The Query Complexity of Preprocessing Attacks

Ashrujit Ghoshal

Carnegie Mellon University

Stefano Tessaro

University of Washington

CRYPTO 2023

Preprocessing attacks [Hellman, '80]

Adversary $\boldsymbol{A} = (\boldsymbol{A}_1, \boldsymbol{A}_2)$

problem instance

S-bit advice

Time T_1 $\boldsymbol{A}_1$ $\xrightarrow{\hspace{2em}}$ $\boldsymbol{A}_2$ Time T_2

offline online

solution

Preprocessing attack

"Classical" interpretation: Advice = Non-uniformity

[Koblitz-Menezes, '13] [Bernstein-Lange, '13]

In this case: offline time T_1 does not matter, only advice size S

Many works embrace this viewpoint and prove lower/upper bounds on space-time trade-offs in <u>ideal</u> models

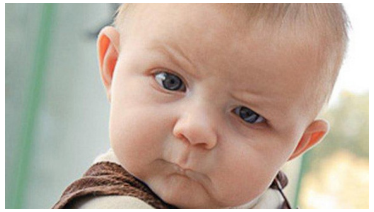
[Hellman ‘80] [Yao ‘90] [Unruh '07][De-Trevisan-Tulsiani ‘10] [Dodis-Guo-Katz ‘17] [Coretti-Dodis-Guo-Steinberger ‘18] [Coretti-Dodis-Guo ‘18] [Corrigan-Gibbs-Kogan ‘18] [Corrigan-Gibbs-Kogan ‘19] [Akshima-Cash-Drucker-Wee ‘20] [Chung-Guo-Liu-Qian ‘20] [Chawin-Haitner-Mazor ‘20] [Guo-Li-Liu-Zhang ‘21] [Gravin-Guo-Chiu-Lu ‘21] [Ghoshal-Komargodski ‘22] [Freitag-Ghoshal-Komargodski ‘22] [Akshima-Guo-Liu ‘22] [Freitag-Ghoshal-Komargodski ‘23] [Golovnev-Guo-Peters-Stephens-Davidowitz '23]

Prototypical theorem

Theorem 5.1. *Let $C = 2^{16} \cdot 6 \cdot e^2$. For any $N, M, B, S, T \in \mathbb{N}_{>0}$ and fixing $\hat{S} := S + \log N$, it holds that*

$$\mathsf{Adv}^{\mathsf{ai\text{-}cr}}_{\mathsf{MD},N,M,B}(S,T) \leqslant C \cdot \max\left\{ \left(\frac{\hat{S}TB^2 \left(\frac{3e \log \hat{S}}{\log\log \hat{S}}\right)^{2(B-2)}}{N} \right), \left(\frac{T^2}{N}\right) \right\} + \frac{1}{N}\ .$$

This talk: should we care about T_1?



(And what can we say about it?)

In some settings, we actually want to run the attack!

For a pre-processing attack to be "practical":

- Feasible T_1
- Worth it to run the attack!

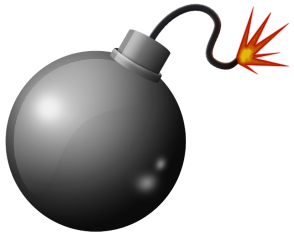
T^* := runtime of best online-only attack to win

To have $T_2 \ll T^*$ we need $T_1 \geq T^*$

When OK?

When is $T_1 \geq T^*$ okay?

Setting 1: Online phase has short time-out and must be fast!



Example: [Adrian et al. ‘15] – breaking (weak) discrete logarithm within TLS session

Setting 2: Advice can be recycled across multiple executions of the attack

Example: Invert $\mathrm{RO(pwd)}$ with N potential pwd’s

Online only: k passwords in time $k \times N$ [memory-less]

Rainbow table: k passwords in time $N + k \times \frac{N}{S}$

Bottom line

There are settings where explicit pre-processing attacks make sense and understanding the necessary offline time complexity is fundamental.

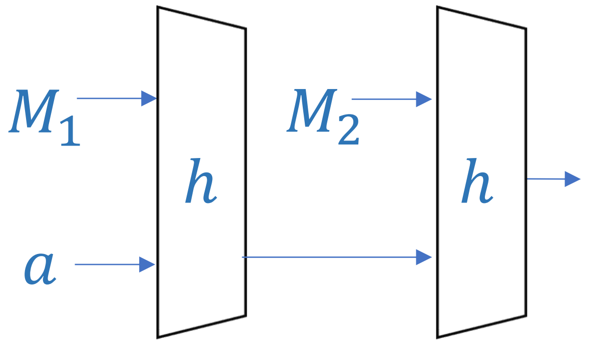
But: can we actually show anything interesting?

- E.g., rainbow tables are easily seen to be optimal (at least one of online and offline phase should take time N)

Interesting example

2-block Merkle-Damgård (MD) collisions

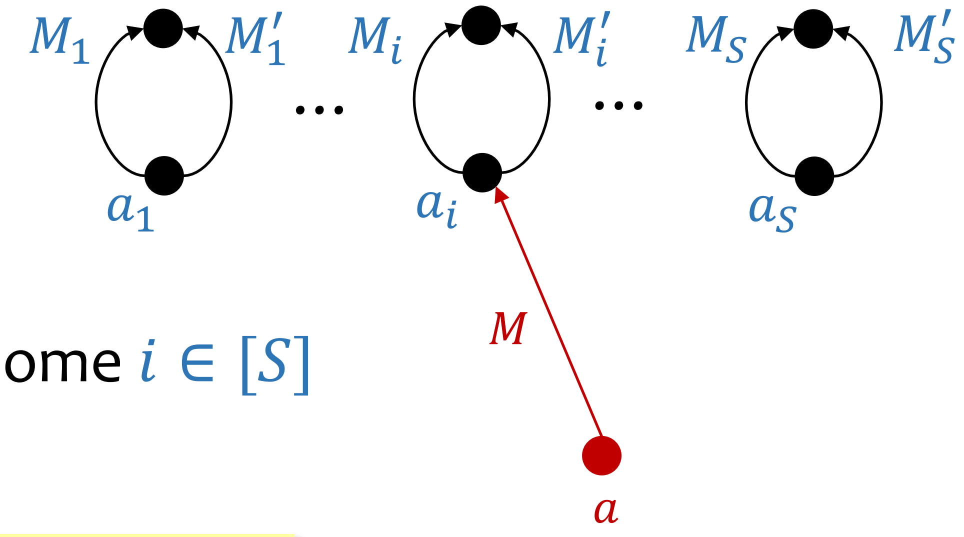
$h: \{0,1\}^{2n} \rightarrow \{0,1\}^{n}$



$2\text{−MD}^h(a, (M_1, M_2))$

Offline

- Advice: S triples (a_i, M_i, M_i') such that $M_i \neq M_i'$, $h(a_i, M_i) = h(a_i, M_i')$ for distinct $a_1, \ldots, a_S$



Online

- Given salt a, find M such that $h(a, M) = a_i$ for some $i \in [S]$
- Return $(M, M_i), (M, M_i')$

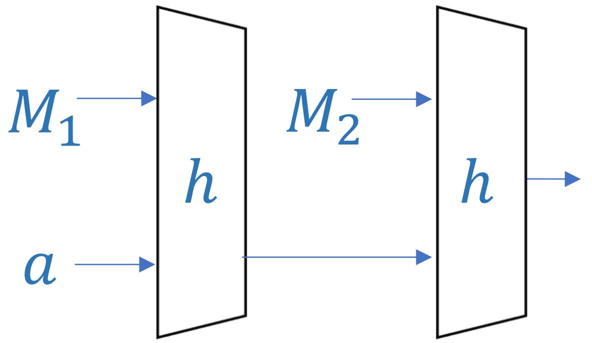
$T_1 \approx S \cdot 2^{0.5n}, T_2 \approx 2^n/S$

$$T_1 \times T_2 \approx 2^{1.5n}$$

Interesting example

2-block Merkle-Damgård (MD) collisions

$h: \{0,1\}^{2n} \to \{0,1\}^n$



$2\text{–MD}^h(a, (M_1, M_2))$

$$T_1 \times T_2 \approx 2^{1.5n}$$

To get $T_2 < 2^{n/2}$, we need $T_1 > 2^n$

e.g., only worth it for more than $2^{n/2}$ collisions

Are there attacks with better trade-offs?

How do we reason about this?

This work!

This work – in a nutshell

Toolkit* to understand inherent relationship between offline and online time in preprocessing attacks.

▷ Generic **salting** defeats preprocessing (qualitatively at least)

▷ **Quantitative** bounds for salted **random oracles**

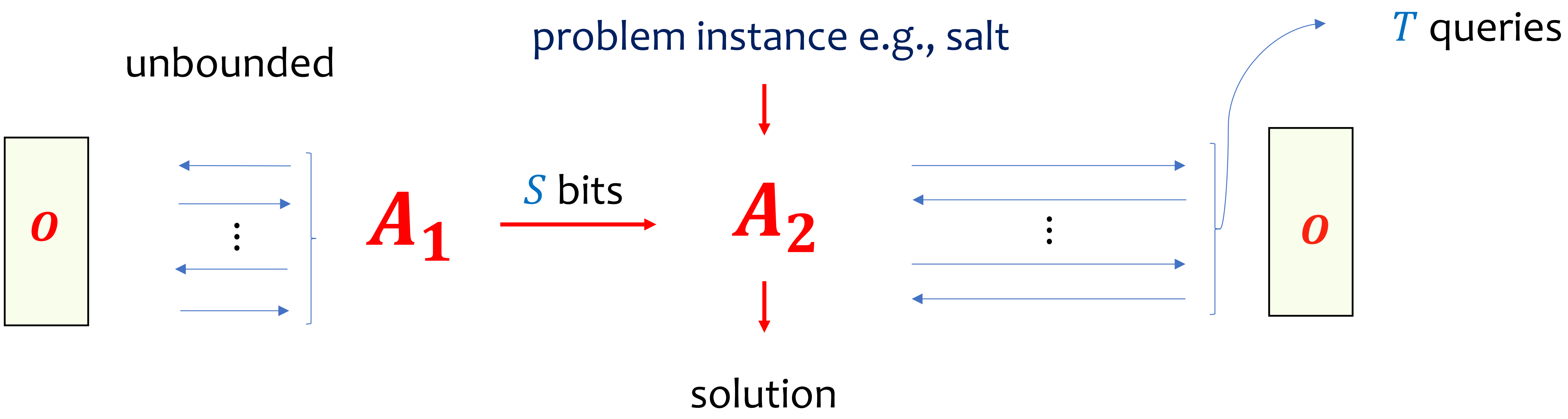
▷ **Quantitative** bounds for **two-block Merkle-Damgård (MD)**

* <u>Only</u> prior work deals with DL with preprocessing [CorriganGibbs-Kogan ’18]

Auxiliary-input (ai) ideal models

$$A = (A_1, A_2)$$

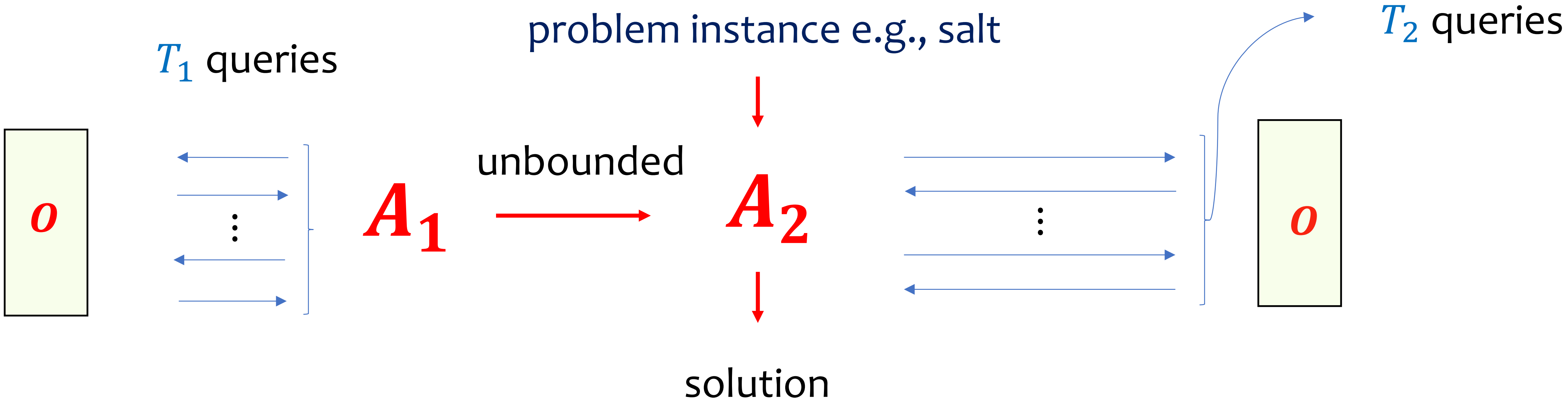
O = RO, ideal cipher, GGM oracle, …



This work -- model

$\boldsymbol{A = (A_1, A_2)}$

$\boldsymbol{O}$ = RO, ideal cipher, GGM oracle, …



Notation: (T_1, T_2)-adversary

Salting defeats preprocessing

Scheme Π^g where $g: \{0,1\}^* \to \{0,1\}^n$ is random oracle

Assume: $\forall T^*$–query B: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^g}(B) \leq 0.4$

Replace g with $h(a, .)$ where $h: \{0,1\}^s \times \{0,1\}^* \to \{0,1\}^n$

Salted hash function, public salt a picked at random

~ Time to break Π on every salt

~ Time to break Π online only!

Theorem. $\forall$ (T_1, T_2)-adversaries A

$$\mathrm{Adv}^{\mathrm{sec}}_{\Pi^{h(a,.)}}(A) \geq 0.9 \Rightarrow T_1 \gtrsim 2^s T^*/4 \text{ or } T_2 \gtrsim T^*/4$$

Two issues:

- only deals with high-advantage regime
- in some cases, not all calls are salted!

Proof idea

$\forall B$ making T^* queries: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^g}(B) \leq 0.4$

$\Downarrow$ (Markov inequality)

Generic technique for concrete bounds!

$\forall B'$ making $T^*/2$ queries <u>in expectation</u>: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^g}(B') \leq 0.9$

$\Downarrow$

$\forall\ (T_1, T_2)$–adversaries A w/ $T_1 \leq 2^s T^*/4$ and $T_2 \leq T^*/4$: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^{h(a,\cdot)}}(A) \leq 0.9$

Generic technique

Use [Jaeger-Tessaro '20] to compute ϵ!

$\forall B'$ making $\frac{T_1}{2^s} + T_2$ queries <u>in expectation</u>: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^g}(B') \leq \epsilon$

$\forall\ (T_1, T_2)$–adversaries A: $\mathrm{Adv}^{\mathrm{sec}}_{\Pi^{h(a,\cdot)}}(A) \leq \epsilon$

Salted Random Oracles – Generic Technique

Example. **Pre-image resistance** of salted random oracle $h: \{0,1\}^s \times \{0,1\}^* \to \{0,1\}^n$

Given $a \overset{\$}{\leftarrow} \{0,1\}^s, y \overset{\$}{\leftarrow} \{0,1\}^n$, find M such that $h(a, M) = y$

Corollary. [Generic + JT20] $\forall (T_1, T_2)$-adversaries A

$$\mathrm{Adv}^{\mathrm{pr}}_{h(a,.)}(A) \leq \frac{5T_1}{2^{s+n}} + \frac{5T_2}{2^n}$$

Matching offline-only attack

Matching online-only attack

Salted Random Oracles – Generic Technique

Example. **Collision resistance** of salted random oracle $h: \{0,1\}^s \times \{0,1\}^* \rightarrow \{0,1\}^n$

Given $a \overset{\$}{\leftarrow} \{0,1\}^s$, find $M \neq M'$ such that $h(a, M) = h(a, M')$

Corollary. [Generic + JT20] $\forall (T_1, T_2)$-adversaries A

$$\mathrm{Adv}^{\mathrm{cr}}_{h(a,.)}(A) \leq \frac{T_1}{2^{s+\frac{n}{2}}} + \frac{T_2}{2^{n/2}}$$

Matching offline-only attack 👍

No matching online-only attack (but close) 🤔

Salted Random Oracles – Direct Proof

Example. **Collision resistance** of salted random oracle $h: \{0,1\}^s \times \{0,1\}^* \to \{0,1\}^n$

Given $a \xleftarrow{\$} \{0,1\}^s$, find $M \neq M'$ such that $h(a, M) = h(a, M')$

Theorem. [This work] $\forall (T_1, T_2)$-adversaries A

$$\mathrm{Adv}^{\mathrm{cr}}_{h(a,.)}(A) \leq \frac{T_1}{2^{s+\frac{n}{2}}} + \frac{T_2^2}{2^n}$$

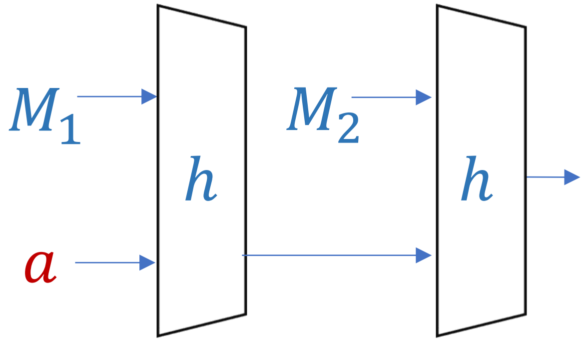
Proof via compression argument [we will come back to this …]

Bottom line: Generic approach does not always give best possible bounds (but gives close enough bounds)

Two-block MD

Two block MD construction does not salt each call to h

$\rightarrow$ prior techniques do not apply & more challenging proofs



$h: \{0,1\}^{2n} \rightarrow \{0,1\}^{n}$

$2\mathrm{-MD}^h\big(a, (M_1, M_2)\big)$

Two-block MD – Pre-image resistance

Theorem. $\forall (T_1, T_2)$-adversaries A

$$\mathrm{Adv}^{\mathrm{pr}}_{2-\mathrm{MD}^h}(A) \leq \frac{T_2}{2^n} + \frac{T_1 T_2}{2^{2n}} + \frac{T_1^2}{2^{3n}}$$

Online-only attack, requires $T_2 = 2^n$

Trade-off. E.g., $T_1 = 2^{1.25n}$ and $T_2 = 2^{0.75n}$

Offline-only attack, requires $T_1 = 2^{1.5n}$

Two-block MD – Collision Resistance

Theorem. $\forall(T_1, T_2)$-adversaries A

$$\mathrm{Adv}^{\mathrm{cr}}_{2-\mathrm{MD}^h}(A) \leq \frac{T_2^2}{2^n} + \frac{T_1 T_2}{2^{1.5n}} + \frac{T_1}{2^{1.25n}} + \frac{T_1^2}{2^{7n/3}}$$

Online-only attack (tight)

Trade-off. (tight)

Offline-only attacks, likely not tight!

What is the main challenge behind these proofs?!

Main challenge = **Offline-only attacks!**

E.g., for collision resistance of salted random oracle

X := # salts a_i for which the adversary can find the following structures

a_1 a_2 … a_X

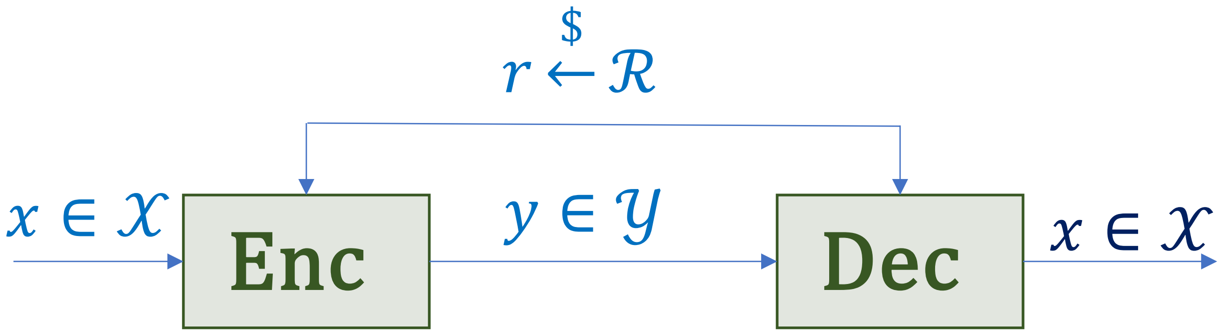
Need to upper bound $E[X]$

Unclear how when queries adaptive

We prove $\Pr\left[X \geq \max\left\{\frac{eT_1}{2^{\frac{n}{2}}}, n\right\}\right]$ is very small, which suffices

Technique: compression argument

Compression lemma



Lemma [DTT10]. Let $\varepsilon := \Pr_{x,r}[\mathrm{Dec}(\mathrm{Enc}(x,r),r) = x]$. Then

$$\log|\mathcal{Y}| \geq \log|\mathcal{X}| - \log\frac{1}{\varepsilon}$$

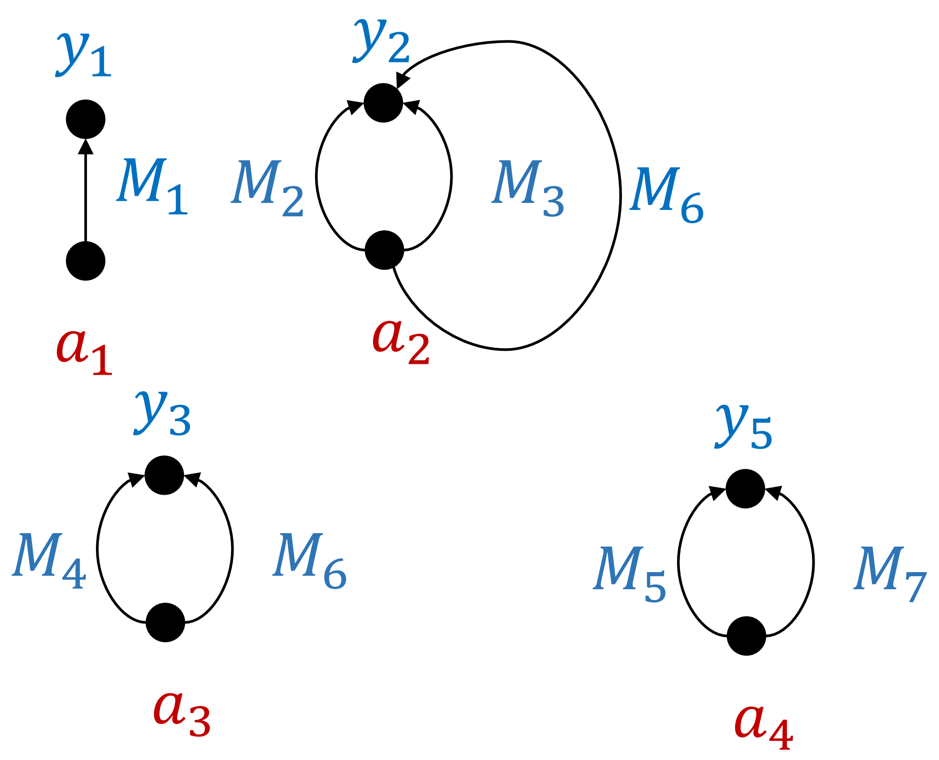
Our strategy: Encode h using A_1

Decoding would succeed as long as A_1^h finds collisions for k different salts

Encoding example

A_1^h's query transcript:

1. $((a_1, M_1), y_1)$
2. $((a_2, M_2), y_2)$
3. $((a_2, M_3), y_2)$
4. $((a_3, M_4), y_3)$
5. $((a_4, M_5), y_5)$
6. $((a_4, M_7), y_5)$
7. $((a_2, M_6), y_2)$
8. $((a_3, M_6), y_3)$



Encoding:

$S = \{2,3,4,5,6,8\}$ (set indices of colliding queries for salts)

$L = (y_1, y_2, y_3, y_5, y_2,$ rest of evaluations of $h)$

Note: only collision pair considered for a_2

Encoding:

$S = \{2,3,4,5,6,8\}$ (set indices of colliding queries for salts)

$L = (\cancel{y_1}, \cancel{y_2}, y_3, y_5, y_2,$ rest of evaluations of $h)$

How does decoding work?

Run A_1

1. $(a_1, M_1) \to y_1$
2. $(a_2, M_2) \to y_2$
3. $(a_2, M_3) \to y_2$

$\vdots$

$2 \in S$, but **no** query j on a_2 earlier such that $j \in S$

$3 \in S$ and query 2 was on a_2 and $2 \in S \Rightarrow$ collision

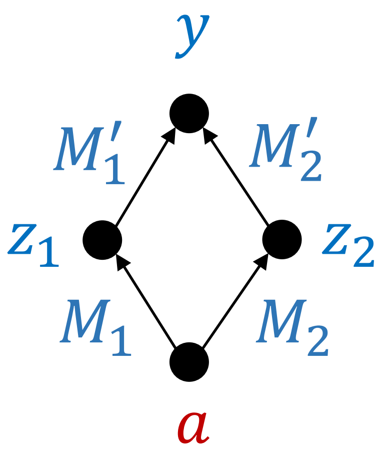
$$\epsilon \coloneqq \Pr_h[A_1^h \text{ finds cols for } k \text{ different salts}]$$

From compression lemma, it follows

$$\log \binom{T_1}{k} \geq kn - \log \frac{1}{\epsilon} \qquad \Rightarrow \epsilon \leq \frac{1}{2^n} \text{ for } k \geq \max\left\{\frac{eT_1}{2^{\frac{n}{2}}}, n\right\}$$

2-block-MD analysis: more challenging

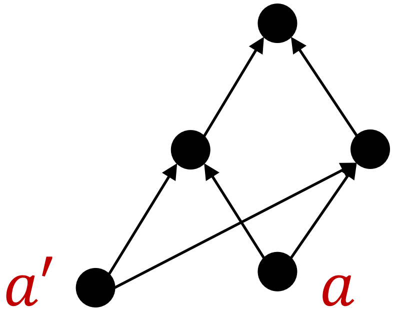
X = # salts for which collision queried in offline phase



$h(a, M_1) = z_1, h(a, M_2) = z_2, h(z_1, M_1'), = y, h(z_2, M_2') = y$

Very challenging to understand for $T_1 \gg 2^n$

Reason: Salts a, a' can share the $h(z_1, M_1)$ and $h(z_2, M_2')$ queries!



Need to be very **careful to avoid double counting**

We give a (loose) analysis using rather **sophisticated** compression arguments

Conclusions and open problems

- Salting generically defeats preprocessing (qualitatively) wrt to time complexity

- Quantitatively precise bounds need ad-hoc analysis

- Open problem: Close the gap for MD collisions? Extend beyond two blocks? Consider <u>both</u> advice size and pre-processing complexity?

ePrint: 2023/856

Thank you!