Cryptographic hash functions IV

Proofs by reduction

Tanja Lange

Eindhoven University of Technology

2MMC10 – Cryptology

More terms from complexity theory: reductions

- A reduction transforms algorithm for problem 1 into an algorithm for problem 2.
- "Reduces problem 2 to problem 1"
  (Can solve problem 2 by solving problem 1)
- Allows to relate the hardness of problems:
  If there exists an efficient reduction that reduces problem 2 to problem 1 then an efficient algorithm solving problem 1 can be used to efficiently solve problem 2.

More terms from complexity theory: reductions

- A reduction transforms algorithm for problem 1 into an algorithm for problem 2.
- "Reduces problem 2 to problem 1"
  (Can solve problem 2 by solving problem 1)
- Allows to relate the hardness of problems:
  If there exists an efficient reduction that reduces problem 2 to problem 1 then an efficient algorithm solving problem 1 can be used to efficiently solve problem 2.

We have seen:
CDHP and DDHP reduce to DLP; DDHP reduces to CDHP.

More terms from complexity theory: reductions

- A reduction transforms algorithm for problem 1 into an algorithm for problem 2.
- "Reduces problem 2 to problem 1"
(Can solve problem 2 by solving problem 1)
- Allows to relate the hardness of problems:
If there exists an efficient reduction that reduces problem 2 to problem 1 then an efficient algorithm solving problem 1 can be used to efficiently solve problem 2.

We have seen:
CDHP and DDHP reduce to DLP; DDHP reduces to CDHP.

- Existence of reduction does not imply that the probabilities of success are equal.
- A reduction might require solving problem 1 multiple times.

These factors control the tightness of the reduction.

More terms from complexity theory: reductions

- A reduction transforms algorithm for problem 1 into an algorithm for problem 2.
- "Reduces problem 2 to problem 1"
  (Can solve problem 2 by solving problem 1)
- Allows to relate the hardness of problems:
  If there exists an efficient reduction that reduces problem 2 to problem 1 then an efficient algorithm solving problem 1 can be used to efficiently solve problem 2.

We have seen:
CDHP and DDHP reduce to DLP; DDHP reduces to CDHP.

- Existence of reduction does not imply that the probabilities of success are equal.
- A reduction might require solving problem 1 multiple times.

These factors control the tightness of the reduction.
In cryptography, reductions relate the security of systems.

"Provable Security": Reduce an assumed to be hard problem to the security of a bigger cryptosystem. No absolute proof.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Need to show how to construct A_{CR} given A_{SPR}.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Need to show how to construct A_{CR} given A_{SPR}.
Proof: Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\mathrm{SPR}}(k,x)$ to get $x' \neq x$ with $H(k,x') = H(k,x)$.
Output (x,x'). □

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Need to show how to construct A_{CR} given A_{SPR}.
Proof: Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\mathrm{SPR}}(k,x)$ to get $x' \neq x$ with $H(k,x') = H(k,x)$.
Output (x,x'). □

Algorithm A_{CR} has same runtime and success probability as A_{SPR}.
Fails if $H(k,x)$ has no second preimage.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Need to show how to construct A_{CR} given A_{SPR}.
Proof: Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\mathrm{SPR}}(k,x)$ to get $x' \neq x$ with $H(k,x') = H(k,x)$.
Output (x,x'). □

Algorithm A_{CR} has same runtime and success probability as A_{SPR}.
Fails if $H(k,x)$ has no second preimage.

Can iterate over $x \leftarrow_R \{0,1\}^{\ell(n)}$, good chance if $\ell(n) \gg n$.

Reductions between hash function properties I

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Collision resistance (CR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, (x,x') \leftarrow A(k) : H(k,x') = H(k,x) \text{ and } x' \neq x]$$

is negligible in n.

CR reduces to SPR.

Need to show how to construct A_{CR} given A_{SPR}.
Proof: Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\mathrm{SPR}}(k,x)$ to get $x' \neq x$ with $H(k,x') = H(k,x)$.
Output (x,x'). □

Algorithm A_{CR} has same runtime and success probability as A_{SPR}.
Fails if $H(k,x)$ has no second preimage.

Can iterate over $x \leftarrow_R \{0,1\}^{\ell(n)}$, good chance if $\ell(n) \gg n$.

This means that a collision resistant function is also second preimage resistant.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.

Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.

Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.

Hope that $x' \neq x$. Output x'.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.
Hope that $x' \neq x$. Output x'.

No chance if H is injective.
If $\ell(n) \gg n$ we have a good chance that $y = H(k,x)$ has a second preimage.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.
Hope that $x' \neq x$. Output x'.

No chance if H is injective.
If $\ell(n) \gg n$ we have a good chance that $y = H(k,x)$ has a second preimage. If so, have at least 50% chance of $x' \neq x$.
Need to use A_{PRE} a few times. Exact numbers depend on $\ell(n)/n$.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.
Hope that $x' \neq x$. Output x'.

No chance if H is injective.
If $\ell(n) \gg n$ we have a good chance that $y = H(k,x)$ has a second preimage. If so, have at least 50% chance of $x' \neq x$.
Need to use A_{PRE} a few times. Exact numbers depend on $\ell(n)/n$.

If we can decide if $H(k,x)$ has a second preimage (DSPR),

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE?

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\text{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.
Hope that $x' \neq x$. Output x'.

No chance if H is injective.
If $\ell(n) \gg n$ we have a good chance that $y = H(k,x)$ has a second preimage. If so, have at least 50% chance of $x' \neq x$.
Need to use A_{PRE} a few times. Exact numbers depend on $\ell(n)/n$.

If we can decide if $H(k,x)$ has a second preimage (DSPR), we can skip $\ell(n) \gg n$ condition.

Reductions between hash function properties II

Preimage resistance: For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, y \leftarrow H(k,x), x' \leftarrow A(k,y) : H(k,x') = y]$$

is negligible in n.

Second preimage resistance (SPR): For any PPT algorithm A

$$\Pr[k \leftarrow_R \{0,1\}^n, x \leftarrow_R \{0,1\}^{\ell(n)}, x' \leftarrow A(k,x) : H(k,x') = H(k,x) \wedge x' \neq x]$$

is negligible in n.

Does SPR reduce to PRE? Sort of.

Attempt at proof: Use A_{PRE} to build A_{SPR}.
Given $k \in \{0,1\}^n$, pick randomly $x \leftarrow_R \{0,1\}^{\ell(n)}$.
Run $A_{\mathrm{PRE}}(k, H(k,x))$ to get x' with $H(k,x') = y$.
Hope that $x' \neq x$. Output x'.

No chance if H is injective.
If $\ell(n) \gg n$ we have a good chance that $y = H(k,x)$ has a second preimage. If so, have at least 50% chance of $x' \neq x$.
Need to use A_{PRE} a few times. Exact numbers depend on $\ell(n)/n$.

If we can decide if $H(k,x)$ has a second preimage (DSPR), we can skip $\ell(n) \gg n$ condition.