The Representation Technique
Cryptanalysis for Dlog, SubsetSum, Decoding

Alexander May

Ruhr-University Bochum

Summer School Kaliningrad, July 2019

Discrete Logarithms

DLP: Discrete Logarithm Problem

Given: Generator g for $G = \langle g \rangle$ with $2^{n-1} \leq |G| < 2^n$, $\beta = g^x$
Find: $x = \mathrm{dlog}_g \beta \in \mathbb{Z}_{|G|}$

Examples:

- $G = (\mathbb{Z}, +) = \langle 1 \rangle$, $x = \mathrm{dlog}_1 \beta = \beta$
- $G = (E(\mathbb{F}_p), +)$, best algorithm $\tilde{\mathcal{O}}(\sqrt{|G|}) = \tilde{\mathcal{O}}(2^{\frac{n}{2}})$.
- $G = (\mathbb{Z}_p^*, \cdot)$, best algorithm sub-exponential
- G generic: $\Omega(\sqrt{|G|})$

Variants: small x, small Hamming weight x, faulty x, many x

DLP Enumeration

Algorithm Brute-Force DLP

Input: g, β

1. $x = 0$.
2. **While** $(g^x \neq \beta)$ **do** $x = x + 1$;

Output: $x = \mathrm{dlog}_g \beta$

Runtime:

- Need x iterations of while-loop, each costs one group operation.
- $\mathcal{O}(x) = \mathcal{O}(|G|) = \mathcal{O}(2^n)$ group operations.
- Each group operation usually costs $\mathcal{O}(\log^c n)$ bit operations.
- **Notice:** Brute-Force not so bad for small x.

Reaching Square Root Complexity

Idea:

- Write $x = x_1 + x_2 2^{n/2}$ with $0 \le x_1, x_2 < 2^{n/2}$.
- Use identity $g^{x_1} = \beta \cdot (g^{-2^{\frac{n}{2}}})^{x_2}$.

Algorithm Meet-in-the-Middle DLP

Input: g, β

1. **For** $0 \le i < 2^{n/2}$ **do** store (i, g^i) in list L.
2. Sort list L according to second entry.
3. **For** $0 \le j < 2^{n/2}$ **do** if $\exists (i, \beta \cdot (g^{-2^{\frac{n}{2}}})^j) \in L$, output $x = i + j2^{n/2}$.

Output: $x = \mathrm{dlog}_g \beta$

Correctness: MitM terminates iff $(i, j) = (x_1, x_2)$.
Run time: $\tilde{\mathcal{O}}(2^{\frac{n}{2}}) = \tilde{\mathcal{O}}(\sqrt{|G|})$. But also memory $\tilde{\Theta}(\sqrt{|G|})$.

Exercise: Modify MitM such that it has runtime $\tilde{\mathcal{O}}(\sqrt{x})$.

Multiple Discrete Logarithms

Multiple DLP

Given: Generator g for $G = \langle g \rangle$ with $2^{n-1} \leq |G| < 2^n$,
$\beta_1 = g^{x_1}, \ldots, \beta_k = g^{x_k}$

Find: $x_1, \ldots, x_k$

Easy: $\tilde{\mathcal{O}}(k \cdot \sqrt{|G|})$.

Exercise: Show that Multiple DLP can be solved in $\tilde{\mathcal{O}}(\sqrt{k \cdot |G|})$.

Small Weight Discrete Logarithms

Small weight DLP

Given: Generator g for $G = \langle g \rangle$ with $2^{n-1} \leq |G| < 2^n$,
$\beta = g^x$ with known Hamming weight $\mathrm{wt}(x) = \alpha n$, $\alpha \in [0, 1]$

Find: x

Algorithm Brute-Force Small weight DLP

Input: g, β, α

1. **For all** x with $\mathrm{wt}(x) = \alpha n$ **do** if $(g^x = \beta)$ output x;

Output: $x = \mathrm{dlog}_g \beta$

Run time: $\tilde{\mathcal{O}}(\binom{n}{\alpha n})$. How good is that?

Bounding Binomial Coefficients

Theorem Binomials

We have $\binom{n}{\alpha n} = \tilde{\Theta}(2^{H(\alpha)n})$ with $H(\alpha) = -\alpha \log(\alpha) - (1-\alpha)\log(1-\alpha)$.

By Stirling's formula $n! \sim \sqrt{2\pi n} \cdot (\frac{n}{e})^n$ we have

$$\binom{n}{\alpha n} = \frac{n!}{(\alpha n)!((1-\alpha)n)!} = \tilde{\Theta}\left(\frac{(\frac{n}{e})^n}{(\frac{\alpha n}{e})^{\alpha n}(\frac{(1-\alpha)n}{e})^{(1-\alpha)n}}\right)$$
$$= \tilde{\Theta}\left(2^{(-\alpha \log \alpha - (1-\alpha)\log(1-\alpha))n}\right) = \tilde{\Theta}(2^{H(\alpha)n})$$

Corollary

For $0 \le \alpha \le \beta \le 1$: $\binom{\beta n}{\alpha n} = \binom{\beta n}{\alpha \frac{1}{\beta} \beta n} = \tilde{\Theta}(2^{H(\frac{\alpha}{\beta}) \cdot \beta n})$.

Small weight Discrete Logarithms

Brute-Force Small Weight DLP: $\tilde{\mathcal{O}}(\binom{n}{\alpha n}) = \tilde{\mathcal{O}}(2^{H(\alpha)n})$, $\alpha = \frac{1}{2} : \tilde{\mathcal{O}}(2^n)$.

Exercise 1: Assume that we get the promise $x = x_1 + x_2 2^{n/2}$ with

$$0 \leq x_1, x_2 < 2^{n/2} \text{ and } \mathrm{wt}(x_1) = \mathrm{wt}(x_2) = \alpha \cdot \frac{n}{2}.$$

Devise a MitM algorithm with run time $\tilde{\mathcal{O}}(2^{\frac{H(\alpha)}{2}n})$.

Exercise 2: Do Exercise 1 without promise.

Faulty Discrete Logarithms

Faulty DLP

Given: Generator g for $G = \langle g \rangle$ with $2^{n-1} \leq |G| < 2^n$,
$\beta = g^x$, faulty $\tilde{x}$ with αn, $\alpha \in [0,1]$ many $1 \to 0$-flips of x

Find: x

Mini Exercise: Show how Faulty DLP relates to Small weight DLP.

Finding a function collision

Collision finding

Given: function $f : \{0,1\}^n \rightarrow \{0,1\}^n$ (with random properties)
Find: $x_1 \neq x_2$ with $f(x_1) = f(x_2)$

- $\Pr_{x_1 \neq x_2}(f(x_1) = f(x_2)) = \frac{1}{2^n}$
- Brute Force: Sample 2^n many pairs (x_1, x_2).

Birthday Paradox – Meet in the Middle

Algorithm List Collision Finding

Input: $f : \{0,1\}^n \to \{0,1\}^n$

1. Compute list L with entries $(x_i, f(x_i))$ for $i = 1, \ldots, 2^{n/2} + 1$.
2. Search for $(x_i, y), (x_j, y) \in L$ with $i \neq j$.

Output: Collision or $\perp$

Run time & Success probabilty:

- Run time $\tilde{\mathcal{O}}(2^{\frac{n}{2}})$ (but also the same memory).
- L does not contain a collision with probability

$$\prod_{i=0}^{2^{n/2}} \left(1 - \frac{i}{2^n}\right) \leq \prod_{i=1}^{2^{n/2}} e^{-\frac{i}{2^n}} = e^{-\sum_{i=1}^{2^{n/2}} \frac{i}{2^n}} = e^{-\frac{2^{n/2}(2^{n/2}+1)}{2 \cdot 2^n}} \leq e^{-\frac{1}{2}} \approx 0.6.$$

- Thus, we succeed with probability ≈ 0.4.

Iterating a function

- Consider sequence: $x, f(x), f(f(x)), f(f(f(x))), \ldots$.
- Let us use notation $f^i(x)$ for i applications.
- Let $\gamma, \lambda > 0$ be minimal with $f^\gamma(x) = f^{\gamma+\lambda}(x)$. Then

$$f^{\gamma+1}(x) = f^{\gamma+\lambda+1}(x), f^{\gamma+2}(x) = f^{\gamma+\lambda+2}(x), \ldots$$

- By the argumentation before we expect that $\gamma + \lambda \approx 2^{\frac{n}{2}}$.

Cycle Finding

Algorithm Cycle Finding

Input: $f : \{0,1\}^n \rightarrow \{0,1\}^n$

1. **Repeat** Choose start point $x \in \{0,1\}$ **until** $x \neq f(x)$.
2. Set $i = 1,\ k_i = f(x), k_{2i} = f(f(x))$.
3. **While** $k_i \neq k_{2i}$ **do**
    1. $k_{i+1} = f(k_i), k_{2(i+1)} = f(f(k_{2i}))$. Set $i = i + 1$.
4. Set $\ell = 0,\ k_\ell = x$.
5. **While** $f(k_\ell) \neq f(k_{\ell+i})$ **do** $k_{\ell+1} = f(k_\ell), k_{\ell+i+1} = f(k_{\ell+i}), \ell = \ell + 1$.

Output: $x_1 = k_\ell, x_2 = k_{\ell+i}$ with $f(x_1) = f(x_2)$ and $x_1 \neq x_2$

Cycle Finding

Correctness

- After the first while-loop we have $k_i = k_{2i}$.
- We already know that $k_j = k_{j+c\lambda}$, $c \in \mathbb{N}$ for all $j \geq \gamma$.
- We conclude that $i = k\lambda$.
- In the second loop we find the minimum γ for which

$$k_\gamma = k_{\gamma+k\lambda}.$$

- At termination we have $f(k_{\gamma-1}) = f(k_{\gamma+k\lambda-1})$ which implies

$$f(x_1) = f(k_{\gamma-1}) = k_\gamma = k_{\gamma+k\lambda} = f(k_{\gamma+k\lambda-1}) = f(x_2).$$

- Furthermore, $x_1 \neq x_2$ by minimality of γ.

Complexity

- Memory consumption $\tilde{\mathcal{O}}(1)$.
- After $\gamma + \lambda \approx 2^{\frac{n}{2}}$ we cycle. The cycle length is λ. (**While**-loop in 3)
- In total, we need $2(\gamma + \lambda) \approx 2^{\frac{n}{2}+1}$ iterations until termination.

Two functions

Theorem Rho Method

In a function $f : \{0,1\}^n \to \{0,1\}^n$ we find a collision in time $\tilde{\mathcal{O}}(2^{\frac{n}{2}})$ with space $\tilde{\mathcal{O}}(1)$.

Two function collision finding

Given: functions $f_1 : \{0,1\}^n \to \{0,1\}^n$, $f_2 : \{0,1\}^n \to \{0,1\}^n$ (with random properties)

Find: x_1, x_2 with $f_1(x_1) = f_2(x_2)$

Theorem Rho Method

In two functions $f_1, f_2 : \{0,1\}^n \to \{0,1\}^n$ we find a collision in time $\tilde{\mathcal{O}}(2^{\frac{n}{2}})$ with space $\tilde{\mathcal{O}}(1)$.

Exercise: Adapt the previous method for two functions.

Rho Method for DLP

Representation of DLP: Define

$$f_1 : \mathbb{Z}_{|G|} \to G, x \mapsto g^{x_1} \text{ and } f_2 : \mathbb{Z}_{|G|} \to G, x_2 \mapsto \beta \cdot g^{-x_2}.$$

- Any collision (x_1, x_2) satisfies $g^{x_1} = \beta \cdot g^{-x_2}$.
- Thus, $x = x_1 + x_2 \bmod \mathbb{Z}_{|G|}$ solves DLP.
- There exist $\mathbb{Z}_{|G|}$ many representations, respectively collisions.
- For solving DLP it suffices to find a *single* representation (x_1, x_2).

Definition Representation

Let $x = x_1 + x_2$. Then (x_1, x_2) is called a *representation* of x.

Theorem Rho Method for DLP

DLP can be solved in any group G in time $\tilde{\mathcal{O}}(\sqrt{|G|})$ and memory $\tilde{\mathcal{O}}(1)$.

Exercise: Show an $\tilde{\mathcal{O}}(x^{\frac{3}{2}})$-algorithm for small x-DLP with memory $\tilde{\mathcal{O}}(1)$.

Small Weight DPL with Low Memory

Promise: $x = x_1 + x_2 2^{n/2}$ with $0 \le x_i < 2^{n/2}$ and $\mathrm{wt}(x_i) = \alpha \cdot \frac{n}{2}$.

- Search space $\mathcal{S} = \{x_i \in \mathbb{Z}_{2^{n/2}} \mid \mathrm{wt}(x_i) = \alpha \cdot \frac{n}{2}\}$.
- Therefore $|\mathcal{S}| = \binom{n/2}{\alpha \cdot n/2} = \tilde{\Theta}(2^{H(\alpha)n/2})$.
- Let $h : G \to \mathcal{S}$. Define $f_i : \mathcal{S} \to \mathcal{S}$ with

$$x_1 \mapsto h(g^{x_1}) \text{ and } x_2 \mapsto h(\beta \cdot g^{-x_2 2^{n/2}}).$$

Algorithm Folklore Low Weight DPL with Low Memory

Input: f_1, f_2, h

1. **Repeat**
   1. Find a random collision (x_1, x_2) in f_1, f_2
2. **Until** $g^{x_1} = \beta \cdot g^{-x_2 2^{n/2}}$

Output: $x = x_1 + x_2 2^{n/2}$

0.75 Algorithm

Run Time:

- Every iteration costs $\tilde{\mathcal{O}}(\sqrt{|S|})$.
- Since $f_i : \mathcal{S} \to \mathcal{S}$, we expect $|\mathcal{S}|$ collisions.
- x has a unique representation as $x = x_1 + x_2 2^{\frac{n}{2}}$.
- Therefore only a single collisions (x_1, x_2) satisfies $g^{x_1} = \beta \cdot g^{-x_2 2^{n/2}}$.
- The probability that an iteration succeeds is thus

$$p = \Pr[(x_1, x_2) \text{ satisfies } g^{x_1} = \beta \cdot g^{-x_2}] = \frac{1}{|\mathcal{S}|}.$$

- We obtain expected run time

$$p^{-1}\tilde{\mathcal{O}}(\sqrt{|S|}) = \tilde{\mathcal{O}}(|S|^{\frac{3}{2}}) = \tilde{\mathcal{O}}(2^{\frac{3}{4}H(\alpha)n})$$

- For $\alpha = \frac{1}{2}$ this is time $2^{\frac{3}{4}n}$ as opposed to $2^{\frac{1}{2}n}$ for Rho.

Improving a bit

Idea: Take the representation $x = x_1 + x_2$ with $x_1, x_2 \in \mathbb{Z}_{|G|}$ as in Rho.

- We choose $\mathrm{wt}(x_1) = \mathrm{wt}(x_2) = \frac{\alpha}{2} n$.
- Search space $\mathcal{S} = \{x_i \in \mathbb{Z}_{|G|} \mid \mathrm{wt}(x_i) = \frac{\alpha}{2} \cdot n\}$.
- Therefore $|\mathcal{S}| = \binom{n}{\alpha/2 \cdot n} = \tilde{\Theta}(2^{H(\alpha/2)n})$.
- Let $h : G \to \mathcal{S}$. Define $f_i : \mathcal{S} \to \mathcal{S}$ with

$$x_1 \mapsto h(g^{x_1}) \text{ and } x_2 \mapsto h(\beta \cdot g^{-x_2}).$$

Algorithm Improved Low Weight DPL with Low Memory

Input: f_1, f_2, h

1. **Repeat**
    1. Find a random collision (x_1, x_2) in f_1, f_2
2. **Until** $g^{x_1} = \beta \cdot g^{-x_2}$

Output: $x = x_1 + x_2 2^{n/2}$

0.72 Algorithm

Run Time:

- Every iteration cost $\tilde{\mathcal{O}}(\sqrt{|S|})$.
- Since $f_i : \mathcal{S} \rightarrow \mathcal{S}$, we expect $|\mathcal{S}|$ collisions.
- x has $\binom{\alpha n}{\frac{\alpha}{2} n} = \tilde{\Theta}(2^{\alpha n})$ many representation as $x = x_1 + x_2$.
- All representations (x_1, x_2) satisfy $g^{x_1} = \beta \cdot g^{-x_2}$.
- The probability that an iteration succeeds is thus

$$p = \Pr[(x_1, x_2) \text{ satisfies } g^{x_1} = \beta \cdot g^{-x_2}] = \frac{\tilde{\Theta}(2^{\alpha n})}{|\mathcal{S}|}.$$

- We obtain expected run time

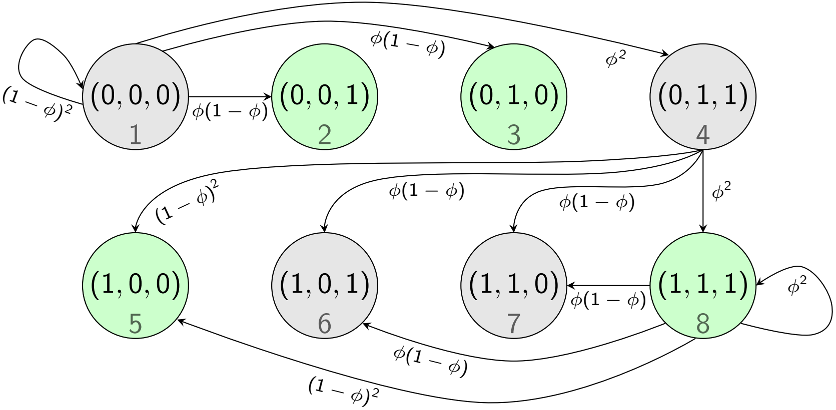
$$p^{-1}\tilde{\mathcal{O}}\left(\sqrt{|S|}\right) = \tilde{\mathcal{O}}\left(\frac{|S|^{\frac{3}{2}}}{2^{\alpha n}}\right) = \tilde{\mathcal{O}}(2^{(\frac{3}{2}H(\alpha/2)-\alpha)n})$$

- For $\alpha = \frac{1}{2}$ this is time $2^{0.72n}$ as opposed to $2^{\frac{1}{2}n}$ for Rho.

Improving a bit more via carries

Idea: Take $\mathrm{wt}(x_1) = \mathrm{wt}(x_2) = \phi n \geq \frac{\alpha}{2} n$ such that $\mathrm{wt}(x_1 + x_2) = \alpha n$.

- Search space $\mathcal{S} = \{x_i \in \mathbb{Z}_{|G|} \mid \mathrm{wt}(x_i) = \phi n\}$.
- Therefore $|\mathcal{S}| = \binom{n}{\phi n} = \tilde{\Theta}(2^{H(\phi)n})$.
- Analysis: Take each 1-coordinate in x_1, x_2 with probability ϕ.



Analysis

- Define matrix M for Markov process.
- Process has a stationary distribution $\pi = (\pi_1, \ldots, \pi_8)$ with $\pi = M\pi$.
- We solve the system of linear equations

$$\pi = M\pi, \ \pi_1 + \ldots + \pi_8 = 1, \ \pi_2 + \pi_3 + \pi_5 + \pi_8 = \alpha.$$

- Obtain $\alpha = 4\phi^4 - 4\phi^3 - \phi^2 + 2\phi$. Check: $\phi = \frac{1}{2} \Rightarrow \alpha = \frac{1}{2}$.
- Number of representations (x_1, x_2): heuristically $\frac{|\mathcal{S}|^2}{\binom{n}{\alpha n}}$.
- This implies $p = \frac{|\mathcal{S}|}{\binom{n}{\alpha n}}$ and run time

$$p^{-1}|\mathcal{S}|^{\frac{1}{2}} = \frac{\binom{n}{\alpha n}}{|\mathcal{S}|^{\frac{1}{2}}} = 2^{(H(\alpha) - \frac{1}{2}H(\phi))n}.$$

- $\alpha = \frac{1}{2}$: Complexity $2^{\frac{1}{2}n}$.

Parallel Collision Search

Theorem PCS

Given functions $f_0, \dots, f_k : \{0,1\}^n \to \{0,1\}^n$. We find a collision between f_0 and all other $f_1, \dots, f_k$ in time $\tilde{\mathcal{O}}(\sqrt{k} 2^{\frac{n}{2}})$ with space $\tilde{\mathcal{O}}(k)$.

Multiple Dlog

- Let $\beta_1 = g^{x_1}, \dots, \beta_k = g^{x_k}$. Define functions $\mathbb{Z}_{|G|} \to G$ with

$$f_0 : x_0 \mapsto g^{x_0} \text{ and } f_i : x_i \mapsto \beta_i \cdot g^{-x_i} \text{ for } i = 1, \dots, k.$$

- Collision (x_0, x_i) solves i^{th} dlog instance.
- Run time is $\tilde{\mathcal{O}}(\sqrt{k|G|})$ with space only $\tilde{\mathcal{O}}(k)$.

Subset Sum

Problem Subset Sum

Given: $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_{2^n}^n$, $t \in \mathbb{Z}_{2^n}$
Find: $\mathbf{e} = (e_1, \ldots, e_n) \in \{0,1\}^n$ with $\sum_{i=1}^n e_i a_i = t \bmod 2^n$

Cryptanalysis basics:

- Brute Force: $\tilde{\mathcal{O}}(2^n)$
- Meet-in-the-Middle: $\tilde{\mathcal{O}}(2^{\frac{n}{2}})$
- **Questions:** Low Memory Algorithms? Faster?

Exercise:
Express DLP as a subset sum problem in $(G, \cdot)$ instead of $(\mathbb{Z}_{2^n}, +)$.

Schroeppel-Shamir algorithm (1979)

Idea: Write $\sum_{i=1}^{\frac{n}{4}} e_i a_i + \sum_{i=\frac{n}{4}+1}^{\frac{1}{2}n} e_i a_i = t - \sum_{i=\frac{3}{4}n}^{\frac{1}{2}n} e_i a_i - \sum_{i=\frac{3}{4}n+1}^{n} e_i a_i$.

Algorithm 4-List algorithm

Input:

1. Generate lists $L_1, \ldots, L_4$ with $L_1 = \{\sum_{i=1}^{\frac{n}{4}} e_i a_i \mid (e_1, \ldots, e_{\frac{n}{4}}) \in \{0,1\}^{\frac{n}{4}}\}$, etc.
2. **Repeat**
   1. Choose $r \in_R \mathbb{Z}_{2^{\frac{n}{4}}}$.
   2. Compute $L_{12} = L_1 \bowtie_{\frac{n}{4}} L_2 := \{\sum_{i=1}^{\frac{n}{2}} e_i a_i \mid \sum_{i=1}^{\frac{n}{2}} e_i a_i = r \bmod 2^{\frac{n}{4}}\}$ and $L_{34} = L_3 \bowtie_{\frac{n}{4}} L_4 := \{t - \sum_{i=\frac{n}{2}+1}^{n} e_i a_i \mid t - \sum_{i=\frac{n}{2}+1}^{n} e_i a_i = r \bmod 2^{\frac{n}{4}}\}$.
   3. Compute $L = L_{12} \bowtie_n L_{34} := \{\sum_{i=1}^{n} e_i a_i \mid \sum_{i=1}^{n} e_i a_i = t \bmod 2^n\}$
3. **Until** $|L| \neq \emptyset$

Output: $\mathbf{e}$ from L

Analysis Shamir-Shroeppel

Correctness (Termination):

- Let **e** be a subset sum solution. Let $r = \sum_{i=1}^{n/2} e_i a_i \bmod 2^{n/4}$.
- Assume that we choose r in Step 2.2.
- Then our algorithm terminates with output **e**.

Run time:

- Each iteration costs on expectation $\tilde{\mathcal{O}}(2^{n/4})$ time/memory.
- On expectation, it takes $2^{n/4}$ iterations for finding r.

Question: Is there a $\tilde{\mathcal{O}}(1)$ memory algorithm faster than brute-force?

0.75 Subset Sum

Idea: Use collision finding in $f_1, f_2 : \{0,1\}^{\frac{n}{2}} \to \mathbb{Z}_{2^{\frac{n}{2}}}$ with

$$f_1 :(e_1, \ldots, e_{n/2}) \mapsto \sum_{i=1}^{n/2} e_i a_i \bmod 2^{n/2} \text{ and}$$

$$f_2 :(e_{n/2+1}, \ldots, e_n) \mapsto t - \sum_{i=n/2+1}^{n} e_i a_i \bmod 2^{n/2}.$$

Algorithm Subset Sum with Low Memory

Input: f_1, f_2

1. **Repeat**
    1. Find a random collision (x_1, x_2) in f_1, f_2
2. **Until** $\sum_{i=1}^{n/2} e_i a_i = t - \sum_{i=n/2+1}^{n} e_i a_i$

Output: **e**

Analysis

Run time:

- We have $f_i : \{0,1\}^{\frac{n}{2}} \to \mathbb{Z}_{2^{\frac{n}{2}}}$ with search space size $|\mathcal{S}| = 2^{\frac{n}{2}}$.
- We expect that f_1, f_2 have $|S| = 2^{\frac{n}{2}}$ many collisions.
- Since we uniquely represent **e**, only a single collision is good.
- Need on expectation $\tilde{\mathcal{O}}(2^{\frac{n}{2}})$ iteration with cost $\tilde{\mathcal{O}}(2^{\frac{n}{4}})$ each.

Exercise: Generalize to solutions **e** with $\mathrm{wt}(\mathbf{e}) = \alpha$.

0.72 Algorithm (Becker, Coron, Joux 2011)

Idea:

- Represent $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$ with $\mathbf{e}_1, \mathbf{e}_2 \in \{0,1\}^n$, $\mathrm{wt}(\mathbf{e}_i) = \frac{n}{4}$.
- Let $\mathcal{S} := \{\mathbf{e}' \in \{0,1\}^n \mid \mathrm{wt}(\mathbf{e}') = \frac{n}{4}\}$ with $|\mathcal{S}| = \binom{n}{n/4} \approx 2^{0.811n}$.
- Use collision finding in $f_1, f_2 : \mathcal{S} \to \mathbb{Z}_{|\mathcal{S}|}$ with

$$f_1 :(e_1, \ldots, e_n) \mapsto \sum_{i=1}^{n} e_i a_i \bmod 2^{0.811n} \text{ and}$$

$$f_2 :(e_1, \ldots, e_n) \mapsto t - \sum_{i=1}^{n} e_i a_i \bmod 2^{0.811n}.$$

Algorithm Subset Sum with Low Memory

Input: f_1, f_2

1. **Repeat**
   1. Find a random collision (x_1, x_2) in f_1, f_2
2. **Until** $\sum_{i=1}^{n/2} e_i a_i = t - \sum_{i=n/2+1}^{n} e_i a_i \bmod 2^{0.811n}$

Output: $\mathbf{e}$

Analysis

Run Time:

- There are $\binom{n/2}{n/4} = \tilde{\Theta}(2^{\frac{n}{2}})$ representations $\mathbf{e}$.
- Overall run time is

$$\tilde{\mathcal{O}}(|\mathcal{S}|) \cdot \frac{\binom{n/2}{n/4}}{|\mathcal{S}|} = \frac{|\mathcal{S}|^{\frac{3}{2}}}{\binom{n/2}{n/4}} = \tilde{\mathcal{O}}(2^{0.72n}).$$

Remarks:

- Hash function h from DLP is now simply the ring homomorphism

$$\mathbb{Z}_{2^n} \to \mathbb{Z}_{2^{0.811n}},\ x \bmod 2^n \mapsto x \bmod 2^{0.811.n}.$$

- Hence subset sum allows more (subgroup) structure than DLP.
- Especially we can do a nested collision finding on the whole $\mathbb{Z}_{2^n}$.

Theorem Esser, May (2019)

Subset Sum can be solved in time $2^{0.65n}$ and space $\tilde{\mathcal{O}}(1)$.

Howgrave-Graham Joux (2010)

Idea:

- Represent $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$ with $\mathbf{e}_1, \mathbf{e}_2 \in \{0,1\}^n$, $\mathrm{wt}(\mathbf{e}_i) = \frac{n}{4}$.
- Let $\mathcal{S}_1 := \{\mathbf{e}' \in \{0,1\}^n \mid \mathrm{wt}(\mathbf{e}') = \frac{n}{4}\}$ with $|\mathcal{S}| = \binom{n}{n/4} \approx 2^{0.811n}$.
- We have $R_1 = \binom{n/2}{n/4}$ representations of $\mathbf{e}$. Then $\log R_1 \approx \frac{n}{2}$. Define

$$L_1 = \left\{ \sum_{i=1}^{n} e_i a_i \mid \mathbf{e} \in \mathcal{S}, \sum_{i=1}^{n} e_i a_i = 0 \bmod 2^{\frac{n}{2}} \right\},$$
$$L_2 = \left\{ t - \sum_{i=1}^{n} e_i a_i \mid \mathbf{e} \in \mathcal{S}, \sum_{i=1}^{n} t - e_i a_i = 0 \bmod 2^{\frac{n}{2}} \right\}.$$

- Then (on expectation) there exists a representation in $L_1 \times L_2$.
- We have $|L_1| = |L_2| = 2^{0.311n}$. Thus, we require at least time $2^{0.311n}$.
- **Observe:** Constructing L_1, L_2 is again a subset sum problem.

Getting below $2^{\frac{n}{2}}$.

Algorithm Subset Sum 1

Input: $a_1, \dots, a_n, t$

1. Construct L_1, L_2 with Schroeppel-Shamir.
2. Construct
$L = (L_1 \times L_2) \cap \{\mathbf{e}' = \mathbf{e}_1 + \mathbf{e}_2 \mid \mathbf{e}_i \in \mathcal{S}, \sum_{i=1}^{n} e_i a_i = t \bmod 2^n\}$.

Output: $L \cap \{0,1\}^n$

Run Time:

- Step 1 runs in time $2^{0.406n}$.
- We expect
$$|L| = \frac{|L_1| \cdot |L_2|}{2^{\frac{n}{2}}} = 2^{0.122n}.$$
- We can construct L in time $\tilde{\mathcal{O}}(\max\{|L_1|, |L_2|, |L|\}) = 2^{0.311n}$.
- Therefore, we obtain total run time $2^{0.406n}$.

Idea: Construct L_1, L_2 recursively with algorithm Subset Sum 1.

One more iteration

- We show how to construct L_1, L_2 is analogous. Recall that $L_1 = \left\{ \sum_{i=1}^n e_i a_i \mid \mathbf{e} \in \{0,1\}^n, \mathrm{wt}(\mathbf{e}) = \frac{n}{4}, \sum_{i=1}^n e_i a_i = 0 \bmod 2^{\frac{n}{2}} \right\}$.
- Represent $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$ with $\mathbf{e}_1, \mathbf{e}_2 \in \{0,1\}^n$, $\mathrm{wt}(\mathbf{e}_i) = \frac{n}{8}$.
- Let $\mathcal{S}_2 := \{\mathbf{e}' \in \{0,1\}^n \mid \mathrm{wt}(\mathbf{e}') = \frac{n}{8}\}$ with $|\mathcal{S}| = \binom{n}{n/8} \approx 2^{0.5435n}$.
- We have $R_2 = \binom{n/4}{n/8}$ representations of $\mathbf{e}$. Then $\log_2 R \approx \frac{n}{4}$. Define

$$L_{11} = \left\{ \sum_{i=1}^n e_i a_i \mid \mathbf{e} \in \mathcal{S}_2, \sum_{i=1}^n e_i a_i = 0 \bmod 2^{\frac{n}{4}} \right\}.$$

- Then (on expectation) there exists a representation in $L_{11} \times L_{11}$.
- We expect that $|L_{11}| = 2^{0.2935n}$.

Getting to $2^{0.337n}$

Algorithm Howgrave-Graham Joux (2010)

Input: $a_1, \ldots, a_n, t$

1. Construct L_1, L_2 with Algorithm Subset Sum 1.
2. Construct
$L = (L_1 \times L_2) \cap \{\mathbf{e}' = \mathbf{e}_1 + \mathbf{e}_2 \mid \mathbf{e}_i \in \mathcal{S}, \sum_{i=1}^{n} e_i a_i = t \bmod 2^n\}$.

Output: $L \cap \{0,1\}^n$

Run Time:

- We expect $|L_1| = \frac{|L_{11}|\cdot|L_{11}|}{2^{\frac{n}{4}}} = 2^{2\cdot 0.2935n - 0.25n} = 2^{0.337n}$.

Theorem Run Time of HGJ algorithm

HGJ solves subset sum instances in $2^{0.337n}$.

The Becker-Coron-Joux algorithm (2011)

Idea of the BCJ algorithm:

- Represent $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$ with $\mathbf{e}_i \in \{-1, 0, 1\}^n$.
- Advantage: Many more representations as in HGJ.
- Disadvantage: Also more sums that do not end up in $\{0, 1\}^n$.

Theorem Becker-Coron-Joux (2011)

BCJ solves subset sum in $2^{0.291n}$.

Theorem Esser, May (2019)

Subset Sum can be solved in $2^{0.255n}$.

- See https://arxiv.org/abs/1907.04295.
- Technique: Sampling instead of enumeration.