進捗報告 2023/4/28

大﨑 俊輔

筑波大学 情報理工学位 P 1 年

2023 年 4 月 28 日

目次

進捗の概要

進捗

1. ハイブリッド化を行うため，途中まで秘密鍵が分かっている場合の格子ベースでの攻撃を行うプログラムを書いた．これを用いて攻撃を行なった．
2. 新しいマシンのセットアップをして，フーリエ解析ベースのプログラムが動くようにした．攻撃を行うのに足りないコードを書いて攻撃を行なった．
3. l が大きいときに必要な署名数がどのようになるのかということについて主に ϵ（エラーレート）について少しだけ考察をした．

ToDo

1. 2-bit 漏洩の場合の論文を読む．
2. 実験の続き．
3. l が大きいときの考察を深める．

目次

必要な署名数の式

M：署名数，ϵ：エラーレート，l：漏洩している nonce の上位 bit 数，r：4–list sum algorithm を繰り返す回数，n_i：4–list sum algorithm の各回で 0 にする上位の bit 数.

$$M \geq \frac{1}{(1-2\epsilon)^2} \times \frac{1}{\{(2^l/\pi)\cdot \sin(\pi/2^l)\}^2} \times 2^{\mathcal{A}}$$

$$\mathcal{A} = \sum_{i=0}^{r-1} 4^{-i-1} n_i + \frac{8}{3}\left(1-4^{-r}\right)$$

目次

秘密鍵の導出

公開鍵等から求まる z_i, h_i と漏洩した k_i の上位 bit から全ての i において，以下の式を満たすような sk を求める．

$$k_i = z_i + h_i \cdot \mathrm{sk} \bmod q$$

このとき，k_i の上位 bit には偏りがあるとし，バイアス関数で計算した値が大きくなれば，正しい sk となる．

定義 1

集合 $K = \{k_i \in \mathbb{Z}_q\}_{i=1}^{M}$ に対するサンプルバイアス

$$B_q(K) := \frac{1}{M} \sum_{i=1}^{M} \exp\left(\frac{2\pi k_i}{q}\mathrm{i}\right)$$

l 毎のバイアスの絶対値

$$M \geq \frac{1}{(1-2\epsilon)^2} \times \frac{1}{\{(2^l/\pi)\cdot \sin(\pi/2^l)\}^2} \times 2^{\mathcal{A}}$$

$$|B_q(K)| \to \left(2^l/\pi\right)\cdot \sin\left(\pi/2^l\right)$$

Table: バイアスの絶対値

l	1	2	3	4	5	10
$\lvert B_q(K)\rvert$	0.6366	0.900	0.974	0.9935	0.9983	0.9999984
$\lvert B_q(K)\rvert^2$	0.405	0.810	0.948	0.987	0.996	0.9999968
$1/\lvert B_q(K)\rvert^2$	2.22	1.23	1.05	1.013	1.004	1.0000031

漏洩した nonce の長さが大きくなると，赤色の項はほとんど 1 になり，必要な署名数がおよそ半分になる．

エラー付きのモジュラバイアスに関する補題

補題 1

$b \in \{0,1\}$，すべての $\epsilon \in [0,1/2]$ と，偶数 $q > 0$ において，以下が成り立つ．
確率変数 $\boldsymbol{K}$ が $\mathbb{Z}_q$ 上の重み付き一様分布 *(weighted uniform distribution)* に従うとすると，

$$\begin{cases} \Pr[\boldsymbol{K} = k_i] = (1-b) \cdot \frac{1-\epsilon}{q/2} + b \cdot \frac{\epsilon}{q/2} & \text{if } \ 0 \le k_i < q/2 \\ \Pr[\boldsymbol{K} = k_i] = b \cdot \frac{1-\epsilon}{q/2} + (1-b) \cdot \frac{\epsilon}{q/2} & \text{if } \ q/2 \le k_i < q \end{cases}$$

$\boldsymbol{K}_b$ を $[0 + bq/2, q/2 + bq/2)$ 上の一様分布として，$\boldsymbol{K}$ のモジュラーバイアスは，

$$B_q(\boldsymbol{K}) = (1 - 2\epsilon) B_q(\boldsymbol{K}_b)$$

Algorithm 1 Bleichenbacher's attack framework

Input: $(h_i, z_i)_{i=1}^{M}$：$\mathbb{Z}_q$ 上の HNP のサンプル，M'：求めたい線形結合の個数，L_{FFT}：FFT テーブルサイズ

Output: sk の上位 bits

1: **Range reduction**

2: 線形結合の組 $\left(h'_j, z'_j\right) = \left(\sum_i \omega_{i,j} h_i, \sum_i \omega_{i,j} z_i\right)$ ． $j \in [1, M']$，係数は $\omega_{i,j} \in \{-1, 0, 1\}$ ，$\left\{\left(h'_j, z'_j\right)\right\}_{j=1}^{M'}$ である M' 個のサンプルを生成する．

(1) Small:$0 \leq h'_j < L_{\mathrm{FFT}}$

(2) Sparse: すべての $j \in [1, M']$ において，$\Omega_j := \sum_i |\omega_{i,j}|$ であり，$|B_q(K)|^{\Omega_j} \gg 1/\sqrt{M'}$

3: **Bias Computation**

4: $Z := \left(Z_0, \ldots Z_{L_{\mathrm{FFT}}-1}\right) \leftarrow (0, \ldots, 0)$

5: **for** $j = 1$ to M' **do**

6: $\quad Z_{h'_j} \leftarrow Z_{h'_j} + \exp\left(2\pi \mathrm{i} z'_j / q\right)$

7: **end for**

8: $w_i = iq/L_{\mathrm{FFT}}$ として，$\{B_q(K_{w_i})\}_{i=0}^{L_{\mathrm{FFT}}-1} \leftarrow \mathrm{FFT}(Z)$

9: $|B_q(K_{w_i})|$ が最大となる i を求める．

10: **return** w_i の上位 $\log L_{\mathrm{FFT}}$bits.

目次

フーリエ解析ベースから格子ベースへの繋ぎ

Table: 既存研究における各攻撃の成功について（緑がフーリエ解析ベース攻撃. マゼンタが格子ベースの攻撃）

	< 1	1	2	3	4
256-bit	–	–	○	○	○
192-bit	○	○	–	–	–
160-bit	○	○	○ ○	○	–

目次

論文について

- Solving BDD by Enumeration: An Update
- Mingjie Liu, Phong Q. Nguyen
- LNCS 2013

貢献

Search-LWE に対して格子攻撃を significant better than Lindner-Peikert attack で行った.
これまでは DSA で 3-bit の nonce が漏洩している場合に破っているものが最も良かったが，この論文では 2-bit の nonce が漏洩した場合に 100 個の署名に対して攻撃が成功している（a few hours）.

目次

LLL アルゴリズム

Algorithm 2 δLLL アルゴリズム

Input: 格子基底 $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) \in \mathbb{Z}^{m \times n}$, パラメータ $\frac{1}{4} < \delta < 1$
Output: $\mathcal{L}(\boldsymbol{B})$ の LLL 簡約基底

(loop)
for $i = 1 \ldots n$ **do**
  for $j = i - 1 \ldots 1$ **do**
    $c_{i,j} \coloneqq \lfloor \langle \boldsymbol{b}_i, \boldsymbol{b}_j \rangle / \langle \boldsymbol{b}_j, \boldsymbol{b}_j \rangle \rceil, \boldsymbol{b}_i \coloneqq \boldsymbol{b}_i - c_{i,j} \boldsymbol{b}_j$
  end for
end for
for $i = 1 \ldots n$ **do**
  if ある i に対して, $\delta \left\| \pi_i(\boldsymbol{b}_i) \right\|^2 \leq \left\| \pi_i(\boldsymbol{b}_{i+1}) \right\|^2$ **then**
    $\boldsymbol{b}_i$ と $\boldsymbol{b}_{i+1}$ を交換し, (loop) へ行く
  end if
end for
$\boldsymbol{B}$ を出力する.