Super-Quadratic Quantum Speed-Ups and Guessing Many Likely Keys

Kaveh Bashiri
Bundesamt für Sicherheit in der Informationstechnik (BSI)
Timo Glaser
Ruhr University Bochum (RUB)
Alexander May
Ruhr University Bochum (RUB)
Julian Nowakowski
Centrum Wiskunde & Informatica (CWI)

Super-Quadratic Quantum Speed-Ups and Guessing Many Likely Keys

Kaveh Bashiri
Bundesamt für Sicherheit in der Informationstechnik (BSI)
Timo Glaser
Ruhr University Bochum (RUB)
Alexander May
Ruhr University Bochum (RUB)
Julian Nowakowski
Centrum Wiskunde & Informatica (CWI)

Key Guessing Problem

Given: oracle access to some function $f_k : K \to \{0,1\}$ with

$$f_k(x) = \begin{cases} 1, & x = k, \\ 0, & \text{else.} \end{cases}$$

Find: key k.

Key Guessing Problem

Given: oracle access to some function $f_k : K \to \{0,1\}$ with

$$f_k(x) = \begin{cases} 1, & x = k, \\ 0, & \text{else.} \end{cases}$$

Find: key k.

Optimal classical algorithm:

- Iterate over key space K in random order.
- Expected runtime $= \dfrac{|K| + 1}{2}$.

Optimal quantum algorithm:

- Grover search.
- Expected runtime $\approx |K|^{1/2}$.

Key Guessing Problem

Given: oracle access to some function $f_k : K \to \{0,1\}$ with

$$f_k(x) = \begin{cases} 1, & x = k, \\ 0, & \text{else.} \end{cases}$$

Find: key k.

Optimal cla

What if we know the probability distribution D of k?

- Iterate over key space K in random order.
- Expected runtime $= \dfrac{|K| + 1}{2}$.

Grover search.

- Expected runtime $\approx |K|^{1/2}$.

Key Guessing Problem

Given: oracle access to some function $f_k : K \to \{0,1\}$ with

$$f_k(x) = \begin{cases} 1, & x = k, \\ 0, & \text{else.} \end{cases}$$

Find: key k.

Optimal cla

- Iterate over key space K in random order. Grover search.

- Expected

What if we know the probability distribution D of k?

Main result:

Super-quadratic quantum speed-up for every non-uniform D.

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th most likely key.

Key Guessing from a Distribution *D*

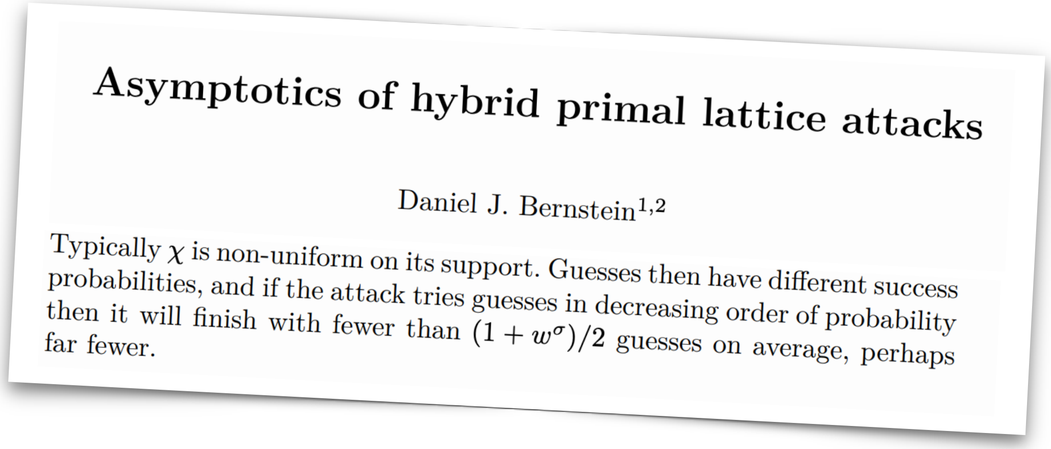
Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th most likely key.



Asymptotics of hybrid primal lattice attacks

Daniel J. Bernstein[1,2]

Typically χ is non-uniform on its support. Guesses then have different success probabilities, and if the attack tries guesses in decreasing order of probability then it will finish with fewer than $(1 + w^\sigma)/2$ guesses on average, perhaps far fewer.

Key Guessing from a Distribution *D*

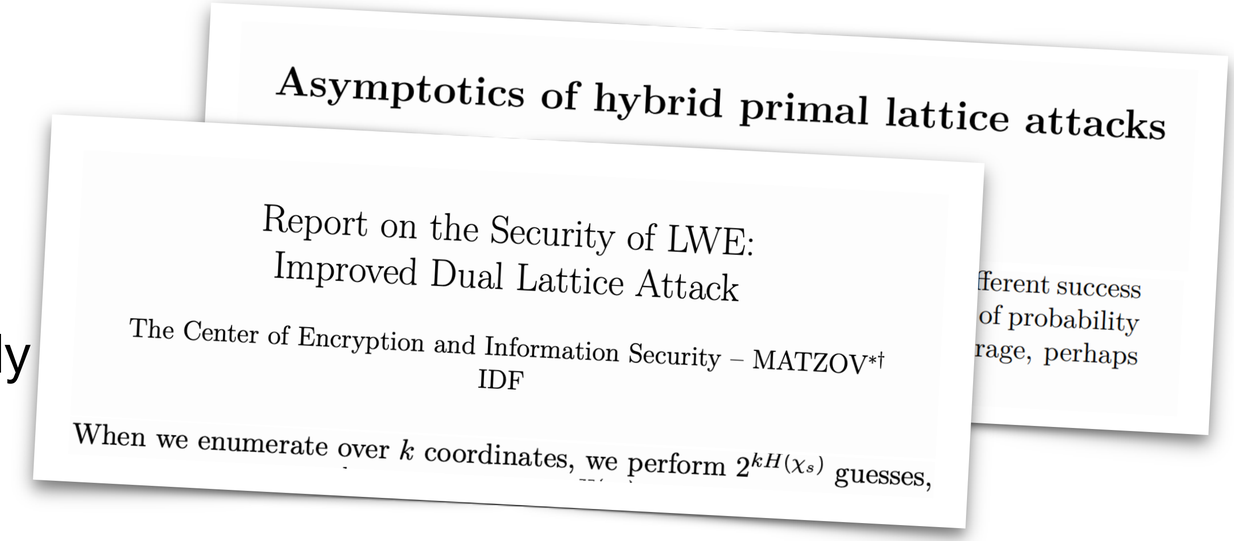
Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th most likely


Asymptotics of hybrid primal lattice attacks

Report on the Security of LWE:
Improved Dual Lattice Attack

The Center of Encryption and Information Security – MATZOV*†
IDF

When we enumerate over k coordinates, we perform $2^{kH(\chi_s)}$ guesses,

fferent success
of probability
rage, perhaps

Key Guessing from a Distribution *D*

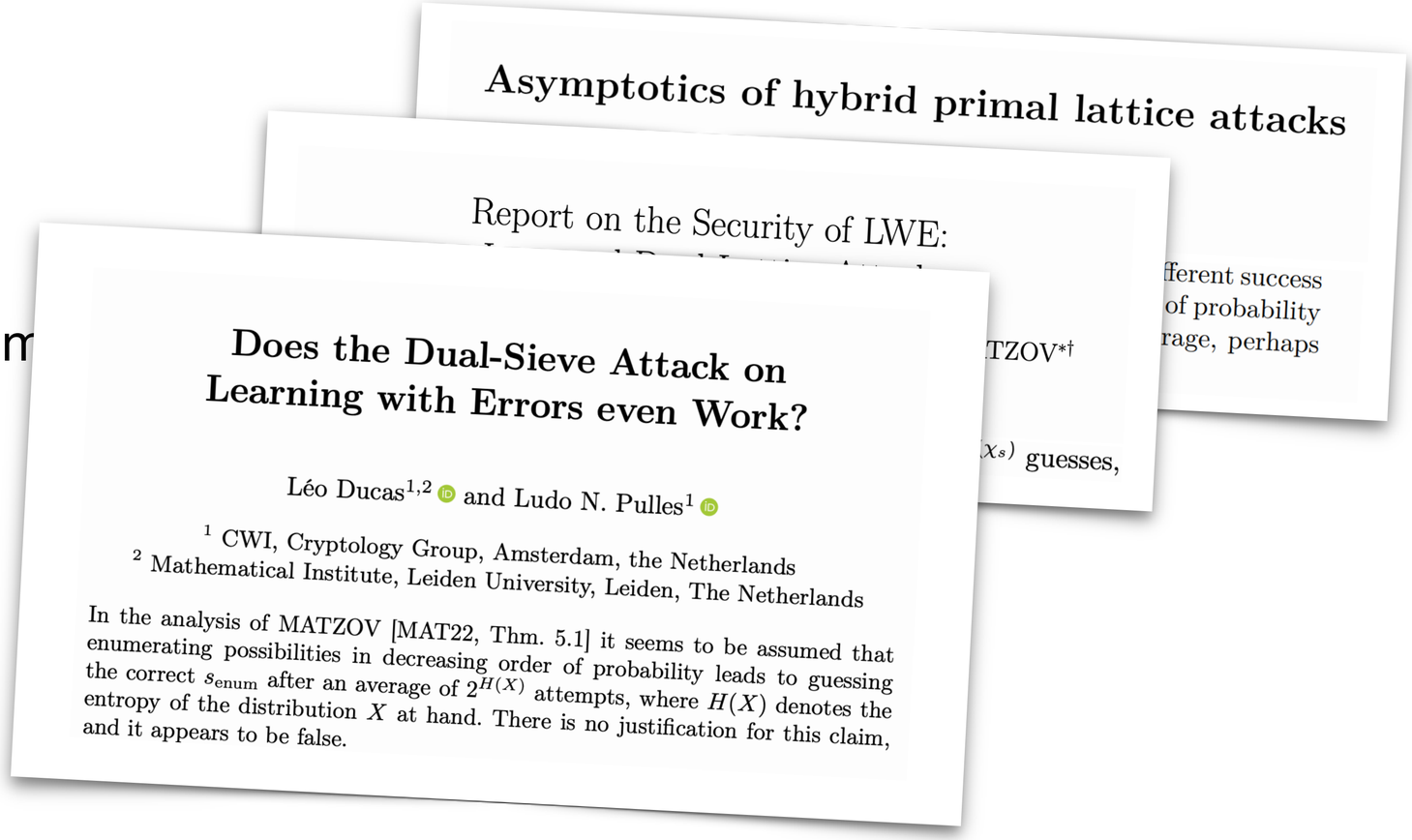
Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th m


Asymptotics of hybrid primal lattice attacks

Report on the Security of LWE:

Does the Dual-Sieve Attack on Learning with Errors even Work?

Léo Ducas[1,2] and Ludo N. Pulles[1]

[1] CWI, Cryptology Group, Amsterdam, the Netherlands
[2] Mathematical Institute, Leiden University, Leiden, The Netherlands

In the analysis of MATZOV [MAT22, Thm. 5.1] it seems to be assumed that enumerating possibilities in decreasing order of probability leads to guessing the correct s_{enum} after an average of $2^{H(X)}$ attempts, where $H(X)$ denotes the entropy of the distribution X at hand. There is no justification for this claim, and it appears to be false.

Key Guessing from a Distribution *D*

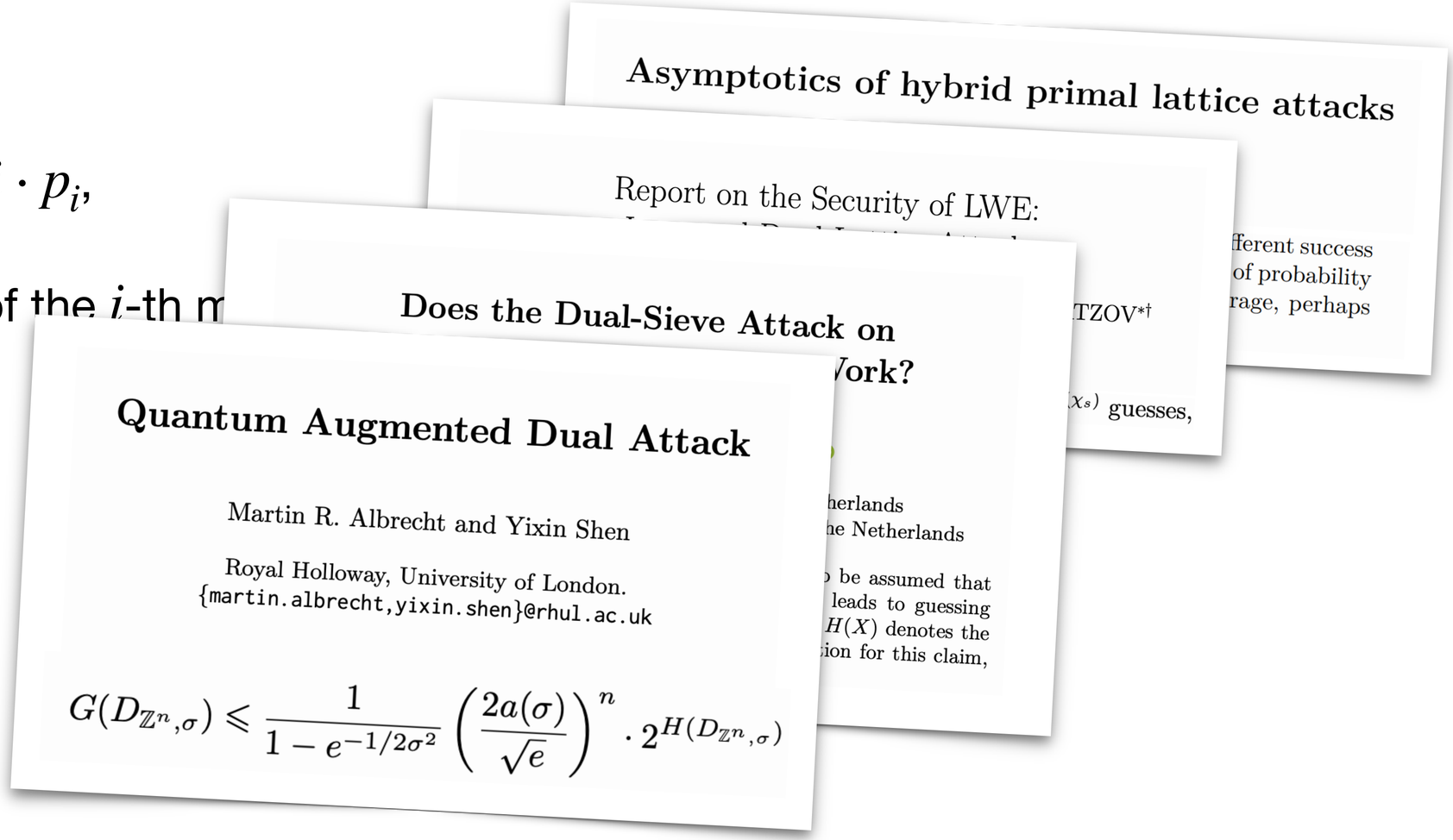
Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th m


Asymptotics of hybrid primal lattice attacks

Report on the Security of LWE:

Does the Dual-Sieve Attack on

Quantum Augmented Dual Attack

Martin R. Albrecht and Yixin Shen

Royal Holloway, University of London.
{martin.albrecht,yixin.shen}@rhul.ac.uk

$$G(D_{\mathbb{Z}^n,\sigma}) \leqslant \frac{1}{1-e^{-1/2\sigma^2}} \left(\frac{2a(\sigma)}{\sqrt{e}}\right)^n \cdot 2^{H(D_{\mathbb{Z}^n,\sigma})}$$

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Iterate over key space K in decreasing order of likelihood.

- Expected runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i,$$

where p_i is the probability of the i-th m

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Ite
- Ex

wh

IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 42, NO. 1, JANUARY 1996 99

An Inequality on Guessing and its Application to Sequential Decoding

Erdal Arikan, *Senior Member, IEEE*

Improved Estimation of Key Enumeration with Applications to Solving LWE

Alessandro Budroni[1] and Erik Mårtensson[2,3]
Cryptography Research Center, Technology Innovation Institute, Abu Dhabi, UAE[1]
alessandro.budroni@tii.ae
Selmer Center, Department of Informatics, University of Bergen, Norway[2]
Department of Electrical and Information Technology, Lund University, Sweden[3]
erik.martensson@{uib.no,eit.lth.se}

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Ite
- Ex

IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 42, NO. 1, JANUARY 1996

An Inequality on Guessing and its Application to Sequential Decoding

Erdal Arikan, *Senior Member, IEEE*

$$\sum_{i=1}^{|K|} i \cdot p_i \leq 2^{H_{1/2}(D)}$$

Rényi entropy with parameter 1/2

Key Guessing from a Distribution *D*

Optimal classical algorithm:

- Ite
- Ex

IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 42, NO. 1, JANUARY 1996

An Inequality on Guessing and its Application to Sequential Decoding

Erdal Arikan, *Senior Member, IEEE*

$$\frac{2^{H_{1/2}(D)}}{1+\log|K|} \leq \sum_{i=1}^{|K|} i \cdot p_i \leq 2^{H_{1/2}(D)}$$

Rényi entropy with parameter 1/2

Rényi Entropy

For a distribution D with support K and probabilities $p_1, \ldots, p_{|K|}$, we define

$$H_\alpha(D) := \lim_{\gamma \to \alpha} \frac{1}{1-\gamma} \log \left(\sum_{i=1}^{|K|} p_i^\gamma \right).$$

Rényi Entropy

For a distribution D with support K and probabilities $p_1, \ldots, p_{|K|}$, we define

$$H_\alpha(D) := \lim_{\gamma \to \alpha} \frac{1}{1-\gamma} \log\left(\sum_{i=1}^{|K|} p_i^\gamma\right).$$

Special cases:

- $H_0(D) = \log(|K|)$

  ☞Max entropy

- $H_1(D) = -\sum_i p_i \log(p_i)$

  ☞ Shannon entropy

- $H_\infty(D) = \min_i -\log\left(p_i\right)$

  ☞ Min entropy

Rényi Entropy

For a distribution D with support K and probabilities $p_1, \ldots, p_{|K|}$, we define

$$H_\alpha(D) := \lim_{\gamma \to \alpha} \frac{1}{1-\gamma} \log\left(\sum_{i=1}^{|K|} p_i^\gamma\right).$$

Special cases:

- $H_0(D) = \log(|K|)$

  ☞Max entropy

- $H_1(D) = -\sum_i p_i \log(p_i)$

  ☞ Shannon entropy

- $H_\infty(D) = \min_i -\log\left(p_i\right)$

  ☞ Min entropy

Brute force: $2^{H_0(D)}$

Guessing in order of likelihood: $2^{H_{1/2}(D)}$

Rényi Entropy

For a distribution D with support K and probabilities $p_1, \ldots, p_{|K|}$, we define

$$H_\alpha(D) := \lim_{\gamma \to \alpha} \frac{1}{1-\gamma} \log \left(\sum_{i=1}^{|K|} p_i^\gamma \right).$$

Special cases:

- $H_0(D) = \log(|K|)$

  ☞Max entropy

- $H_1(D) = -\sum_i p_i \log(p_i)$

  ☞ Shannon entropy

- $H_\infty(D) = \min_i -\log(p_i)$

  ☞ Min entropy

Brute force: $2^{H_0(D)}$

Guessing in order of likelihood: $2^{H_{1/2}(D)}$

Properties:

- D uniform

  $\implies H_\alpha(D) = H_\beta(D)$, for all $\alpha, \beta \in [0, \infty]$

- D non-uniform

  $\implies H_\alpha(D)$ decreases as α increases

- Product distributions $D = \chi^n$

  $\implies H_\alpha(D) = n \cdot H_\alpha(\chi)$

Quantum Key Guessing from a Distribution *D*

Optimal quantum algorithm:

- Sort keys in decreasing order of likelihood $k_1, k_2, \ldots, k_{|K|}$.
- Partition keys as
  - $K_0 = \{k_1\}$,
  - $K_1 = \{k_2, k_3\}$,
  - $K_2 = \{k_4, k_5, k_6, k_7\}$,
  - $K_3 = \{k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}\}$
  - $\ldots$
- Run Grover search on $K_0, K_1, K_2, K_3, \ldots$

Montanaro, TQC 2011

Quantum Key Guessing from a Distribution *D*

Optimal quantum algorithm:

- Sort keys in decreasing order of likelihood $k_1, k_2, \ldots, k_{|K|}$.
- Partition keys as
  - $K_0 = \{k_1\}$,
  - $K_1 = \{k_2, k_3\}$,
  - $K_2 = \{k_4, k_5, k_6, k_7\}$,
  - $K_3 = \{k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}\}$
  - $\ldots$
- Run Grover search on $K_0, K_1, K_2, K_3, \ldots$

Montanaro, TQC 2011

Runtime analysis:

- For the i-th most likely key, we run Grover search on $K_0, K_1, \ldots, K_t$, where $t = \lfloor \log(i) \rfloor$.

Quantum Key Guessing from a Distribution *D*

Optimal quantum algorithm:

- Sort keys in decreasing order of likelihood $k_1, k_2, \ldots, k_{|K|}$.
- Partition keys as
  - $K_0 = \{k_1\}$,
  - $K_1 = \{k_2, k_3\}$,
  - $K_2 = \{k_4, k_5, k_6, k_7\}$,
  - $K_3 = \{k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}\}$
  - $\ldots$
- Run Grover search on $K_0, K_1, K_2, K_3, \ldots$

Montanaro, TQC 2011

Runtime analysis:

- For the i-th most likely key, we run Grover search on $K_0, K_1, \ldots, K_t$, where $t = \lfloor \log(i) \rfloor$.
- $\Longrightarrow$ We find i-th most likely key in time

$$\sum_{j=1}^{t} |K_j|^{1/2}$$

Quantum Key Guessing from a Distribution *D*

Optimal quantum algorithm:

- Sort keys in decreasing order of likelihood $k_1, k_2, \ldots, k_{|K|}$.
- Partition keys as
  - $K_0 = \{k_1\}$,
  - $K_1 = \{k_2, k_3\}$,
  - $K_2 = \{k_4, k_5, k_6, k_7\}$,
  - $K_3 = \{k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}\}$
  - $\ldots$
- Run Grover search on $K_0, K_1, K_2, K_3, \ldots$

Montanaro, TQC 2011

Runtime analysis:

- For the i-th most likely key, we run Grover search on $K_0, K_1, \ldots, K_t$, where $t = \lfloor \log(i) \rfloor$.
- $\Longrightarrow$ We find i-th most likely key in time

$$\sum_{j=1}^{t} |K_j|^{1/2} = \sum_{j=1}^{t} 2^{j/2} \approx 2^{t/2} \approx \sqrt{i}\,.$$

Quantum Key Guessing from a Distribution *D*

Optimal quantum algorithm:

- Sort keys in decreasing order of likelihood $k_1, k_2, \ldots, k_{|K|}$.
- Partition keys as
  - $K_0 = \{k_1\}$,
  - $K_1 = \{k_2, k_3\}$,
  - $K_2 = \{k_4, k_5, k_6, k_7\}$,
  - $K_3 = \{k_8, k_9, k_{10}, k_{11}, k_{12}, k_{13}, k_{14}, k_{15}\}$
  - $\ldots$
- Run Grover search on $K_0, K_1, K_2, K_3, \ldots$

Montanaro, TQC 2011

Runtime analysis:

- For the i-th most likely key, we run Grover search on $K_0, K_1, \ldots, K_t$, where $t = \lfloor \log(i) \rfloor$.
- $\Longrightarrow$ We find i-th most likely key in time

$$\sum_{j=1}^{t} |K_j|^{1/2} = \sum_{j=1}^{t} 2^{j/2} \approx 2^{t/2} \approx \sqrt{i}\,.$$

- $\Longrightarrow$ Expected runtime is

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i,$$

where p_i is the probability of the i-th most likely key.

Key Guessing from a Distribution *D*

Classical runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i$$

Quantum runtime:

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i$$

Key Guessing from a Distribution *D*

Classical runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i \approx 2^{H_{1/2}(D)}$$

Quantum runtime:

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i$$

Key Guessing from a Distribution *D*

Classical runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i \approx 2^{H_{1/2}(D)}$$

Quantum runtime:

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i$$

Arikan, 1996:

For every $\rho \geq 0$, we have $\sum_{i=1}^{|K|} i^{\rho} \cdot p_i \approx 2^{\rho H_{1/(1+\rho)}(D)}$

Key Guessing from a Distribution *D*

Classical runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i \approx 2^{H_{1/2}(D)}$$

Quantum runtime:

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i \approx 2^{H_{2/3}(D)/2}$$

$\rho = 1/2$

Arikan, 1996:

For every $\rho \geq 0$, we have $\sum_{i=1}^{|K|} i^{\rho} \cdot p_i \approx 2^{\rho H_{1/(1+\rho)}(D)}$

Key Guessing from a Distribution *D*

Classical runtime:

$$\sum_{i=1}^{|K|} i \cdot p_i \approx 2^{H_{1/2}(D)}$$

Quantum runtime:

$$\sum_{i=1}^{|K|} \sqrt{i} \cdot p_i \approx 2^{H_{2/3}(D)/2}$$

$\rho = 1/2$

Arikan, 1996:

For every $\rho \geq 0$, we have $\sum_{i=1}^{|K|} i^{\rho} \cdot p_i \approx 2^{\rho H_{1/(1+\rho)}(D)}$

Quantum speed-up: $2\frac{H_{1/2}(D)}{H_{2/3}(D)} > 2$

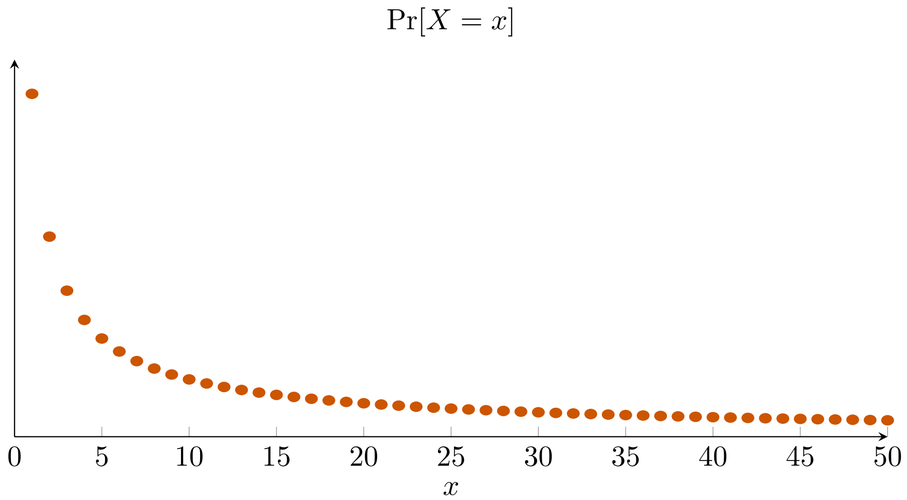
for every non-uniform D

Example: Zipf Distribution (Passwords)

Definition:

- Fix parameters N and t.
- $\Pr[X = x] \propto \dfrac{1}{x^t}$ for $x = 1, \ldots, N$.

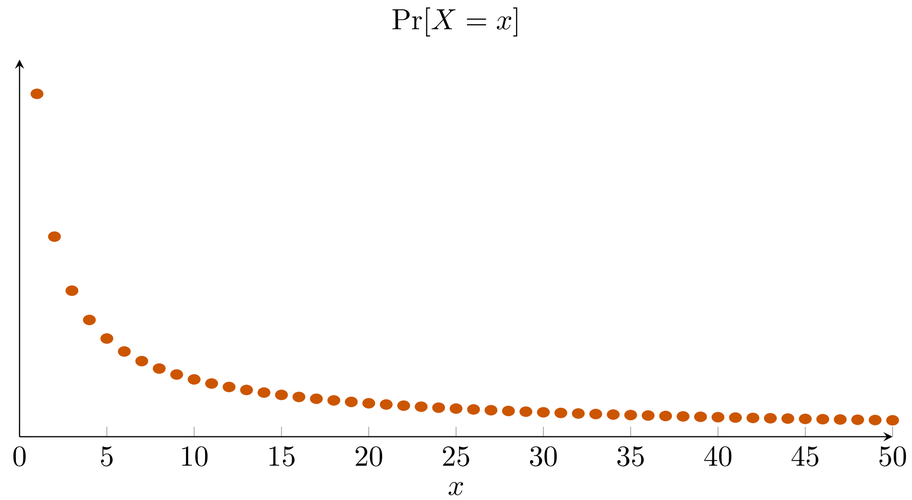
- Leaked LinkedIn passwords (approximately) follow Zipf distribution with $N = 2^{27.2\cdots}$ and $t = 0.777$.



Example: Zipf Distribution (Passwords)

Definition:

- Fix parameters N and t.
- $\Pr[X = x] \propto \dfrac{1}{x^t}$ for $x = 1, \ldots, N$.
- Leaked LinkedIn passwords (approximately) follow Zipf distribution with $N = 2^{27.2\ldots}$ and $t = 0.777$.



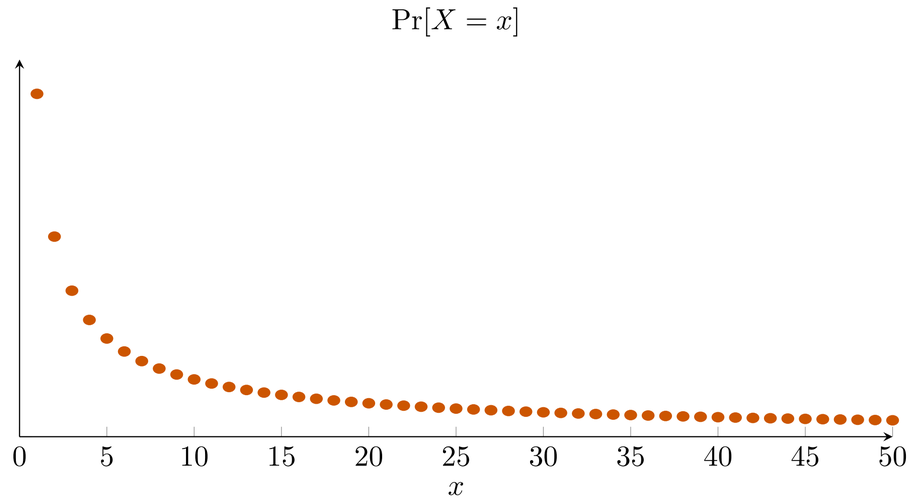
Classical guessing:

- $2^{H_{1/2}(D)} = 2^{26.5\ldots}$

Example: Zipf Distribution (Passwords)

Definition:

- Fix parameters N and t.
- $\Pr[X = x] \propto \dfrac{1}{x^t}$ for $x = 1, \ldots, N$.
- Leaked LinkedIn passwords (approximately) follow Zipf distribution with $N = 2^{27.2\ldots}$ and $t = 0.777$.



Classical guessing:

- $2^{H_{1/2}(D)} = 2^{26.5\ldots}$

Quantum guessing:

- $2^{H_{2/3}(D)/2} = 2^{13.1\ldots}$

Speed-up:

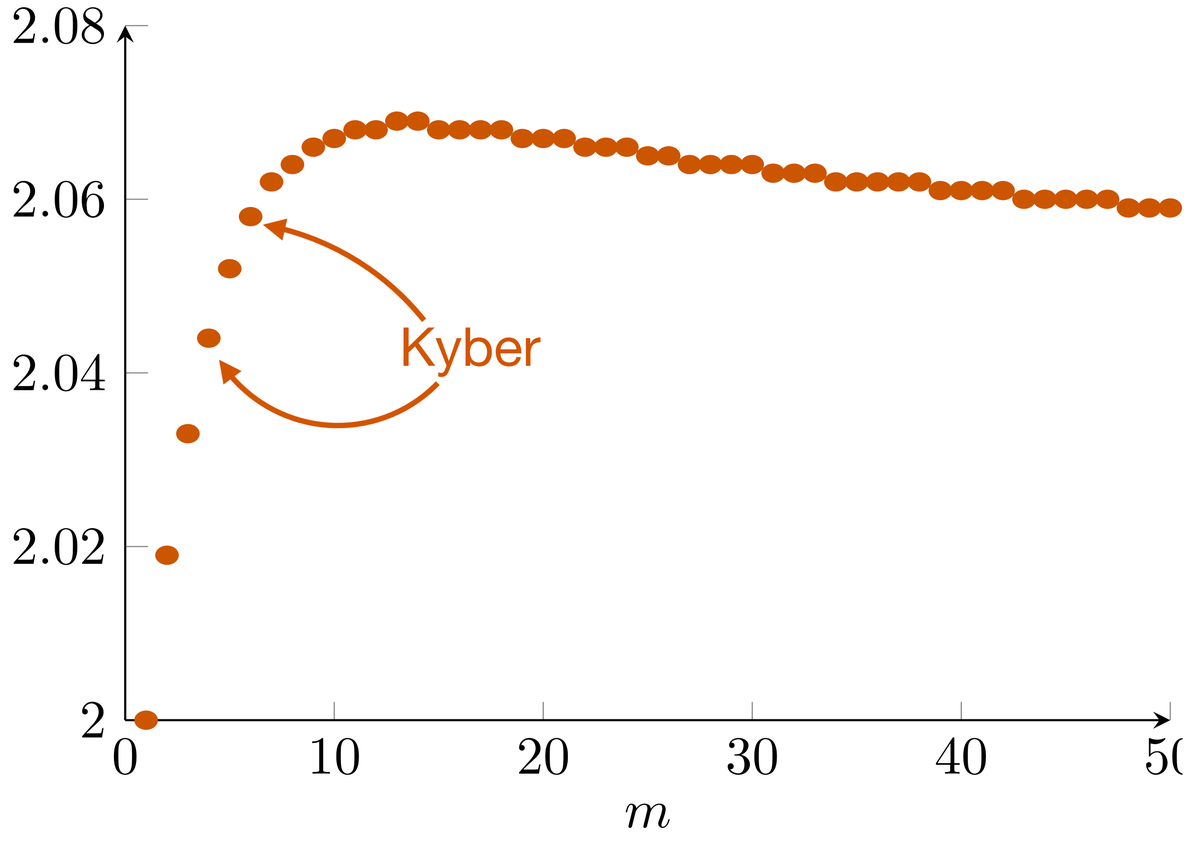
- 2.03

Example: Centered Binomial (Kyber)

Definition:

- Sample $(X_1, \ldots, X_m) \leftarrow \{0,1\}^m$ uniformly.
- Output $X_1 + \ldots + X_m - \frac{m}{2}$.

Quantum speed-up:



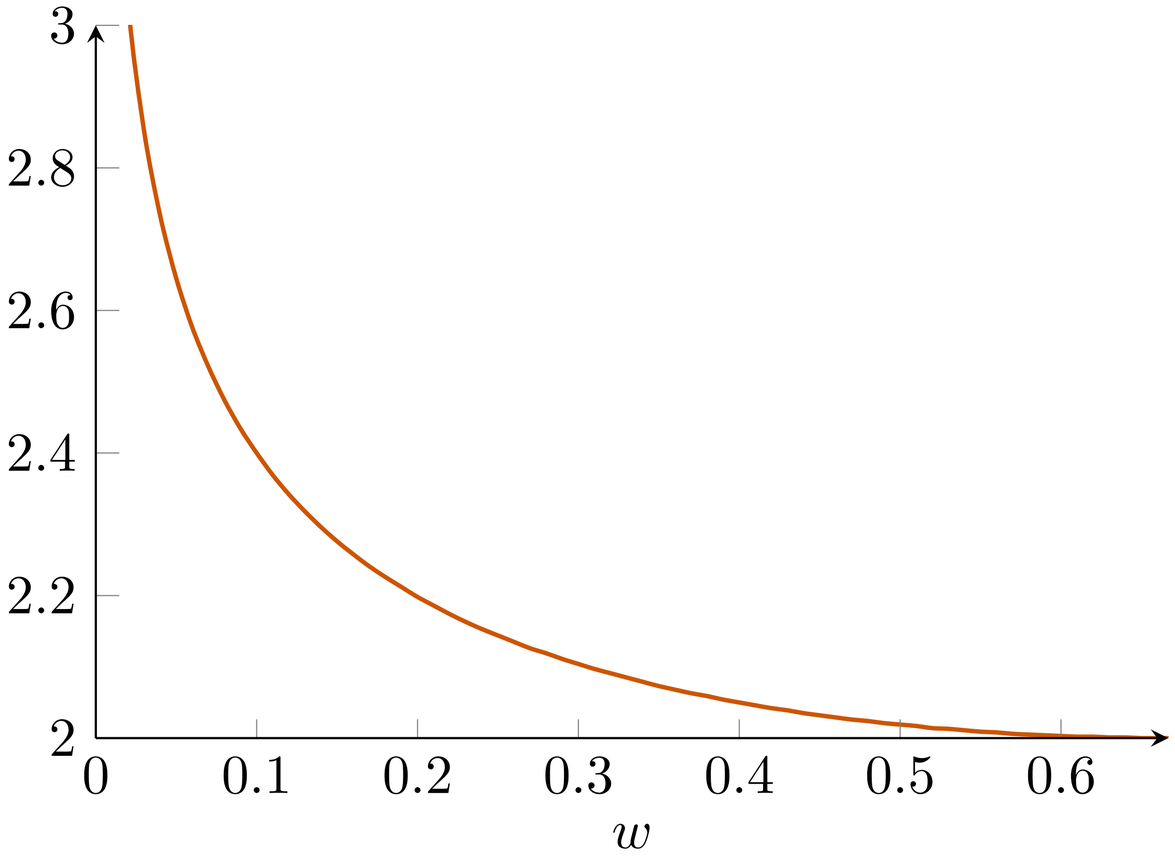
Example: Weighted Ternary (FHE)

Definition:

- $\Pr[X = 1] = w/2$ ☞$w \in (0,2/3]$
- $\Pr[X = -1] = w/2$
- $\Pr[X = 0] = 1 - w$

- Used for sampling ternary n-dimensional vectors of expected Hamming weight wn.

Quantum speed-up:

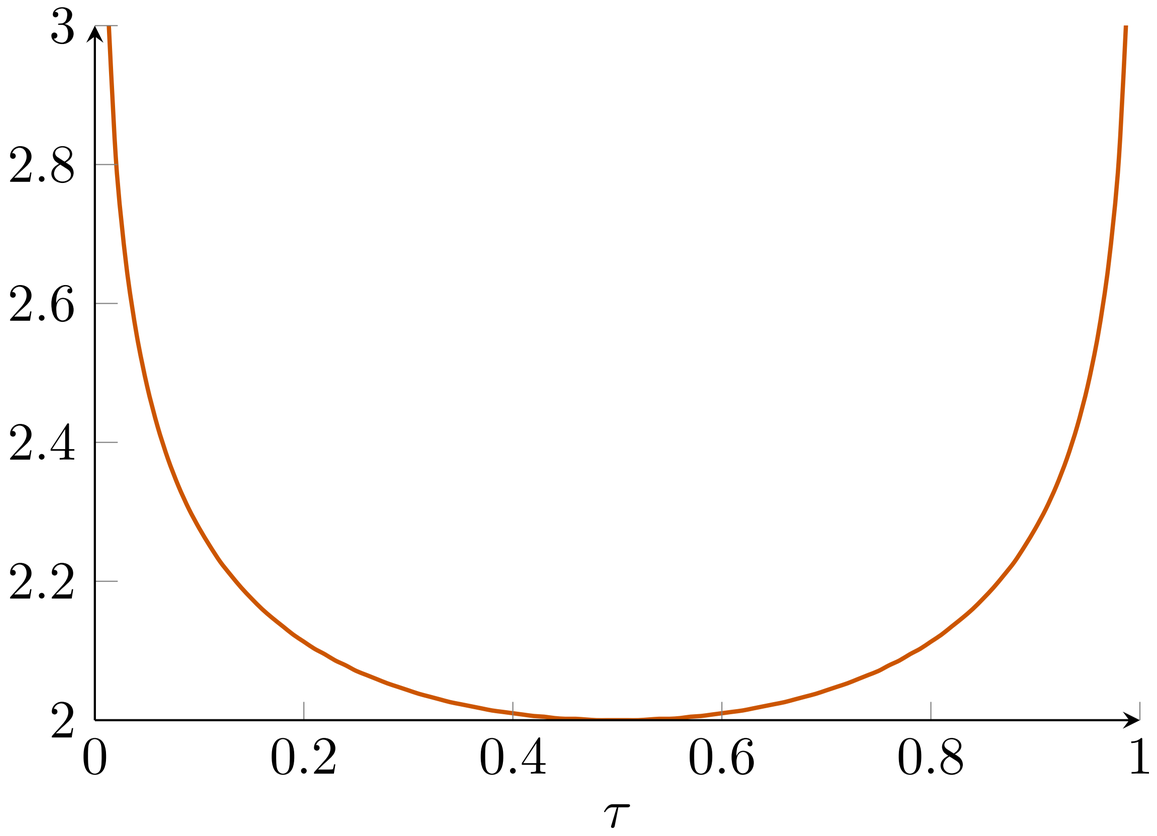


Example: Bernoulli Distribution (LPN)

Definition:

- $\Pr[X = 1] = \tau$
- $\Pr[X = 0] = 1 - \tau$

Quantum speed-up:



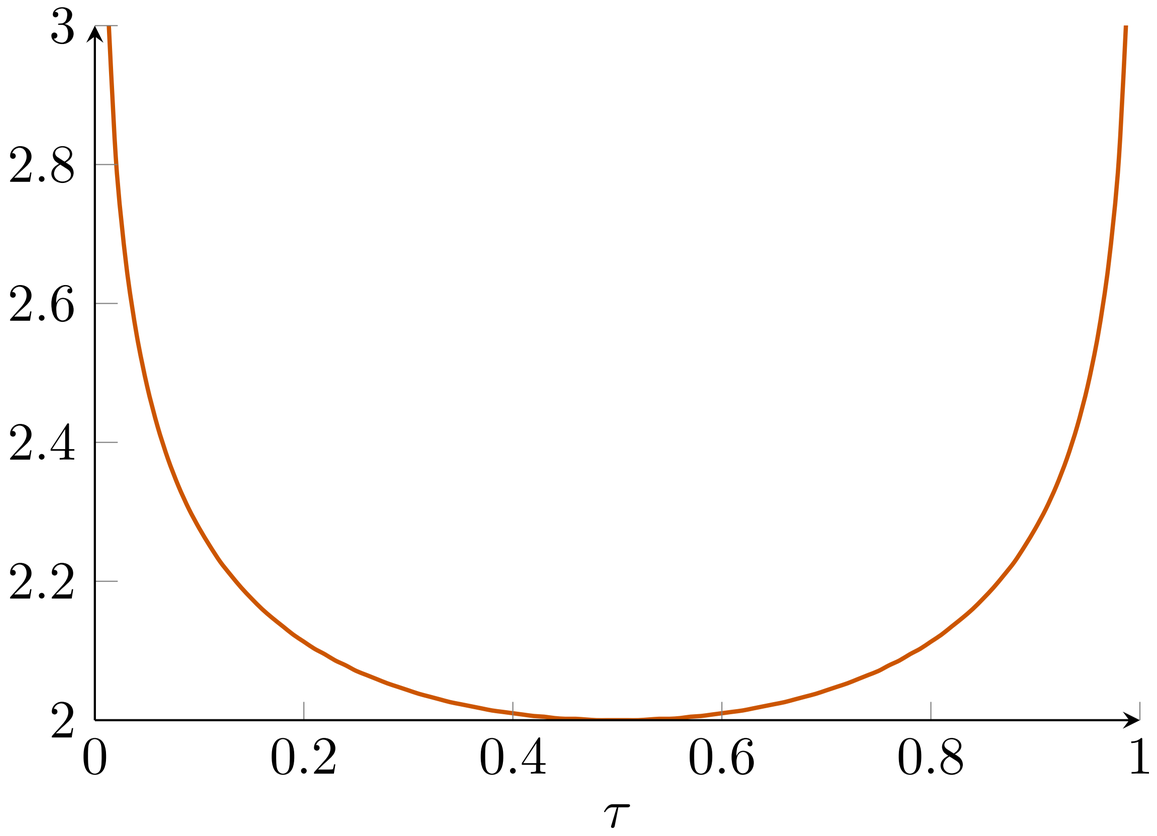
Example: Bernoulli Distribution (LPN)

Definition:

- $\Pr[X = 1] = \tau$
- $\Pr[X = 0] = 1 - \tau$

- LPN-based encryption uses n-fold Bernoulli with $\tau = n^{-1/2}$.
- Quantum speed-up: $\approx n^{1/12}$.

Quantum speed-up:



Super-Quadratic Quantum Speed-Ups and Guessing Many Likely Keys

Super-Quadratic Quantum Speed-Ups and Guessing Many Likely Keys

Likely Keys

Example:

- LPN subkey drawn from n-fold Bernoulli distribution with $n = 15$, $\tau = 0.2$.
- Let T be RV for number of required guesses.

- Arikan: $\mathbb{E}[T] \approx 2^{H_{1/2}(D)} = 6476.64\ldots$

Likely Keys

Example:

- LPN subkey drawn from n-fold Bernoulli distribution with $n = 15, \tau = 0.2$.
- Let T be RV for number of required guesses.

- Arikan: $\mathbb{E}[T] \approx 2^{H_{1/2}(D)} = 6476.64\ldots$

$\boxed{\text{Does } T \text{ concentrate around } \mathbb{E}[T]?}$

- Markov's inequality: $T \gg \mathbb{E}[T]$ is unlikely.
- What about $T \ll \mathbb{E}[T]$?

Likely Keys

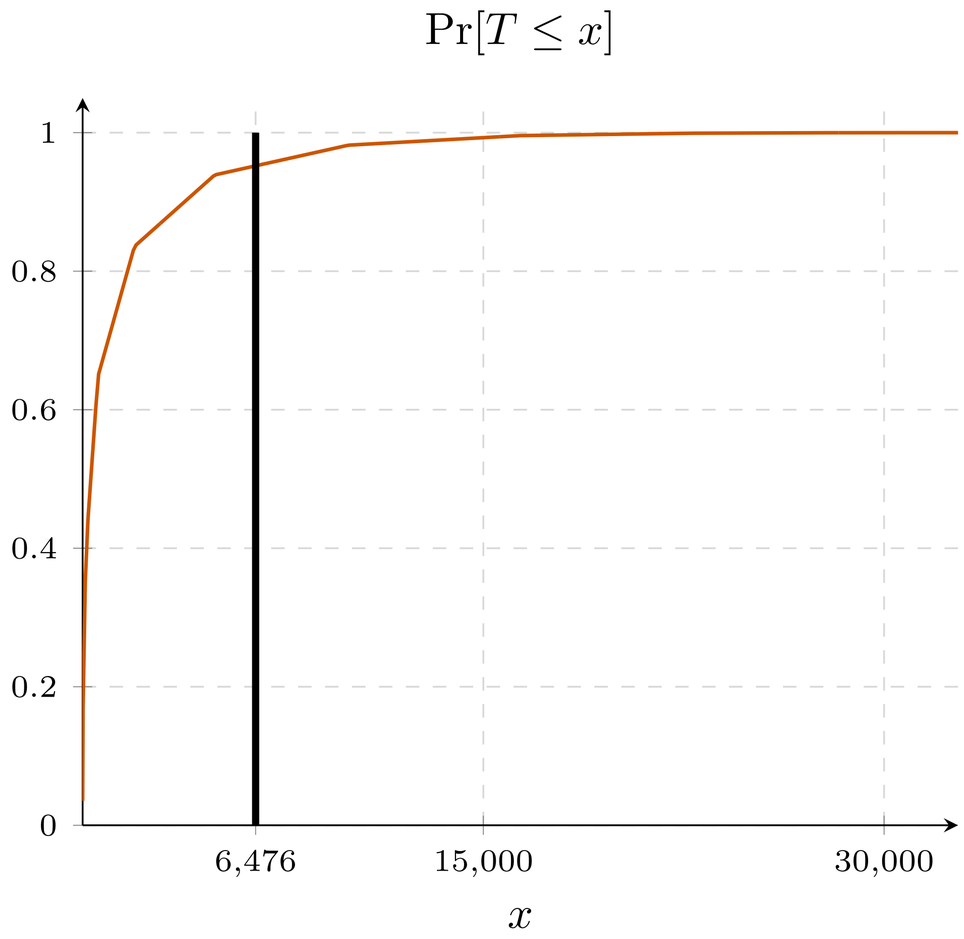
Example:

- LPN subkey drawn from n-fold Bernoulli distribution with $n = 15$, $\tau = 0.2$.
- Let T be RV for number of required guesses.

- Arikan: $\mathbb{E}[T] \approx 2^{H_{1/2}(D)} = 6476.64\ldots$

$$\boxed{\text{Does } T \text{ concentrate around } \mathbb{E}[T]?}$$

- Markov's inequality: $T \gg \mathbb{E}[T]$ is unlikely.
- What about $T \ll \mathbb{E}[T]$?



Likely Keys

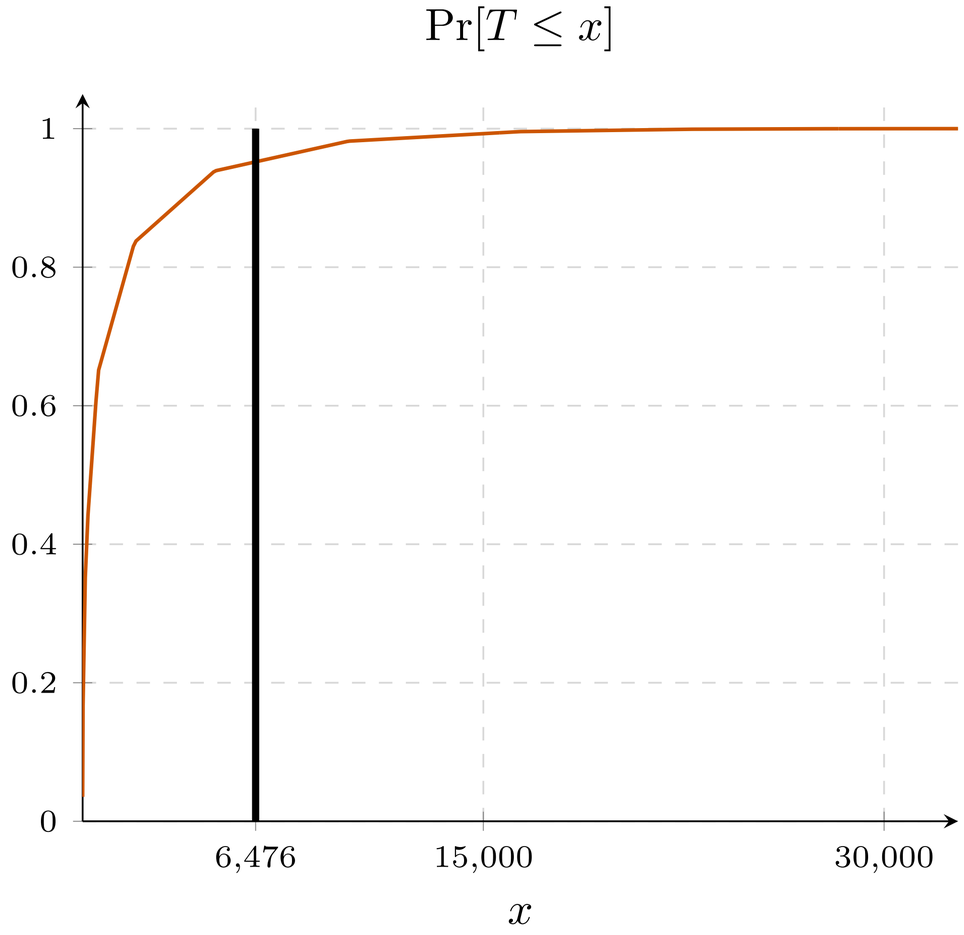
Example:

- LPN subkey drawn from n-fold Bernoulli distribution with $n = 15$, $\tau = 0.2$.
- Let T be RV for number of required guesses.

- Arikan: $\mathbb{E}[T] \approx 2^{H_{1/2}(D)} = 6476.64\ldots$

$$\boxed{\text{Does } T \text{ concentrate around } \mathbb{E}[T]?}$$

- Markov's inequality: $T \gg \mathbb{E}[T]$ is unlikely.
- What about $T \ll \mathbb{E}[T]$?



⚠ **32,768** possible keys.
307 most likely keys carry 50% of probability mass.

Beyond Expected Runtime

Alternative measures of complexity:

- ratio T/ε of runtime T and success probability ε
- expected time to find one out of m keys

Beyond Expected Runtime

Alternative measures of complexity:

- ratio T/ε of runtime T and success probability ε
- expected time to find one out of m keys

Results:

- T/ε is minimized when trying only the most likely key.

Beyond Expected Runtime

Alternative measures of complexity:

- ratio T/ε of runtime T and success probability ε
- expected time to find one out of m keys

Results:

- T/ε is minimized when trying only the most likely key. ☞ $T/\varepsilon = 2^{H_\infty(D)}$

Beyond Expected Runtime

Alternative measures of complexity:

- ratio T/ε of runtime T and success probability ε
- expected time to find one out of m keys

Results:

- T/ε is minimized when trying only the most likely key. ☞ $T/\varepsilon = 2^{H_\infty(D)}$
- Upper bounds for one-out-of-m:

$$m \cdot 2^{H_{1-1/(m+1)}(D)}$$

classically

$$m \cdot \sqrt{2^{H_{1-1/(2m+1)}(D)}}$$

quantumly

Beyond Expected Runtime

Alternative measures of complexity:

- ratio T/ε of runtime T and success probability ε
- expected time to find one out of m keys

Results:

- T/ε is minimized when D is the most likely key. ☞ $T/\varepsilon = 2^{H_\infty(D)}$
- Upper bounds for one-out-of-m:

m	1	∞
classically	$2^{H_{1/2}(D)}$	$2^{H_1(D)}$
quantumly	$\sqrt{2^{H_{2/3}(D)}}$	$\sqrt{2^{H_1(D)}}$

$$m \cdot 2^{H_{1-1/(m+1)}(D)}$$

classically

$$m \cdot \sqrt{2^{H_{1-1/(2m+1)}(D)}}$$

quantumly

Is H_1 the *correct* measure of complexity?

Is H_1 the *correct* measure of complexity?

For product distributions $D = \chi^n$, we have two results pointing in this direction.

Is H_1 the *correct* measure of complexity?

For product distributions $D = \chi^n$, we have two results pointing in this direction.

Lower bound for all classical algorithms

enumerating $\ll 2^{H_1(D)}$ candidates

$\Longrightarrow$ succ. probability ≈ 0

Algorithm matching the lower bound

sampling $2^{H_1(D)}$ times from D

$\Longrightarrow$ succ. probability $\gtrapprox 5/16$

$\ll$, $\approx$, $\gtrapprox$ depend on n

Follow-up Work: Impact for Lattices

[Karenin, Kirshanova, May, N., AC'25]:

- First practical implementaion of CVPP-style hybrid attack against LWE/BDD

[Albrecht, Curtis, Wunderer SAC'19]

[Laarhoven, Ducas, van Woerden, PKC'20]

[Bernstein EPRINT'23]

Follow-up Work: Impact for Lattices

[Karenin, Kirshanova, May, N., AC'25]:

- First practical implementaion of CVPP-style hybrid attack against LWE/BDD

[Albrecht, Curtis, Wunderer SAC'19]

[Laarhoven, Ducas, van Woerden, PKC'20]
[Bernstein EPRINT'23]

- Enhances previous hybrid attacks by using our $2^{H_1(D)}$-time algorithm as a subroutine.

Follow-up Work: Impact for Lattices

[Karenin, Kirshanova, May, N., AC'25]:

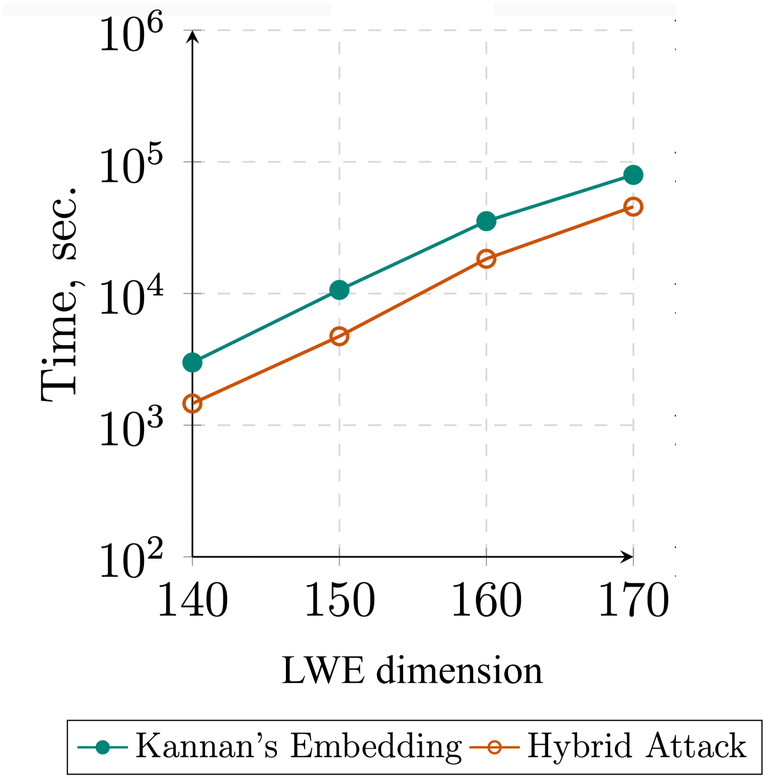
- First practical implementaion of CVPP-style hybrid attack against LWE/BDD

[Albrecht, Curtis, Wunderer SAC'19]

[Laarhoven, Ducas, van Woerden, PKC'20]
[Bernstein EPRINT'23]

- Enhances previous hybrid attacks by using our $2^{H_1(D)}$-time algorithm as a subroutine.

Practical performance on down-scaled Kyber



Summary

	Single Key	One-out-of-m
classically	$H_{1/2}(D)$	$H_{1-\frac{1}{m+1}}(D)$
quantumly	$H_{2/3}(D)/2$	$H_{1-\frac{1}{2m+1}}(D)/2$

https://ia.cr/2023/797