Variations on the Knapsack Generator

Florette Martinez

ENS-PSL

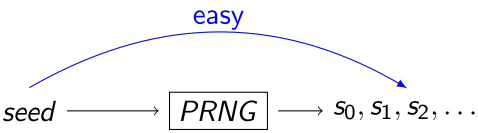
March 11th

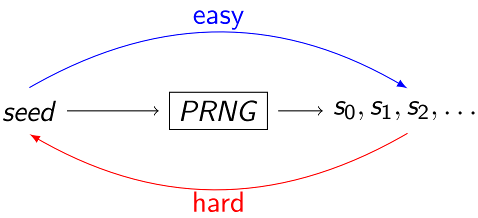
ENS

PRNG

$$seed \longrightarrow \boxed{PRNG} \longrightarrow s_0, s_1, s_2, \ldots$$

PRNG




easy
seed
PRNG
s0, s1, s2, . . .
hard

1. Definition of the Knapsack Generator
2. Attacks on the Knapsack Generator
3. Generalized Knapsack Generator

Table of Contents

1 Definition of the Knapsack Generator

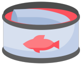
2 Attacks on the Knapsack Generator

3 Generalized Knapsack Generator

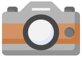
Knapsack Problem

Optimization Problem

$\leq C$



$\omega_1, \; p_1$



$\omega_2, \; p_2$

$\omega_3, \; p_3$

$\omega_4, \; p_4$

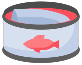
---

images made by surang from www.flaticon.com

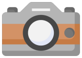
Knapsack Problem

Optimization Problem

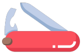
$\leq C$



$\omega_1, \; p_1$



$\omega_2, \; p_2$



$\omega_3, \; p_3$

$\omega_4, \; p_4$

Goal: Finding bits u_i

$$\sum_{i=1}^{4} u_i \omega_i \leq C \text{ and } \sum_{i=1}^{4} u_i p_i \text{ maximal}$$

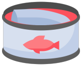
---

images made by surang from www.flaticon.com

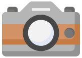
Subset Sum Problem (SSP)

Guessing Problem

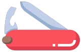
$= C$



ω_1



ω_2



ω_3

ω_4

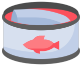
---

images made by surang from www.flaticon.com

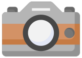
Subset Sum Problem (SSP)

Guessing Problem

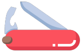
$= C$



ω_1



ω_2



ω_3

ω_4

Goal: Finding bits u_i

$$\sum_{i=1}^{4} u_i \omega_i = C$$

---

images made by surang from www.flaticon.com

Parameters:

- an integer n
- a vector of weights $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_{n-1})$
- a target C
- a modulo M

The goal is finding $\mathbf{u}$ such that

$$\langle \mathbf{u}, \boldsymbol{\omega} \rangle = C \bmod M$$

Parameters:

- an integer n
- a vector of weights $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_{n-1})$
- a target C
- a modulo M

The goal is finding $\mathbf{u}$ such that

$$\langle \mathbf{u}, \boldsymbol{\omega} \rangle = C \bmod M$$

The closer M is to 2^n, the harder the problem is. For now $M = 2^n$

Knapsack Generator by Rueppel and Massey[1]

$$seed \longrightarrow \boxed{PRNG} \longrightarrow s_0, s_1, s_2, \ldots$$

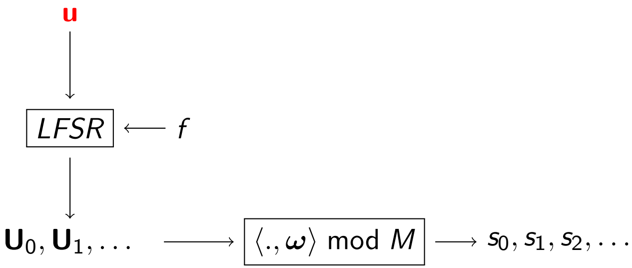
Knapsack Generator by Rueppel and Massey[1]

$$\mathbf{u} \longrightarrow \boxed{\langle ., \boldsymbol{\omega} \rangle \text{ mod } M} \longrightarrow s_0, s_1, s_2, \ldots$$

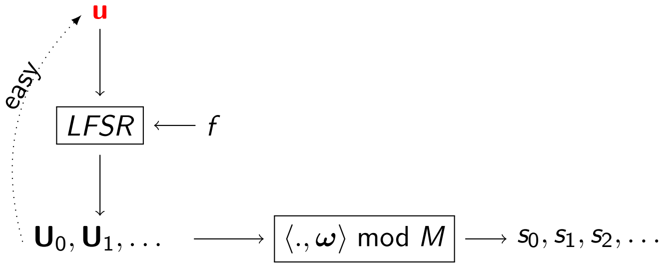
Knapsack Generator by Rueppel and Massey[1]

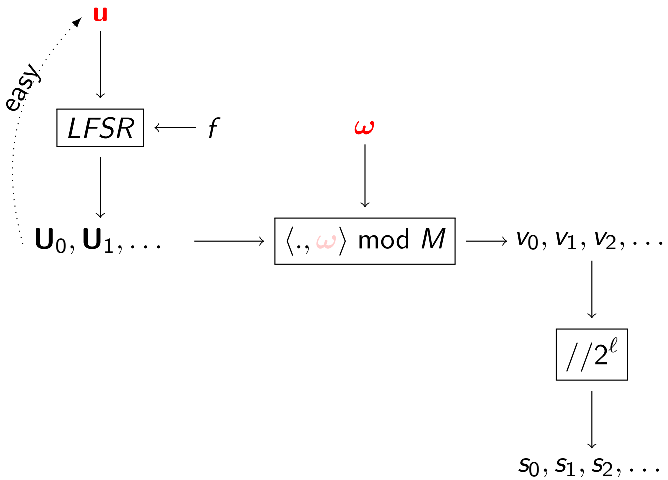
$$\mathbf{u} \longrightarrow \boxed{\langle ., \omega \rangle \bmod M} \longrightarrow s_0, \cancel{s_1, s_2, \ldots}$$

Knapsack Generator by Rueppel and Massey[1]



Knapsack Generator by Rueppel and Massey[1]





[1]Rueppel, R.A., Massey, J.L.: Knapsack as a nonlinear function. In: IEEE Intern. Symp. of Inform. Theory, vol. 46 (1985)

Formalization of the Knapsack Generator

Public	Secret
n and $\ell \in \mathbb{N}$	$\mathbf{u} \in \{0,1\}^n$
$f \in \mathbb{F}_2[X_1, \ldots, X_n]$	$\boldsymbol{\omega} \in \{0, \ldots, 2^n - 1\}^n$

Public	Secret
n and $\ell \in \mathbb{N}$	$\mathbf{u} \in \{0,1\}^n$
$f \in \mathbb{F}_2[X_1, \ldots, X_n]$	$\boldsymbol{\omega} \in \{0, \ldots, 2^n - 1\}^n$

m is the number of outputs

Public	Secret
n and $\ell \in \mathbb{N}$ $f \in \mathbb{F}_2[X_1, \ldots, X_n]$	$\mathbf{u} \in \{0,1\}^n$ $\boldsymbol{\omega} \in \{0, \ldots, 2^n - 1\}^n$

m is the number of outputs

Intermediate states	
$(u_i)_{i \geq n}$	$u_{n+i} = f(u_i, \ldots, u_{n+i-1})$
$(\mathbf{U}_i)_{0,\ldots,m-1}$	$\mathbf{U}_i = (u_i, \ldots u_{n+i-1})$

Public	Secret
n and $\ell \in \mathbb{N}$	$\mathbf{u} \in \{0,1\}^n$
$f \in \mathbb{F}_2[X_1,\ldots,X_n]$	$\boldsymbol{\omega} \in \{0,\ldots,2^n-1\}^n$

m is the number of outputs

Intermediate states	
$(u_i)_{i\geq n}$	$u_{n+i} = f(u_i,\ldots,u_{n+i-1})$
$(\mathbf{U}_i)_{0,\ldots,m-1}$	$\mathbf{U}_i = (u_i,\ldots u_{n+i-1})$
$\mathbf{v} = (v_0,\ldots,v_{m-1})$	$v_i = \langle \mathbf{U}_i, \omega\rangle \bmod M$

Public	Secret
n and $\ell \in \mathbb{N}$	$\mathbf{u} \in \{0,1\}^n$
$f \in \mathbb{F}_2[X_1, \ldots, X_n]$	$\boldsymbol{\omega} \in \{0, \ldots, 2^n - 1\}^n$

m is the number of outputs

Intermediate states	
$(u_i)_{i \geq n}$	$u_{n+i} = f(u_i, \ldots, u_{n+i-1})$
$(\mathbf{U}_i)_{0,\ldots,m-1}$	$\mathbf{U}_i = (u_i, \ldots u_{n+i-1})$
$\mathbf{v} = (v_0, \ldots, v_{m-1})$	$v_i = \langle \mathbf{U}_i, \omega \rangle \bmod M$
$\mathbf{s} = (s_0, \ldots, s_{m-1})$	$s_i = v_i // 2^\ell$
$\boldsymbol{\delta} = (\delta_0, \ldots, \delta_{m-1})$	$v_i = 2^\ell s_i + \delta_i$, $\lvert \boldsymbol{\delta} \rvert_\infty \leq 2^\ell$

Table of Contents

1. Definition of the Knapsack Generator
2. Attacks on the Knapsack Generator
3. Generalized Knapsack Generator

The main flaw

$= $  $(\mathbf{u})$ $+$  (ω)

$n(1+n)$ bits = n bits + n^2 bits

The main flaw

$n(1+n)$ bits

$=$

$(\mathbf{u})$

n bits

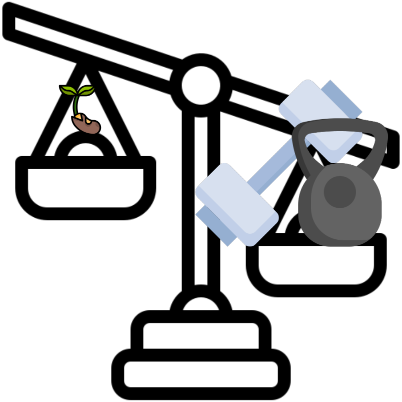
$\uparrow$

SMALL

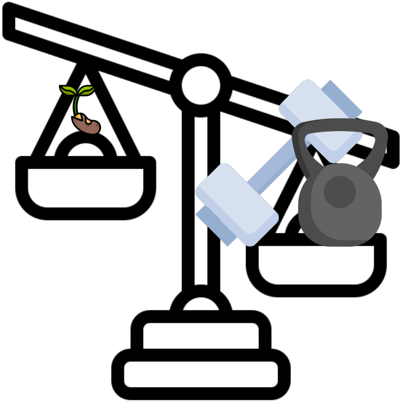
$+$

(ω)

n^2 bits



The secret is unbalanced.



The secret is unbalanced.
For a secret of ~ 1024 bits, the seed $(\mathbf{u})$ is only made of 32 bits.

Layout

ApproxWeights($\mathbf{u}, \mathbf{s}(\mathit{short})$):

???

Return($\boldsymbol{\omega}'$)

Check Consistency ($\mathbf{u}', \boldsymbol{\omega}', \mathbf{s}(\mathit{long})$):

$\mathbf{s}' = \mathit{PRNG}(\mathbf{u}', \boldsymbol{\omega}')$

Return Boolean($\mathbf{s}'$ is close to $\mathbf{s}$)

ApproxWeights($\mathbf{u}, \mathbf{s}(\textit{short})$):

???

Return($\boldsymbol{\omega}'$)

Check Consistency ($\mathbf{u}', \boldsymbol{\omega}', \mathbf{s}(\textit{long})$):

$\mathbf{s}' = \textit{PRNG}(\mathbf{u}', \boldsymbol{\omega}')$

Return Boolean($\mathbf{s}'$ is close to $\mathbf{s}$)

Full Attack($\mathbf{s}$):

For $\mathbf{u}' \in \{0,1\}^n$:

$\boldsymbol{\omega}' =$ ApproxWeights($\mathbf{u}', \mathbf{s}(\textit{short})$)

If Check Consistency($\mathbf{u}', \boldsymbol{\omega}', \mathbf{s}(\textit{long})$) $=$ True

Return ($\mathbf{u}', \boldsymbol{\omega}'$)

End If

End For

- If $\mathbf{v} = (v_0, \ldots v_{n-1})$, $\|\mathbf{v}\|_\infty = \max_{i \in \{0,\ldots,n-1\}} |v_i|$
- If M is a matrix, $\|M\|_\infty = \max_{\|\mathbf{v}\|_\infty = 1} \|\mathbf{v}M\|_\infty$

Hence

$$\|\mathbf{v}M\|_\infty \leq \|\mathbf{v}\|_\infty \|M\|_\infty$$

Attack of Knellwolf and Meier [2]

$$U = \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \dots \\ \mathbf{U}_{m-1} \end{pmatrix}$$

[2] Knellwolf, S., & Meier, W. (2011). Cryptanalysis of the knapsack generator. FSE 2011

Attack of Knellwolf and Meier [2]

$$U = \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \dots \\ \mathbf{U}_{m-1} \end{pmatrix}$$

$$\begin{aligned} \omega U &= \mathbf{v} \bmod M \\ &= 2^{\ell}\mathbf{s} + \boldsymbol{\delta} \bmod M \end{aligned}$$

[2] Knellwolf, S., & Meier, W. (2011). Cryptanalysis of the knapsack generator. FSE 2011

Attack of Knellwolf and Meier [2]

$$U = \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \dots \\ \mathbf{U}_{m-1} \end{pmatrix}$$

$$\begin{aligned} \omega U &= \mathbf{v} \bmod M \\ &= 2^{\ell}\mathbf{s} + \boldsymbol{\delta} \bmod M \end{aligned}$$

T such that $UT = I_n \bmod M$

[2] Knellwolf, S., & Meier, W. (2011). Cryptanalysis of the knapsack generator. FSE 2011

$$U = \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \dots \\ \mathbf{U}_{m-1} \end{pmatrix}$$

$$\begin{aligned} \omega U &= \mathbf{v} \bmod M \\ &= 2^\ell \mathbf{s} + \boldsymbol{\delta} \bmod M \end{aligned}$$

T such that $UT = I_n \bmod M$

$$\begin{aligned} \omega &= \mathbf{v} T \bmod M \\ &= 2^\ell \mathbf{s} T + \boldsymbol{\delta} T \bmod M \end{aligned}$$

$$\omega - 2^\ell \mathbf{s} T = \boldsymbol{\delta} T \bmod M$$

[2] Knellwolf, S., & Meier, W. (2011). Cryptanalysis of the knapsack generator. FSE 2011

Attack of Knellwolf and Meier [2]

$$U = \begin{pmatrix} \mathbf{U}_0 \\ \mathbf{U}_1 \\ \dots \\ \mathbf{U}_{m-1} \end{pmatrix}$$

$$\begin{aligned} \boldsymbol{\omega} U &= \mathbf{v} \bmod M \\ &= 2^{\ell}\mathbf{s} + \boldsymbol{\delta} \bmod M \end{aligned}$$

T such that $UT = I_n \bmod M$

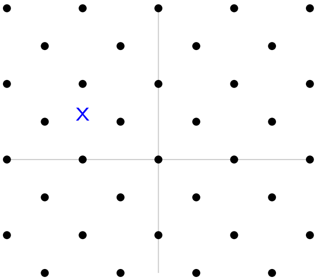
$$\begin{aligned} \boldsymbol{\omega} &= \mathbf{v}T \bmod M \\ &= 2^{\ell}\mathbf{s}T + \boldsymbol{\delta}T \bmod M \end{aligned}$$

$$\boldsymbol{\omega} - 2^{\ell}\mathbf{s}T = \boldsymbol{\delta}T \bmod M$$

Goal : Construct small $\hat{T}$ such that $\|\boldsymbol{\delta}\hat{T}\|_{\infty} < M$

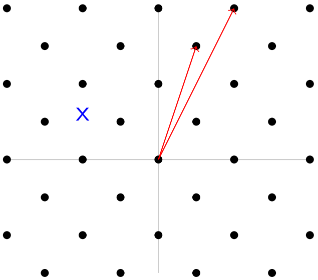
[2]Knellwolf, S., & Meier, W. (2011). Cryptanalysis of the knapsack generator. FSE 2011

Lattice Interlude: CVP and Babai Rounding



$\mathsf{x} = (-2, 1.1)$

Lattice Interlude: CVP and Babai Rounding

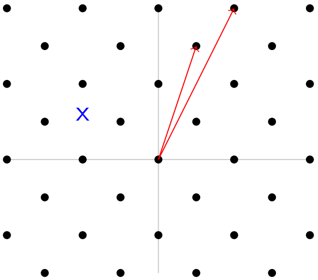


$x = (-2, 1.1)$

$$M = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \text{ and}$$

$$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$$

Lattice Interlude: CVP and Babai Rounding



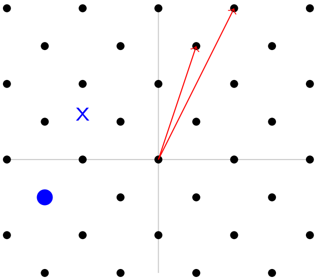
$$M = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \text{ and}$$

$$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$$

$\mathsf{x} = (-2, 1.1)$

β such that $\mathsf{x} = \beta M$, $\beta = (5.1, -3.55)$

Lattice Interlude: CVP and Babai Rounding



$$M = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \text{ and}$$

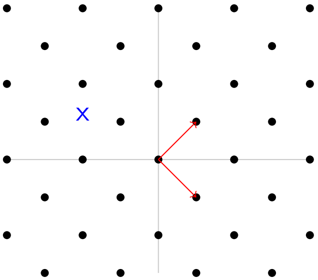
$$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$$

$\mathsf{x} = (-2, 1.1)$

β such that $\mathsf{x} = \beta M$, $\beta = (5.1, -3.55)$

$x' = \lfloor \beta \rceil M = (-3, -1)$

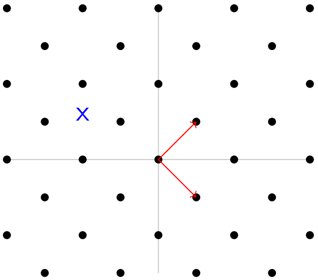
Lattice Interlude: CVP and Babai Rounding



$\mathsf{x} = (-2, 1.1)$

$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ and

$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$



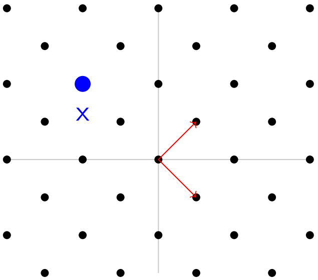
$\mathsf{x} = (-2, 1.1)$

$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ and

$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$

β such that x$= \beta M$, $\beta = (-0.45, -1.55)$

Lattice Interlude: CVP and Babai Rounding



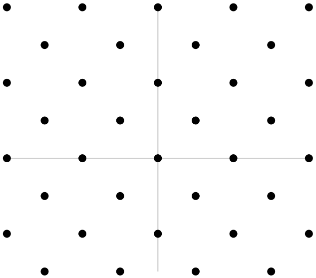
$$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \text{ and}$$
$$\mathcal{L} = \{\alpha M \mid \alpha \in \mathbb{Z}^2\}$$

$x = (-2, 1.1)$

β such that $x = \beta M$, $\beta = (-0.45, -1.55)$

$x' = \lfloor \beta \rceil M = (-2, 2)$

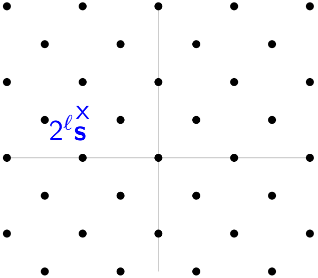
I have $\mathbf{v} = \boldsymbol{\omega} U \bmod M$



$\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$

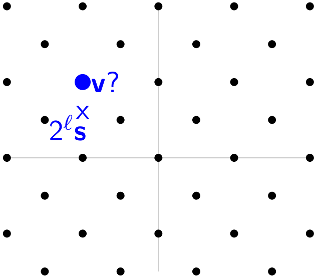
My variation

I have $\mathbf{v} = \boldsymbol{\omega} U \bmod M$ and $\mathbf{v} = 2^\ell \mathbf{s} + \boldsymbol{\delta}$ with $\boldsymbol{\delta}$ small



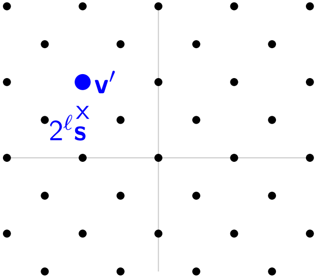
$\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$

I have $\mathbf{v} = \boldsymbol{\omega} U \bmod M$ and $\mathbf{v} = 2^\ell \mathbf{s} + \boldsymbol{\delta}$ with $\boldsymbol{\delta}$ small



$\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$

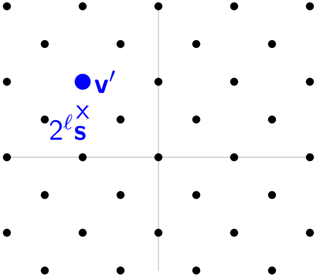
I have $\mathbf{v} = \boldsymbol{\omega} U \bmod M$ and $\mathbf{v} = 2^{\ell}\mathbf{s} + \boldsymbol{\delta}$ with $\boldsymbol{\delta}$ small



$\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$

Failed, this is not $\mathbf{v}$, we call it $\mathbf{v}'$

I have $\mathbf{v} = \boldsymbol{\omega} U \bmod M$ and $\mathbf{v} = 2^\ell \mathbf{s} + \boldsymbol{\delta}$ with $\boldsymbol{\delta}$ small



$\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$

Failed, this is not $\mathbf{v}$, we call it $\mathbf{v}'$

We compute $\boldsymbol{\omega}$' as

$$\boldsymbol{\omega}' U = \mathbf{v}' \bmod M$$

Why is $\boldsymbol{\omega}'$ close to $\boldsymbol{\omega}$?

Why does it work ? First Explanation

$(\omega - \omega')U = \mathbf{v} - \mathbf{v}' \bmod M$

Why does it work ? First Explanation

$$(\omega - \omega')U = \mathbf{v} - \mathbf{v}' \bmod M \quad \Leftrightarrow (\omega - \omega') = (\mathbf{v} - \mathbf{v}')\hat{T} \bmod M$$

Why does it work ? First Explanation

$$\begin{aligned}(\omega - \omega')U = \mathbf{v} - \mathbf{v}' \bmod M \quad &\Leftrightarrow (\omega - \omega') = (\mathbf{v} - \mathbf{v}')\hat{T} \bmod M \\ &\Rightarrow \|\omega - \omega'\|_\infty \leq \|\hat{T}\|_\infty \|\mathbf{v} - \mathbf{v}'\|_\infty\end{aligned}$$

Why does it work ? First Explanation

$$
\begin{aligned}
(\omega - \omega')U = \mathbf{v} - \mathbf{v}' \bmod M \quad &\Leftrightarrow (\omega - \omega') = (\mathbf{v} - \mathbf{v}')\hat{T} \bmod M \\
&\Rightarrow \|\omega - \omega'\|_\infty \leq \|\hat{T}\|_\infty \|\mathbf{v} - \mathbf{v}'\|_\infty
\end{aligned}
$$

In KW case: $\|\omega - 2^\ell \mathbf{s}\hat{T}\|_\infty \simeq \|\hat{T}\|_\infty \|\boldsymbol{\delta}\|_\infty$

Why does it work ? First Explanation

$$
\begin{aligned}
(\omega - \omega')U = \mathbf{v} - \mathbf{v}' \bmod M \quad &\Leftrightarrow (\omega - \omega') = (\mathbf{v} - \mathbf{v}')\hat{T} \bmod M \\
&\Rightarrow \|\omega - \omega'\|_\infty \leq \|\hat{T}\|_\infty \|\mathbf{v} - \mathbf{v}'\|_\infty
\end{aligned}
$$

In KW case: $\|\omega - 2^\ell \mathbf{s}\hat{T}\|_\infty \simeq \|\hat{T}\|_\infty \|\boldsymbol{\delta}\|_\infty$
But in our case $\|\omega - \omega'\|_\infty \ll \|\hat{T}\|_\infty \|\mathbf{v} - \mathbf{v}'\|_\infty$, precisely
$\|\omega - \omega'\|_\infty \leq \|\mathbf{v} - \mathbf{v}'\|_\infty$

Why does it work ? Second Explanation

I already have $\|\mathbf{v} - \mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

Why does it work ? Second Explanation

I already have $\|\mathbf{v} - \mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

Why does it work ? Second Explanation

I already have $\|\mathbf{v} - \mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

If I call $\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$, then

$$(\mathbf{v} - \mathbf{v}') \in \mathcal{A} = \mathcal{L} \cap B_{m,\infty}(2^{\ell+1})$$

$$(\boldsymbol{\omega} - \boldsymbol{\omega}') \in \mathcal{B} = \mathbb{Z}^n \cap B_{n,\infty}\left(\frac{2^{\ell+1}}{\|U\|_\infty}\right)$$

Why does it work ? Second Explanation

I already have $\|\mathbf{v} - \mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

If I call $\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$, then

$$(\mathbf{v} - \mathbf{v}') \in \mathcal{A} = \mathcal{L} \cap B_{m,\infty}(2^{\ell+1})$$

$$(\boldsymbol{\omega} - \boldsymbol{\omega}') \in \mathcal{B} = \mathbb{Z}^n \cap B_{n,\infty}\left(\frac{2^{\ell+1}}{\|U\|_\infty}\right)$$

By (1), $\mathcal{B} \times U \subseteq \mathcal{A}$

Why does it work ? Second Explanation

I already have $\|\mathbf{v}-\mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega}-\boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

If I call $\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$, then

$$(\mathbf{v}-\mathbf{v}') \in \mathcal{A} = \mathcal{L} \cap B_{m,\infty}(2^{\ell+1})$$

$$(\boldsymbol{\omega}-\boldsymbol{\omega}') \in \mathcal{B} = \mathbb{Z}^n \cap B_{n,\infty}\left(\frac{2^{\ell+1}}{\|U\|_\infty}\right)$$

By (1), $\mathcal{B} \times U \subseteq \mathcal{A}$ and I want $\mathcal{A} \subseteq \mathcal{B} \times U$

Why does it work ? Second Explanation

I already have $\|\mathbf{v}-\mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega}-\boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

If I call $\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$, then

$$(\mathbf{v}-\mathbf{v}') \in \mathcal{A} = \mathcal{L} \cap B_{m,\infty}(2^{\ell+1})$$

$$(\boldsymbol{\omega}-\boldsymbol{\omega}') \in \mathcal{B} = \mathbb{Z}^n \cap B_{n,\infty}\left(\frac{2^{\ell+1}}{\|U\|_\infty}\right)$$

By (1), $\mathcal{B} \times U \subseteq \mathcal{A}$ and I want $\mathcal{A} \subseteq \mathcal{B} \times U$
We will show that $|\mathcal{B}| \geq |\mathcal{A}|$

Why does it work ? Second Explanation

I already have $\|\mathbf{v} - \mathbf{v}'\|_\infty \leq 2^{\ell+1} \Leftarrow \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_\infty \leq \frac{2^{\ell+1}}{\|U\|_\infty}$ (1)

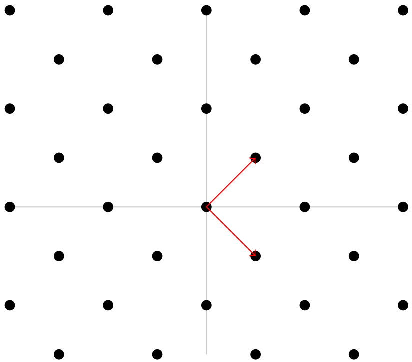
If I call $\mathcal{L} = \{\alpha U \bmod M \mid \alpha \in \mathbb{Z}^n\}$, then

$$(\mathbf{v} - \mathbf{v}') \in \mathcal{A} = \mathcal{L} \cap B_{m,\infty}(2^{\ell+1})$$

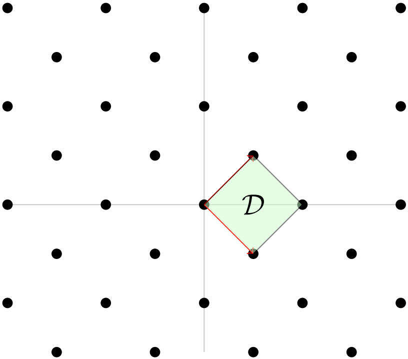
$$(\boldsymbol{\omega} - \boldsymbol{\omega}') \in \mathcal{B} = \mathbb{Z}^n \cap B_{n,\infty}\left(\frac{2^{\ell+1}}{\|U\|_\infty}\right)$$

By (1), $\mathcal{B} \times U \subseteq \mathcal{A}$ and I want $\mathcal{A} \subseteq \mathcal{B} \times U$
We will show that $|\mathcal{B}| \geq |\mathcal{A}|$

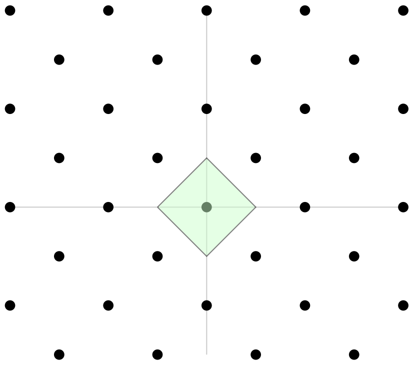
$$|\mathcal{B}| = (2\lfloor \frac{2^{\ell+1}}{\|U\|_\infty} \rfloor - 1)^n$$



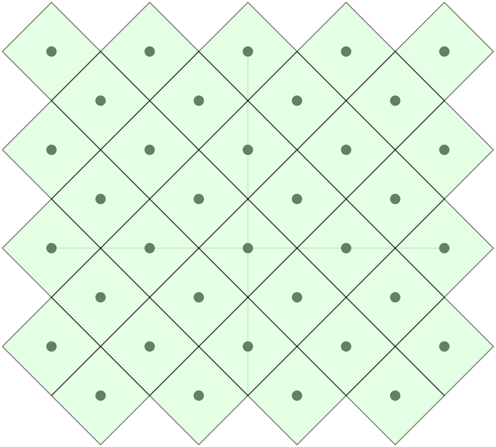
Lattice Interlude n2: Fundamental domain



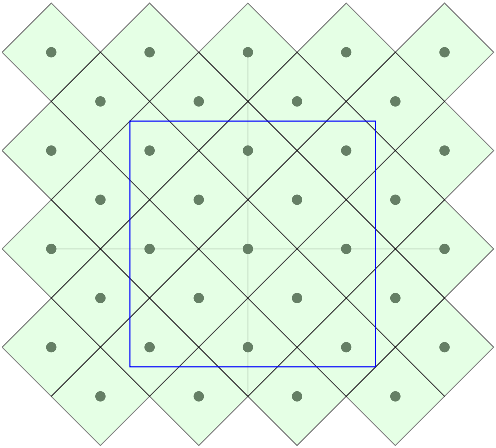
Lattice Interlude n2: Fundamental domain



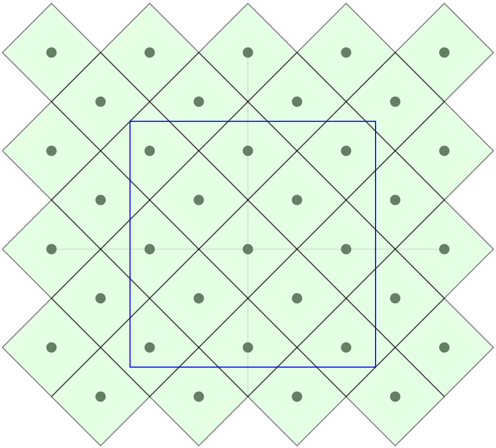
Lattice Interlude n2: Fundamental domain



Lattice Interlude n2: Fundamental domain



Lattice Interlude n2: Fundamental domain



$$\frac{vol(\textcolor{blue}{rectangle})}{vol(\mathcal{D})} = 12.5 \sim 13$$

$$|\mathcal{B}| = (2\lfloor \frac{2^{\ell+1}}{\|U\|_\infty} \rfloor - 1)^n$$

$$|\mathcal{A}| \simeq \frac{2^n(2^{\ell+1} - 1)^n}{2^{n-m}}$$

For $n = 32$ and $m = 40$ we obtain $|\mathcal{B}| \geq |\mathcal{A}|$ for $\ell \leq 14$.

$$|\mathcal{B}| = (2\lfloor \frac{2^{\ell+1}}{\|U\|_\infty} \rfloor - 1)^n$$

$$|\mathcal{A}| \simeq \frac{2^n(2^{\ell+1}-1)^n}{2^{n-m}}$$

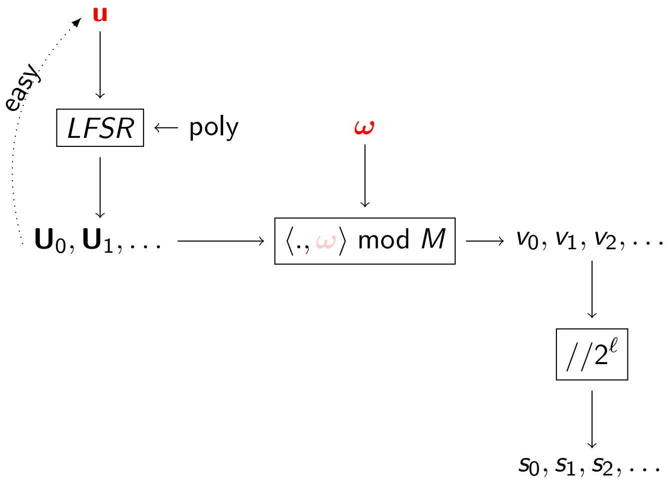
For $n = 32$ and $m = 40$ we obtain $|\mathcal{B}| \geq |\mathcal{A}|$ for $\ell \leq 14$.

ℓ	5	10	15	20	25
$log_2(\|\omega - 2^\ell \hat{T}\|_\infty)$	9.9	14.9	19.8	24.7	~~31~~
$log_2(\|\omega - \omega'\|_\infty)$	3.6	8.7	13.6	18.7	~~31~~

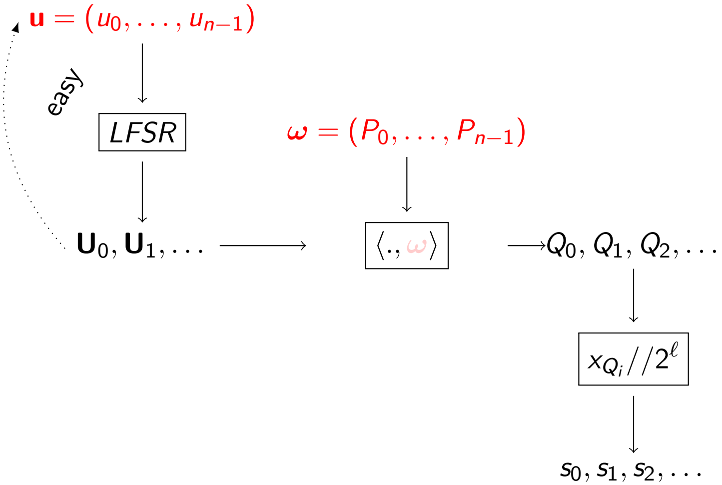
Table of Contents

1. Definition of the Knapsack Generator
2. Attacks on the Knapsack Generator
3. Generalized Knapsack Generator

Knapsack Generator by Rueppel and Massey



Generalized Knapsack Generator by Von zur Gathen and Shparlinski[3]



[3] von zur Gathen, J., & Shparlinski, I. E. . Predicting subset sum pseudorandom generators. In Selected Areas in Cryptography: 11th International Workshop, SAC 2004.

Formalization of the Generalized Knapsack Generator

Public	Secret
n and $\ell \in \mathbb{N}$	$\mathbf{u} = (u_0, \ldots, u_{n-1}) \in \{0,1\}^n$
$f \in \mathbb{F}_2[X_1, \ldots, X_n]$	$\boldsymbol{\omega} = (P_0, \ldots, P_{n-1}) \in \mathcal{E}^n$
$\mathcal{E}$ elliptic curve over $\mathbb{F}_p$	

Formalization of the Generalized Knapsack Generator

Public	Secret
n and $\ell \in \mathbb{N}$ $f \in \mathbb{F}_2[X_1, \ldots, X_n]$ $\mathcal{E}$ elliptic curve over $\mathbb{F}_p$	$\mathbf{u} = (u_0, \ldots, u_{n-1}) \in \{0,1\}^n$ $\boldsymbol{\omega} = (P_0, \ldots, P_{n-1}) \in \mathcal{E}^n$

m is the number of outputs

Intermediate states	
$(u_i)_{i \geq n}$	$u_{n+i} = f(u_i, \ldots, u_{n+i-1})$
Q_j	$Q_j = \sum_{i=0}^{n-1} u_{i+j} P_i$
s_i	$s_i = x_{Q_i} // 2^\ell$
δ_i	$x_{Q_i} = 2^\ell s_i + \delta_i$, $\delta_i \leq 2^\ell$

$$(x, y) \text{ such that } y^2 = x^3 + ax + b \bmod p$$

For x_0:

$$(x, y) \text{ such that } y^2 = x^3 + ax + b \bmod p$$

For x_0:

- there is no P such that $x_P = x_0$

$$(x, y) \text{ such that } y^2 = x^3 + ax + b \bmod p$$

For x_0:

- there is no P such that $x_P = x_0$
- there exists P such that $x_P = x_{-P} = x_0$

$$(x, y) \text{ such that } y^2 = x^3 + ax + b \bmod p$$

For x_0:

- there is no P such that $x_P = x_0$
- there exists P such that $x_P = x_{-P} = x_0$

For $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$

- $s = \frac{y_P - y_Q}{x_P - x_Q}$
- $x_R = s^2 - x_P - x_Q$
- $y_R = y_P - s(x_P - x_R)$
- $P + Q = -R$

$\mathbb{Z}$	$\mathcal{E}$
$\|x - x'\| < 2^\ell$ $\|y - y'\| < 2^\ell$	$\|x_P - x_{P'}\| < 2^\ell$ $\|x_Q - x_{Q'}\| < 2^\ell$

$\mathbb{Z}$	$\mathcal{E}$
$\lvert x - x' \rvert < 2^\ell$ $\lvert y - y' \rvert < 2^\ell$	$\lvert x_P - x_{P'} \rvert < 2^\ell$ $\lvert x_Q - x_{Q'} \rvert < 2^\ell$
$\lvert (x + y) - (x' + y') \rvert < 2^{\ell+1}$	

Elliptic curve over $\mathbb{F}_p$ Interlude - part 2

$\mathbb{Z}$	$\mathcal{E}$
$\lvert x - x' \rvert < 2^{\ell}$ $\lvert y - y' \rvert < 2^{\ell}$	$\lvert x_P - x_{P'} \rvert < 2^{\ell}$ $\lvert x_Q - x_{Q'} \rvert < 2^{\ell}$
$\lvert (x + y) - (x' + y') \rvert < 2^{\ell+1}$ $\mathbb{P}_{x',y'}(\lvert (x + y) - (x' + y') \rvert < 2^{\ell}) = \frac{1}{2}$	

$\mathbb{Z}$	$\mathcal{E}$
$\lvert x - x' \rvert < 2^\ell$ $\lvert y - y' \rvert < 2^\ell$	$\lvert x_P - x_{P'} \rvert < 2^\ell$ $\lvert x_Q - x_{Q'} \rvert < 2^\ell$
$\lvert (x+y) - (x'+y') \rvert < 2^{\ell+1}$ $\mathbb{P}_{x',y'}(\lvert (x+y) - (x'+y') \rvert < 2^\ell) = \frac{1}{2}$	$\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell)$ $=??$

$\mathbb{Z}$	$\mathcal{E}$
$\lvert x - x' \rvert < 2^\ell$ $\lvert y - y' \rvert < 2^\ell$	$\lvert x_P - x_{P'} \rvert < 2^\ell$ $\lvert x_Q - x_{Q'} \rvert < 2^\ell$
$\lvert (x+y) - (x'+y') \rvert < 2^{\ell+1}$ $\mathbb{P}_{x',y'}(\lvert (x+y) - (x'+y') \rvert < 2^\ell) = \frac{1}{2}$	$\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell)$ $=??$

- If $P' + Q' = \pm(P+Q)$, $\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell) = 1$

Elliptic curve over $\mathbb{F}_p$ Interlude - part 2

$\mathbb{Z}$	$\mathcal{E}$
$\lvert x - x' \rvert < 2^\ell$ $\lvert y - y' \rvert < 2^\ell$	$\lvert x_P - x_{P'} \rvert < 2^\ell$ $\lvert x_Q - x_{Q'} \rvert < 2^\ell$
$\lvert (x+y) - (x'+y') \rvert < 2^{\ell+1}$ $\mathbb{P}_{x',y'}(\lvert (x+y) - (x'+y') \rvert < 2^\ell) = \frac{1}{2}$	$\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell)$ $=??$

- If $P' + Q' = \pm(P+Q)$, $\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell) = 1$
- If $P' + Q' \neq \pm(P+Q)$, $\mathbb{P}_{P',Q'}(\lvert x_{P+Q} - x_{P'+Q'} \rvert < 2^\ell)$
  $= \mathbb{P}_R(\lvert x_{P+Q} - x_R \rvert < 2^\ell) = \frac{2^\ell}{\lvert \mathcal{E} \rvert}$

The problem

$$\begin{pmatrix} P_0 & P_1 & \dots & P_{n-1} \end{pmatrix} \times \begin{pmatrix} u_0 & u_1 & \dots & u_{n-1} \\ u_1 & u_2 & \dots & u_n \\ & & \ddots & \\ u_{n-1} & u_n & \dots & u_{2n-2} \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ \vdots \\ Q_{n-1} \end{pmatrix}$$

$$\begin{pmatrix} P_0 & P_1 & \dots & P_{n-1} \end{pmatrix} \times \begin{pmatrix} u_0 & u_1 & \dots & u_{n-1} \\ u_1 & u_2 & \dots & u_n \\ & & \ddots & \\ u_{n-1} & u_n & \dots & u_{2n-2} \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ \vdots \\ Q_{n-1} \end{pmatrix}$$

Naive attack : Guess $\mathbf{u}$ and $\boldsymbol{\delta}$: $\mathcal{O}(2^n \times 2^{n\ell})$ operations

The problem

$$\begin{pmatrix} P_0 & P_1 & \dots & P_{n-1} \end{pmatrix} \times \begin{pmatrix} u_{i_1} & u_{i_1+1} & \dots & u_{i_1+n-1} \\ u_{i_2} & u_{i_2+1} & \dots & u_{i_2+n-1} \\ & & \ddots & \\ u_{i_n} & u_{i_n+1} & \dots & u_{i_n+n-1} \end{pmatrix} = \begin{pmatrix} Q_{i_1} \\ Q_{i_2} \\ \vdots \\ Q_{i_n} \end{pmatrix}$$

$$\begin{pmatrix} P_0 & P_1 & \dots & P_{n-1} \end{pmatrix} \times \begin{pmatrix} u_{i_1} & u_{i_1+1} & \dots & u_{i_1+n-1} \\ u_{i_2} & u_{i_2+1} & \dots & u_{i_2+n-1} \\ & & \ddots & \\ u_{i_n} & u_{i_n+1} & \dots & u_{i_n+n-1} \end{pmatrix} = \begin{pmatrix} Q_{i_1} \\ Q_{i_2} \\ \vdots \\ Q_{i_n} \end{pmatrix}$$

I want to go here in less than: $\mathcal{O}(2^n \times 2^{n\ell})$ operations

Two steps:

- Finding $n/2$ "good triplets" i, j, k such that $\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k$ (in $\mathbb{Z}$!)
- For each triplet, retrieving Q_i, Q_j by bruteforce.

Bruteforcing Q_i, Q_j

I have 3 points Q_i, Q_j, Q_k that I do not know but I know:

- s_i, s_j, s_k the leading bits of xQ_i, xQ_j, xQ_k
- the relation $Q_i + Q_j = Q_k$

Bruteforcing Q_i, Q_j

I have 3 points Q_i, Q_j, Q_k that I do not know but I know:

- s_i, s_j, s_k the leading bits of xQ_i, xQ_j, xQ_k
- the relation $Q_i + Q_j = Q_k$

$A_i = \{R_i \mid x_{R_i}//2^\ell = s_i\}$ and $A_j = \{R_j \mid x_{R_j}//2^\ell = s_j\}$

I have 3 points Q_i, Q_j, Q_k that I do not know but I know:

- s_i, s_j, s_k the leading bits of xQ_i, xQ_j, xQ_k
- the relation $Q_i + Q_j = Q_k$

$A_i = \{R_i \mid x_{R_i}//2^\ell = s_i\}$ and $A_j = \{R_j \mid x_{R_j}//2^\ell = s_j\}$

$$\mathbb{P}(\exists (R_i, R_j) \in A_1 \times A_2 \mid x_{R_i+R_j}//2^\ell = s_k \wedge (R_i, R_j) \neq \pm(Q_i, Q_j))$$
$$\simeq |A_i \times A_j| \times \frac{2^\ell}{|\mathcal{E}|} \simeq \frac{2^{3\ell}}{|\mathcal{E}|}$$

Bruteforcing Q_i, Q_j

I have 3 points Q_i, Q_j, Q_k that I do not know but I know:

- s_i, s_j, s_k the leading bits of xQ_i, xQ_j, xQ_k
- the relation $Q_i + Q_j = Q_k$

$A_i = \{R_i \mid x_{R_i} // 2^\ell = s_i\}$ and $A_j = \{R_j \mid x_{R_j} // 2^\ell = s_j\}$

$$\mathbb{P}(\exists (R_i, R_j) \in A_1 \times A_2 \mid x_{R_i + R_j} // 2^\ell = s_k \wedge (R_i, R_j) \neq \pm(Q_i, Q_j))$$
$$\simeq |A_i \times A_j| \times \frac{2^\ell}{|\mathcal{E}|} \simeq \frac{2^{3\ell}}{|\mathcal{E}|}$$

If ℓ small enough, I can bruteforce (Q_i, Q_j) and $(-Q_i, -Q_j)$ out of $A_i \times A_j$ in $\mathcal{O}(2^{2\ell})$ operations using s_k as a filter. They are not distinguishable.

$(\mathbf{U}_i, \mathbf{U}_j, \mathbf{U}_k) \in \{0,1\}^n$, if $n = 1$:

$\mathbf{U}_i$	0	0	0	0	1	1	1	1
$\mathbf{U}_j$	0	0	1	1	0	0	1	1
$\mathbf{U}_k$	0	1	0	1	0	1	0	1
$\mathbf{U}_i + \mathbf{U}_j$	0	0	1	1	1	1	2	2

$$\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = 3/8$$

$(\mathbf{U}_i, \mathbf{U}_j, \mathbf{U}_k) \in \{0,1\}^n$, if $n = 1$:

$\mathbf{U}_i$	0	0	0	0	1	1	1	1
$\mathbf{U}_j$	0	0	1	1	0	0	1	1
$\mathbf{U}_k$	0	1	0	1	0	1	0	1
$\mathbf{U}_i + \mathbf{U}_j$	0	0	1	1	1	1	2	2

$$\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = 3/8$$

If $n \geq 1$, $\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = (3/8)^n$

$(\mathbf{U}_i, \mathbf{U}_j, \mathbf{U}_k) \in \{0,1\}^n$, if $n = 1$:

$\mathbf{U}_i$	0	0	0	0	1	1	1	1
$\mathbf{U}_j$	0	0	1	1	0	0	1	1
$\mathbf{U}_k$	0	1	0	1	0	1	0	1
$\mathbf{U}_i + \mathbf{U}_j$	0	0	1	1	1	1	2	2

$$\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = 3/8$$

If $n \geq 1$, $\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = (3/8)^n$

$\mathbf{U}_i \in A$, $\mathbf{U}_j \in B$, $\mathbf{U}_k \in C$, $|A| = |B| = |C| = N$

$$\mathbb{E}(\text{ good triplets }) = N^3 \left(\frac{3}{8}\right)^n$$

$(\mathbf{U}_i, \mathbf{U}_j, \mathbf{U}_k) \in \{0,1\}^n$, if $n = 1$:

$\mathbf{U}_i$	0	0	0	0	1	1	1	1
$\mathbf{U}_j$	0	0	1	1	0	0	1	1
$\mathbf{U}_k$	0	1	0	1	0	1	0	1
$\mathbf{U}_i + \mathbf{U}_j$	0	0	1	1	1	1	2	2

$$\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = 3/8$$

If $n \geq 1$, $\mathbb{P}(\mathbf{U}_i + \mathbf{U}_j = \mathbf{U}_k) = (3/8)^n$

$\mathbf{U}_i \in A$, $\mathbf{U}_j \in B$, $\mathbf{U}_k \in C$, $|A| = |B| = |C| = N$

$$\mathbb{E}(\text{ good triplets }) = N^3 \left(\frac{3}{8}\right)^n$$

For now $N \simeq \left(\frac{8}{3}\right)^{n/3}$

The good triplets have a bias as $w(\mathbf{U}_k) = w(\mathbf{U}_i) + w(\mathbf{U}_j)$.

The good triplets have a bias as $w(\mathbf{U}_k) = w(\mathbf{U}_i) + w(\mathbf{U}_j)$.
Then we often have $w(\mathbf{U}_i) \simeq w(\mathbf{U}_j) \simeq n/3$ and $w(\mathbf{U}_k) \simeq 2n/3$

A Sub-Quadratic Algorithm

The good triplets have a bias as $w(\mathbf{U}_k) = w(\mathbf{U}_i) + w(\mathbf{U}_j)$.
Then we often have $w(\mathbf{U}_i) \simeq w(\mathbf{U}_j) \simeq n/3$ and $w(\mathbf{U}_k) \simeq 2n/3$

```
function FindTriplet(A, B, C, ε)
    A′ ← {U_i ∈ A | w(U_i) ≤ n/3 + ε}
    B′ ← {U_j ∈ B | w(U_j) ≤ n/3 + ε}
    for all U_i, U_j ∈ A′ × B′ do
        if U_i + U_j ∈ C then
            return (U_i, U_j, U_k)
    return ⊥
```

For $\epsilon = 1/6$, the algorithm succeed with overwhelming probability in time $\mathcal{O}(N^{1.654}) \simeq \mathcal{O}(2^{0.78n})$.

For all $(u_0, \dots, u_{n-1})$ in $\{0,1\}^n$:

- derive all the $\mathbf{U}_i$ and find $n/2$ good triplets in $\mathcal{O}(2^{0.78n})$
- for each good triplet derive (Q_i, Q_j) and $(-Q_i, -Q_j)$ in $\mathcal{O}(2^{2\ell})$
- derive the P_i's for the $2^{n/2-1}$ possible signs combinations
- check consistency

For all $(u_0, \ldots, u_{n-1})$ in $\{0,1\}^n$:

- derive all the $\mathbf{U}_i$ and find $n/2$ good triplets in $\mathcal{O}(2^{0.78n})$
- for each good triplet derive (Q_i, Q_j) and $(-Q_i, -Q_j)$ in $\mathcal{O}(2^{2\ell})$
- derive the P_i's for the $2^{n/2-1}$ possible signs combinations
- check consistency

The complexity is

$$\mathcal{O}(2^n \times (2^{0.78n} + (n/2 \times 2^{2\ell}) + 2^{n/2-1}))$$

that is to say $\mathcal{O}(2^{1.78n})$ binary operations (with $\ell = \log_2(n)$).

When $n = 16$ and the initial sequence $(u_0, \ldots, u_{n-1})$ is known.

- When $|\mathcal{E}| = 65111$.

ℓ	1	2	3	4	5	6
m	1000	1000	1000	1000	1000	1885
time	$6.9s$	$5.3s$	$5.6s$	$5.02s$	$5.7s$	$26.7s$

- When $|\mathcal{E}| = 1099510687747$.

ℓ	1	2	3	4	5	6	7	8	9
m	1885	1885	1885	1885	1885	1885	1885	1885	1750
time	$2.1s$	$2.1s$	$2.08s$	$2.5s$	$2.6s$	$2.1s$	$3.5s$	$8.3s$	$26.7s$