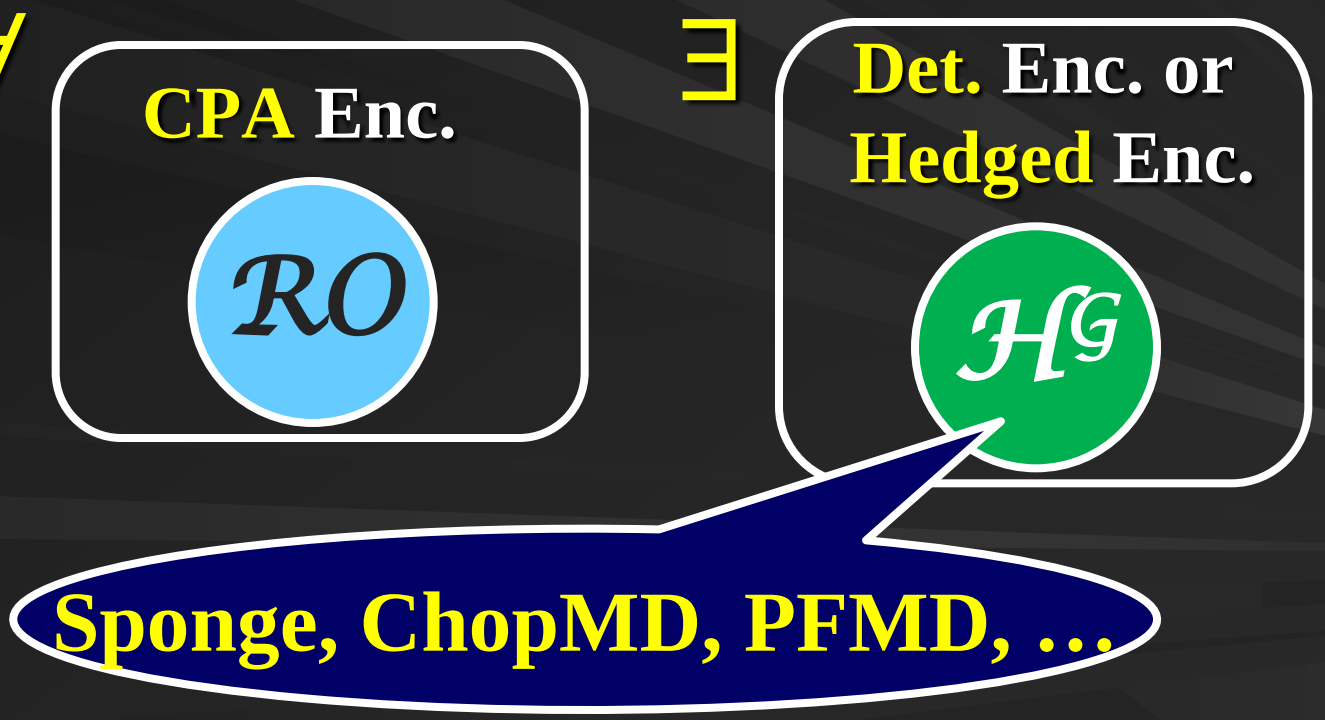
確定的，またはヘッジ暗号における強識別不可能ハッシュ関数の適用について

内藤 祐介
(三菱電機)

米山 一樹
(NTT)

本発表の概要

- ランダムオラクル(RO)の**強識別不可能ハッシュ**による置き換えに関する**肯定的な結果**
- **確定的暗号、またはヘッジ暗号**において、置き換え可能性を示した

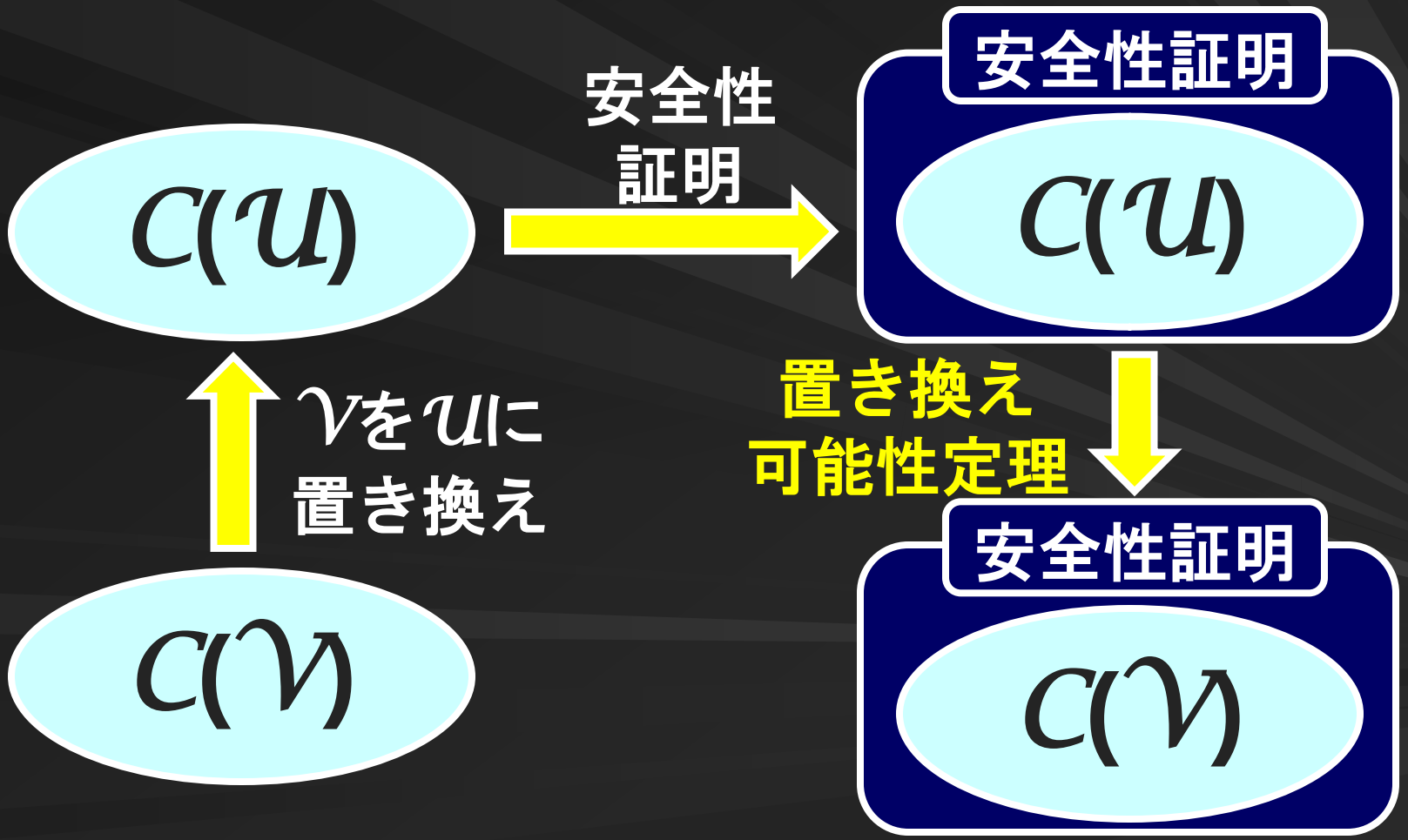


目次

- 導入
  - 強識別不可能性フレームワーク
  - 強識別不可能性ハッシュ関数
  - 関連研究
- 強識別不可能性ハッシュのサルベージ
  - 弱いランダムオラクルモデル
  - リセット強識別不可能性証明
  - 確定的暗号、ヘッジ暗号の構成と証明

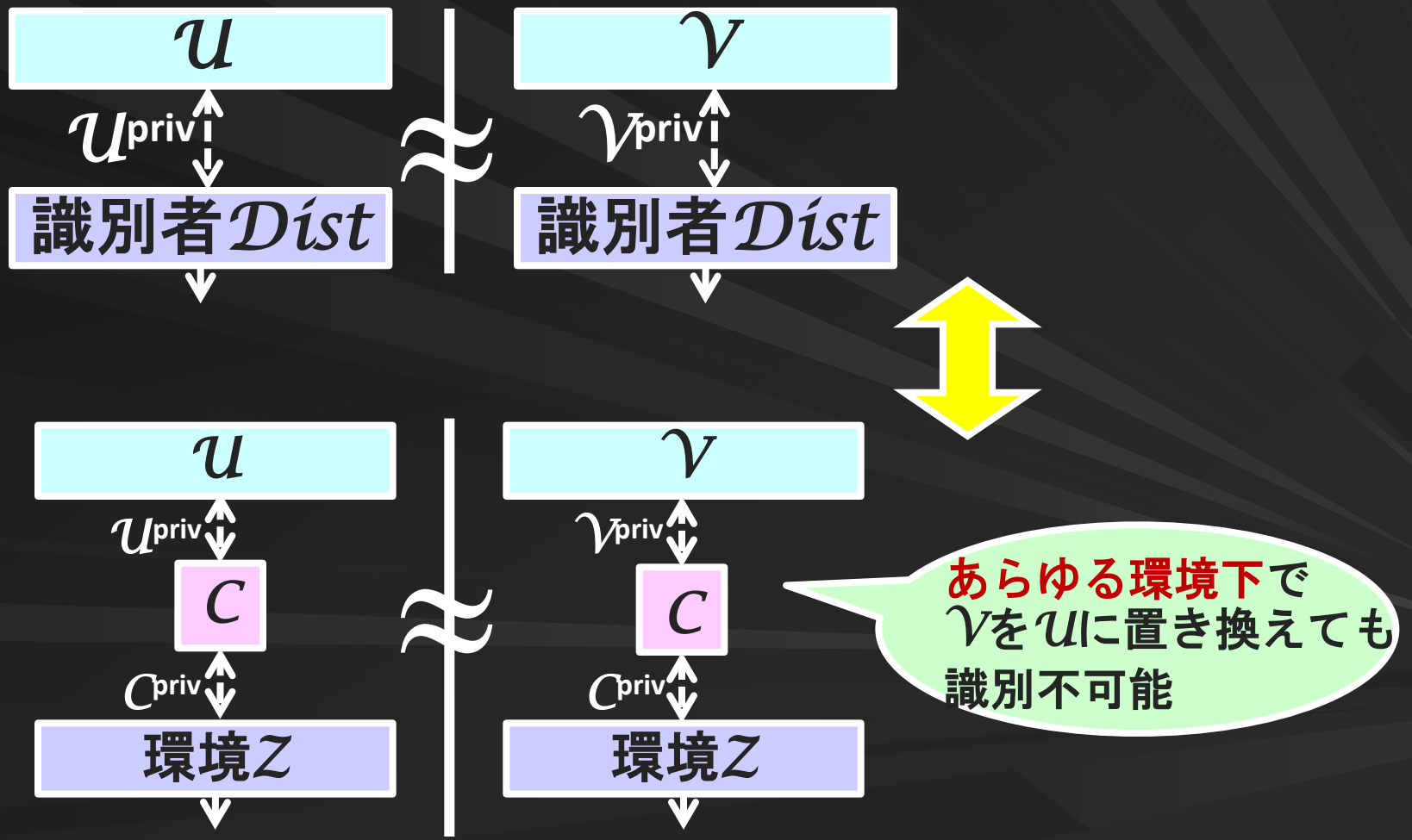
「置き換え」による安全性証明の簡易化

C ： ある暗号システム
$\mathcal{V}$ ： 構造が**複雑**な資源（プロセス）
$\mathcal{U}$ ： 構造が**単純**な資源（プロセス）



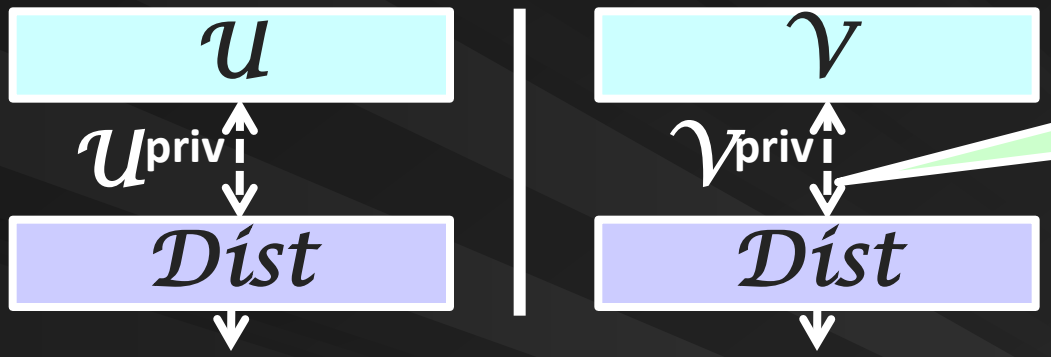
置き換え可能性の証明

- 資源$\mathcal{U}$が資源$\mathcal{V}$との**識別不可能性**を満たすならば$\mathcal{V}$は$\mathcal{U}$と置き換え可能である.



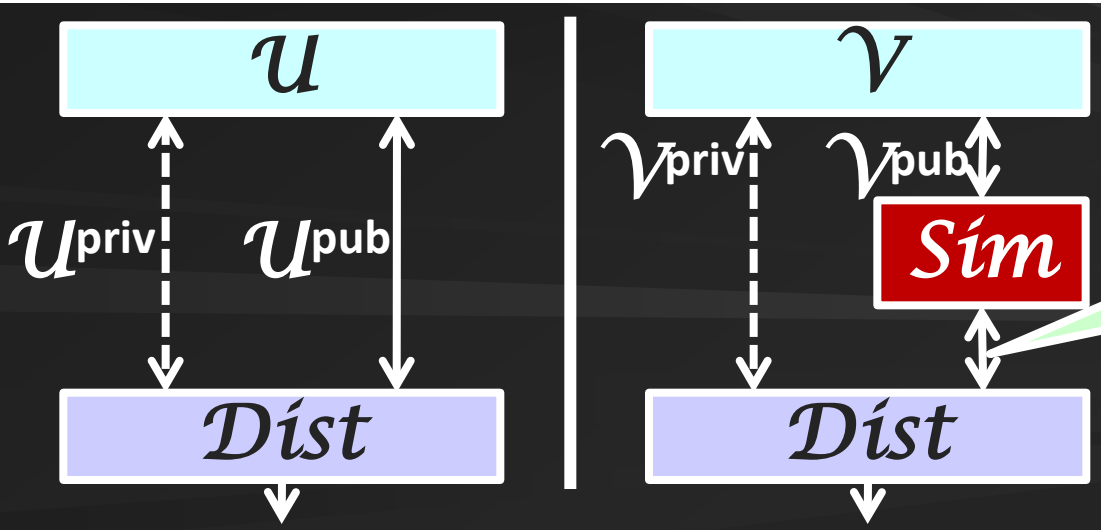
識別不可能性と強識別不可能性

識別不可能性（indistinguishability）



識別者はhonestなやり取りだけを見て識別

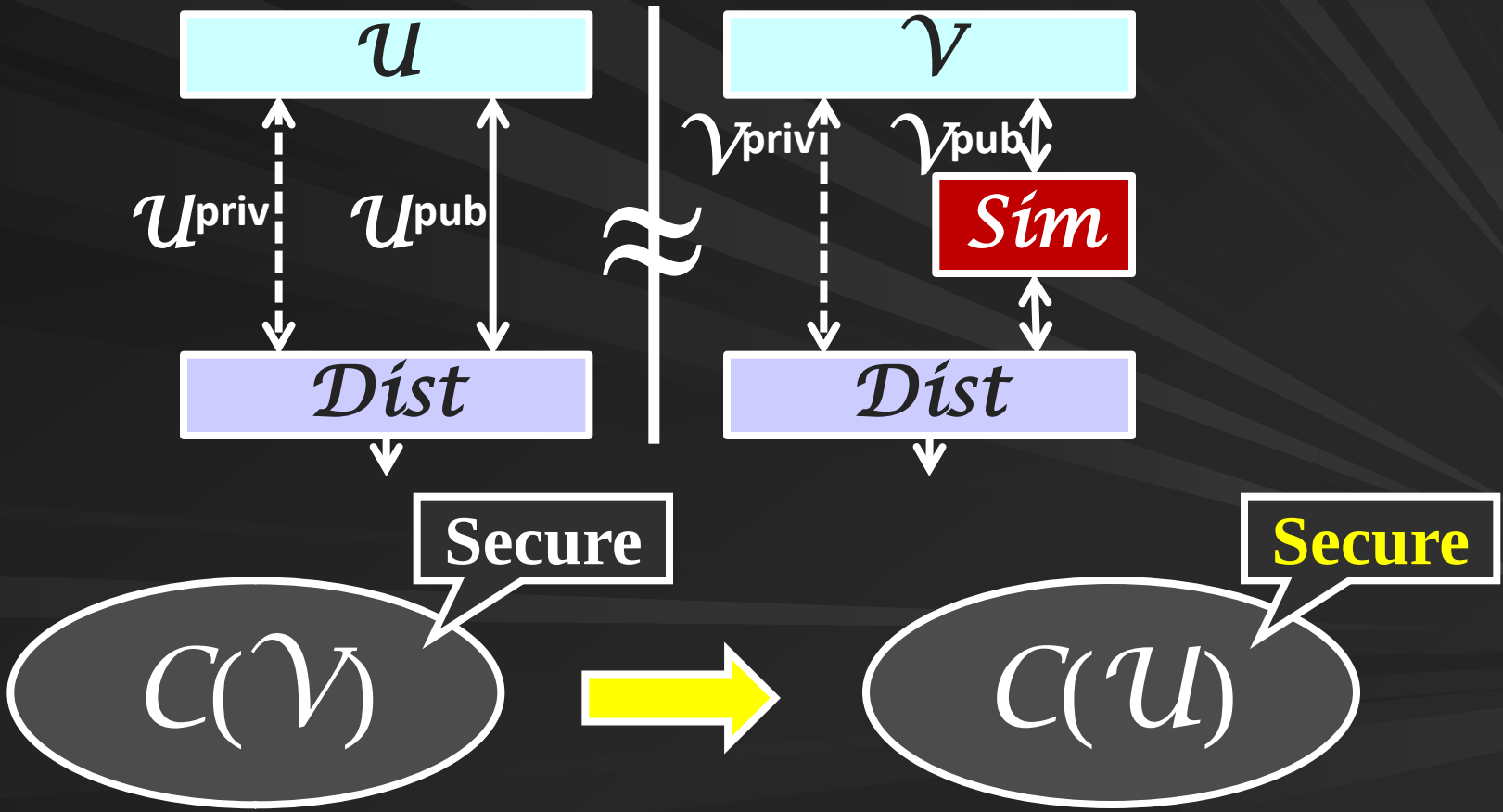
強識別不可能性（indifferentiability）[MRH04]



識別者は攻撃者用I/Fも観察可能

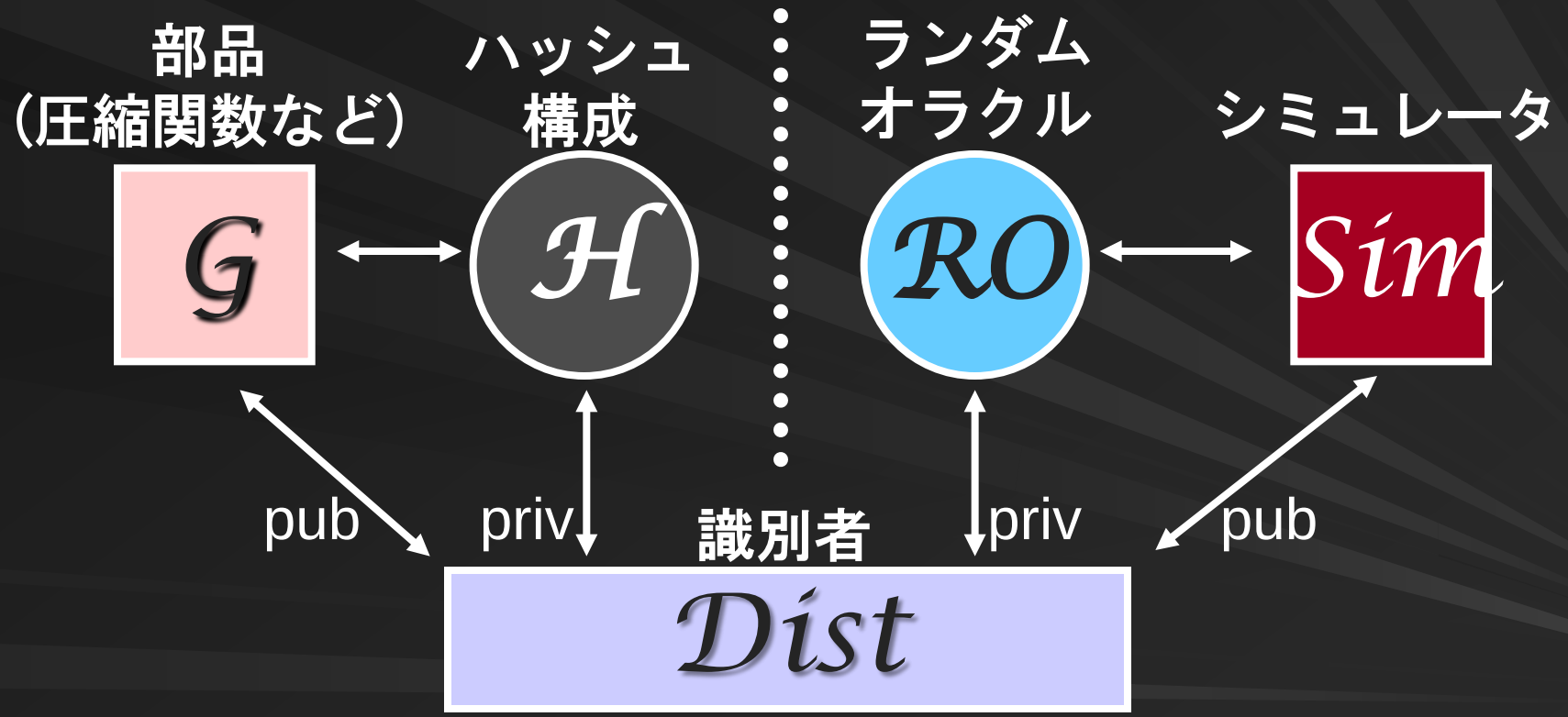
強識別不可能性フレームワーク[MRH04]

- $\mathcal{U}$が$\mathcal{V}$と強識別不可能
  ⇒ 任意の暗号プロトコルCに対して、
  $C(\mathcal{V})$が安全ならば$C(\mathcal{U})$も安全



強識別不可能ハッシュ関数 [CDMP05]

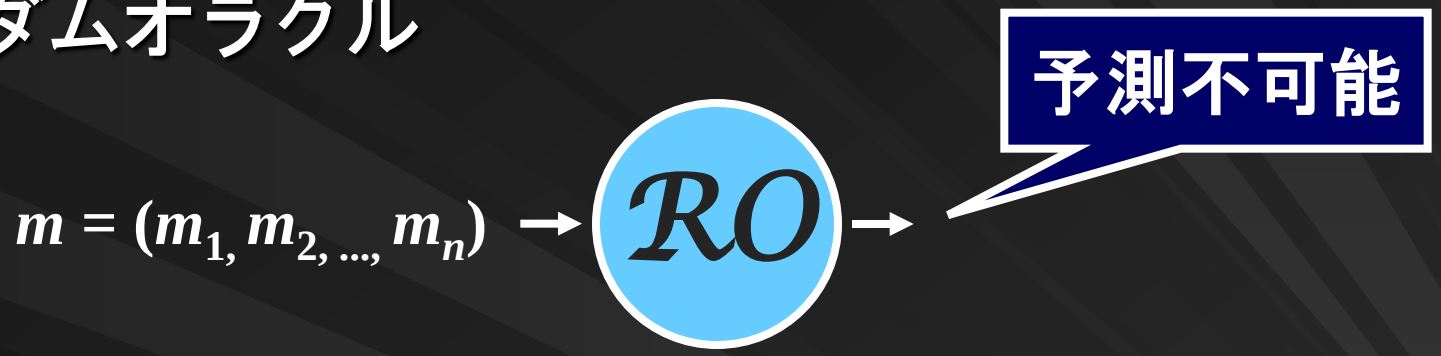
- ランダムオラクル($\mathcal{RO}$)をハッシュ関数($\mathcal{H}$)を使って置き換えたい！



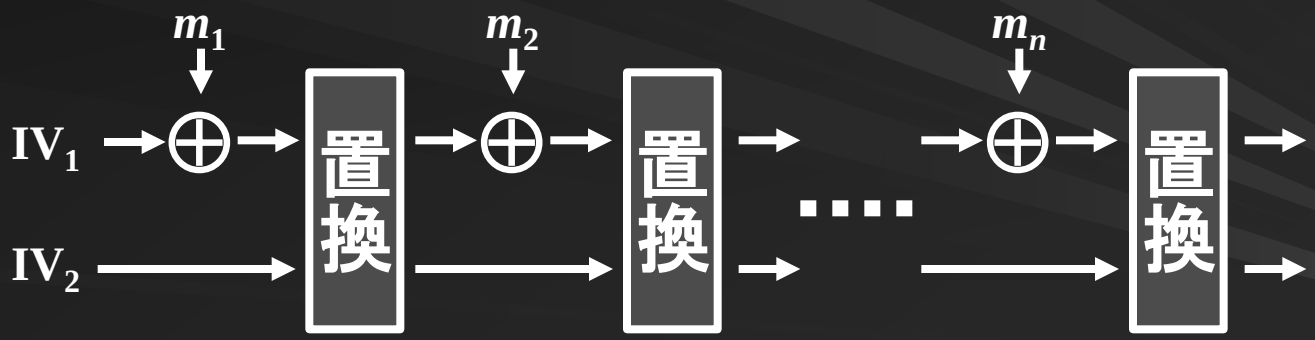
$$| \Pr[\mathcal{Dist}(\mathcal{H},\mathcal{G}) = 1] - \Pr[\mathcal{Dist}(\mathcal{F},\mathcal{S}) = 1] | < \text{negl. iff 強識別不可能}$$

ROと強識別不可能ハッシュ

- ランダムオラクル



- $\mathcal{H}$: **Sponge** (SHA-3に採用), $\mathcal{G}$: ランダム置換



任意の暗号プロトコルにおいて置き換え可能だと信じられていた

置き換え可能性の否定的結果 [RSS11]

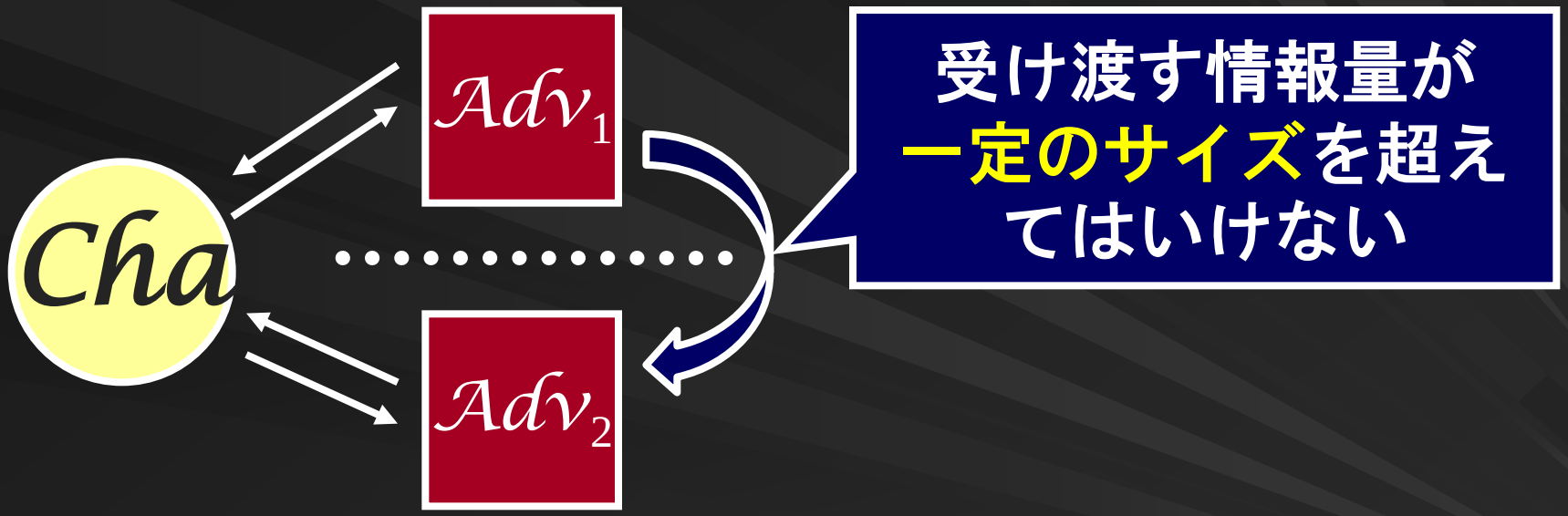
- ∃暗号プロトコル C s.t. $\mathcal{H}^G$が$\mathcal{RO}$と強識別不可能なのに、$C(\mathcal{RO})$は安全だが**$C(\mathcal{H}^G)$は安全でない**
  - C : ストレージ証明チャレンジレスポンスプロトコル

サーバ(mを保持)		クライアント(mを保持)
	c ←	c: ランダムチャレンジ
$z = hash(m\|\|c)$		
	z →	$z = hash(m\|\|c)$をチェック

- *hash*がROならmを知らないサーバは答えられない
  - しかし、*hash*がSpongeの場合、サーバはmを知らなくても***hash(m)*を保持**しておけば正しいzを答えられる！

なぜ置き換え不能なの？

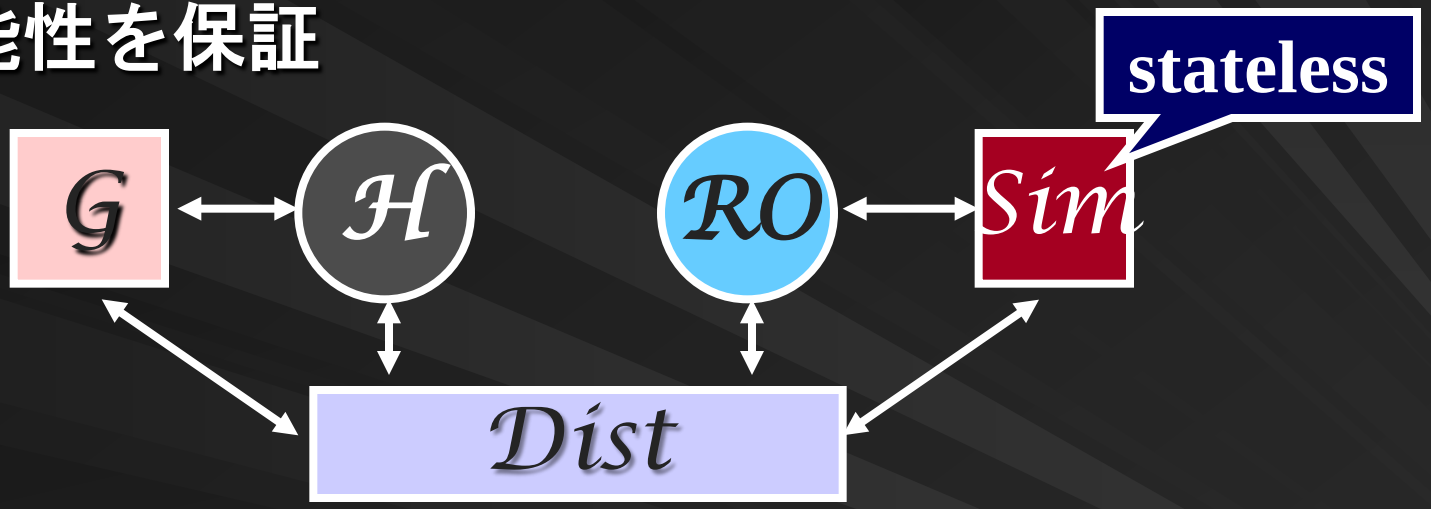
- オリジナルのフレームワークは**複数ステージゲーム**を捉えていない



- 代表的な複数ステージゲーム
  - 確定的暗号、またはヘッジ暗号に対する**選択分布攻撃(Chosen Distribution Attack, CDA)ゲーム**

リセット強識別不可能性 [RSS11]

- 任意の複数ステージゲームに対して置き換え可能性を保証



- しかし。。。
  - one-passなハッシュ関数は証明不可能

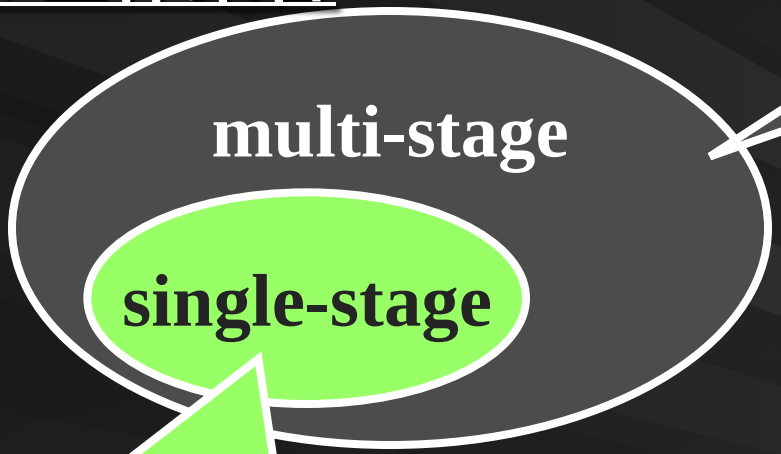
実用的な強識別不可能ハッシュは死んだの?

。。。

研究動機

NEVER NEVER NEVER SURRENDER

Cの全体集合



強識別不可能ハッシュで置き換え可能

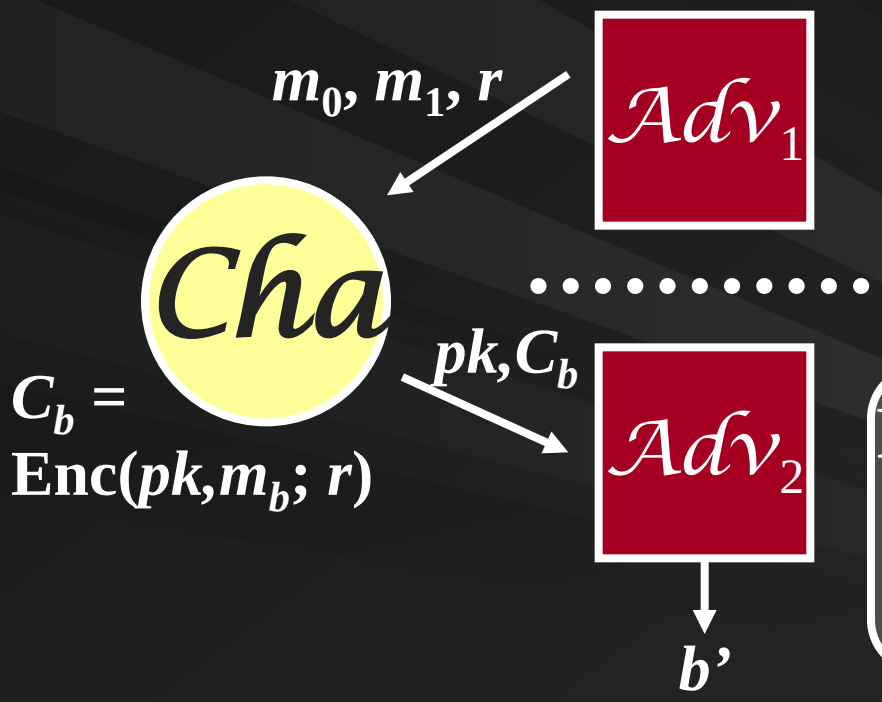
リセット強識別不可能性経由では保証不可能

「全てのCと$\mathcal{HG}$に対して$C(\mathcal{HG})$が安全でない」という意味ではない！

$\mathcal{HG}$で置き換え可能な複数ステージゲームは？

CDAゲームにおける肯定的結果 [RSS11]

- CDAゲーム



- NMAC
  (one-passハッシュ)

ROをNMACに置き換えても
CDA安全な方式が存在する
ことを**直接**示した

- より重要なSpongeやChopMDは？
- 強識別不可能性を活かした解決法は無いのか？

結果

- ROモデルでCPA安全な任意の公開鍵暗号を$\mathcal{HG}$に置き換えた上でCDA安全な確定的暗号、またはヘッジ暗号に（効率を落とすこと無く）変換可能

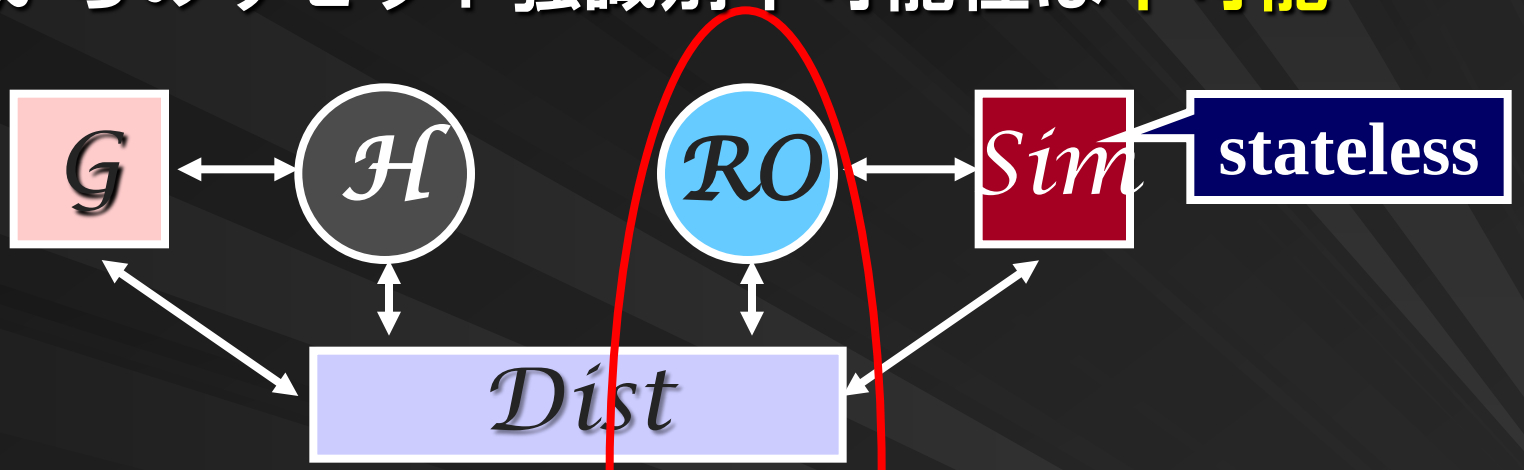
$\mathcal{HG}$：Sponge, ChopMD, PFMD, EMD, MDP, …
(実用的な強識別不可能ハッシュを網羅)

目次

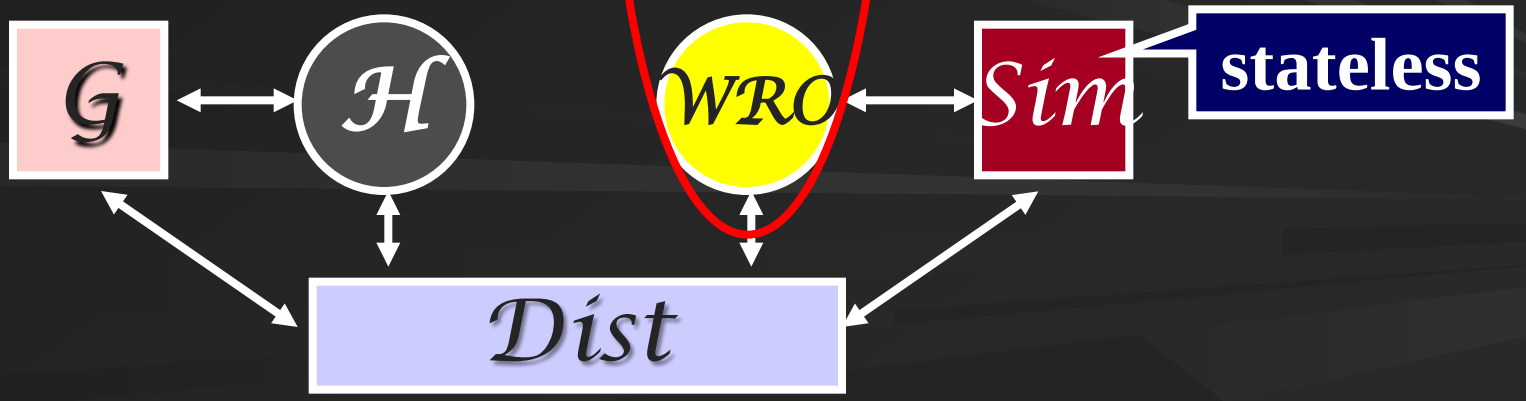
- 導入
  - 強識別不可能性フレームワーク
  - 強識別不可能性ハッシュ関数
  - 関連研究
- 強識別不可能性ハッシュのサルベージ
  - 弱いランダムオラクルモデル
  - リセット強識別不可能性証明
  - 確定的暗号、ヘッジ暗号の構成と証明

アイディア

- ROからのリセット強識別不可能性は**不可能**

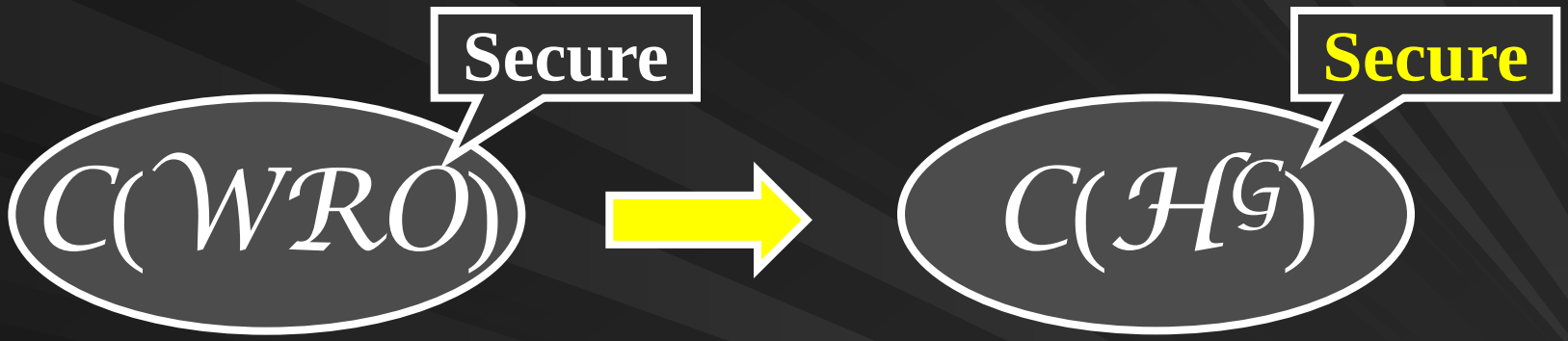


- **弱いRO**からのリセット強識別不可能性を示す



分割して統治せよ

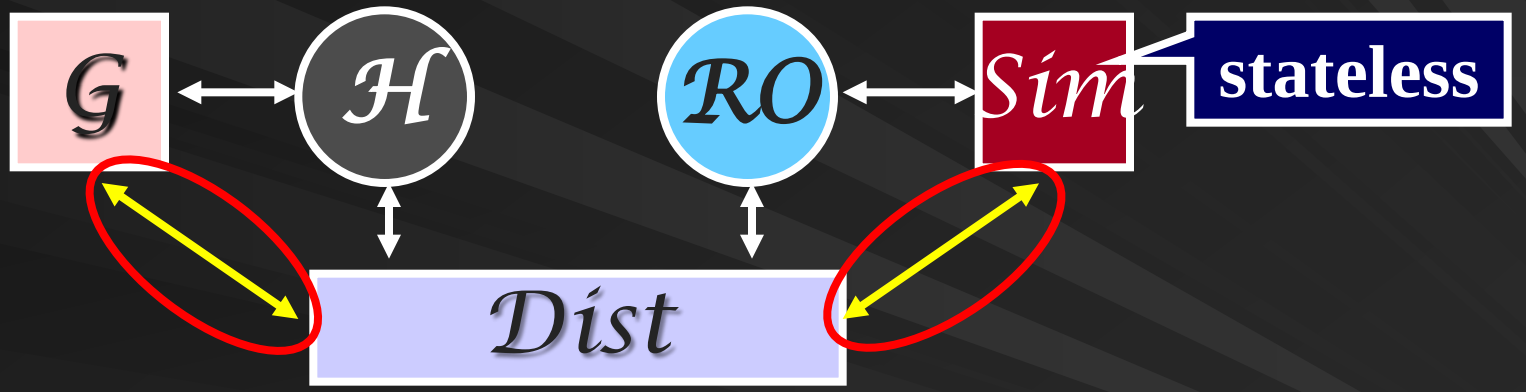
Step 1: $\mathcal{H}^{\mathcal{G}}$がリセット強識別不可能となるような**ちょうどいい**$\mathcal{WRO}$を探す



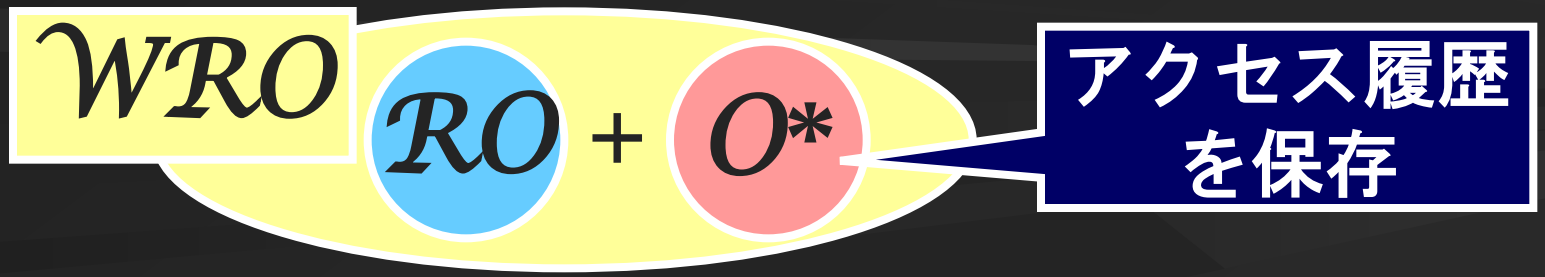
Step 2: ある**一般的変換法**が$\mathcal{WRO}$モデルでCDA安全となることを証明する

WROの定義の方針

- 何が問題だったか？
  - Simがstatelessなので、DistのGへのアクセス履歴をうまくシミュレートできない



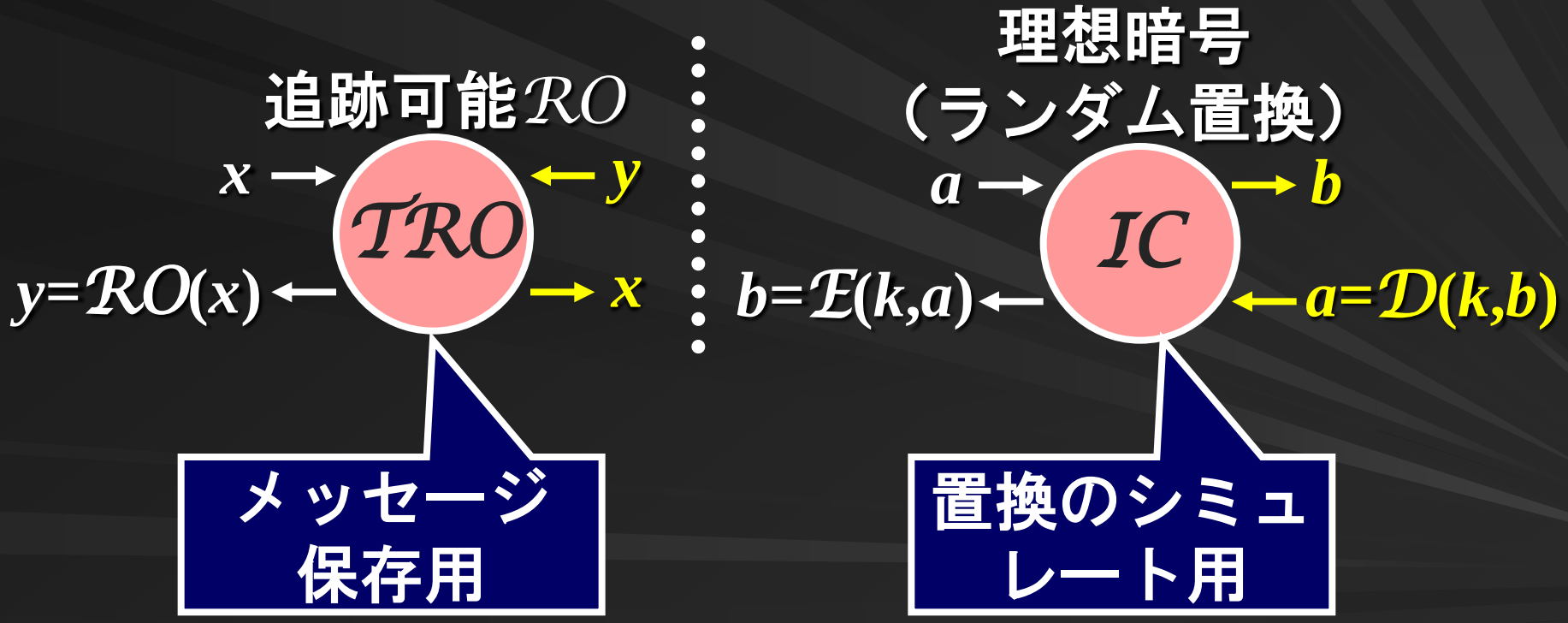
- 基本的な考え方
  - Simのメモリの代わりをするサブオラクルを加える



$\mathcal{WRO}$の具体的定義

- サブオラクルの構成

$\mathcal{D}ist$が$\mathcal{RO}$や置換に聞いた値を抽出できるようにする



(Spongeの場合)

一般的変換法 (REwH1)

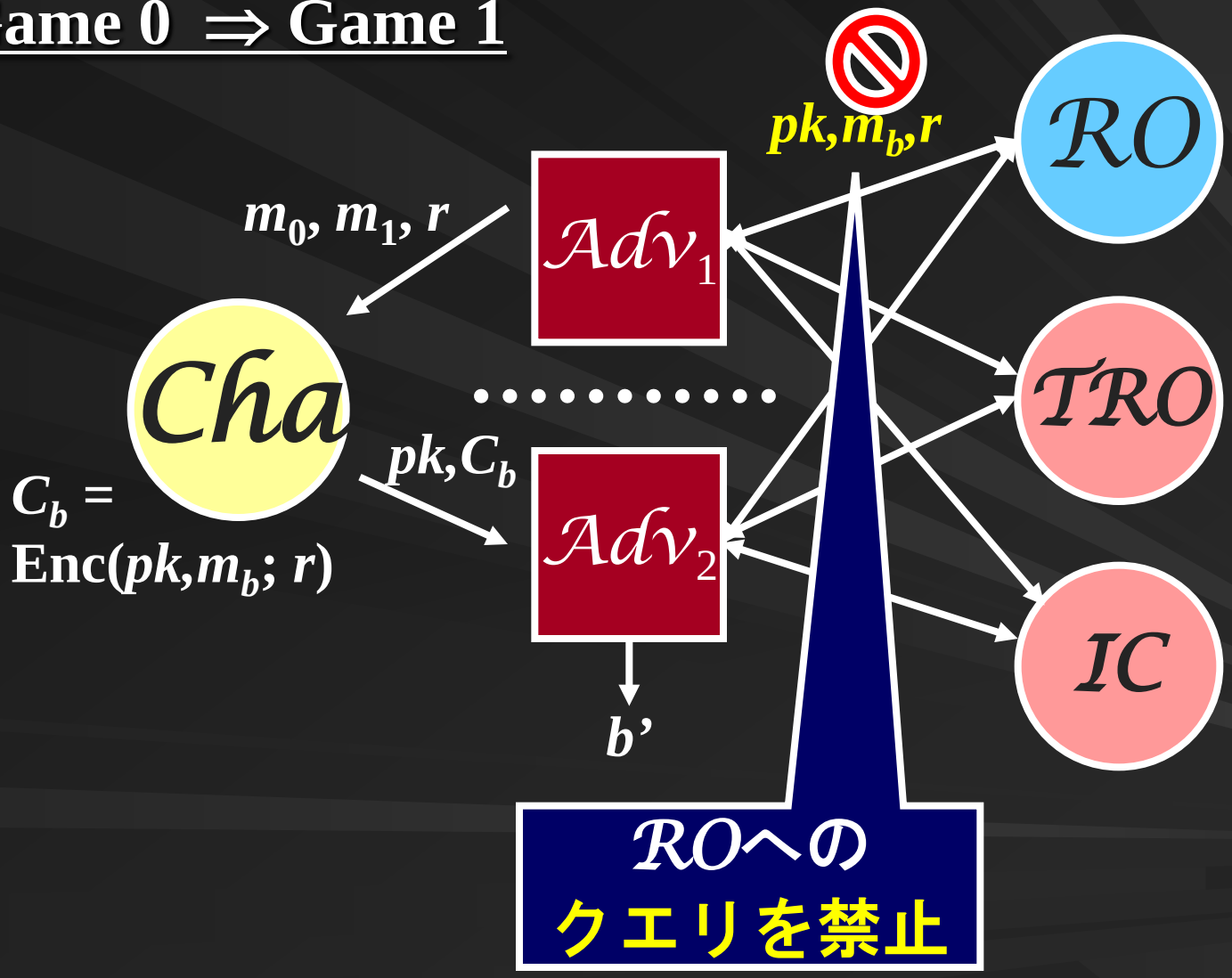
- **Randomized-Encrypt-with-Hash** (REwH1)
  - 普通の公開鍵暗号(KeyGen, Enc, Dec)を確定的暗号、またはヘッジ暗号に変換
  - KeyGenとDecは共通
  - $\mathrm{Enc}_{\mathrm{REwH1}}(pk,m;r) := \mathrm{Enc}(pk,m;\mathcal{RO}(pk\|m\|r))$

暗号文サイズ：完全に一致
増加計算量：ハッシュ1回分

redundancy free!

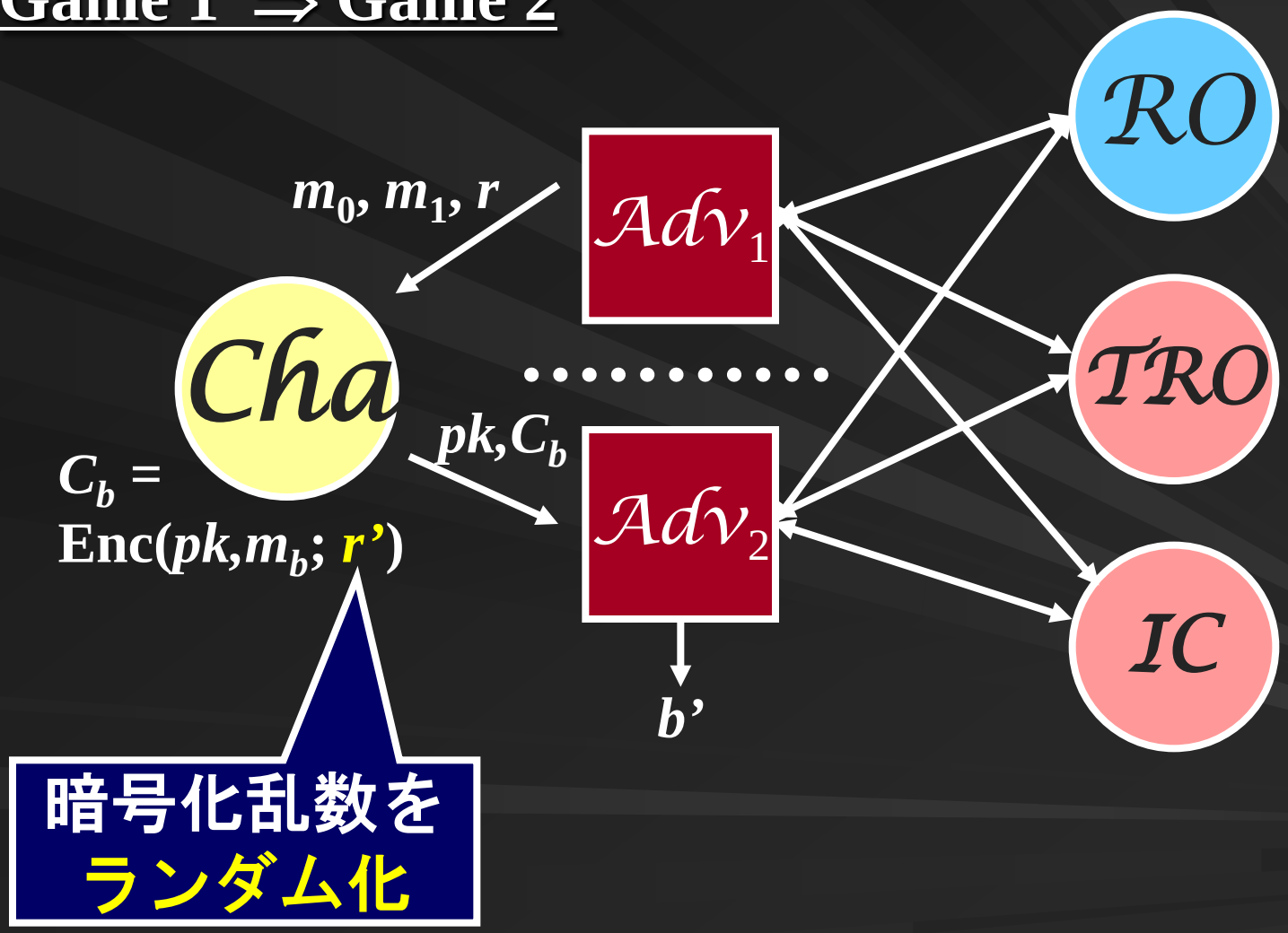
証明の方針（ゲーム変換）

- **Game 0 ⇒ Game 1**



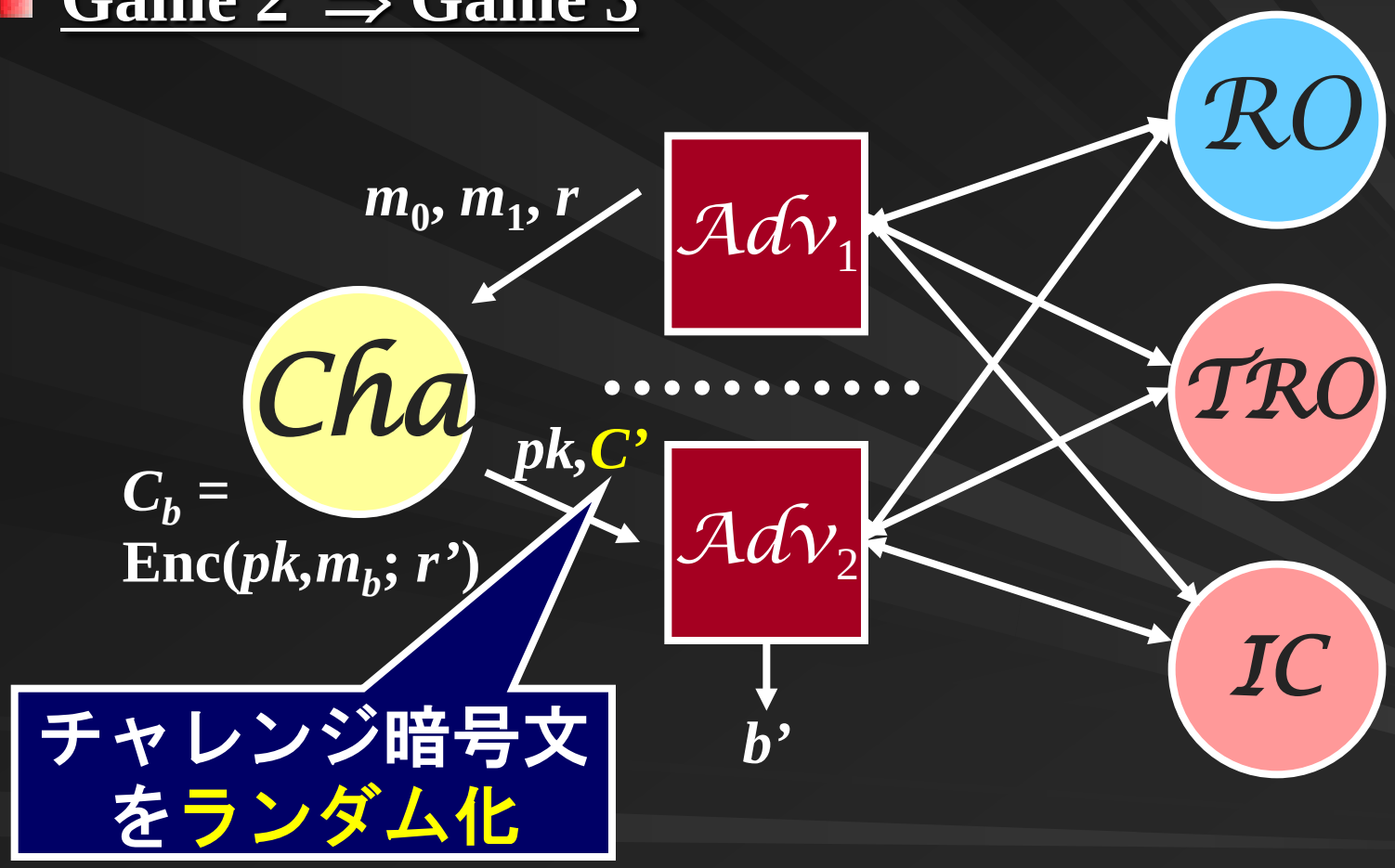
証明の方針（ゲーム変換）

- **Game 1 ⇒ Game 2**



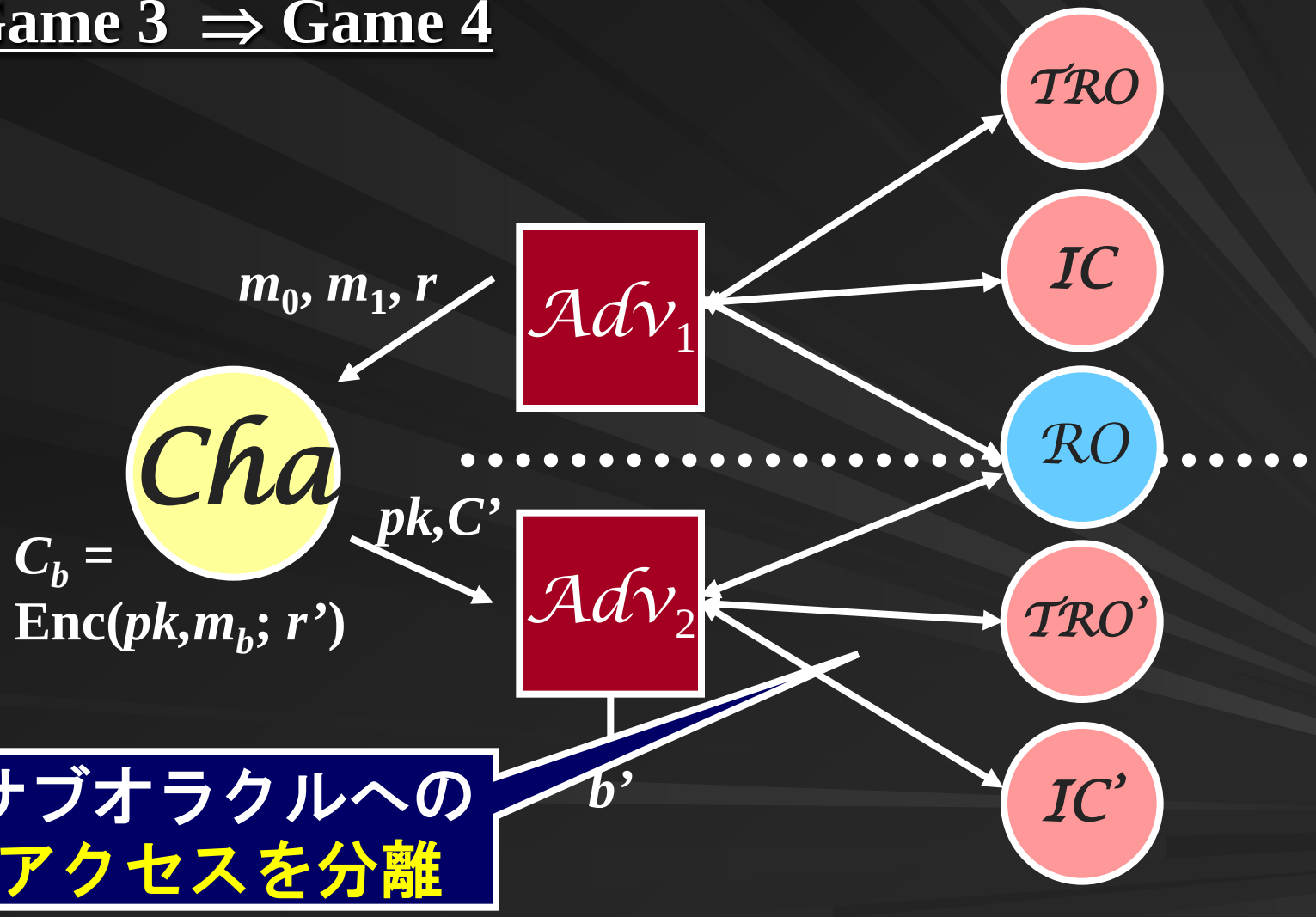
証明の方針（ゲーム変換）

- **Game 2 ⇒ Game 3**



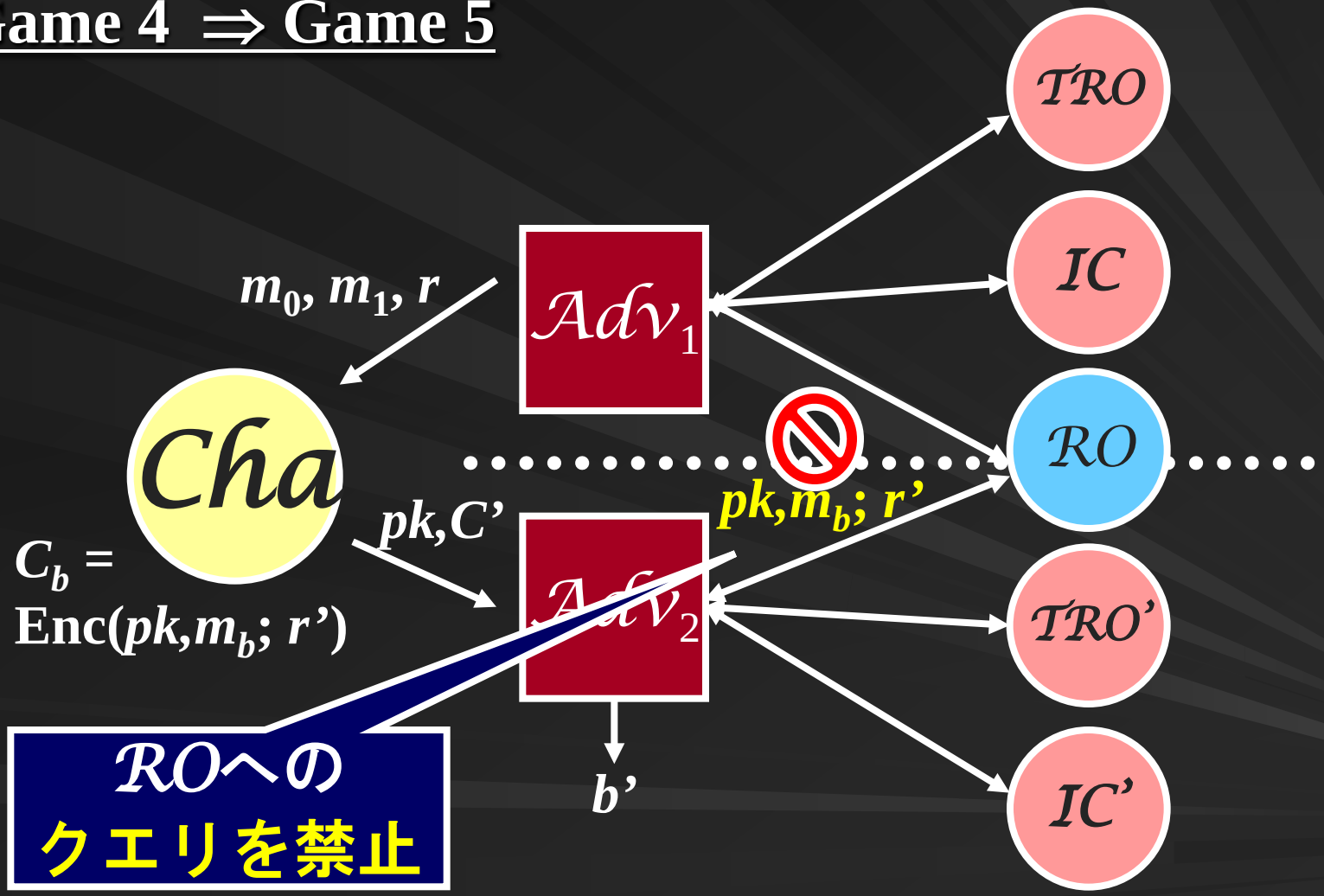
証明の方針（ゲーム変換）

- **Game 3 ⇒ Game 4**

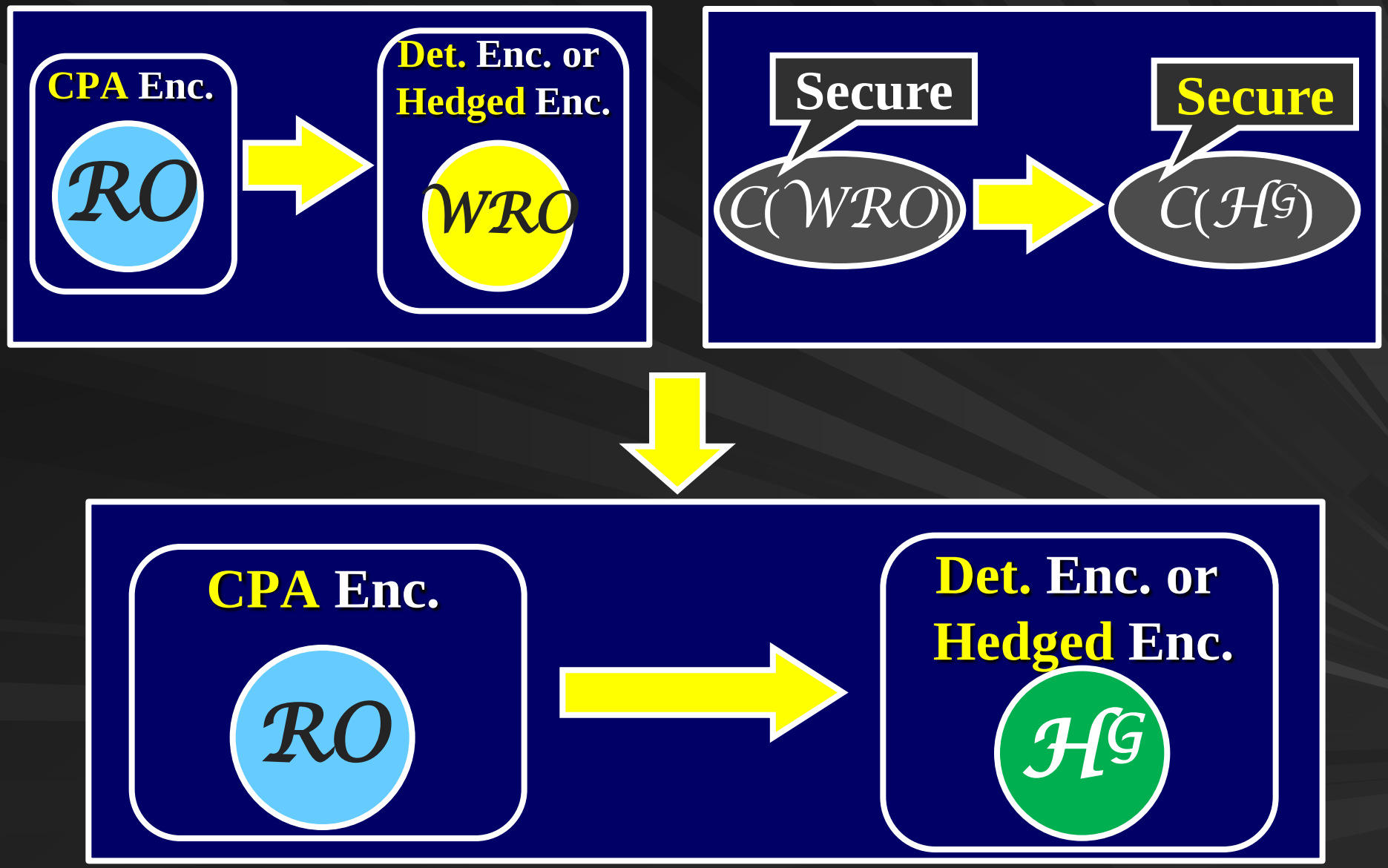


証明の方針（ゲーム変換）

- **Game 4 ⇒ Game 5**

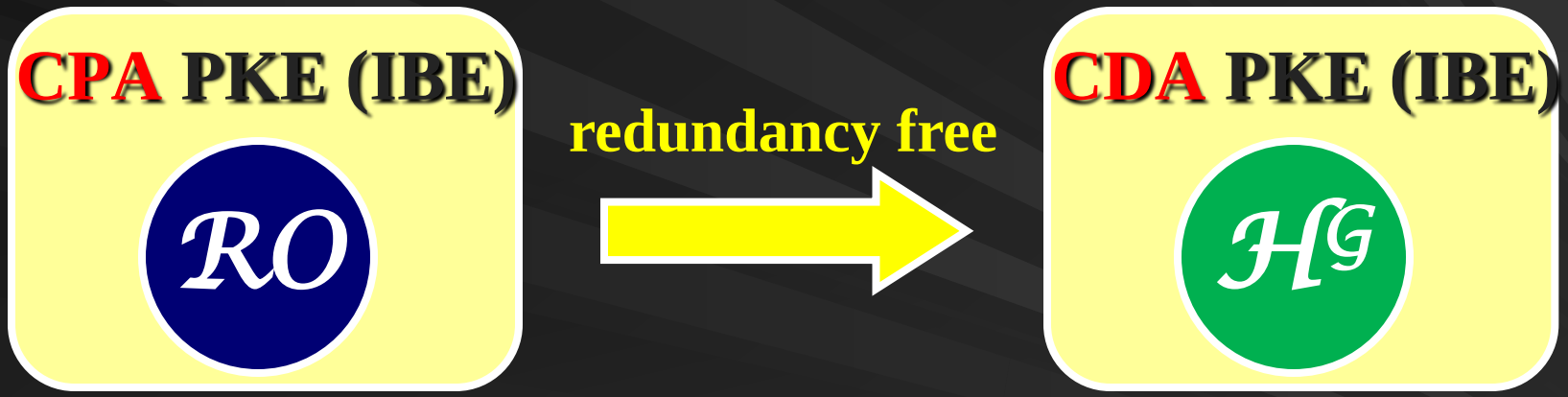


リセット強識別不可能性証明のまとめ



まとめ

- 重要なハッシュ関数 (Sponge, ChopMD, PFMD, EMD, MDP, etc.) はCDAゲームにおいて置き換え可能性を満たす



詳細はePrint 2012/014に掲載

Thank you!