

# CRYE 6138 Security Models

Jorge L. Villar

UBa Cyber Crypto Center, Fall 2025

## Security Statements

### Hard Computational Problems

#### Definition

A problem family  $P$  is hard (on average) if the success probability (or advantage) of any PPTM solving a random problem instance is negligible

Proving that  $P$  is easy means showing the existence of a PPTM solving random instances with a non-negligible probability.

Types of computational problems:

- **Search:** There is a single or a few solutions and they are large objects. E.g. DLOG
- **Flexible:** There are many different solutions.
- **Decision:** The solution is just a single bit. E.g. DDH

### Outline

- 1 Reducing Computational Problems
- 2 Security Statements

### Reductions

Meaning of “ $P$  is hard  $\Rightarrow P'$  is hard”?

Or equivalently, “ $P'$  is easy  $\Rightarrow P$  is easy”

“ $P$  is easy” means there **exists** a PPTM  $\mathcal{A}_P$  that solves  $P$  with a non-negligible probability

Showing only the existence is a non-constructive proof

**Constructive proof:** Explicitly builds a PPTM  $\mathcal{A}_P$  solving  $P$  from any PPTM  $\mathcal{A}_{P'}$  solving  $P'$

$$\mathcal{A}_P = \mathcal{R}[\mathcal{A}_{P'}] \quad \text{or} \quad P \stackrel{\mathcal{R}}{\Rightarrow} P'$$

It reads: “Solving  $P$  reduces to solving  $P'$ ”

# Reduction Example: $CDH \stackrel{\mathcal{R}}{\Rightarrow} DLOG$

Transform any PPTM  $\mathcal{A}_{DLOG}$  into another  $\mathcal{A}_{CDH} = \mathcal{R}[\mathcal{A}_{DLOG}]$

**Syntax:**

$\mathcal{A}_{DLOG}$  receives  $(g, X = g^x)$  and computes  $x$ .

$\mathcal{A}_{CDH}$  receives  $(g, X = g^x, Y = g^y)$  and computes  $Z = g^{xy}$ .

**Reduction:**

$$\mathcal{A}_{CDH}(g, X, Y) = Y^{\mathcal{A}_{DLOG}(g, X)}$$

**Correctness:**

$$\mathcal{A}_{DLOG}(g, X) = x \Rightarrow Y^{\mathcal{A}_{DLOG}(g, X)} = Y^x = (g^y)^x = g^{xy} = Z$$

# Reduction Example: $CDH \stackrel{\mathcal{R}}{\Rightarrow} DLOG$

Common parameters:  $pp = (G, q, g)$

$\mathcal{C}_{CDH}(pp) :$   
 $x, y \leftarrow \mathbb{Z}q;$   
 $X \leftarrow g^x;$   
 $Y \leftarrow g^y;$   
 $Z' \leftarrow \mathcal{A}_{CDH}(pp, X, Y);$

**if  $Z' = g^{xy}$  output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{DLOG}](pp, X, Y) :$   
 $x' \leftarrow \mathcal{A}_{DLOG}(pp, X);$   
**output  $Y^{x'}$ ;**

# Reduction Example: $CDH \stackrel{\mathcal{R}}{\Rightarrow} DLOG$

Common parameters:  $pp = (G, q, g)$

$\mathcal{C}_{CDH}(pp) :$   
 $x, y \leftarrow \mathbb{Z}q;$   
 $X \leftarrow g^x;$   
 $Y \leftarrow g^y;$   
 $Z' \leftarrow \mathcal{A}_{CDH}(pp, X, Y);$

**if  $Z' = g^{xy}$  output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{DLOG}](pp, X, Y) :$   
 $x' \leftarrow \mathcal{A}_{DLOG}(pp, X);$   
**output  $Y^{x'}$ ;**

$\mathcal{C}_{DLOG}(pp) :$   
 $x \leftarrow \mathbb{Z}q;$   
 $X \leftarrow g^x;$

$x' \leftarrow \mathcal{A}_{DLOG}(pp, X);$

**if  $x' = x$  output 1;**  
**else output 0;**

Perfect simulation of  $\mathcal{C}_{DLOG}$  for  $\mathcal{A}_{DLOG}$

# Reduction Example: $CDH \stackrel{\mathcal{R}}{\Rightarrow} DLOG$

Common parameters:  $pp = (G, q, g)$

$\mathcal{C}_{CDH}(pp) :$   
 $x, y \leftarrow \mathbb{Z}q;$   
 $X \leftarrow g^x;$   
 $Y \leftarrow g^y;$   
 $Z' \leftarrow \mathcal{A}_{CDH}(pp, X, Y);$

**if  $Z' = g^{xy}$  output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{DLOG}](pp, X, Y) :$   
 $x' \leftarrow \mathcal{A}_{DLOG}(pp, X);$   
**output  $Y^{x'}$ ;**

$\mathcal{C}_{DLOG}(pp) :$   
 $x \leftarrow \mathbb{Z}q;$   
 $X \leftarrow g^x;$

$x' \leftarrow \mathcal{A}_{DLOG}(pp, X);$

**if  $x' = x$  output 1;**  
**else output 0;**

- Time overhead of only one mod. exp.
- Same success probability
- "Oracle access" to  $\mathcal{A}_{DLOG}$

## Black-Box Reductions

$\mathcal{R}$  is just a Oracle PPTM and now  $\mathcal{A} = \mathcal{R}[\mathcal{A}'] = \mathcal{R}^{\mathcal{A}'}$

- $\mathcal{R}$  has no access to the internals of  $\mathcal{A}'$ , but only to its input-output behavior
- Recall that  $\mathcal{A}'$  is non-perfect, i.e., it solves  $P'$  with a non-negligible probability/advantage
- $\mathcal{R}$  can run several instances of  $\mathcal{A}'$  on different inputs, **but then success probabilities of  $\mathcal{A}$  and  $\mathcal{A}'$  are hard to relate!**

A typical reduction: Black-Box with a **single call** to  $\mathcal{A}'$ :

- $\mathcal{R}[\mathcal{A}']$  transforms its input  $x \in P$  into  $x' \in P'$
- $\mathcal{R}[\mathcal{A}']$  runs  $\mathcal{A}'$  with input  $x'$
- $\mathcal{R}[\mathcal{A}']$  computes its output from the output of  $\mathcal{A}'$

## Self-Reductions: Examples

### Repetition for a checkable search problem

'**checkable**' means there is an efficient way to verify a solution

$\mathcal{A} = \mathcal{R}[\mathcal{A}']$  runs  $n$  times  $\mathcal{A}'$  on the same input and checks the  $n$  outputs to find a correct one.

For a small  $\text{Succ}_{P,\mathcal{A}'}(\ell)$  and not too large  $n$ :

$$\text{Succ}_{P,\mathcal{A}}(\ell) \approx n \cdot \text{Succ}_{P,\mathcal{A}'}(\ell), \quad \text{time}(\mathcal{A}, x) \approx n \cdot \text{time}(\mathcal{A}', x)$$

The ratio probability/time is a good efficiency measure.

## Self-Reductions: Examples

### Probability Amplification by Repetition (decision problems)

$\mathcal{A} = \mathcal{R}[\mathcal{A}']$  runs  $n$  times  $\mathcal{A}'$  on the same input and decides by majority voting among the  $n$  outputs (bits)

For small  $\text{Adv}_{P,\mathcal{A}'}(\ell)$  and  $n$  not too large:

$$\text{Adv}_{P,\mathcal{A}}(\ell) \approx \sqrt{\frac{2n}{\pi}} \text{Adv}_{P,\mathcal{A}'}(\ell), \quad \text{time}(\mathcal{A}, x) \approx n \cdot \text{time}(\mathcal{A}', x)$$

For very large  $n \approx \frac{1}{\text{Adv}_{P,\mathcal{A}'}(\ell)^2}$  we obtain  $\text{Adv}_{P,\mathcal{A}}(\ell) \approx 1$ .

## Random Self-Reducibility

### Definition

The **decision** problem family  $P$  is **random self-reducible** if there exists a PPTM  $T$  that transforms any particular instance  $x \in P_\ell$  into a random instance in  $P_\ell$ .

Using  $T$  as a self-reduction  $\mathcal{R}_T$ ,

$$\mathcal{A}(x) = \mathcal{R}_T[\mathcal{A}'](x) = \mathcal{A}'(T(x))$$

shows that no particular instance of  $P$  is harder to solve than a random instance.

For a random self-reducible problem **average hardness is equivalent to worst-case hardness**

# Random Self-Reducibility Examples

**DDH:** Common parameters  $(G, q, g)$

Randomizing transformation:

$$T(X, Y, Z) = (g^a X, g^b Y^c, g^{ab} X^b Y^{ac} Z^c)$$

for random  $a, b, c \in \mathbb{Z}_q, c \neq 0$ .

It must work for both “0” and “1” instances of DDH!

- For any “0” instance  $(X = g^x, Y = g^y, Z = g^{xy})$   
 $T(X, Y, Z) = (g^{a+x}, g^{b+cy}, g^{(a+x)(b+cy)})$   
 is a uniformly distributed “0” instance.
- For a “1” instance  $(X = g^x, Y = g^y, Z = g^z)$   
 $T(X, Y, Z) = (g^{a+x}, g^{b+cy}, g^{(a+x)(b+cy)+c(z-xy)})$   
 is a uniformly distributed “1” instance.

# More Reduction Examples: $IF \stackrel{\mathcal{R}}{\Rightarrow} SQRT$

Integer factorization reduces to computing square roots

$\mathcal{C}_{IF}$  :  
 $p, q \leftarrow \text{Primes}_{3 \bmod 4}$ ;  
 $n \leftarrow pq$ ;  
 $r \leftarrow \mathcal{A}_{IF}(n)$ ;

if  $\gcd(n, r) \notin \{1, n\}$   
**output 1**;  
 else **output 0**;

$\mathcal{R}[\mathcal{A}_{SQRT}](n)$  :  
 $x \leftarrow \mathbb{Z}_n^\times; y \leftarrow x^2 \bmod n$ ;  
 $z \leftarrow \mathcal{A}_{SQRT}(n, y)$ ;  
**output**  $\gcd(n, z - x)$ ;

# Random Self-Reducibility Examples

**Search problems:** more complex reductions are required

**CDH:** Common parameters  $pp = (G, q, g)$

$\mathcal{R}[\mathcal{A}'_{CDH}](pp, X, Y)$  :

$a, b \leftarrow \mathbb{Z}_q; X' \leftarrow g^a X; Y' \leftarrow g^b Y;$  // randomization

$Z' \leftarrow \mathcal{A}'_{CDH}(pp, X', Y');$

$Z = Z' g^{-ab} X^{-b} Y^{-a};$  // post processing

**output**  $Z$ ;

- $X = g^x, Y = g^y \Rightarrow X' = g^{a+x}, Y' = g^{b+y}$
- $(pp, X', Y')$  is a random CDH instance
- $\mathcal{A}'_{CDH}$  succeeds  $\Rightarrow \mathcal{R}[\mathcal{A}'_{CDH}]$  succeeds  
 $Z' = g^{(a+x)(b+y)} = g^{ab} X^b Y^a g^{xy}$  and  $Z = g^{xy}$

# More Reduction Examples: $IF \stackrel{\mathcal{R}}{\Rightarrow} SQRT$

Integer factorization reduces to computing square roots

$\mathcal{C}_{IF}$  :  
 $p, q \leftarrow \text{Primes}_{3 \bmod 4}$ ;  
 $n \leftarrow pq$ ;  
 $r \leftarrow \mathcal{A}_{IF}(n)$ ;

if  $\gcd(n, r) \notin \{1, n\}$   
**output 1**;  
 else **output 0**;

$\mathcal{R}[\mathcal{A}_{SQRT}](n)$  :  
 $x \leftarrow \mathbb{Z}_n^\times; y \leftarrow x^2 \bmod n$ ;  
 $z \leftarrow \mathcal{A}_{SQRT}(n, y)$ ;  
**output**  $\gcd(n, z - x)$ ;

$\mathcal{C}_{SQRT}$  :  
 $p, q \leftarrow \text{Primes}_{3 \bmod 4}$ ;  
 $n \leftarrow pq$ ;  
 $t \leftarrow \mathbb{Z}_n^\times; x \leftarrow t^2 \bmod n$ ;  
 $y \leftarrow x^2 \bmod n$ ;  
 $z \leftarrow \mathcal{A}_{SQRT}(n, y)$ ;  
 if  $z = x$   
**output 1**;  
 else **output 0**;

Perfect simulation of  $\mathcal{C}_{SQRT}$  for  $\mathcal{A}_{SQRT}$

## More Reduction Examples: $\mathcal{IF} \xrightarrow{\mathcal{R}} \text{SQRT}$

Integer factorization reduces to computing square roots

$\mathcal{C}_{\text{IF}}$  :  
 $p, q \leftarrow \text{Primes3mod4};$   
 $n \leftarrow pq;$   
 $r \leftarrow \mathcal{A}_{\text{IF}}(n);$

if  $\gcd(n, r) \notin \{1, n\}$   
**output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{\text{SQRT}}](n)$  :  
 $x \leftarrow \mathbb{Z}_n^\times; y \leftarrow x^2 \bmod n;$   
 $z \leftarrow \mathcal{A}_{\text{SQRT}}(n, y);$   
**output**  $\gcd(n, z - x);$

$\mathcal{C}_{\text{SQRT}}$  :  
 $p, q \leftarrow \text{Primes3mod4};$   
 $n \leftarrow pq;$   
 $t \leftarrow \mathbb{Z}_n^\times; x \leftarrow t^2 \bmod n;$   
 $y \leftarrow x^2 \bmod n;$   
 $z \leftarrow \mathcal{A}_{\text{SQRT}}(n, y);$   
**if**  $z = x$   
**output 1;**  
**else output 0;**

- Time overhead of one modular square and one gcd.
- $\text{Succ}(\mathcal{R}[\mathcal{A}_{\text{SQRT}}]) = \text{Succ}(\mathcal{A}_{\text{SQRT}})/2$
- “Oracle access” to  $\mathcal{A}_{\text{SQRT}}$

## More Reduction Examples: $\text{CDH} \xrightarrow{\mathcal{R}} \text{SQE}$

CDH reduces to squaring the exponent

Common parameters:  $pp = (G, q, g)$

$\mathcal{C}_{\text{CDH}}(pp)$  :  
 $x, y \leftarrow \mathbb{Z}_q;$   
 $X \leftarrow g^x; Y \leftarrow g^y;$   
 $Z \leftarrow \mathcal{A}_{\text{CDH}}(pp, X, Y);$

if  $Z = g^{xy}$  **output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{\text{SQE}}](pp, X, Y)$  :  
 $t \leftarrow 4^{-1} \bmod q;$   
 $U \leftarrow XY; V \leftarrow XY^{-1};$   
 $A \leftarrow \mathcal{A}_{\text{SQE}}(pp, U);$   
 $B \leftarrow \mathcal{A}_{\text{SQE}}(pp, V);$   
**output**  $(AB^{-1})^t;$

## Why $\text{Succ}(\mathcal{R}[\mathcal{A}_{\text{SQRT}}]) = \text{Succ}(\mathcal{A}_{\text{SQRT}})/2?$

- $p, q = 3 \bmod 4$  implies  $-1$  is not a square modulo  $p$  or  $q$ .
- Modulo  $n$ , any square has **exactly** 4 square roots, and only one is itself a square.

Let's denote  $\{x, -x, u, -u\}$  the 4 square roots of  $y$  modulo  $n$ , such that  $u = x \pmod{p}$  and  $u = -x \pmod{q}$ .

- The square root  $z$  given by  $\mathcal{A}_{\text{SQRT}}$  will equal  $x$  or  $-x$  with probability  $1/2$ .
- $x^2 = z^2 = y \pmod{n}$ , then  $(x - z)(x + z) = 0 \pmod{n}$ , and whenever  $z \notin \{x, -x\}$  we have  $\gcd(n, x - z) \in \{p, q\}$ .

## More Reduction Examples: $\text{CDH} \xrightarrow{\mathcal{R}} \text{SQE}$

CDH reduces to squaring the exponent

Common parameters:  $pp = (G, q, g)$

$\mathcal{C}_{\text{CDH}}(pp)$  :  
 $x, y \leftarrow \mathbb{Z}_q;$   
 $X \leftarrow g^x; Y \leftarrow g^y;$   
 $Z \leftarrow \mathcal{A}_{\text{CDH}}(pp, X, Y);$

if  $Z = g^{xy}$  **output 1;**  
**else output 0;**

$\mathcal{R}[\mathcal{A}_{\text{SQE}}](pp, X, Y)$  :  
 $t \leftarrow 4^{-1} \bmod q;$   
 $U \leftarrow XY; V \leftarrow XY^{-1};$   
 $A \leftarrow \mathcal{A}_{\text{SQE}}(pp, U);$   
 $B \leftarrow \mathcal{A}_{\text{SQE}}(pp, V);$   
**output**  $(AB^{-1})^t;$

$\mathcal{C}_{\text{SQE}}(pp)$  :  
 $x \leftarrow \mathbb{Z}_q;$   
 $X \leftarrow g^x;$   
 $Z \leftarrow \mathcal{A}_{\text{CDH}}(pp, X);$   
**if**  $Z = g^{x^2}$  **output 1;**  
**else output 0;**

Perfect simulation of  $\mathcal{C}_{\text{SQE}}$  in the 2 calls to  $\mathcal{A}_{\text{SQE}}$

$$X = g^x, Y = g^y \Rightarrow U = g^{x+y}, V = g^{x-y}$$

$$A = g^{x^2+y^2+2xy}, B = g^{x^2+y^2-2xy}, (AB^{-1})^t = g^{xy}$$

# More Reduction Examples: $CDH \xrightarrow{\mathcal{R}} SQE$

CDH reduces to squaring the exponent

Common parameters:  $pp = (G, q, g)$

$C_{CDH}(pp)$  :  
 $x, y \leftarrow \mathbb{Z}_q$ ;  
 $X \leftarrow g^x$ ;  $Y \leftarrow g^y$ ;  
 $Z \leftarrow \mathcal{A}_{CDH}(pp, X, Y)$ ;

$\mathcal{R}[\mathcal{A}_{SQE}](pp, X, Y)$  :  
 $t \leftarrow 4^{-1} \bmod q$ ;  
 $U \leftarrow XY$ ;  $V \leftarrow XY^{-1}$ ;  
 $A \leftarrow \mathcal{A}_{SQE}(pp, U)$ ;  
 $B \leftarrow \mathcal{A}_{SQE}(pp, V)$ ;  
**output**  $(AB^{-1})^t$ ;

$C_{SQE}(pp)$  :  
 $x \leftarrow \mathbb{Z}_q$ ;  
 $X \leftarrow g^x$ ;  
 $Z \leftarrow \mathcal{A}_{CDH}(pp, X)$ ;  
**if**  $Z = g^{x^2}$  **output** 1;  
**else output** 0;

**if**  $Z = g^{xy}$  **output** 1;  
**else output** 0;

- $\text{time}(\mathcal{R}[\mathcal{A}_{SQE}]) \approx 2 \cdot \text{time}(\mathcal{A}_{SQE})$
- $\text{Succ}(\mathcal{R}[\mathcal{A}_{SQE}]) \geq (\text{Succ}(\mathcal{A}_{SQE}))^2$
- Two oracle accesses to  $\mathcal{A}_{SQE}$  with **independent** inputs

## Outline

- 1 Reducing Computational Problems
- 2 Security Statements

# A Remark About Reduction Tightness

$P \Rightarrow P'$  means “If  $P'$  is easy, then so is  $P$ ”

More precisely:

“There exists  $\mathcal{R}$  such that if  $\mathcal{A}'$  solves  $P'$  in time  $t'$  with probability/advantage  $\epsilon' > \text{negl}(\ell)$ , then  $\mathcal{R}[\mathcal{A}']$  solves  $P$  in time  $t$  with probability/advantage  $\epsilon > \text{negl}(\ell)$ ”

- If  $t \approx t'$  and  $\epsilon \approx \epsilon'$ ,  $\mathcal{R}$  is **tight**  
Meaningful reduction!
- If  $t \approx t'$  but  $\epsilon \approx C\epsilon'$  for some  $C \ll 1$ ,  $\mathcal{R}$  is **almost tight**  
Quite meaningful reduction!
- If  $t \approx t'$  but  $\epsilon/\epsilon' \rightarrow 0$ ,  $\mathcal{R}$  is **not tight**  
It depends...
- If  $t' \gg t$ , compare the ratios  $\epsilon/t$  and  $\epsilon'/t'$

## Applications of Reductions (I)

Security definitions are (interactive) problem families.

**Reductions between security notions** show implications, or relative hardness, e.g.,

$$\text{IND-CCA}(\Pi) \Rightarrow \text{IND-CPA}(\Pi) \Rightarrow \text{OW-CPA}(\Pi)$$

(strongest)

(weakest)

A reduction  $\text{SEC1}(\Pi) \xrightarrow{\mathcal{R}} \text{SEC2}(\Pi)$  transforms any adversary  $\mathcal{A}_2$  breaking  $\text{SEC2}(\Pi)$  into another one  $\mathcal{A}_1 = \mathcal{R}[\mathcal{A}_2]$  breaking  $\text{SEC1}(\Pi)$ .

$\mathcal{R}$  must **simulate** any oracle given in  $\text{SEC2}(\Pi)$  for  $\mathcal{A}_2$ , but it can **use** the oracles given in  $\text{SEC1}(\Pi)$ .

# Applications of Reductions (II)

Reductions between computational problems show relative strongness of the different security assumptions, e.g.,

(easiest) DDH  $\Rightarrow$  CDH  $\Rightarrow$  DLOG (hardest)  
(strongest) (weakest)

**Security proofs by reduction:** A reduction of a computational problem family  $P$  to the security notion  $SEC(\Pi)$ , proves security of  $\Pi$  under the assumption that  $P$  is hard

$$P \Rightarrow SEC(\Pi)$$

It means “if  $SEC$  is broken for  $\Pi$ , then  $P$  is easy”

The reduction must simulate all oracles in the definition of  $SEC$ .

# Provable Security

**Main goal in provable security:** Give a proof by reduction under a well-known and well-studied assumption

- The same assumption can be used for several cryptosystems
- ... even if they are of different type (e.g., encryption and signatures)
- It makes easier comparing them
- Cryptanalysis focus on computational problems and not on specific schemes

## CRYE 6138 Security Models

Jorge L. Villar

UBa Cyber Crypto Center, Fall 2025

## Security Statements

—END—