On Bounded Distance Decoding with Predicate: Breaking the "Lattice Barrier" for the Hidden Number Problem

Martin Albrecht and Nadia Heninger

June 14, 2021

The hidden number problem

[Boneh Venkatesan 96]

Secret: Integer α. **Public parameter:** Integer n

Input: Pairs (t_i, a_i) where a_i are most significant bits of $t_i\alpha \bmod n$.

Desired Output: α

The hidden number problem

[Boneh Venkatesan 96]

Secret: Integer α. **Public parameter:** Integer n

Input: Pairs (t_i, a_i) where a_i are most significant bits of $t_i\alpha \bmod n$.

Desired Output: α

Can formulate system of equations in unknowns $r_1, \ldots, r_m, \alpha$:

$$\begin{aligned} r_1 - t_1\alpha + a_1 &\equiv 0 \bmod n \\ r_2 - t_2\alpha + a_2 &\equiv 0 \bmod n \\ &\vdots \\ r_m - t_m\alpha + a_m &\equiv 0 \bmod n \end{aligned}$$

Here the r_i are small.

HNP Application: (EC)DSA Key Recovery

Global Parameters Group of order n with generator G.

Private Key Integer d Public Key $Q = dG$

Signature Generation

Message Hash: h

Per-Signature "nonce": Integer k

Signature on h: (r, s) $r = x(kG)$ $s = k^{-1}(h + dr) \bmod n$

HNP Application: (EC)DSA Key Recovery

Global Parameters Group of order n with generator G.

Private Key Integer d Public Key $Q = dG$

Signature Generation

Message Hash: h

Per-Signature "nonce": Integer k

Signature on h: (r, s) $r = x(kG)$ $s = k^{-1}(h + dr) \bmod n$

Formulation as a HNP instance:

Attacker learns some MSBs of nonces via a side channel.

(Assume 0 wlog, so k_i are "small".)

HNP instance:

$$k_1 - s_1^{-1} r_1 d - s_1^{-1} h_1 \equiv 0 \bmod n$$

$$k_2 - s_2^{-1} r_2 d - s_2^{-1} h_2 \equiv 0 \bmod n$$

$$\vdots$$

$$k_m - s_m^{-1} r_m d - s_m^{-1} h_m \equiv 0 \bmod n$$

Solving HNP with Lattices

Usual approach: Use BKZ to find solution vector

Input:

$$r_1 - t_1\alpha + a_1 \equiv 0 \bmod n$$
$$\vdots$$
$$r_m - t_m\alpha + a_m \equiv 0 \bmod n$$

in unknowns $r_1, \ldots, r_m, \alpha$, where $|r_i| < R$.

Construct the lattice basis (rows)

$$M = \begin{bmatrix} n & & & & & \\ & n & & & & \\ & & \ddots & & & \\ & & & n & & \\ t_1 & t_2 & \ldots & t_m & R/n & \\ a_1 & a_2 & \ldots & a_m & & R \end{bmatrix}$$

$v_r = (r_1, r_2, \ldots, r_m, R\alpha/n, R)$ is a short vector in this lattice.

Solving HNP with lattices

Construct the lattice

$$M = \begin{bmatrix} n & & & & & \\ & n & & & & \\ & & \ddots & & & \\ & & & n & & \\ t_1 & t_2 & \dots & t_m & R/n & \\ a_1 & a_2 & \dots & a_m & & R \end{bmatrix}$$

Want vector
$v_r = (r_1, r_2, \dots, r_m, R\alpha/n, R)$

We expect to find vector with a SVP solver when it is the shortest vector.

- $\det \Lambda = R^2 n^{m-1}$
- $|v_r| \leq \sqrt{m+2}B$.

  Gaussian Heuristic for length of shortest vector
- $\mathrm{GH}(\Lambda) \approx \sqrt{\frac{m+2}{2\pi e}} \det \Lambda^{1/\dim \Lambda}$

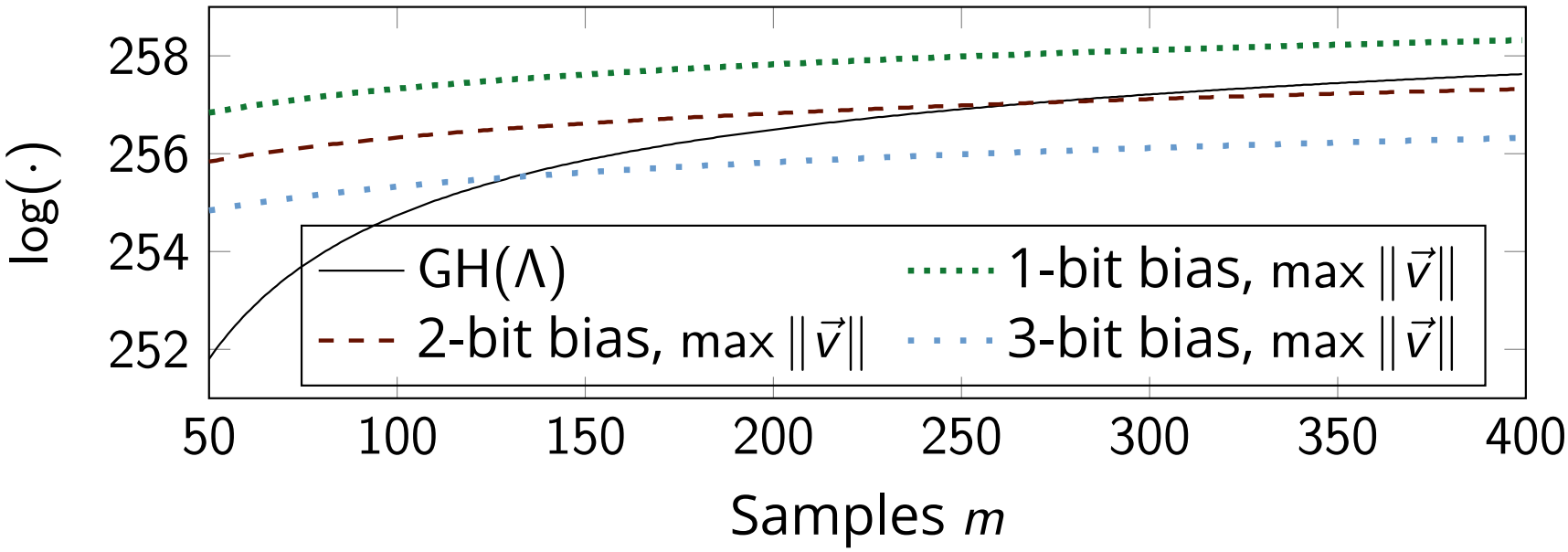
The “lattice barrier”

> *“[T]here is a hard limit to what can be achieved using lattice reduction: due to the underlying structure of the HNP lattice, it is impossible to attack (EC)DSA using a single-bit nonce leak with lattice reduction. In that case, the ’hidden lattice point’ corresponding to the HNP solution will not be the closest vector even under the Gaussian heuristic (see [NT12]), so that lattice techniques cannot work.” [AFGKTZ14]*

Similar points are made in [DHMP13,M17,TTA18,ANTTY20]

The “lattice barrier”

Compare the upper bound for $\|\vec{v}\|$ for $\log(n) = 256$ as we vary the number of samples m.



Crossover points estimate required number of samples (and lattice dimension to solve SVP in).

Our work: Breaking the lattice barrier

Three observations:

1. In practical applications, there is still a unique solution. (e.g. the attacker has a known public key and can compare against the target vector)

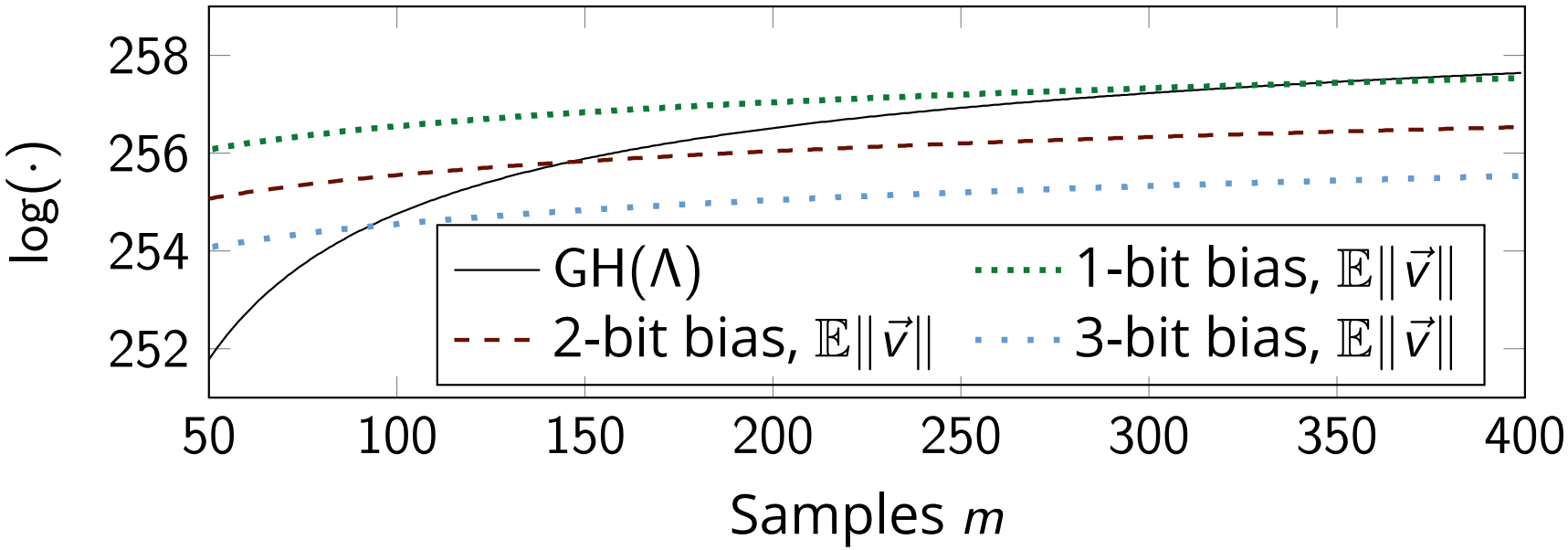
   Thus even if the target vector isn't the closest vector, we can use this extra information to search for solutions.

2. Typical lattice behavior follows expected vector length, not upper bound. ($\mathrm{E}(v) \approx \sqrt{m/3}B \leq \sqrt{m}B$)

3. To get the above to work, we must apply optimizations that are inconsistently applied in practice. (Recentering, variable eliminiation, etc.)

Lattice barrier behavior with expected vector length

Lattice dimensions become much more tractable.



Later: We experimentally confirm the analysis.

Bounded Distance Decoding with Predicate

Definition (Bounded Distance Decoding with predicate)

Given a lattice basis $\vec{B}$, a vector $\vec{t}$, a predicate $f(\cdot)$, and a parameter $0 < \alpha$ such that the Euclidean distance $\text{dist}(\vec{t}, \vec{B}) < \alpha \cdot \lambda_1(\vec{B})$, find the lattice vector $\vec{v} \in \Lambda(\vec{B})$ satisfying $f(\vec{v} - \vec{t}) = 1$ which is closest to $\vec{t}$.

Definition (unique SVP with predicate)

Given a lattice Λ and a predicate $f(\cdot)$ find the shortest nonzero vector $\vec{v} \in \Lambda$ satisfying $f(\vec{v}) = 1$.

Using Kannan's embedding and a simple transform on the predicate we can solve the former using the latter.

Concretely: ECDSA predicate computes a curve scalar multiplication on a candidate nonce and compares to signature

BKZ with Predicate

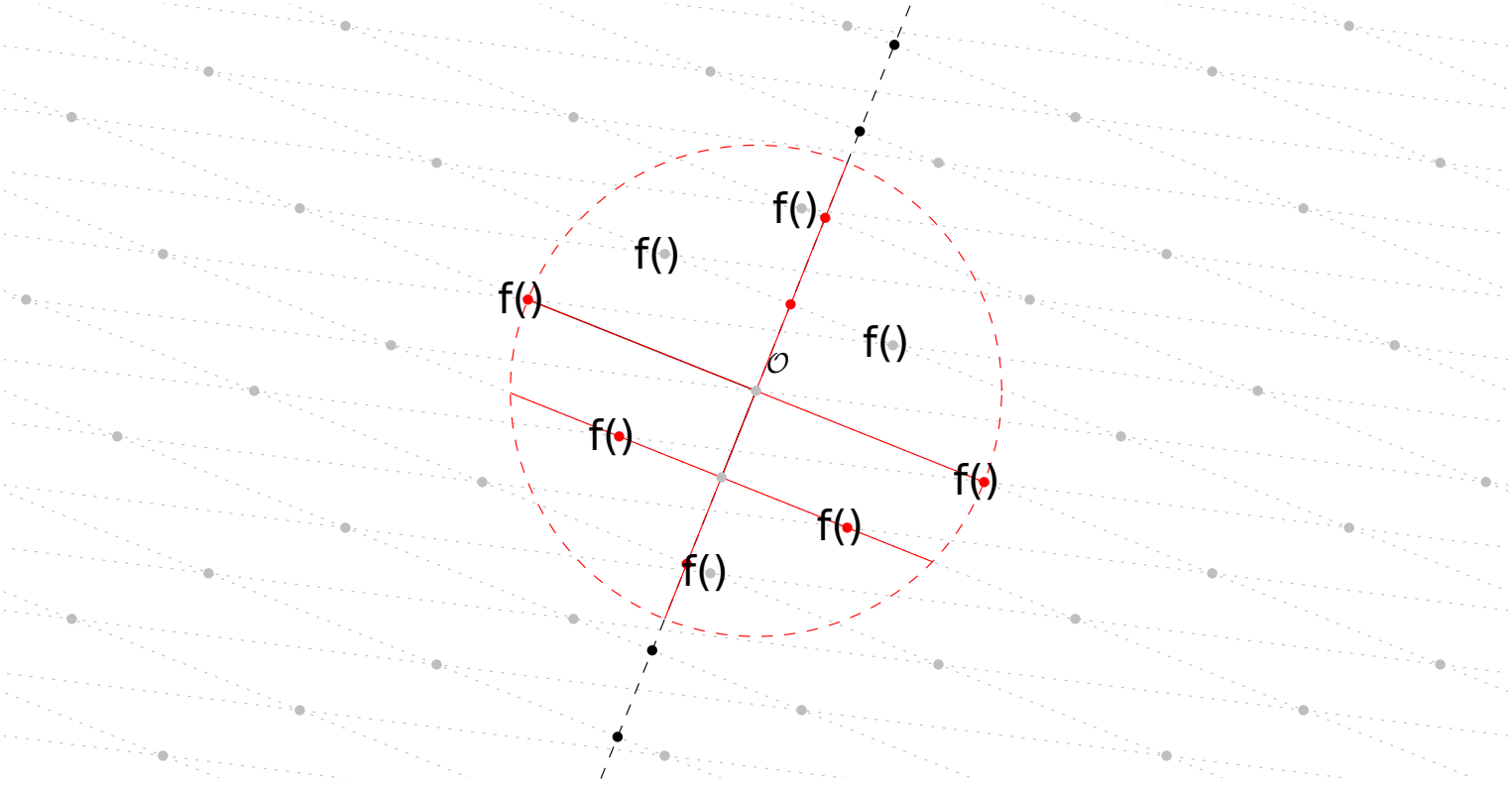
Baseline algorithm: It is folklore in the literature to use BKZ and search through the reduced basis for the presence of the target vector.

- When $\vec{v}$ is expected to be shorter than any other vector in Λ we call BKZ algorithm with the appropriate block size β.
- When $\beta = d$ this computes an HKZ reduced basis and thus a shortest vector in the basis.
- We will consider these algorithms to have succeeded if the target is contained in the reduced basis.

Enumeration with Predicate

Enumeration algorithms can exhaustively search within a given radius.

We augment with predicate to search for target.



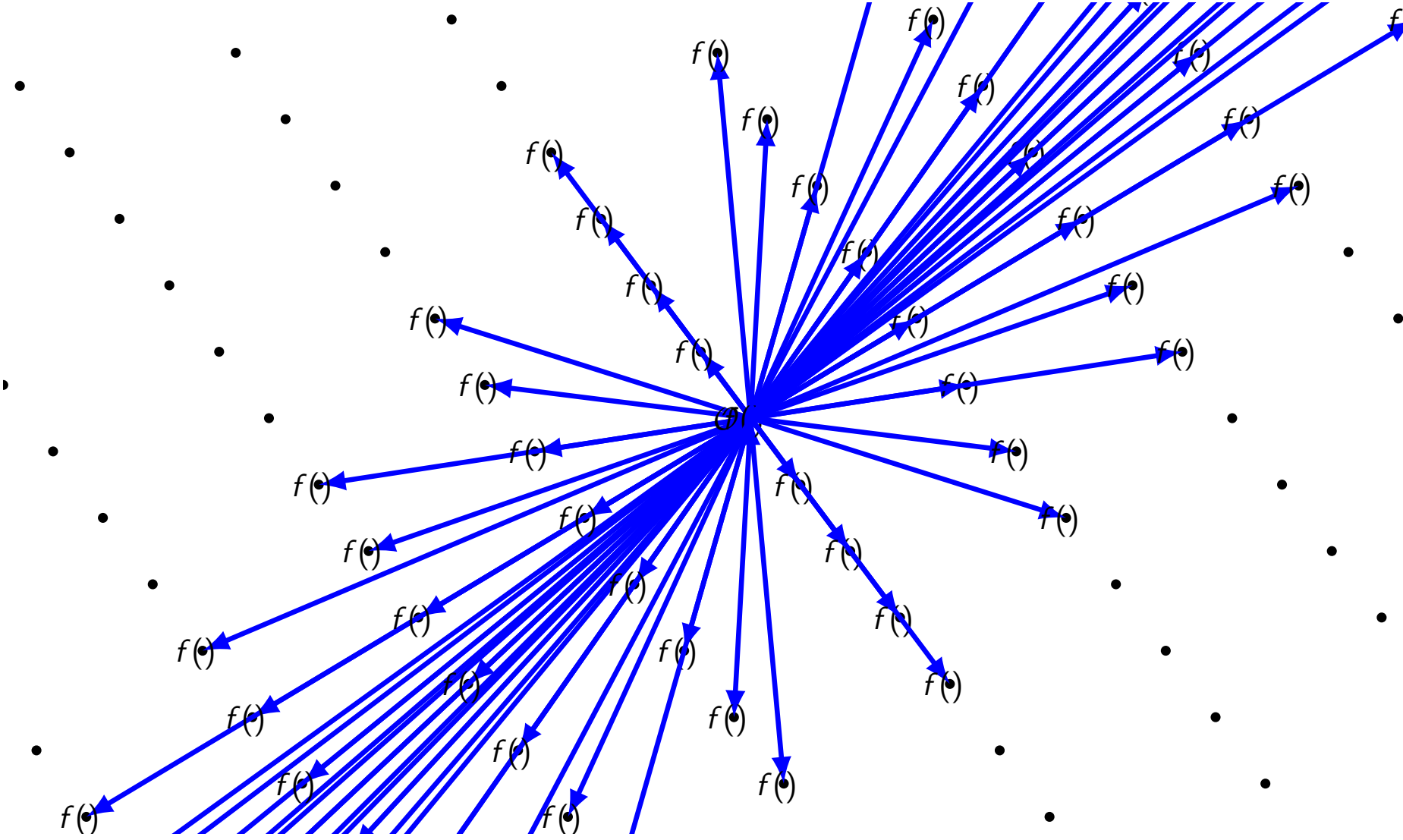
Enumeration with Predicate

Theorem

Let $\Lambda \subset \mathbb{R}^d$ be a lattice containing vectors $\vec{v}$ such that $\|\vec{v}\| \leq R = \xi \cdot \mathsf{GH}(\Lambda)$ and $f(\vec{v}) = 1$. Assuming the Gaussian heuristic, then enumeration with predicate finds the shortest vector $\vec{v}$ satisfying $f(\vec{v}) = 1$ in $\xi^d \cdot d^{d/(2e)+o(d)}$ steps. Enumeration with predicate will make $\xi^{d+o(d)}$ calls to $f(\cdot)$.

Sieving with Predicate

We can similarly augment sieving algorithms with a predicate.



Sieving with Predicate II

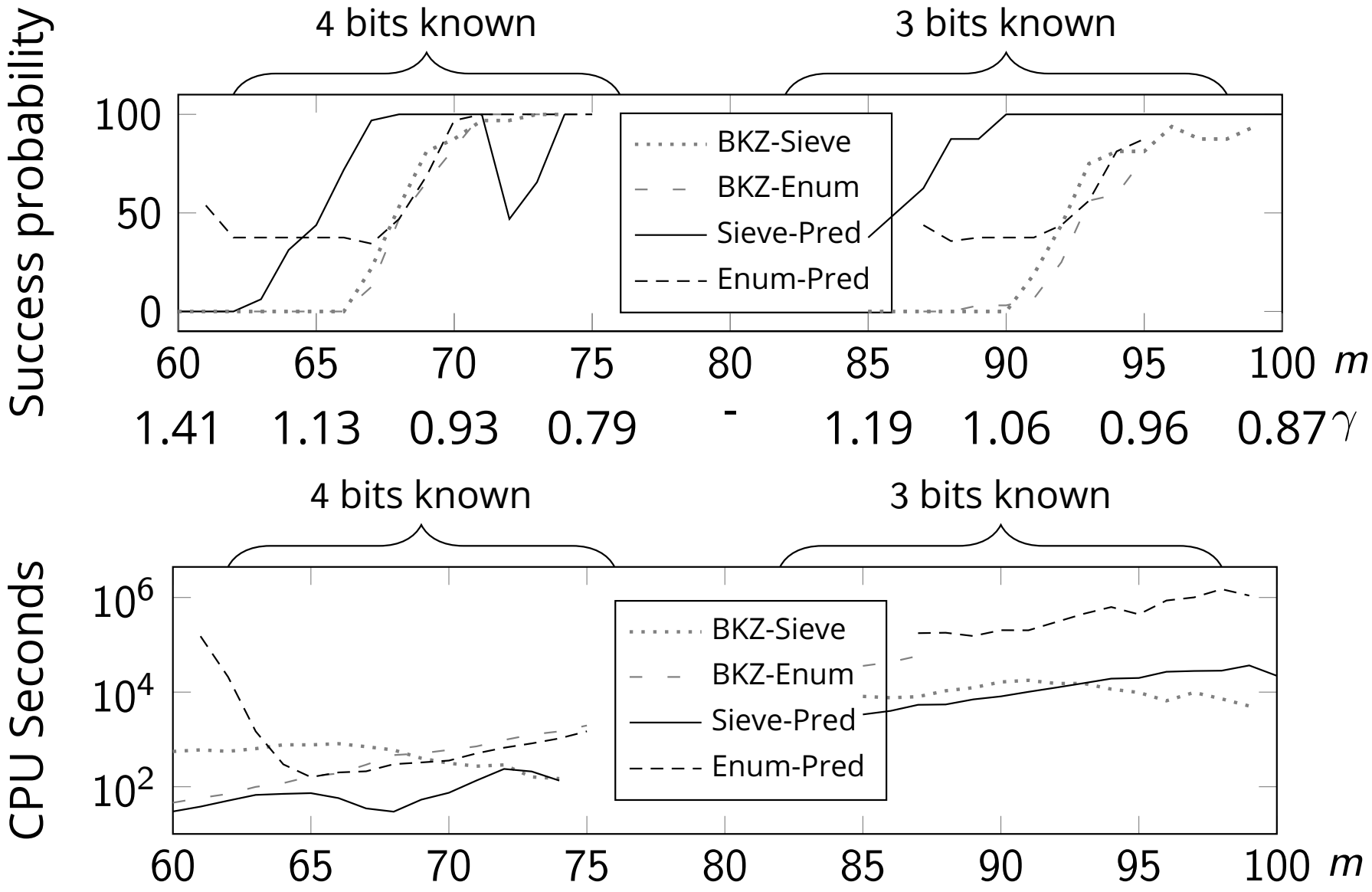
Assumption

When a 2-sieve algorithm terminates, it outputs a database L containing all vectors with norm $\leq \sqrt{4/3} \cdot \mathrm{GH}(\Lambda)$.

Theorem

Let $\Lambda \subset \mathbb{R}^d$ be a lattice containing a vector $\vec{v}$ such that $\|\vec{v}\| \leq R = \sqrt{4/3} \cdot \mathrm{GH}(\Lambda)$. Under our assumption sieving with predicate is expected to find the minimal $\vec{v}$ satisfying $f(\vec{v}) = 1$ in $2^{0.292\,d+o(d)}$ steps and $(4/3)^{d/2+o(d)}$ calls to $f(\cdot)$.

ECDSA Success Rates / Running Time for $\log(n) = 256$



Comparison to previous work for $\log(n) = 256$

$\log(n)$	bias	m	time	alg.	s/r	previous work
256	4 bits	63	2122s	E	41%	below information-theoretic barrier
256	4 bits	65	76s	S	66%	BKZ-25, $m \approx 82$, $s/r = 90\%$ in [Ryan18]
256	3.6 bits	73	69s	S	66%	BKZ-30, $m = 80$, $s/r = 94.5\%$ in [GB17]
256	3 bits	87	5400s	S	63%	enum, $m = 100$, $s/r = 21\%$ in [LCLi14]
256	2 bits	–	–	–	–	Bleichenbacher, $m \approx 2^{26}$, in [TTA18]

We can solve HNP instances with fewer samples than reported in the literature.

Practical impact: In side-channel attacks, sample collection can be expensive—often thousands of measurements for a single usable sample.

Cost estimates for $\log(n) = 256$

bits known	8	7	6	5	4	3	2	1
Sieve m/d	33/20	38/26	45/33	54/42	69/56	93/79	146/128	341/310
Sieve-Pred m/d	33/34	37/38	43/44	52/53	65/66	87/88	131/132	267/268
Sieve-Pred cost	34.9	34.1	33.6	33.9	35.7	41.5	57.6	108.6
limit m	32	37	43	52	64	86	128	256
limit -1 cost	27.2	27.4	29.8	32.3	38.7	48.2	73.7	169.7

Sieve #samples m required for solving uSVP and sieve dim.

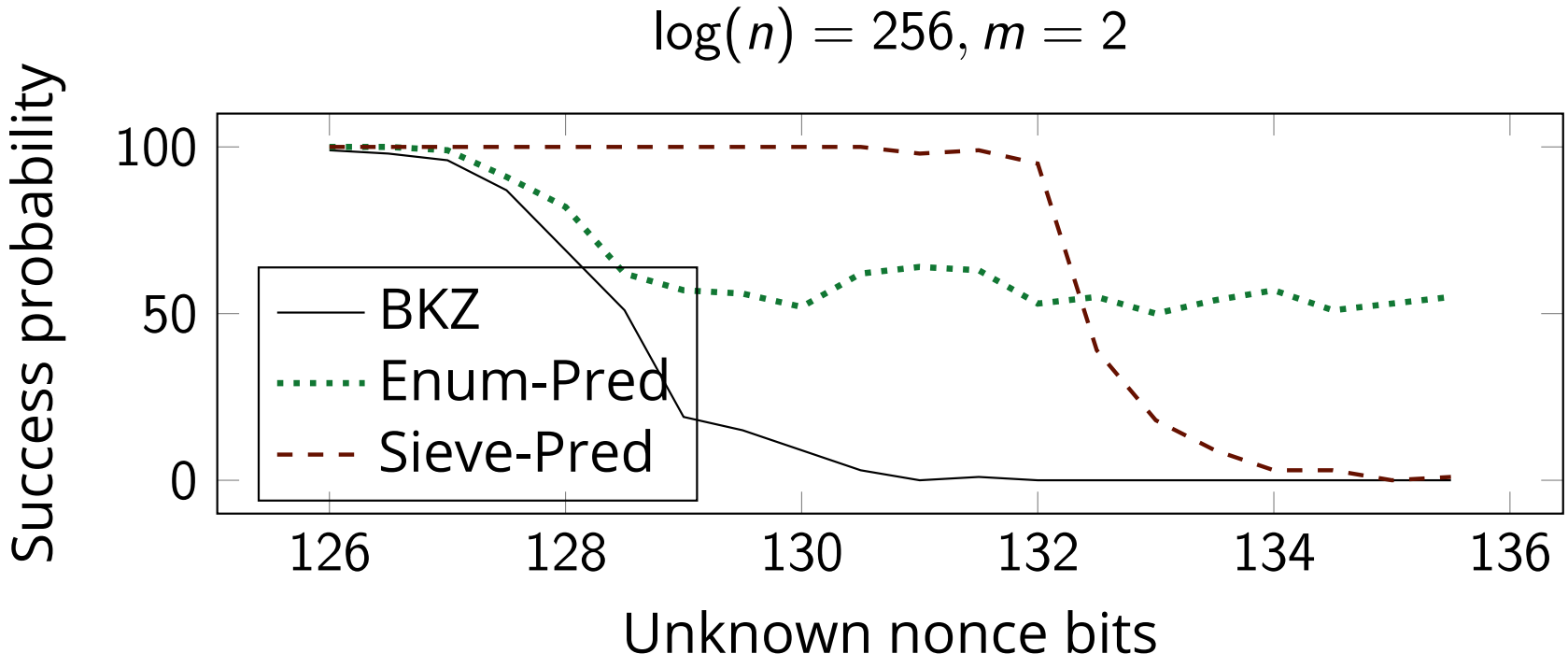
Sieve-Pred #samples m required for and sieving dimension $d = m + 1$.

Sieve-Pred cost Log of expected cost in CPU cycles

limit Information theoretic limit for m of pure lattice approach: $\lceil \log(n)/\text{bits known} \rceil$.

limit -1 cost Log of expected cost for in CPU cycles with $m = \lceil \log(n)/\text{bits known} \rceil - 1$ samples.

Earning some Bitcoin



BH19 reported using BKZ in small dimensions to find 287 Bitcoin signing keys.

- We applied sieving with predicate and were able to compute 9 more signing keys.

Additional Benefit: Handling Errors

In side-channel applications, measurement errors are common.

Prior works state that lattice algorithms do not deal well with noisy data.

Natural approach to errors given our results:

1. Estimate error rate and use it to estimate length of target vector.
2. Use estimated length of target vector to choose block size, enumeration, or sieving parameters accordingly.
3. This works well in experiments.

Conclusions

- You can break the “lattice barrier” if you are willing to spend more computational time.
- All our code available at `https://github.com/malb/bdd-predicate/`
- Future work: 2-bit bias on a 256-bit curve with lattices.