Playing “Hide-and-Seek” in Finite Fields: Hidden Number Problem and Its Applications

Igor E. Shparlinski

Macquarie University

Introduction

We describe a rather surprising, yet powerful, combination of

- **exponential sums**

- **lattice basis reduction algorithms**.

This combination has led to a number of cryptographic applications, helping to make rigorous several heuristic approaches.

It provides a two edge sword to:

- prove important **security** results;

- create powerful **attacks**

Examples:

- Bit security of the
  - Diffie–Hellman key exchange system,
  - Shamir message passing scheme,
  - XTR cryptosystem,
  - Rivest–Shamir–Wagner timed-release crypto.
- Attacks on the
  - Digital Signature Scheme (DSA),
  - Nyberg–Rueppel Signature Scheme.

Notation

p = prime number

$\mathbb{F}_p$ = finite field of p elements.

$\lfloor s \rfloor_m$ = the remainder of s on division by m.

For $\ell > 0$, $\mathrm{MSB}_{\ell,p}(x)$ denotes any integer u such that

$$|\lfloor x \rfloor_p - u| \le p/2^{\ell+1}.$$

$\mathrm{MSB}_{\ell,p}(x) \approx \ell$ most significant bits of x.

However this definition is more flexible.

In particular, ℓ **need not be an integer**.

$\ell = 0$ gives no information
$\ell = \lceil \log p / \log 2 \rceil$ identifies $\lfloor x \rfloor_p$ uniquely.

Everything in between is nontrivial.

Hidden Number Problem (HNP)

Boneh & Venkatesan (1996):

HNP: *Recover $\alpha \in \mathbb{F}_p$ such that for many known random $t \in \mathbb{F}_p$ we are given* $\mathrm{MSB}_{\ell,p}(\alpha t)$ *for some $\ell > 0$.*

Boneh & Venkatesan (1996): a polynomial time algorithm to solve **HNP** with $\ell \approx \log^{1/2} p$.

Note: $\ell \approx \log p$ is trivial.

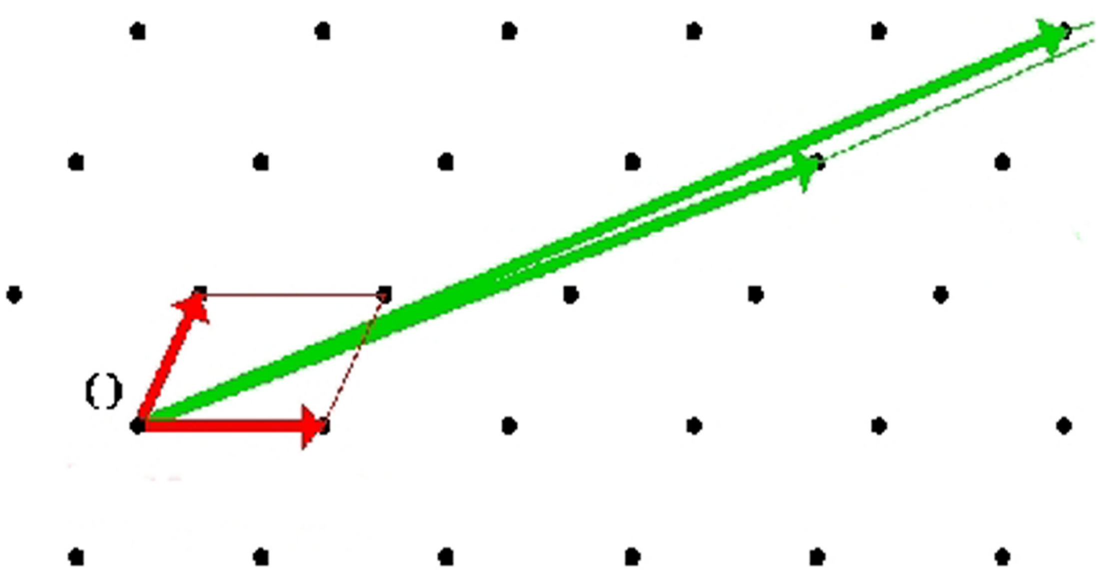
The algorithm is based on the **lattice basis reduction.**

Lattices

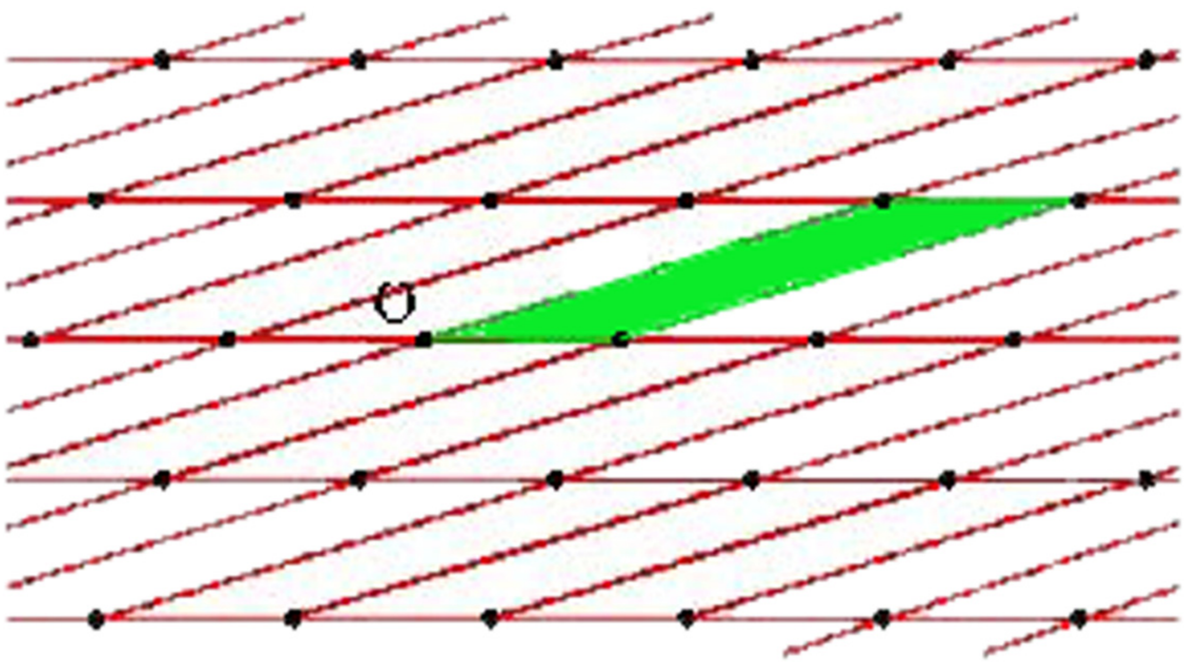
Let $\{\mathbf{b}_1, \ldots, \mathbf{b}_s\}$ be a set of linearly independent vectors in $\mathbb{R}^s$. The set of vectors

$$L = \{\mathbf{z} \mid \mathbf{z} = \sum_{i=1}^{s} c_i \mathbf{b}_i, \quad c_1, \ldots, c_s \in \mathbf{Z}\}$$

is called an s-dimensional full rank lattice. The set $\{\mathbf{b}_1, \ldots, \mathbf{b}_s\}$ is called a *basis* of L.



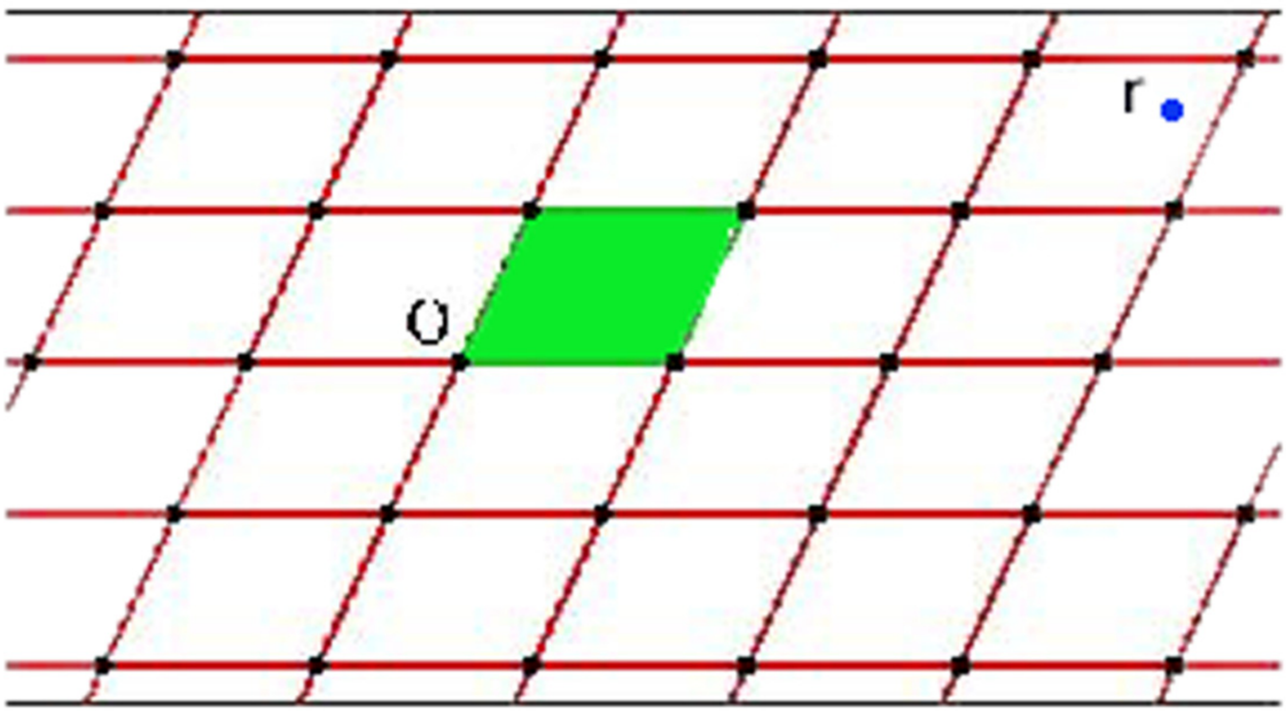
The volume of the parallelogram defined by the basic vectors is the invariant, called the **discriminant**.



The closest vector problem

CVP: Given a vector $\mathbf{r} \in \mathbb{R}^s$ find a lattice vector $\mathbf{v} \in L$ with

$$\|\mathbf{r} - \mathbf{v}\| = \min_{\mathbf{z} \in L} \|\mathbf{r} - \mathbf{z}\|.$$



CVP is **NP**-complete.

Approximate solution?

Lenstra, Lenstra & Lovász (1982)
Kannan (1987)
Schnorr (1987)

Lemma 1 *There exists a deterministic polynomial time algorithm which, for a given lattice L and a vector $\mathbf{r} \in \mathbb{R}^s$, finds a lattice vector $\mathbf{v} \in L$ satisfying the inequality*

$$\|\mathbf{r} - \mathbf{v}\| \le \exp\left(C \frac{s \log^2 \log s}{\log s}\right) \min_{\mathbf{z} \in L} \|\mathbf{r} - \mathbf{z}\|$$

for some absolute constant $C > 0$.

LLL: stretch factor $2^{s/2}$ (can be used as well)

Working with $2^{o(s)}$ is technically easier

HNP and CVP

Boneh & Venkatesan (1996) :

Let $d \geq 1$ be integer. Given t_i, $u_i = \mathrm{MSB}_{\ell,p}(\alpha t_i)$, $i = 1, \ldots, d$, we build the lattice $\mathcal{L}(p, \ell, t_1, \ldots, t_d)$ spanned by the rows of the matrix:

$$\begin{pmatrix} p & 0 & \ldots & 0 & 0 \\ 0 & p & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & 0 & \ldots & p & 0 \\ t_1 & t_2 & \ldots & t_d & 1/2^{\ell+1} \end{pmatrix}.$$

The **unknown** vector $\mathbf{v} = (\lfloor \alpha t_1 \rfloor_p, \ldots, \lfloor \alpha t_d \rfloor_p, \alpha/2^{\ell+1})$

- belongs to $\mathcal{L}(p, \ell, t_1, \ldots, t_d)$;

- is close to the **known** vector $\mathbf{u} = (u_1, \ldots, u_d, 0)$:

$$\|\mathbf{v} - \mathbf{u}\| = O\left(p 2^{-\ell}\right).$$

Idea: Apply a CVP algorithm and *hope* that it will output $\mathbf{v}$.

How to make it rigorous?

We show that for almost all $t_1, \ldots, t_d$, $\mathbf{v}$ is the only lattice vector which can be so close to $\mathbf{u}$.

In fact, even within the approximation factor of Lemma 1, that is within the distance of order $p 2^{-\ell + o(d)}$, this is still the **only** lattice vector.

Analysiz

Note that any vector

$$\mathbf{w} = (w_1, \ldots, w_d, w_{d+1} \in \mathcal{L}(p, \ell, t_1, \ldots, t_d)$$

satisfies

$$(w_1, \ldots, w_d) \equiv (\beta t_1, \ldots, \beta t_d) \pmod p$$

with some integer β

Assume that $\mathbf{w} \in \mathcal{L}(p, \ell, t_1, \ldots, t_d)$, with $\beta \not\equiv \alpha \pmod p$ is another lattice vector with

$$\|\mathbf{w} - \mathbf{u}\| \le p2^{-\ell+o(d)}.$$

Then, by the triangle inequality

$$\|\mathbf{w} - \mathbf{v}\| \le p2^{-\ell+o(d)}. \qquad (1)$$

Therefore for each $i = 1, \ldots, d$

$$(\alpha - \beta)t_i \in [-p2^{-\ell+o(d)}, p2^{-\ell+o(d)}] \pmod p$$

For every fixed $\gamma \not\equiv 0 \pmod p$

$$\Pr_{t \in \mathbb{F}_p} (\gamma t \in [-h, h] \pmod p) \le \frac{2h+1}{p} \qquad (2)$$

Thus

$$\Pr_{t_1,\ldots,t_d \in \mathbb{F}_p} (\gamma t_i \in [-h, h] \pmod p),\ i = 1, \ldots, d)$$
$$\le \left(\frac{2h+1}{p}\right)^d.$$

In our settings

$$\gamma = \alpha - \beta \qquad \text{and} \qquad h = p2^{-\ell+o(d)}.$$

Because β (and thus $\gamma = \alpha - \beta$) may belong to $p-1$ distinct residue classes we conclude that (1) holds with probability at most

$$P \le p\left(2^{-\ell+o(d)}\right)^d.$$

Choose $\ell = d = 2\left\lceil \log^{1/2} p \right\rceil$. Then

$$P \le \frac{1}{p}.$$

CVP algorithm returns $\mathbf{v}$ with prob. $\ge 1 - 1/p$

Extended HNP

HNP: *Recover $\alpha \in \mathbb{F}_p$ such that for many known random $t \in \mathbb{F}_p$ we are given* $\mathsf{MSB}_{\ell,p}(\alpha t)$ *for some $\ell > 0$.*

The condition that t is selected uniformly at random from $\mathbb{F}_p$ is too restrictive for applications.

Typically t is selected from some finite sequence $\mathcal{T}$ of elements of $\mathbb{F}_p$ which:

- may have a nice and well-studied number theoretic structure (bit security of Diffie–Hellman key),

- may be rather "ugly" looking (attacks on DSA).

EHNP: *Recover $\alpha \in \mathbb{F}_p$ such that for many known random $t \in \mathcal{T}$ we are given* $\mathsf{MSB}_{\ell,p}(\alpha t)$ *for some $\ell > 0$.*

The same arguments as above apply to the **EHNP** ... but one needs an analogue of (2).

$$\Downarrow$$

$\mathcal{T}$ must have some **uniformity of distribution** properties.

$$\Downarrow$$

Nontrivial bounds of exponential sums

$$\left|\sum_{t\in\mathcal{T}} \exp\left(2\pi i c t/p\right)\right| \le \delta\#\mathcal{T}, \quad \gcd(c,p)=1, \quad (3)$$

with some nontrivial saving $\delta < 1$.

We say that $\mathcal{T}$ is δ-good is (3) holds.

Koksma (1950) and *Szüsz (1950)* independently

$$\Downarrow$$

For a δ-good sequence $\mathcal{T}$ instead of (2) we get

$$\Pr_{t\in\mathcal{T}}\left(\gamma t \in [-h,h] \pmod p\right) \le \frac{2h+1}{p} + O\left(\delta \log(\delta^{-1})\right)$$

Putting Together

Nguyen & Shparlinski (2000):

Theorem 2 *Let* $\ell = \lceil \log^{1/2} p \rceil + \lceil \log\log p \rceil$ *and* $d = 2\left\lceil \log^{1/2} p \right\rceil$. *Let* $\mathcal{T}$ *be* $2^{-\log^{1/2} p}$*-good. There exists a deterministic polynomial time algorithm* $\mathcal{A}$ *such that for any fixed integer* $\alpha \in [0, p-1]$, *given* $2d$ *integers*

$$t_i \qquad \text{and} \qquad u_i = \mathrm{MSB}_{\ell,p}(\alpha t_i), \qquad i = 1, \ldots, d,$$

its output satisfies

$$\Pr_{t_1,\ldots,t_d \in \mathcal{T}} [\mathcal{A}(t_1, \ldots, t_d; u_1, \ldots, u_d) = \alpha] \ge 1 - 2^{-(\log p)^{1/2} \log\log p}$$

if $t_1, \ldots, t_d$ *are chosen uniformly and independently at random from the elements of* $\mathcal{T}$.

Using Very Weak Bounds

Usually we prove that $\mathcal{T}$ if δ-good with $\delta \sim \#\mathcal{T}^{-\alpha}$ for some fixed $\alpha > 0$ or nothing at all. However in some important cases (e.g. $\mathcal{T}$ = a small subgroup of $\mathbb{F}_p^*$) only very weak bounds are know with δ very close to 1.

Shparlinski & Winterhof (2003):
Modifications to the Algorithm

Choose

$$t_{11}, \ldots, t_{1k}, \ldots, t_{d1}, \ldots, t_{dk} \in \mathcal{G}$$

and get integers u_{ij} with

$$\left| \left\lfloor \alpha t_{ij} \right\rfloor_p - u_{ij} \right| < p/2^{\ell+1}, \quad i = 1, \ldots, d, \ j = 1, \ldots, k.$$

For $i = 1, 2, \ldots, d$ we put

$$v_i = \sum_{j=1}^{k} \left\lfloor \alpha t_{ij} \right\rfloor_p, \quad t_i = \left\lfloor \sum_{j=1}^{k} t_{ij} \right\rfloor_p, \quad u_i = \sum_{j=1}^{k} u_{ij}$$

The rest of the algorithm remains the same.

We work with k-fold Cartesian product $\mathcal{T}^k$ of $\mathcal{T}$. So we have

$$\left|\sum_{t\in\mathcal{T}} \exp\left(2\pi ict/p\right)\right| \qquad \text{vs.} \qquad \left|\sum_{t\in\mathcal{T}} \exp\left(2\pi ict/p\right)\right|^k$$

If

$$\left|\sum_{t\in\mathcal{T}} \exp\left(2\pi ict/p\right)\right| \le \delta \#\mathcal{T}$$

then

$$\left|\sum_{t\in\mathcal{T}} \exp\left(2\pi ict/p\right)\right|^k \le \delta^k \left(\#\mathcal{T}\right)^k = \delta^k \#\mathcal{T}^k$$

If $\mathcal{T}$ if δ-good (but δ is close to 1) then $\mathcal{T}^k$ if δ^k-good and adjusting k one can make it work.

Good News: Bit Security of the Diffie–Hellman Key

Diffie–Hellman (DH) problem:

Given an element g of order τ modulo p, recover $K = \lfloor g^{xy} \rfloor_p$ from $\lfloor g^x \rfloor_p$ and $\lfloor g^y \rfloor_p$.

Typically, either $\tau = p - 1$ or $\tau = q$ – a large prime divisor of $p - 1$

The size of p and τ is determined by the present state of art in the **discrete logarithm problem**. Typically, p is about 500 bits, τ is at least 160 bits.

However after the common DH key $K = g^{xy}$ is established, only a small portion of bits of K will be used as a common key for some **private** key cryptosystem.

Private Key |	Public Key

Question: Assume that finding K is infeasible. Is it still infeasible to find certain bits of K?

Boneh & Venkatesan (1996) :
for $\tau = p - 1$ (- small gap in the proof)

González Vasco & Shparlinski (2000) :
for "any" τ (+ fixing the gap in BV)

YES!!!

Assume we know how to recover ℓ most significant bits of $\lfloor g^{xy} \rfloor_p$ from from $X = \lfloor g^x \rfloor_p$ and $Y = \lfloor g^y \rfloor_p$.

Select a random $u \in [0, \tau - 1]$ and apply this algorithm to $X = \lfloor g^x \rfloor_p$ and $U = \lfloor Y g^u \rfloor_p = \left\lfloor g^{y+u} \right\rfloor_p$:

$$\mathsf{MSB}_{\ell,p}\left(g^{x(y+u)}\right) = \mathsf{MSB}_{\ell,p}\left(g^{xy} g^{xu}\right) = \mathsf{MSB}_{\ell,p}(\alpha t)$$

EHNP with $\alpha = g^{xy}$ and $t = g^{xu}$, $u \in [0, \tau - 1]$!!!

When is γ^u $2^{-\log^{1/2} p}$-good? $\quad (\gamma = g^x)$

Shparlinski & Winterhof (2003):

Theorem 3 *For any $\varepsilon > 0$ there exists $c > 0$ such that for $k = c \log^2 p$ any $\gamma \in \mathbb{F}_p$ of order $\tau \geq (\log p)^{1+\varepsilon}$ the sequence*

$$\mathcal{T}_k = \{\gamma^{u_1} + \ldots + \gamma^{u_k},\ u_1, \ldots, u_k = 0, \ldots, \tau - 1\}$$

is $p^{-\delta}$-good.

If p is an n-bit prime and $\tau \geq (\log p)^{1+\varepsilon}$ then $\approx n^{1/2}$ most significant bits of the DH key are as secure as the whole key.

Bad News: Attack on DSA

DSA: Proposed NIST, August 1991; US Federal Information Processing Standard 186, May 1994

Public Data:

q and p = primes with $q|p-1$
$g \in \mathbb{F}_p$ = a fixed element of order q.
$\mathcal{M}$ = set of messages to be signed
$h : \mathcal{M} \to \mathbb{F}_q$ = a hash-function.

The **secret key** is $\alpha \in \mathbb{F}_q^*$ which is known only to the **signer** (and publishes $A = \lfloor g^\alpha \rfloor_p$ – to be used for signature verification).

To sign a message $\mu \in \mathcal{M}$, the signer chooses a random integer $k \in \mathbb{F}_q^*$ usually called the *nonce*, and which must be kept **secret** and computes:

$$r(k) = \left\lfloor \left\lfloor g^k \right\rfloor_p \right\rfloor_q, \quad s(k,\mu) = \left\lfloor k^{-1}\left(h(\mu) + \alpha r(k)\right) \right\rfloor_q$$

$(r(k), s(k,\mu))$ is the *DSA signature* of the message μ with a nonce k.

Assume that some bits of k are "leaked"

Howgrave-Graham & Smart (1998)
Heuristic lattice based attack.

Nguyen (1999):
Simpler and more powerful but still **heuristic** lattice based attack.

Nguyen & Shparlinski (1999):
Rigorous lattice based attack.

Idea *Nguyen (1999)*:

$$s(k,\mu) \equiv k^{-1}(h(\mu) + \alpha r(k)) \pmod q$$

$$\Downarrow$$

$$\alpha\, r(k)s(k,\mu)^{-1} \equiv k - h(\mu)s(k,\mu)^{-1} \pmod q.$$

If ℓ most significant bits of k are known then we know $\mathrm{MSB}_{\ell,q}\left(\alpha r(k)s(k,\mu)^{-1}\right)$.

EHNP with

$$t(k,\mu) = \left\lfloor r(k)s(k,\mu)^{-1}\right\rfloor_q, \quad (k,\mu) \in [1, q-1] \times \mathcal{M}.$$

Nguyen & Shparlinski (1999) + Recent bounds of *Bourgain, Glibichuk & Konyagin (2004)*:
Let

$$W = \#\{h(\mu_1) = h(\mu_2), \quad \mu_1, \mu_2 \in \mathcal{M}\}$$

$$\boxed{W/\#\mathcal{M}^2 = \text{probability of } \textit{collision}}$$

Typically

$$W/|\mathcal{M}|^2 \approx q^{-1}.$$

Theorem 4 *For any $\varepsilon > 0$ there exists $\delta > 0$ such that for any $g \in \mathbb{F}_p$ of order $q \ge p^\varepsilon$ the sequence*

$$t(k,\mu) = \left\lfloor r(k)s(k,\mu)^{-1} \right\rfloor_q, \quad (k,\mu) \in [1, q-1] \times \mathcal{M}.$$

is $q^{-\delta}$-good, provided

$$W \le \frac{\#\mathcal{M}^2}{q^{1-\delta}}.$$

We need to estimnate double exponential sums

$$\sum_{k \in \mathbb{F}_q} \sum_{\mu \in \mathcal{M}} \exp\left(2\pi i c r(k) s(k,\mu)^{-1}/q\right),$$

with $\gcd(c, q) = 1$.

The proof uses:

- bounds of exponential sums with exponential functions: *Konyagin & Shparlinski (1999)* in the original work, nowaday one should use *Bourgain, Glibichuk & Konyagin (2004)*;

- **Weil's** bound;

- **Vinogradov's** method of estimates of double sums.

Main difficulty: The double modular reduction modulo p then modulo q destroys any number theoretic structure among the values of $r(k)$.

Theoretically: If q is an n-bit prime and $\approx n^{1/2}$ most significant bits of k are known for $\approx n^{1/2}$ signatures then α can be recovered in polynomial time.

Practically (dates back to 2000): 4 bits of k are always enough, 3 bits are often enough, 2 bits are possibly enough as well.

Moral:

1. Do not use **small** k (to cut the cost of exponentiation in $r(k)$).

2. Protect your software/hardware against **timing/power attacks** when the attacker measures the time/power consumption and selects the signatures for which this value is smaller than "on average" – these signatures are likely to correspond to small k ($\sim$ faster exponentiation in $r(k)$).

3. Use quality **PRNG**'s to generate k, biased generators are dangerous.

4. Do not use **Arazi's cryptosystem** which combines DSA and Diffie-Hellman protocol – it leakes some bits of k (*Brown & Menezes*).

5. Do not buy CryptoLib from **AT&T**, it always uses odd values of k thus one bit is leaked immediately, one more and . . . .

Nonlinear Variants

Shparlinski, 2001

HNP with sparse polynomials: "Noisy Interpolation"

Recover the coefficients of a sparse polynomial

$$f(X) = \sum_{j=1}^{m} \alpha_j X^{e_j} \in \mathbb{F}_p[X]$$

with *known* exponents e_j given $\mathrm{MSB}_{\ell,p}(f(t))$ for many known random $t \in \mathbb{F}_p$.

Shparlinski & Winterhof, 2003:

Under some natural (and very wide) conditions on e_j, including the dense case $e_j = j$, results of the same level as for $m = 1$, $e_1 = 1$:

About $m \log^{1/2} p$ queries with $\ell \sim \log^{1/2} p$

Howgrave-Graham, Nguyen & Shparlinski, 2000

HNP with approximations to the "test" points t, i.e. We are given

$$\mathrm{MSB}_{\ell,p}(\alpha t) \qquad \text{and} \qquad \mathrm{MSB}_{\ell,p}(t).$$

Results are naturally weaker.

Applications to

- bit security of the "timed-release crypto", *Rivest, Shamir & Wagner (1996)*
- "correcting" noisy exponentiation black-boxes
- "correcting" noisy Weil pairing on elliptic curves

There are many lose ends which have never been exploited:
E.g. *polynomial interpolation* with noisy both values and arguments.

Boneh, Halevi & Howgrave-Graham (2001):
HNP with inversions:

Recover the hidden shift α given

$$\mathsf{MSB}_{\ell,p}\left(\frac{1}{t+\alpha}\right)$$

for many known random $t \in \mathbb{F}_p$.

Boneh, Halevi & Howgrave-Graham (2001):
A heuristic algorithms with

$$\ell \sim \frac{2}{3}\log p$$

and, using Coppersmith's trick with considering higher powers and this congruences modulo p^k with some $k \geq 1$, a heuristic algorithms with

$$\ell \sim \frac{1}{3}\log p$$

Applications to MAC's (Message Authemtication Codes) and PRNG (Pseudorandom Number Generators).

Ling, Shparlinski, Steinfeld & Wang (2010):
A rigorous algorithms with

$$\ell \sim \frac{2}{3}\log p$$

Recent Developments

- *Akavia (2009)* :
  New approach to HNP via Fourier coefficients of $t \mapsto \text{MSB}_{\ell,p}(\alpha t)$. May even work for any $\ell > 0$? Has to be understood better. . . .

  It may also work when if we are given $\text{MSB}_{\ell,p}(\alpha t)$ with propobaility $1 - \rho$ for some small (???) ρ and a random integer otherwise.

• *Lyubashevsky, Peikert & Regev (2010)*: LWE, Learning With Errors

Find $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbb{F}_p^m$ given

$$\mathrm{MSB}_{\ell,p}(\langle \alpha \cdot \mathbf{t} \rangle)$$

for many known random $\mathbf{t} \in \mathbb{F}_p^m$.

If m is fixed (or grows slowly with p) the HNP technique applies and seems to lead (to be checked!) to an algorithm that uses:

about $m \log^{1/2} p$ queries with $\ell \sim \log^{1/2} p$

Lyubashevsky, Peikert & Regev (2010): Hardness results in the case of growing m?

What is in between?

Open Problems

- HNP with rational functions?

  Recover the coefficients of a rational function $f(X) \in \mathbb{F}_p(X)$ given $\mathsf{MSB}_{\ell,p}(f(t))$ for many known random $t \in \mathbb{F}_p$.

  HNP with polynomials + HNP with inversions:

- HNP with unknown modulus?

  All know algorithms build a lattice which depends on the modulus p. Once p is unknown **exactly**, the lattice is **wrong** and everything falls apart.

- HNP on elliptic curves?

  Recover $P \in E(\mathbb{F}_p)$ given $\mathsf{MSB}_{\ell,p}(x(tP))$?

  Some related results by:
  Boneh & Shparlinski (2003) :
  Jao, Jetchev & Venkatesan (2009)