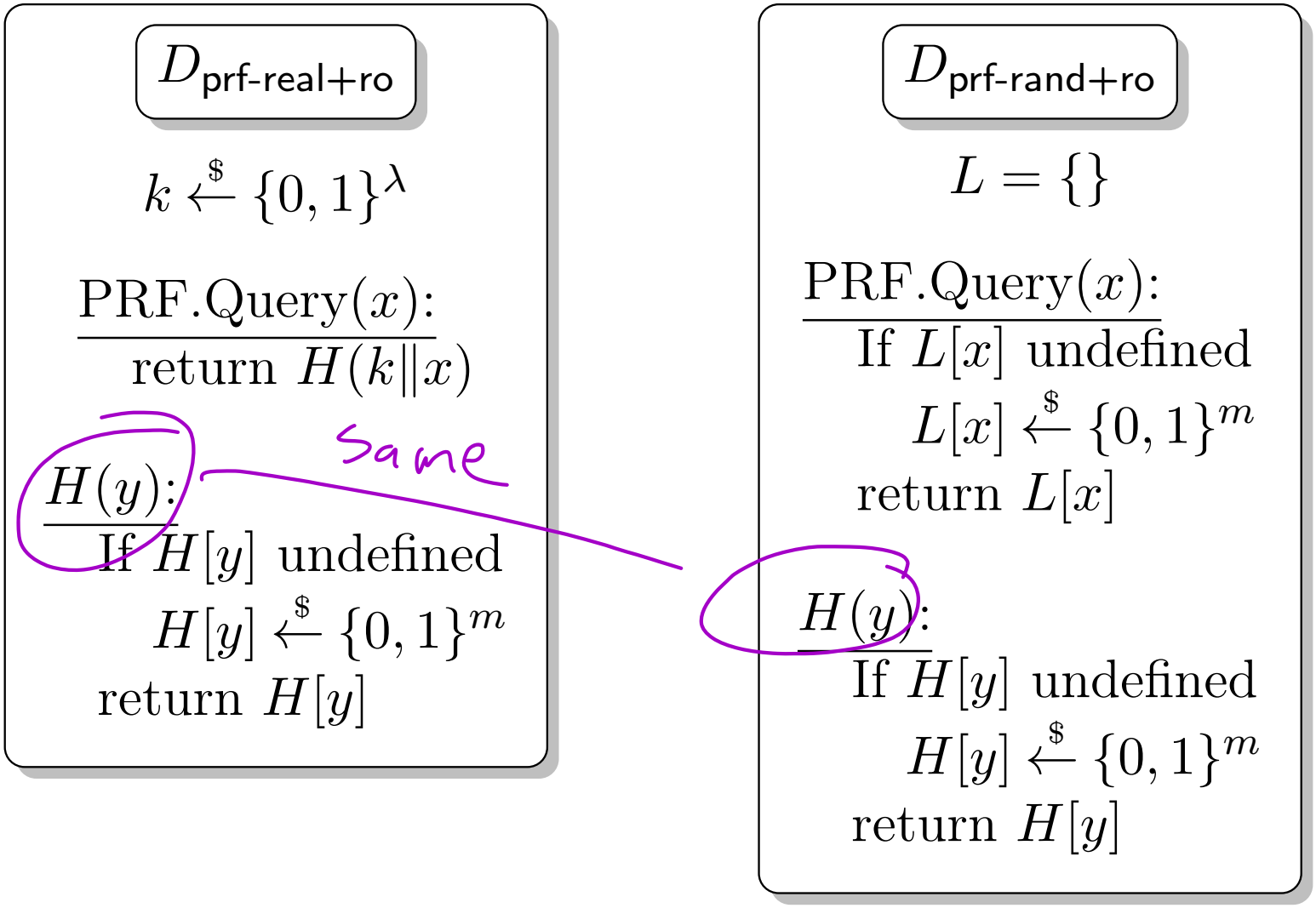
$D_{\mathsf{prf\text{-}rand+ro}}$

$L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

PRFs from Random Oracles

To prove the lemma, we show the following distributions are indistinguishable



Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.

Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.

$D_{\mathsf{prf\text{-}real+ro}}$

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

PRF.Query(x):
  return $H(k \| x)$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_0$ (Hybrid 0)

Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.

$D_{\mathsf{prf\text{-}real+ro}}$

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

$\underline{\mathrm{PRF.Query}(x)\text{:}}$
return $H(k\|x)$

$\underline{H(y)\text{:}}$
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^m$
return $H[y]$

$\mathcal{H}_0$ (Hybrid 0)

$D_{\mathsf{prf\text{-}inline\text{-}ro}}$

$\mathcal{H}_1$

Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.

$D_{\mathsf{prf\text{-}real+ro}}$

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

$\underline{\mathrm{PRF.Query}(x)\text{:}}$
return $H(k\|x)$

$\underline{H(y)\text{:}}$
If $H[y]$ undefined
$H[y] \xleftarrow{\$} \{0,1\}^m$
return $H[y]$

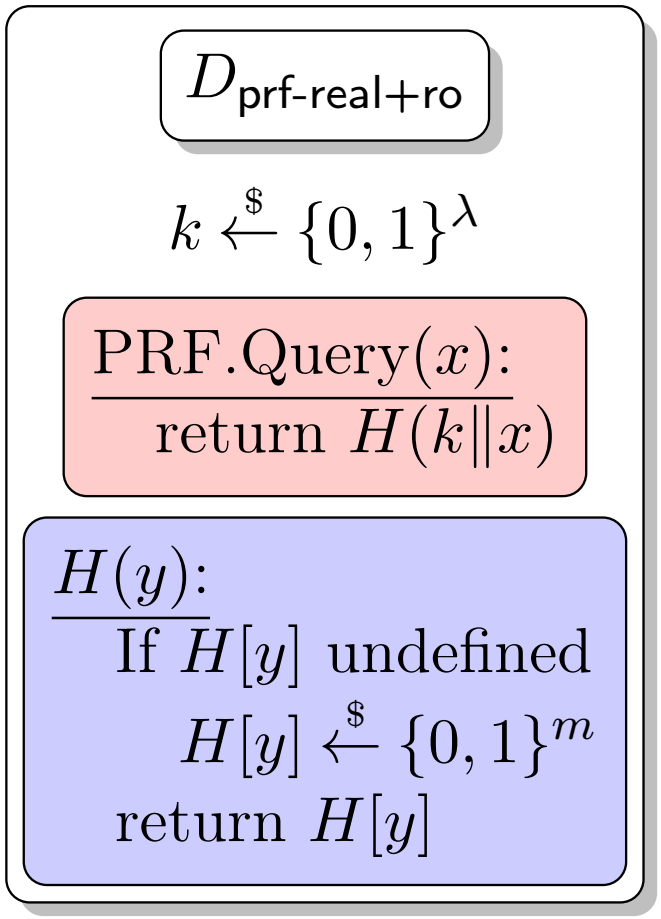
$\mathcal{H}_0$ (Hybrid 0)

$D_{\mathsf{prf\text{-}inline\text{-}ro}}$

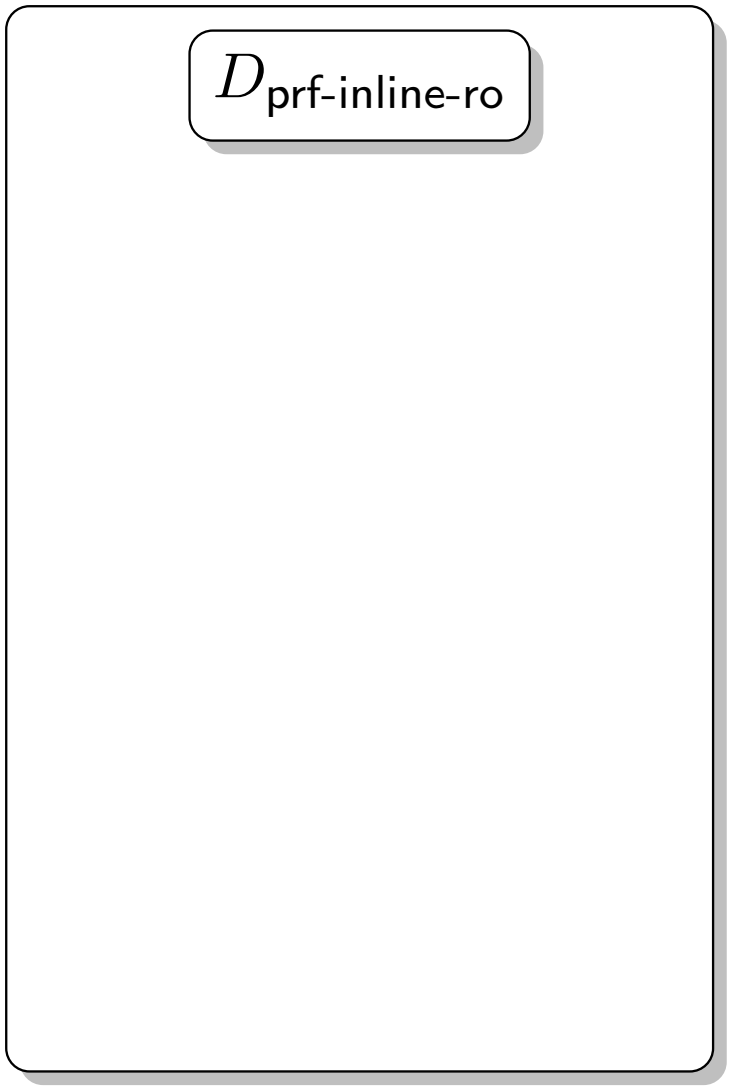
$\mathcal{H}_1$

Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.



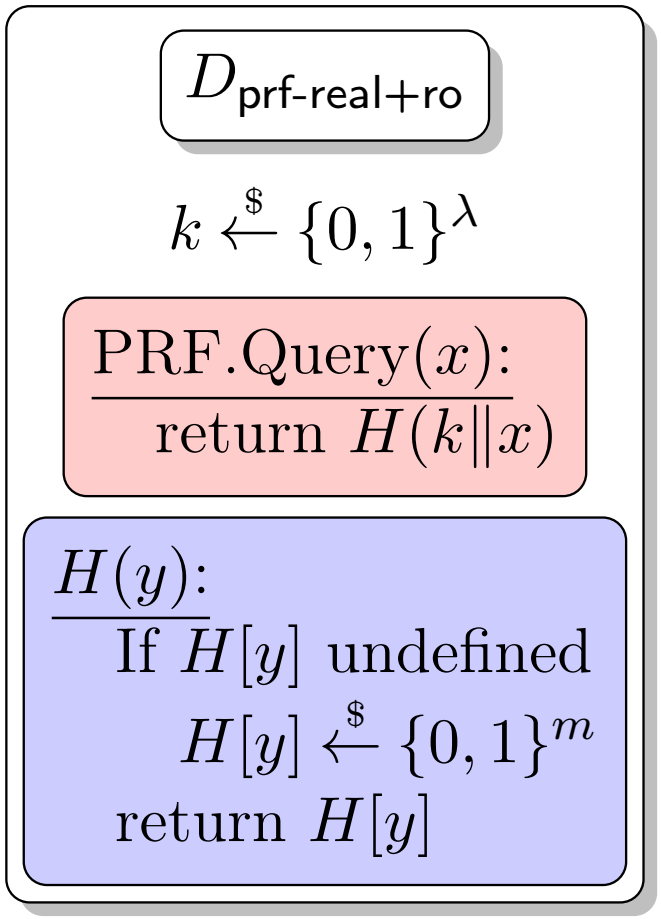
$\mathcal{H}_0$ (Hybrid 0)



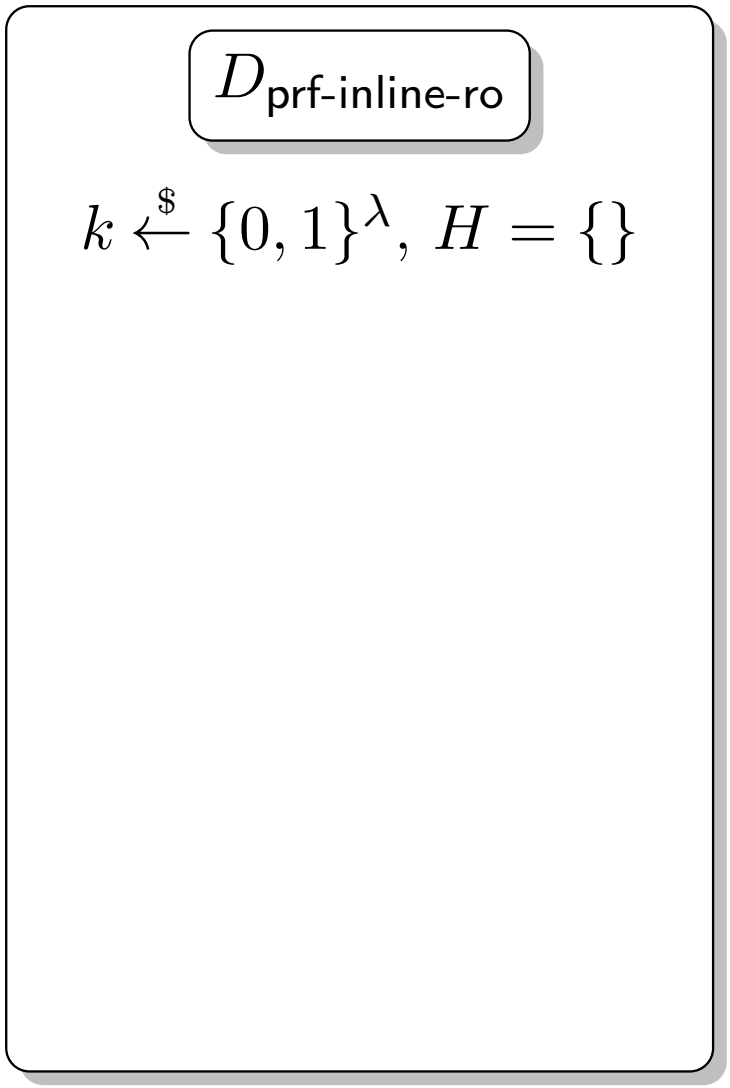
$\mathcal{H}_1$

Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.



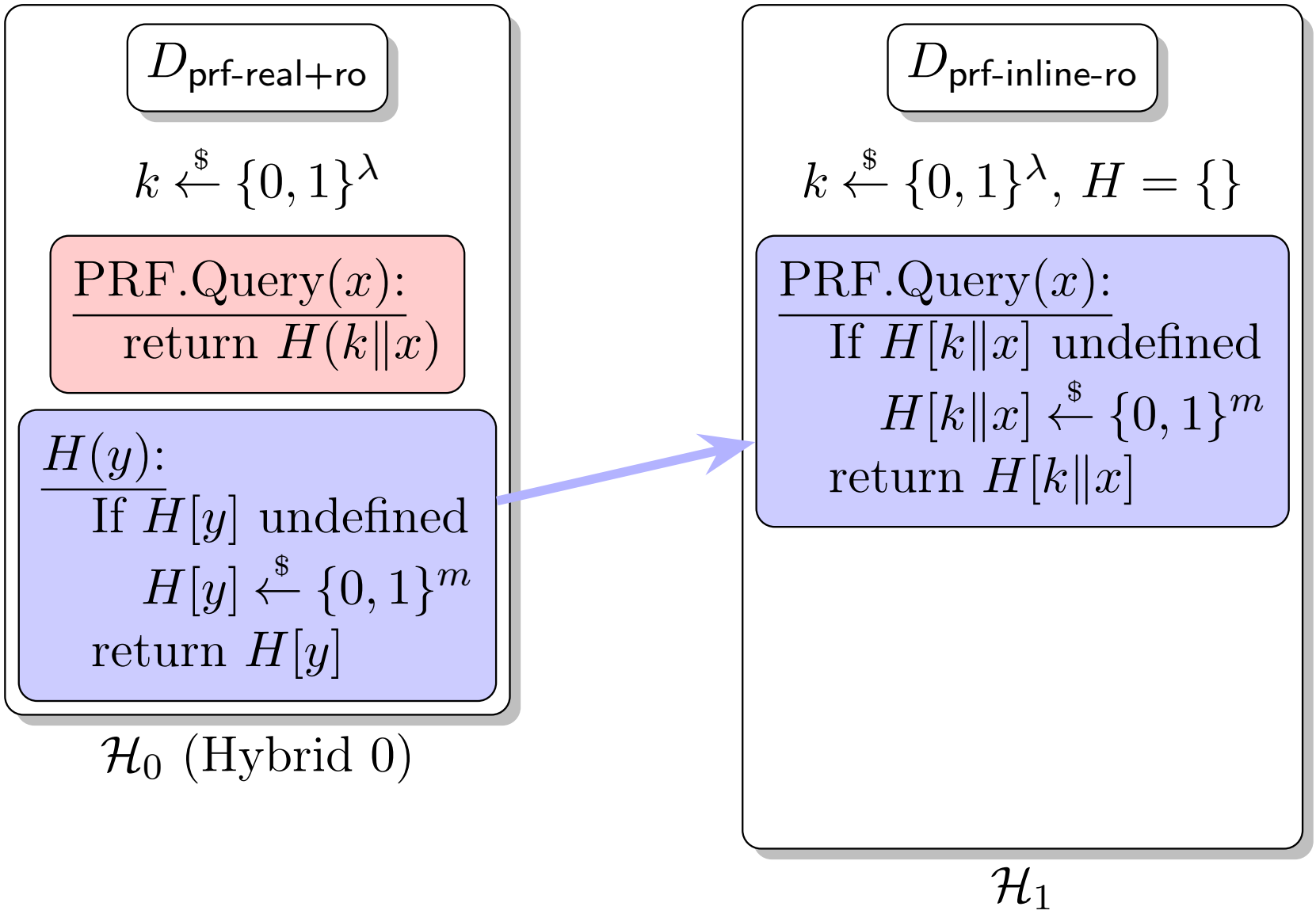
$\mathcal{H}_0$ (Hybrid 0)



$\mathcal{H}_1$

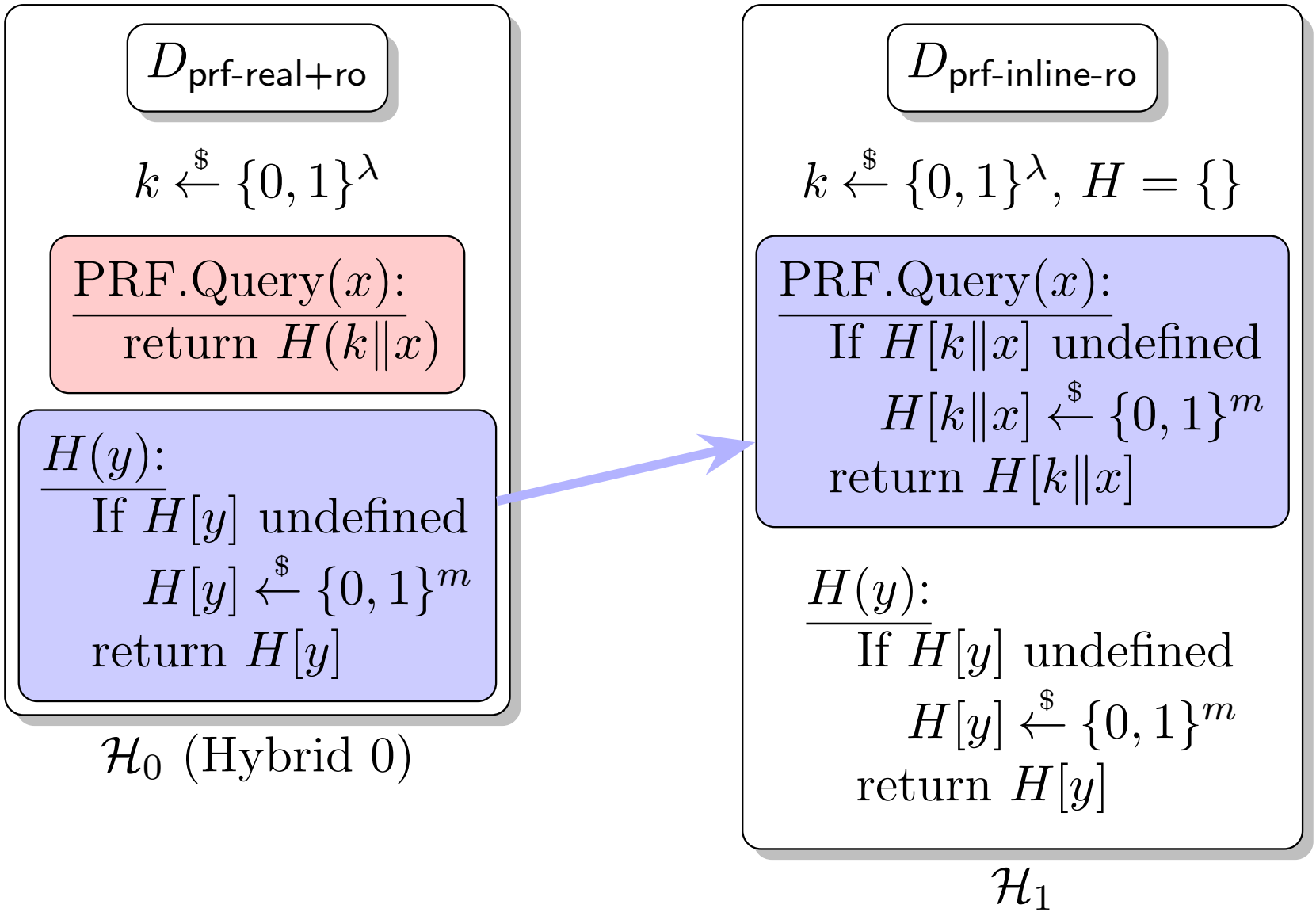
Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.



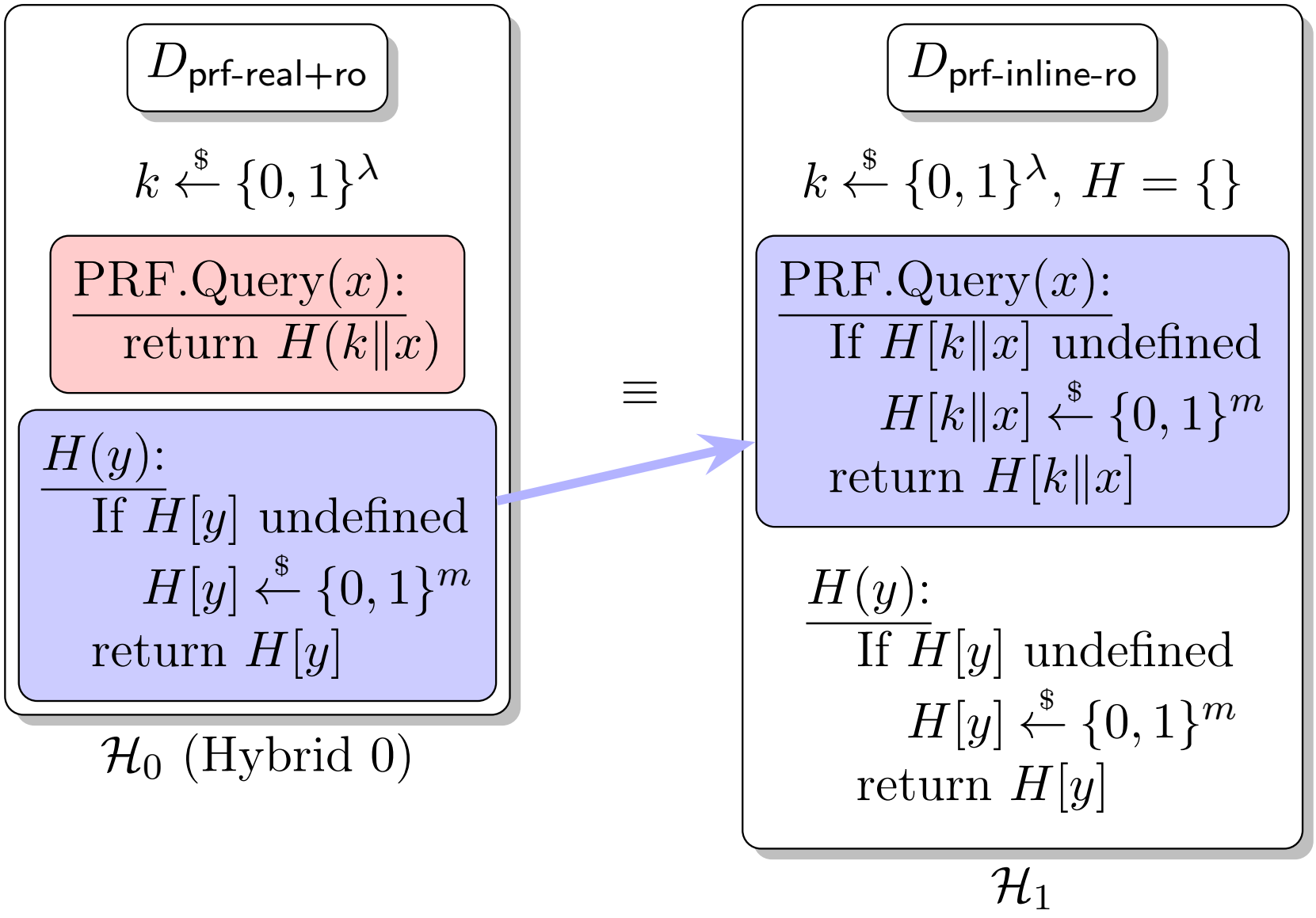
Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.



Proof via Hybrid Argument

To prove indistinguishability, we employ a *hybrid argument*.



Proof via Hybrid Argument

Claim 2

$\mathcal{H}_0$ and $\mathcal{H}_1$ are identical distributions.

Proof via Hybrid Argument

Claim 2

$\mathcal{H}_0$ and $\mathcal{H}_1$ are identical distributions.

Proof.

Follows immediately from the construction of our function F_k. □

Proof via Hybrid Argument

Claim 2

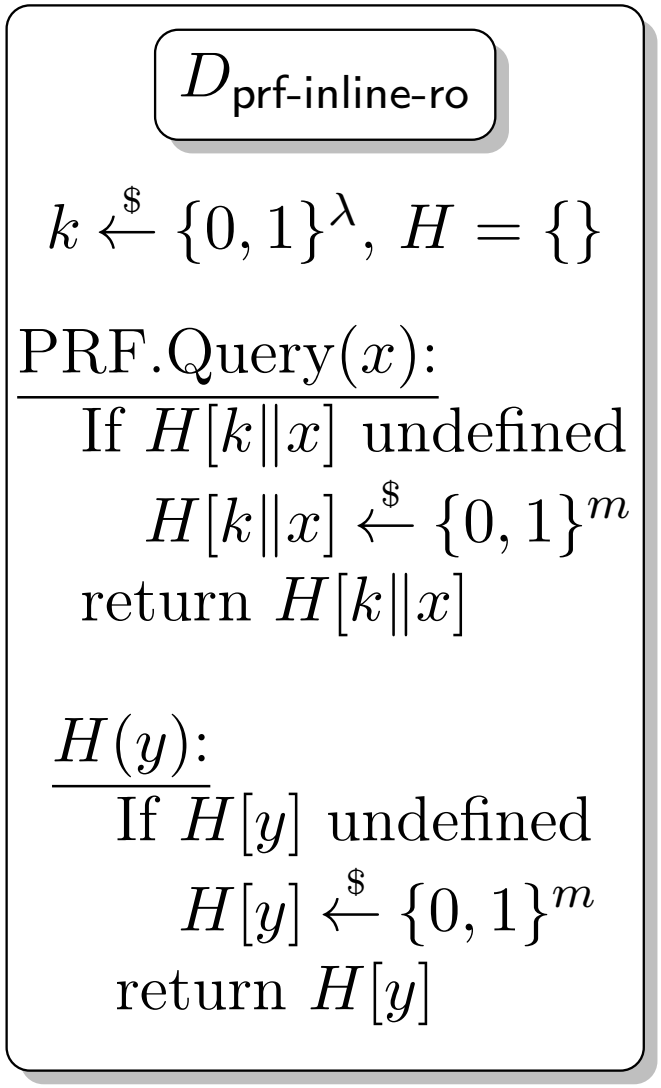
$\mathcal{H}_0$ and $\mathcal{H}_1$ are identical distributions.

Proof.

Follows immediately from the construction of our function F_k. □

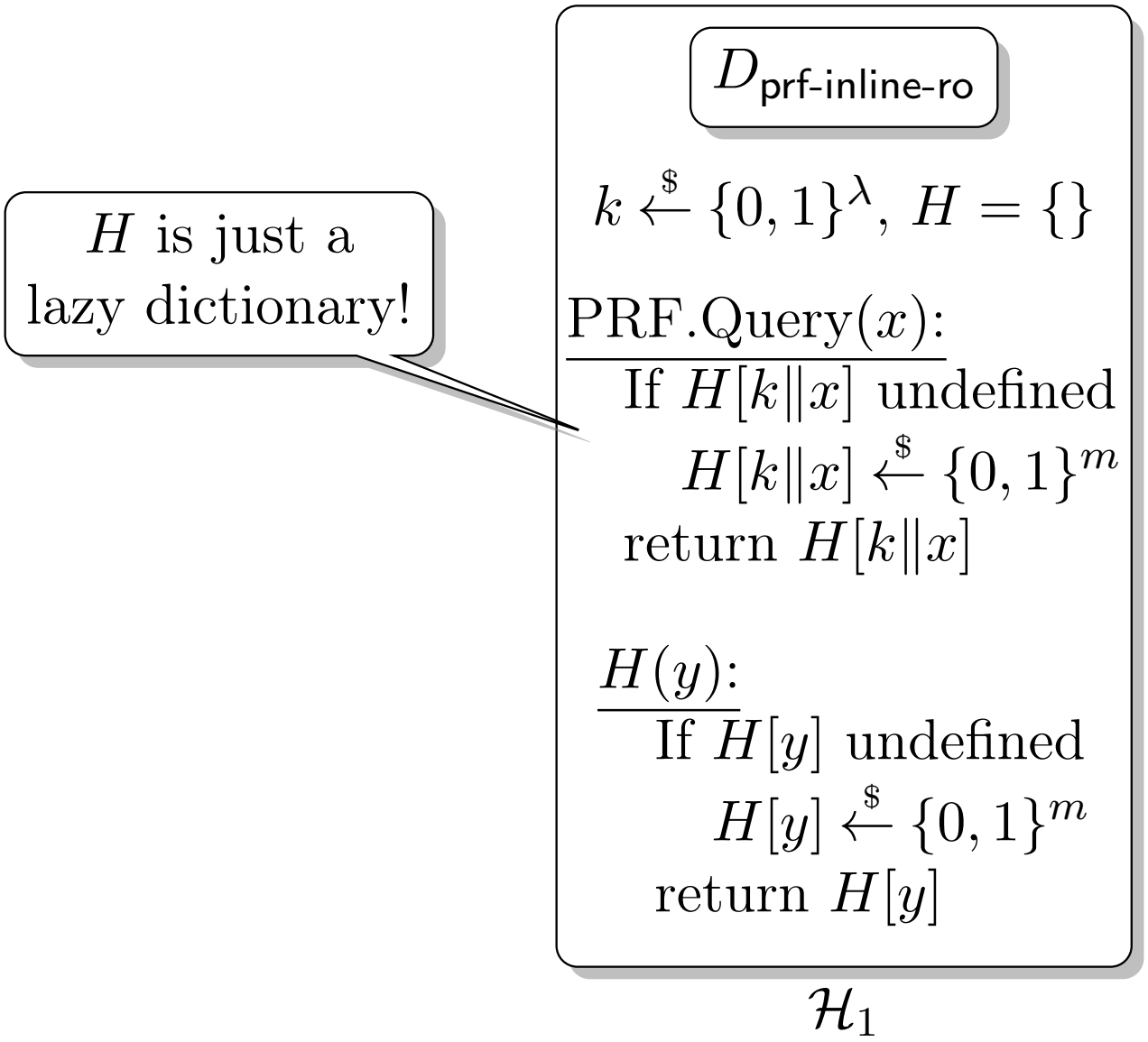
- Intuition: All we've done is replaced "return $H(k\|x)$" with how the Random Oracle H is defined!

Proof via Hybrid Argument

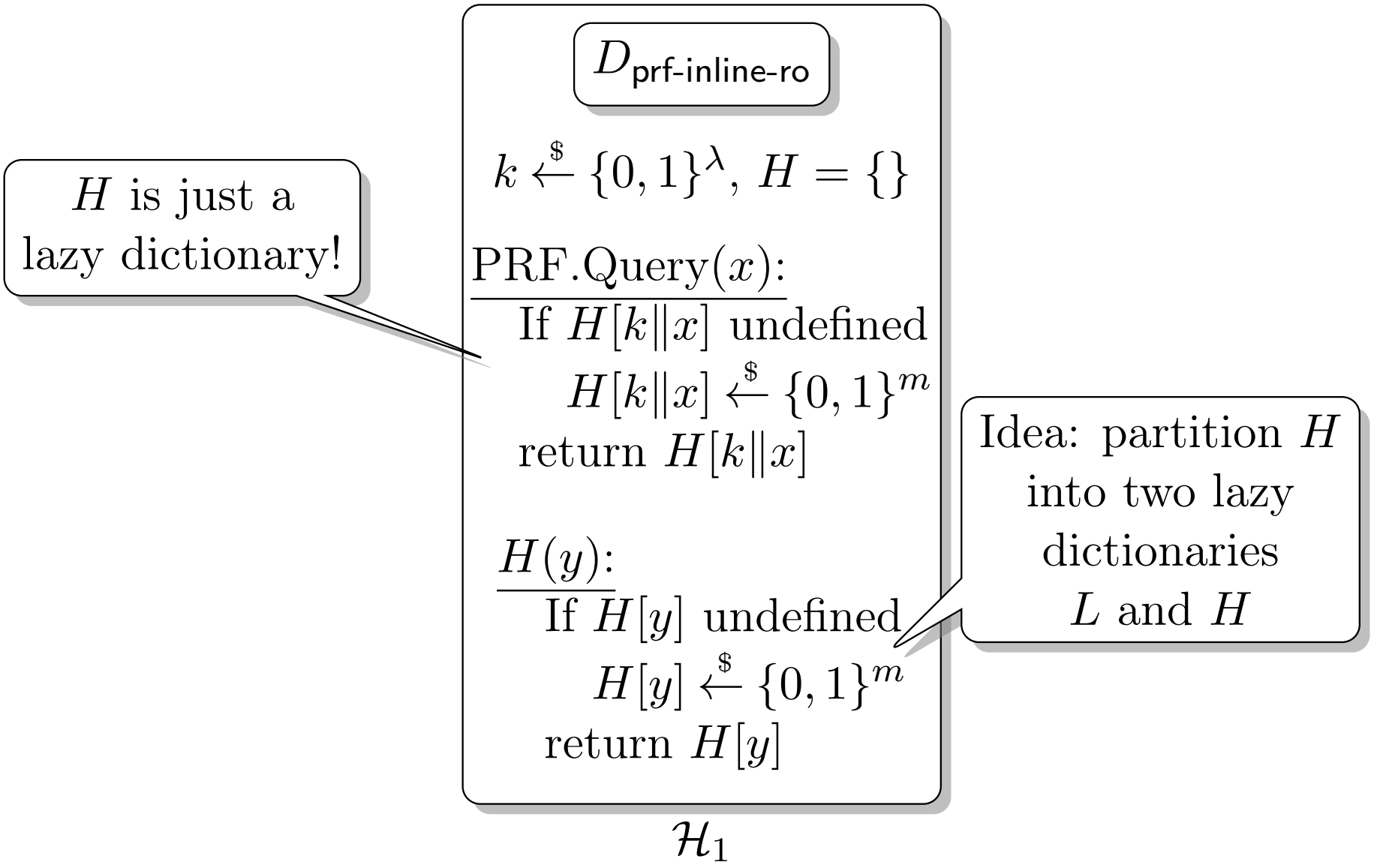


$\mathcal{H}_1$

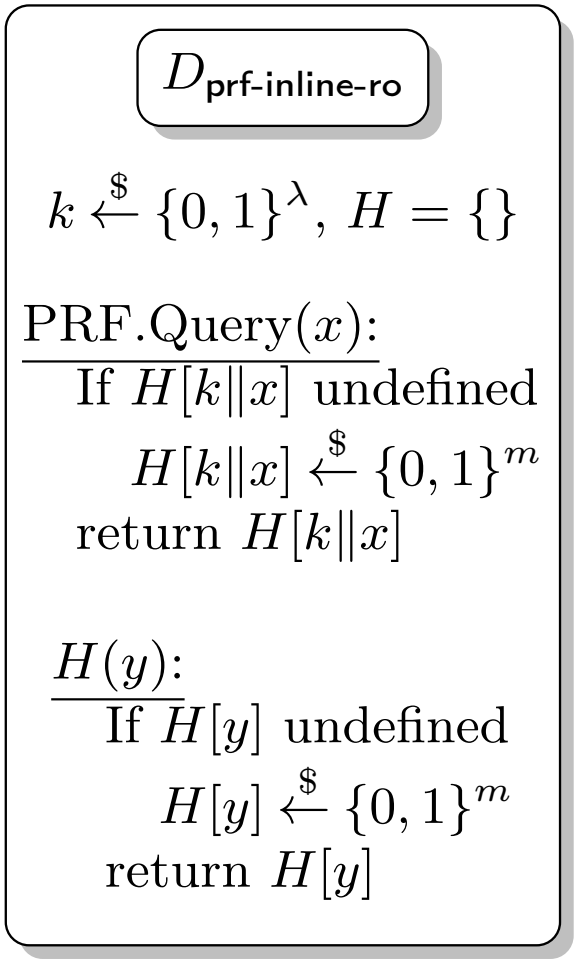
Proof via Hybrid Argument



Proof via Hybrid Argument



Proof via Hybrid Argument



$\mathcal{H}_1$

Proof via Hybrid Argument

$D_{\text{prf-inline-ro}}$

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

PRF.Query(x):
  If $H[k\|x]$ undefined
    $H[k\|x] \xleftarrow{\$} \{0,1\}^m$
  return $H[k\|x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_1$

$D_{\text{prf-dict-ro}}$

$\mathcal{H}_2$

Proof via Hybrid Argument

$D_{\text{prf-inline-ro}}$

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

PRF.Query(x):
  If $H[k\|x]$ undefined
    $H[k\|x] \xleftarrow{\$} \{0,1\}^m$
  return $H[k\|x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_1$

$D_{\text{prf-dict-ro}}$

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

$\mathcal{H}_2$

Proof via Hybrid Argument

$D_{\text{prf-inline-ro}}$

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

PRF.Query(x):
  If $H[k\|x]$ undefined
    $H[k\|x] \xleftarrow{\$} \{0,1\}^m$
  return $H[k\|x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_1$

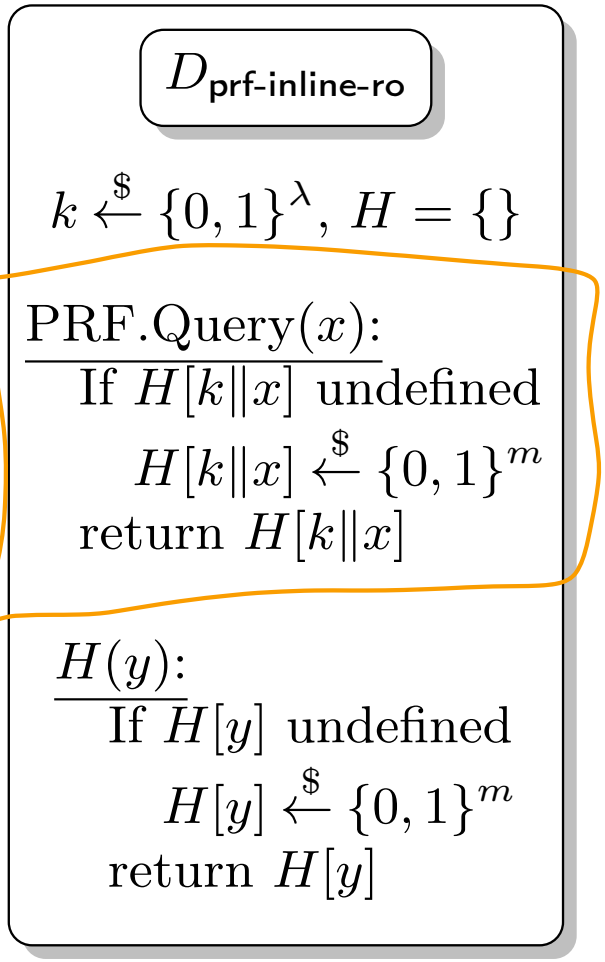
$D_{\text{prf-dict-ro}}$

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

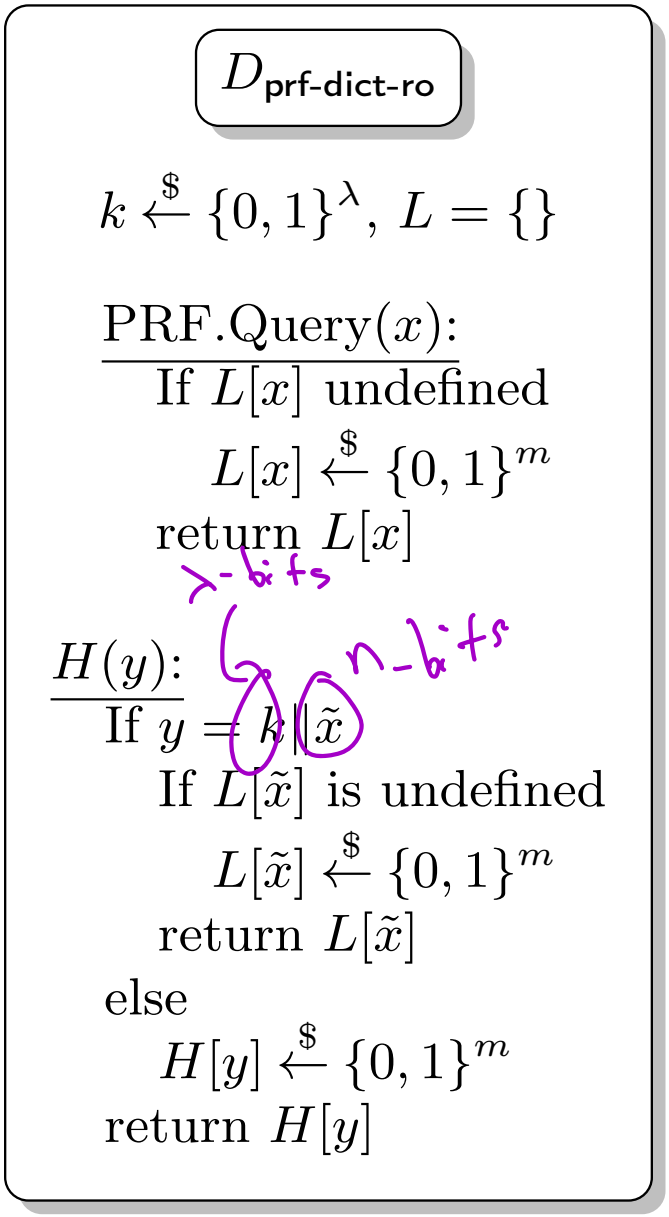
PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$\mathcal{H}_2$

Proof via Hybrid Argument



$\mathcal{H}_1$



$\mathcal{H}_2$

Proof via Hybrid Argument

$D_{\text{prf-inline-ro}}$

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

PRF.Query(x):
  If $H[k\|x]$ undefined
    $H[k\|x] \xleftarrow{\$} \{0,1\}^m$
  return $H[k\|x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_1$

$\equiv$

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k\|\tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.

m-bits m-bits

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.
- Any call to the random oracle H from within PRF.Query *always* are of the form $k\|x$.

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.
- Any call to the random oracle H from within PRF.Query *always* are of the form $k\|x$. So the behavior of PRF.Query is identical in both hybrids.

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.
- Any call to the random oracle H from within PRF.Query *always* are of the form $k\|x$. So the behavior of PRF.Query is identical in both hybrids.
- An adversary interacting with random oracle H in $\mathcal{H}_1$ makes arbitrary queries, and only can tell the different between PRF.Query and the random oracle if they query $y = k\|x$.

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.
- Any call to the random oracle H from within PRF.Query *always* are of the form $k\|x$. So the behavior of PRF.Query is identical in both hybrids.
- An adversary interacting with random oracle H in $\mathcal{H}_1$ makes arbitrary queries, and only can tell the different between PRF.Query and the random oracle if they query $y = k\|x$.
- In $\mathcal{H}_2$, this statement is identical!

□

Proof via Hybrid Argument

Claim 3

$\mathcal{H}_1$ and $\mathcal{H}_2$ are identical distributions.

Proof.

- In $\mathcal{H}_1$, we directly stored queried the dictionary H at $k\|x$ on any input x.
- In $\mathcal{H}_2$, values of the form $H[k\|x]$ are now stored in $L[x]$.
- Any call to the random oracle H from within PRF.Query *always* are of the form $k\|x$. So the behavior of PRF.Query is identical in both hybrids.
- An adversary interacting with random oracle H in $\mathcal{H}_1$ makes arbitrary queries, and only can tell the different between PRF.Query and the random oracle if they query $y = k\|x$.
- In $\mathcal{H}_2$, this statement is identical!
- So $\mathcal{H}_1$ and $\mathcal{H}_2$ are identical.

□

Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

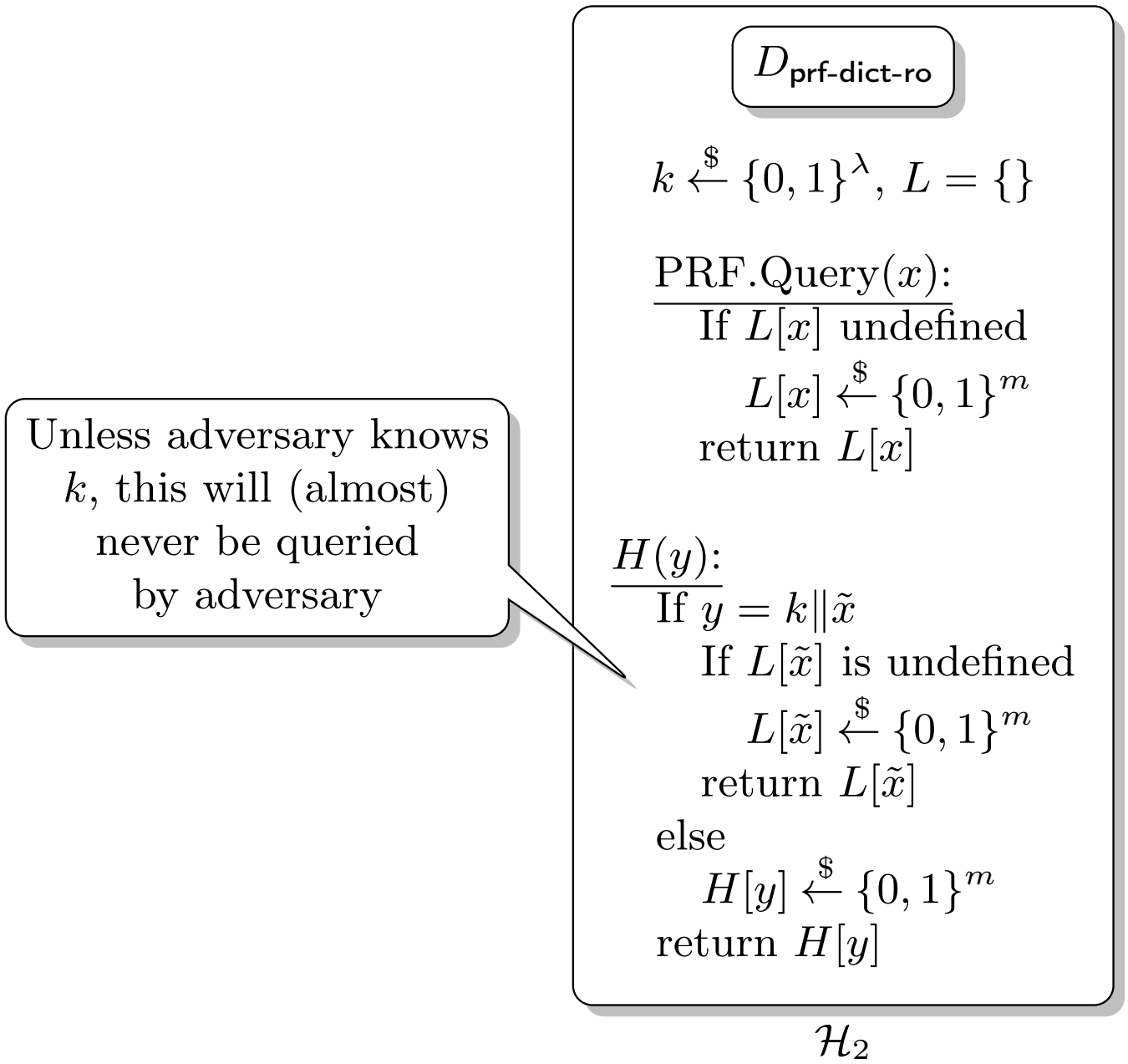
$k \xleftarrow{\$} \{0,1\}^\lambda$, $L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

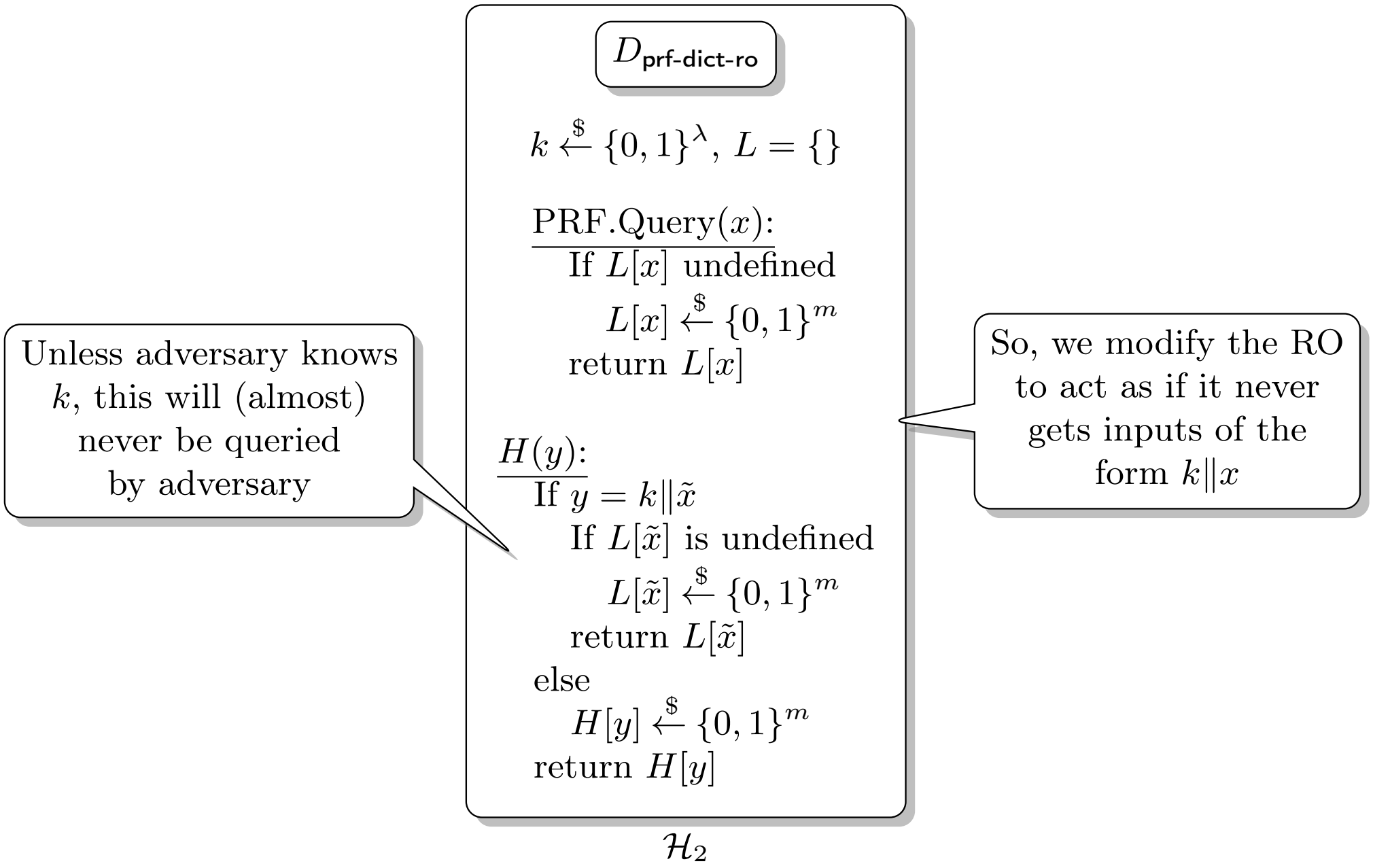
$H(y)$:
  If $y = k \| \tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

Proof via Hybrid Argument



Proof via Hybrid Argument



Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k\|\tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

Unless adversary knows k, this will (almost) never be queried by adversary

So, we modify the RO to act as if it never gets inputs of the form $k\|x$

However, we will track if RO is queried on an input $k\|x$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

$D_{\text{prf-dict-ro-bad}}$

$\mathcal{H}_3$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$\mathcal{H}_3$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    If $L[\tilde{x}]$ is undefined
      $L[\tilde{x}] \xleftarrow{\$} \{0,1\}^m$
    return $L[\tilde{x}]$
  else
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_2$

$D_{\text{prf-dict-ro-bad}}$

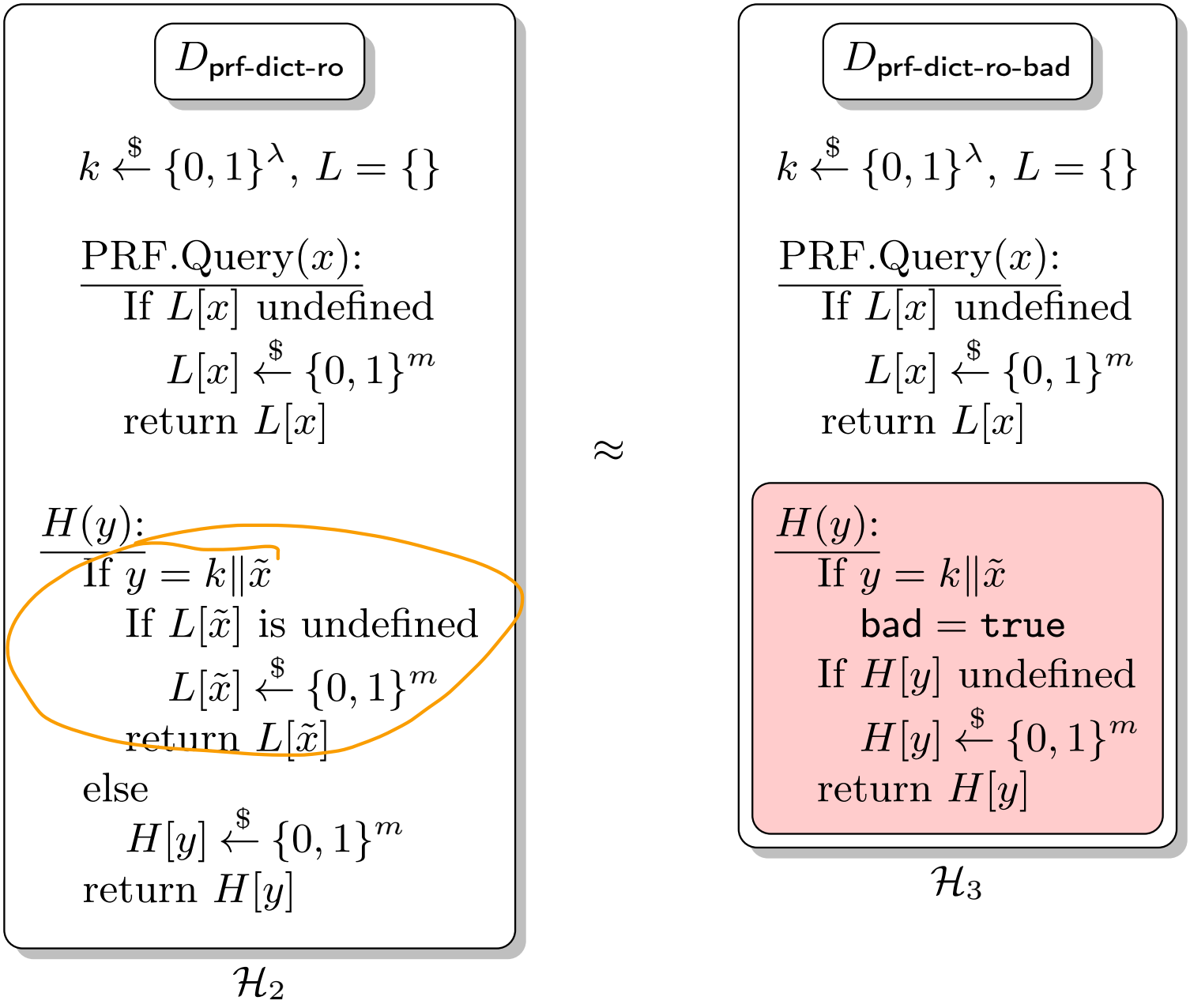
$k \xleftarrow{\$} \{0,1\}^\lambda$, $L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    bad = `true`
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

Proof via Hybrid Argument



Proof via Hybrid Argument

Claim 4

$\mathcal{H}_2$ and $\mathcal{H}_3$ are computationally indistinguishable.

Proof via Hybrid Argument

Claim 4

$\mathcal{H}_2$ and $\mathcal{H}_3$ are computationally indistinguishable.

Proof.

□

Proof via Hybrid Argument

Claim 4

$\mathcal{H}_2$ and $\mathcal{H}_3$ are computationally indistinguishable.

Proof.

- An adversary can distinguish between $\mathcal{H}_2$ and $\mathcal{H}_3$ if and only if they query H at $y = k \| \widetilde{x}$.

□

Proof via Hybrid Argument

Claim 4

$\mathcal{H}_2$ and $\mathcal{H}_3$ are computationally indistinguishable.

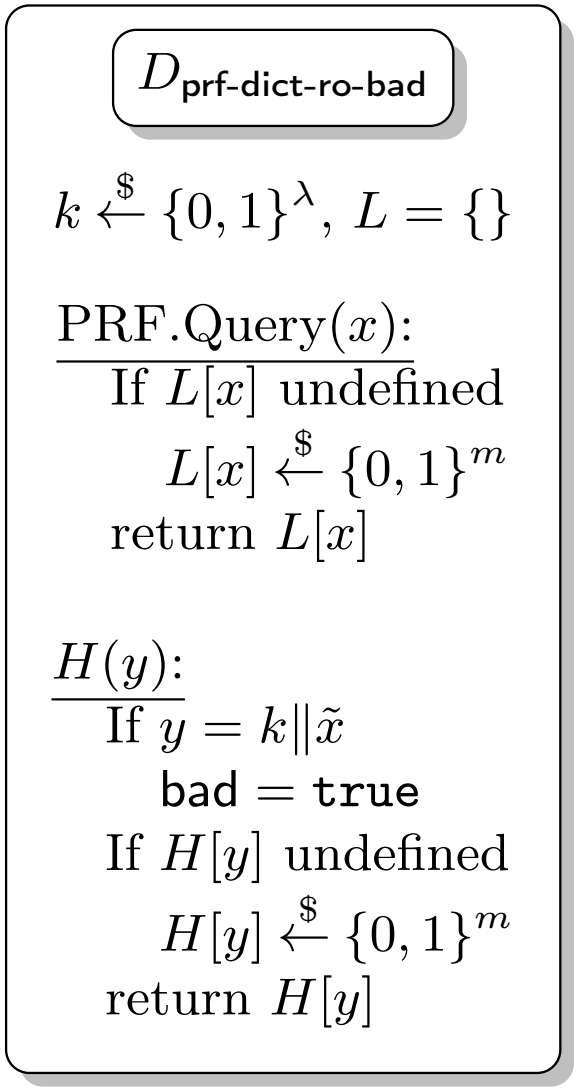
Proof.

- An adversary can distinguish between $\mathcal{H}_2$ and $\mathcal{H}_3$ if and only if they query H at $y = k \| \widetilde{x}$.
- Since $k \xleftarrow{\$} \{0,1\}^\lambda$ and k is hidden from any adversary, the probability and adversary can guess k is $2^{-\lambda}$, which is negligible.

$\leq 2^{-\lambda}(1 - 2^{-m}) \leq 2^{-\lambda}$

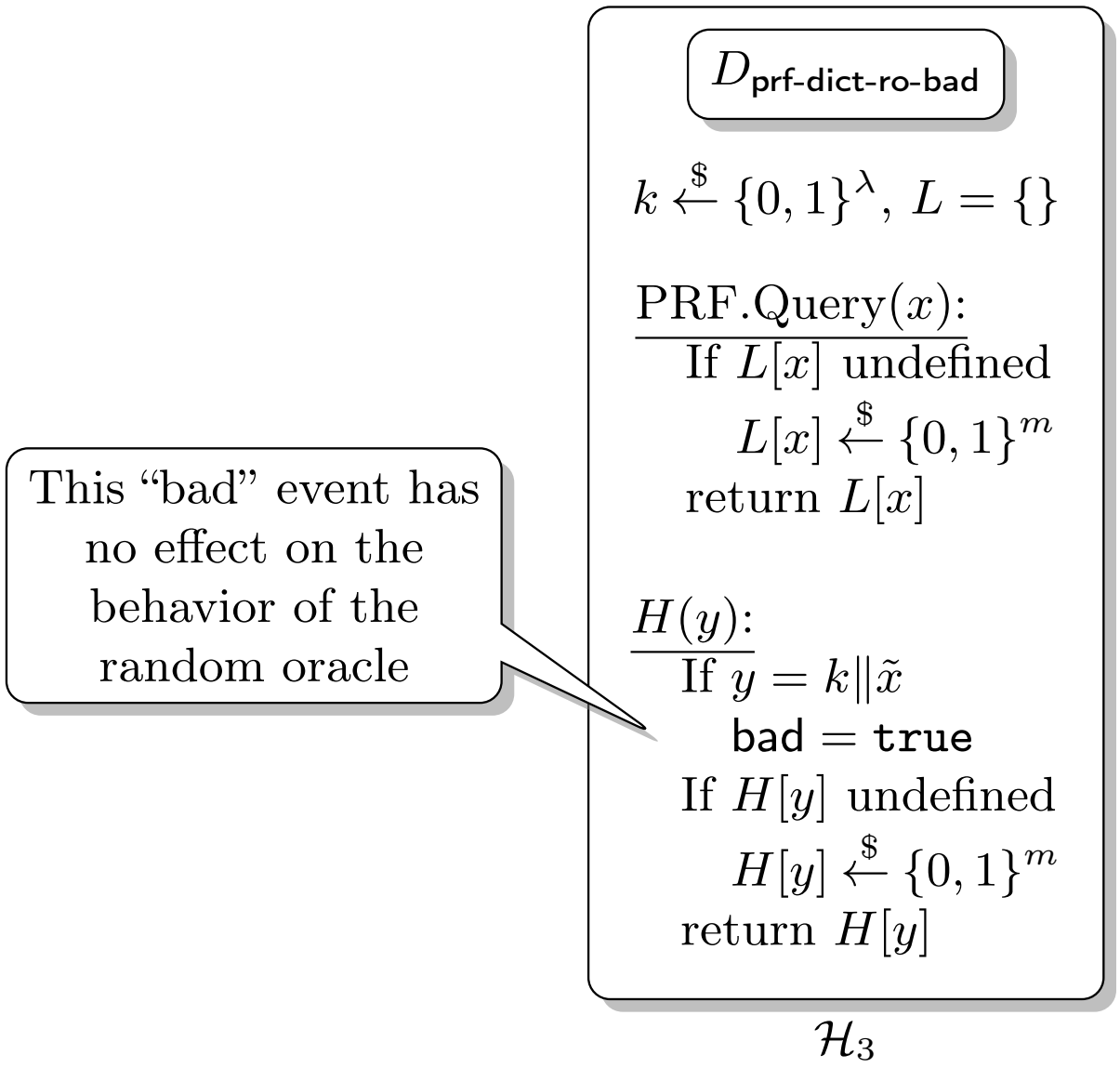
□

Proof via Hybrid Argument

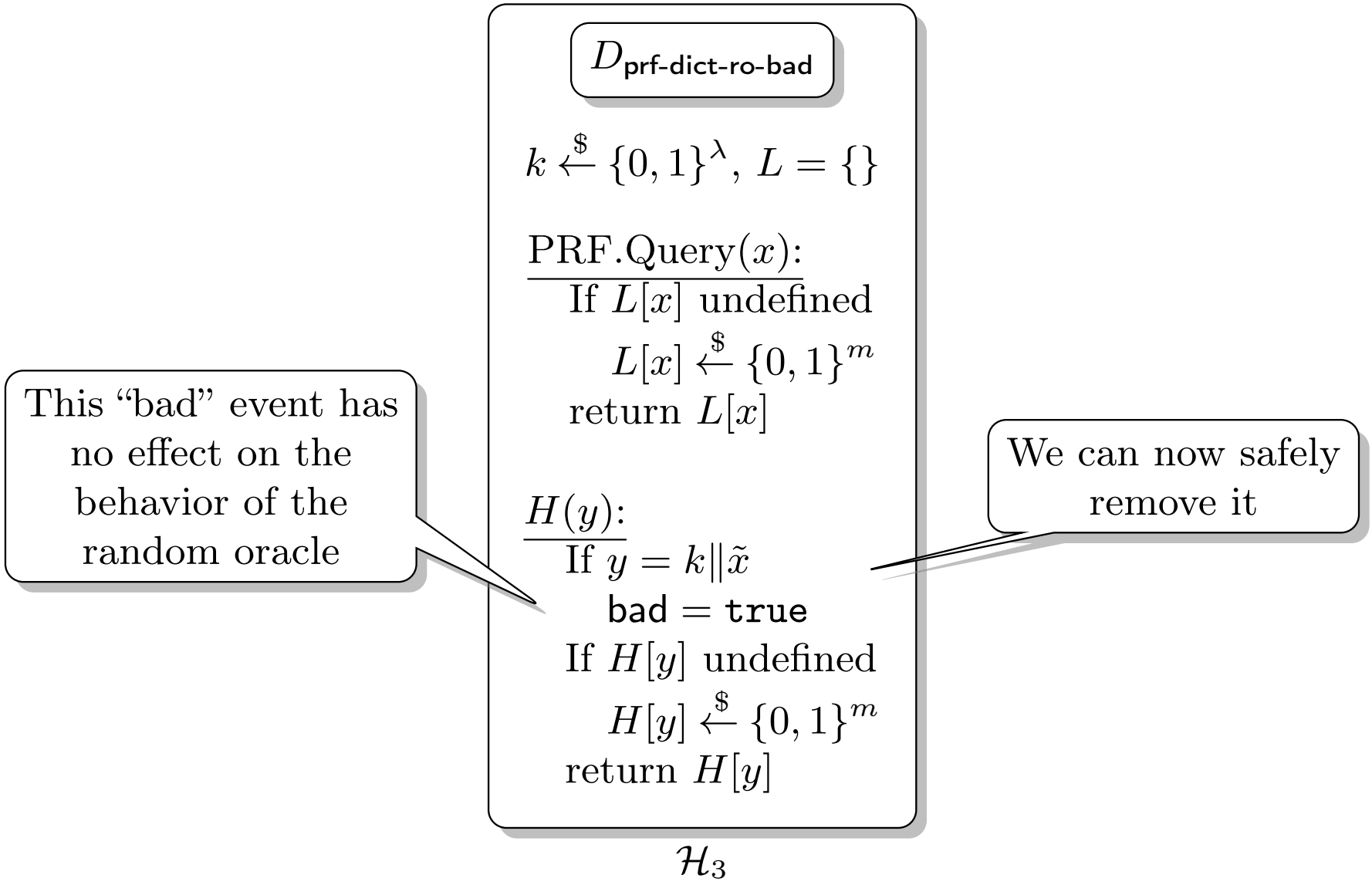


$\mathcal{H}_3$

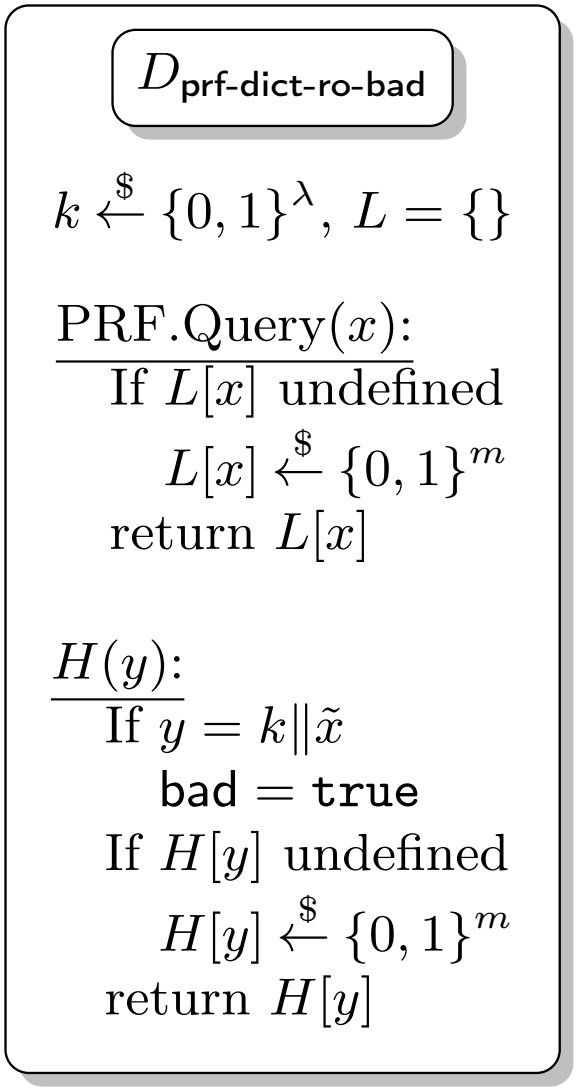
Proof via Hybrid Argument



Proof via Hybrid Argument



Proof via Hybrid Argument



$\mathcal{H}_3$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    bad = true
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

$D_{\text{prf-rand+ro}}$

$\mathcal{H}_4$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    $\mathsf{bad} = \mathtt{true}$
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

$D_{\text{prf-rand+ro}}$

$L = \{\}$

$\mathcal{H}_4$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    $\mathsf{bad} = \mathtt{true}$
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

$D_{\text{prf-rand+ro}}$

$L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$\mathcal{H}_4$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    bad = true
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

$D_{\text{prf-rand+ro}}$

$L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_4$

Proof via Hybrid Argument

$D_{\text{prf-dict-ro-bad}}$

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

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $y = k \| \tilde{x}$
    bad = true
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_3$

$\equiv$

$D_{\text{prf-rand+ro}}$

$L = \{\}$

PRF.Query(x):
  If $L[x]$ undefined
    $L[x] \xleftarrow{\$} \{0,1\}^m$
  return $L[x]$

$H(y)$:
  If $H[y]$ undefined
    $H[y] \xleftarrow{\$} \{0,1\}^m$
  return $H[y]$

$\mathcal{H}_4$

Proof via Hybrid Argument

Claim 5

$\mathcal{H}_3$ and $\mathcal{H}_4$ are identical distributions.

Proof via Hybrid Argument

Claim 5

$\mathcal{H}_3$ and $\mathcal{H}_4$ are identical distributions.

Proof.

The "bad" event does not change the behavior of the random oracle, so it has no effect on any adversary trying to distinguish between these two hybrids. □

Proof via Hybrid Argument (Conclusion)

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

PRF Family

$$\mathcal{H}_0 \equiv \mathcal{H}_1 \equiv \mathcal{H}_2 \approx \mathcal{H}_3 \equiv \mathcal{H}_4.$$

PRF sec. Def.

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

$$\mathcal{H}_0 \equiv \mathcal{H}_1 \equiv \mathcal{H}_2 \approx \mathcal{H}_3 \equiv \mathcal{H}_4.$$

- $\mathcal{H}_0$ is the distribution corresponding to the PRF family.

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

$$\mathcal{H}_0 \equiv \mathcal{H}_1 \equiv \mathcal{H}_2 \approx \mathcal{H}_3 \equiv \mathcal{H}_4.$$

- $\mathcal{H}_0$ is the distribution corresponding to the PRF family.
- $\mathcal{H}_4$ is the distribution corresponding to random functions $\mathcal{F}_{n,m}$.

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

$$\mathcal{H}_0 \not\equiv \mathcal{H}_1 \not\equiv \mathcal{H}_2 \approx \mathcal{H}_3 \not\equiv \mathcal{H}_4.$$

$$\approx \quad \approx \qquad \approx$$

- $\mathcal{H}_0$ is the distribution corresponding to the PRF family.
- $\mathcal{H}_4$ is the distribution corresponding to random functions $\mathcal{F}_{n,m}$.
- To get from $\mathcal{H}_0$ to $\mathcal{H}_4$, we make a new hybrid $\mathcal{H}_i$ for $i \in [4]$ which only *slightly* changes $\mathcal{H}_{i-1}$.

Proof via Hybrid Argument (Conclusion)

- We have proved that the function family $\{F_k^H : \{0,1\}^n \to \{0,1\}^m\}_{k \in \{0,1\}^\lambda}$ such that $F_k^H(x) := H(k \| x)$ is a secure PRF family.
- We did so via a hybrid argument, which showed that:

$$\mathcal{H}_0 \equiv \mathcal{H}_1 \equiv \mathcal{H}_2 \approx \mathcal{H}_3 \equiv \mathcal{H}_4.$$

- $\mathcal{H}_0$ is the distribution corresponding to the PRF family.
- $\mathcal{H}_4$ is the distribution corresponding to random functions $\mathcal{F}_{n,m}$.
- To get from $\mathcal{H}_0$ to $\mathcal{H}_4$, we make a new hybrid $\mathcal{H}_i$ for $i \in [4]$ which only *slightly* changes $\mathcal{H}_{i-1}$.
- Then we argue that $\mathcal{H}_{i-1}$ and $\mathcal{H}_i$ are *close*.

Next Time

- More discussion on the Random Oracle Model.
- Overview of other Idealized Models.