CRYE 6138 Security Models

Jorge L. Villar

UBa Cyber Crypto Center, Fall 2025

Security Statements

Outline

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.

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

Reductions

Meaning of "P is hard $\Rightarrow$ P' is hard"?

Or equivalently, "P' is easy $\Rightarrow$ P is easy"

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

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'

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

It reads: "Solving P reduces to solving P'"

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

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

Syntax:

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

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

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

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

Syntax:

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

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

Reduction:

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

Correctness:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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;

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

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

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

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

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

Black-Box Reductions

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

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!

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

Probability Amplification by Repetition (decision problems)

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)

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

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

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

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

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

Self-Reductions: Examples

Repetition for a checkable search problem

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

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.

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

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

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

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

The ratio probability/time is a good efficiency measure.

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_ℓ.

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_ℓ.

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.

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_ℓ.

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!

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.

Random Self-Reducibility Examples

Search problems: more complex reductions are required

Random Self-Reducibility Examples

Search problems: more complex reductions are required

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

$\mathcal{R}[\mathcal{A}'_{\mathsf{CDH}}](pp, X, Y)$:
$a, b \leftarrow \mathbb{Z}_q$; $X' \leftarrow g^a X$; $Y' \leftarrow g^b Y$; // randomization
$Z' \leftarrow \mathcal{A}'_{\mathsf{CDH}}(pp, X', Y')$;
$Z = Z' g^{-ab} X^{-b} Y^{-a}$; // post processing
output Z;

Random Self-Reducibility Examples

Search problems: more complex reductions are required

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

$\mathcal{R}[\mathcal{A}'_{\mathsf{CDH}}](pp, X, Y)$:
$a, b \leftarrow \mathbb{Z}_q$; $X' \leftarrow g^a X$; $Y' \leftarrow g^b Y$; // randomization
$Z' \leftarrow \mathcal{A}'_{\mathsf{CDH}}(pp, X', Y')$;
$Z = Z' g^{-ab} X^{-b} Y^{-a}$; // 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
- $\mathcal{A}'_{\mathsf{CDH}}$ succeeds $\Rightarrow$ $\mathcal{R}[\mathcal{A}'_{\mathsf{CDH}}]$ 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

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

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

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

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

Integer factorization reduces to computing square roots

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

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

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

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

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

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

Integer factorization reduces to computing square roots

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

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

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

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

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

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

- $p, q = 3 \bmod 4$ 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.

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

- $p, q = 3 \bmod 4$ 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 $u = -x \pmod{q}$.

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

- $p, q = 3 \bmod 4$ 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 $\mathcal{A}_{\mathrm{SQRT}}$ 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: CDH $\overset{\mathcal{R}}{\Rightarrow}$ SQE

CDH reduces to squaring the exponent

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

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

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

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

More Reduction Examples: CDH $\overset{\mathcal{R}}{\Rightarrow}$ SQE

CDH reduces to squaring the exponent

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

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

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

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

$\mathcal{C}_{\mathrm{SQE}}(pp)$:
$x \leftarrow \mathbb{Z}_q$;
$X \leftarrow g^x$;

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

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

$$X = g^X, Y = g^y \Rightarrow U = g^{x+y}, V = g^{x-y}$$
$$A = g^{x^2+y^2+2xy}, B = g^{x^2+y^2-2xy}, (AB^{-1})^t = g^{xy}$$

More Reduction Examples: CDH $\overset{\mathcal{R}}{\Rightarrow}$ SQE

CDH reduces to squaring the exponent

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

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

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

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

$\mathcal{C}_{\mathsf{SQE}}(pp)$:
$x \leftarrow \mathbb{Z}_q$;
$X \leftarrow g^x$;

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

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

A Remark About Reduction Tightness

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

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 $\varepsilon' > \mathbf{negl}(\ell)$, then $\mathcal{R}[\mathcal{A}']$ solves P in time t with probability/advantage $\varepsilon > \mathbf{negl}(\ell)$"

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 $\varepsilon' > \mathbf{negl}(\ell)$, then $\mathcal{R}[\mathcal{A}']$ solves P in time t with probability/advantage $\varepsilon > \mathbf{negl}(\ell)$"

- If $t \approx t'$ and $\varepsilon \approx \varepsilon'$, $\mathcal{R}$ is **tight**
- If $t \approx t'$ but $\varepsilon \approx C\varepsilon'$ for some $C \ll 1$, $\mathcal{R}$ is **almost tight**
- If $t \approx t'$ but $\varepsilon/\varepsilon' \to 0$, $\mathcal{R}$ is **not tight**
- If $t' \gg t$, compare the ratios ε/t and ε'/t'

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 $\varepsilon' > \mathbf{negl}(\ell)$, then $\mathcal{R}[\mathcal{A}']$ solves P in time t with probability/advantage $\varepsilon > \mathbf{negl}(\ell)$"

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

Outline

Applications of Reductions (I)

Security definitions are (interactive) problem families.

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)

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 $\text{SEC1}(\Pi) \overset{\mathcal{R}}{\Rightarrow} \text{SEC1}(\Pi)$ 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) (hardest)

$$\text{DDH} \Rightarrow \text{CDH} \Rightarrow \text{DLOG}$$

(strongest) (weakest)

Applications of Reductions (II)

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

(easiest) (hardest)

$$\text{DDH} \Rightarrow \text{CDH} \Rightarrow \text{DLOG}$$

(strongest) (weakest)

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

$$P \Rightarrow \text{SEC}(\Pi)$$

It means "if SEC is broken for Π, 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

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
- Cryptoanalysis focus on computational problems and not on specific schemes

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
- Cryptoanalysis focus on computational problems and not on specific schemes

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
- Cryptoanalysis focus on computational problems and not on specific schemes

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
- Cryptoanalysis 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—