SCIS 2003 The 2003 Symposium on Cryptography and Information Security
Hamamatsu,Japan, Jan.26-29,2003
The Institute of Electronics,
Information and Communication Engineers

高速離散対数検証法について
On the Fast DL Verifiaction

星野 文学* 小林 鉄太郎 *
Fumitaka Hoshino Tetsutaro Kobayashi

あらまし あらまし 高速離散対数検証法 [1] は署名, ゼロ知識証明等の検証コストを削減するアルゴリズムである. 本論文では高速離散対数検証法の有意性について安全性と演算コストの観点から検討する.

キーワード 高速離散対数検証法, 格子基底縮小アルゴリズム

1 はじめに

近年, 電子現金, 電子投票, オンラインゲームといった様々な ネットワークサービス が実用化されつつあり, そうしたアプリケーションを実現する為の大規模な暗号プロトコルもしばしば提案されている.

これらの暗号プロトコルでは, 署名やゼロ知識証明のような何らかの検証を必要とするステートメントがしばしば大量に利用される. このような大量の署名検証を一括して行う方法として batch 検証などが検討されており, この方法を使えば冪乗演算に於いて従来は律速段階となっていた自乗演算のコストを大幅に削減できる.

しかしながら, batch 検証は単一の署名検証には適用できないため, オンラインで対話を繰り返すようなプロトコルの場合には必ずしも向いていない.

高速離散対数検証法 [1] は, 離散対数ベースの batch 検証にしばしば用いられるテクニックと類似の方法を用いて, 単一の署名検証を高速化する技術である.

本論文では 従来は精密な検討がなされなかった高速離散対数検証法の安全性と演算コストについて精密な定式化を行い, その有意性を検討し, 高速離散対数検証法の使い方の指針を与える.

2 準備

何らかのシステムに於いて, あるステートメント $\mathcal{S}$ を検証する手段が V_1 及び V_2 と 2 通りあると仮定する.

V_1, V_2 の内, 敢えて V_2 で検証したくなる為には, 検証法 V_2 には V_1 に無い何らかのメリットが無くてはならない. 本論文では $\mathcal{T}(X)$ を X の実行時間として

$$\tau \equiv \mathcal{T}(V_1(\mathcal{S})) - \mathcal{T}(V_2(\mathcal{S})) > 0 \tag{1}$$

を検証法 V_2 のメリットとする. 即ち, 高速離散対数検証法は元の検証法より高速でなくてはならない. τ を V_1 に対する V_2 による利得と呼ぶことにする.

2.1 高速離散対数検証法の概要

p を素数として $h, h_1, h_2, \ldots$ を位数 p の巡回群の元, $e_1, e_2, \ldots$ を正整数とする. 通常, 離散対数ベースの署名等の検証は

$$h \stackrel{?}{=} \prod_{i=1}^{k-1} h_i^{e_i} \tag{2}$$

で与えられることが多い (但し $k \geq 2$). 高速離散対数検証法 [1] は (2) の型の検証処理を, e'_0 を整数, $h_0 = h^{-1}$ として

$$1 \stackrel{?}{=} \prod_{i=0}^{k-1} h_i^{e'_i}, \tag{3}$$

但し $\quad i = 1, 2, 3, \ldots$ に対し $e'_i \equiv e'_0 e_i \pmod{p}$

の型に変換する. M, S をそれぞれ群演算及び群自乗の演算コスト, ω を定数として通常 (2) の右辺の演算コストは, $l = \log_2(\max|e_i|)$ とすると

$$\left(\frac{k-1}{\omega}M + S\right) l \tag{4}$$

と見積もることが出来る. 但し ω は冪乗アルゴリズムによって決まる定数で binary method なら $\omega = 2$ である (従って $\omega \geq 2$). 冪指数が $\sim 2^{160}$ 程度の数なら, 普通は $\omega \lesssim 5$ 程度である. 本論文では ω を (抽象的な) ウィンドウサイズと呼ぶ. (3) の右辺の演算コストは,

* NTT 情報流通プラットフォーム研究所 情報セキュリティプロジェクト, 〒 239-0847 神奈川県 横須賀市 光の丘 1-1 NTT Information Sharing Platform Laboratories, NTT Corporation, 1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa-ken, 239-0847 Japan

$l' = \log_2(\max|e'_i|)$ とすると

$$\left(\frac{k}{\omega}M + S\right) l' \tag{5}$$

であるから, (1), (4),(5) より $\sigma \equiv l'/l$ として利得 τ は

$$\tau = \frac{Ml}{\omega}\left[\left(k + \omega\frac{S}{M}\right)(1-\sigma) - 1\right] \tag{6}$$

となる. 即ち,(6) が正となるような e'_i を群の冪演算に比べて十分小さな演算コストで求めることが出来れば, 署名等の検証コストを抑えることが出来る. 一般に k 個のベクトル

$$\begin{array}{ccccccc}
(& 1, & e_1, & e_2, & \ldots, & e_{k-2}, & e_{k-1} &) \\
(& 0, & p, & 0, & \ldots, & 0, & 0 &) \\
(& 0, & 0, & p, & & 0, & 0 &) \\
 & & \vdots & & \ddots & & \vdots & \\
(& 0, & 0, & 0, & & p, & 0 &) \\
(& 0, & 0, & 0, & \ldots & 0, & p &)
\end{array}$$

を何らかの格子基底縮小アルゴリズム (例えば LLL[2]) に入力すれば

$$\sigma \sim \frac{k-1}{k} \tag{7}$$

程度の e'_i が得られると期待できる. 此の時, 利得 τ は

$$\tau = \frac{S}{k}l \tag{8}$$

で, 必ず正の値を取る. 従って群演算の演算コストが冪指数どうしの演算コストよりずっと大きい極限では 高速離散対数検証法 [1] は k に依らず高速であると云える. 本論文では

$$\sigma = \frac{\log_2(\max|e'_i|)}{\log_2(\max|e_i|)} \sim \frac{\log_2(\max|e'_i|)}{\log_2 p}$$

を圧縮率と呼ぶことにする.

2.2 $O((\log p)^2)$ の格子基底縮小アルゴリズム

乗法群や楕円曲線といった群の群演算のコストは, 確かに冪指数どうしの演算コストより大きい事も多いと云えるが, 格子基底縮小アルゴリズムとしてどんなアルゴリズムを使用しても良いほど大きいとは云えない.

通常, リアルな署名や認証に用いられる巡回群での冪演算のコストは群の位数 p に対して概ね $(\log p)^3$ 程度, あるいは漸近的には $(\log p)^2 \log\log p$ 程度と考えるのが妥当である.

従って, 巡回群の冪演算コストに比べて十分小さな格子基底縮小アルゴリズムは $(\log p)^2$ 程度で動作する事が望ましい. これは, 格子基底縮小に単なる乗除算並のスピードが要求されていることを意味しており 次元 k が大きいときには難しい要求になる. さらに (8) は次元 k が大きい時, 利得 τ が減少することを意味しており, 現実に高速離散対数検証法 [1] が機能するのは k が小さい場合のみである.

2.2.1 $k = 2$ の場合

$k = 2$ の場合, 即ち

$$h \stackrel{?}{=} h_1^{e_1} \tag{9}$$

の型の検証を行う場合は t を $\sqrt{p}$ 程度の正整数たとえば $\lceil\sqrt{p}\rceil$ や $2^{\lceil(\log_2 p)/2\rceil}$ 等として以下の手続き **xgcd** を

$$(e'_0, e'_1) \leftarrow \mathbf{xgcd}(e_1, p, t)$$

と呼び出すことによって (7) 程度の e'_i を得ることが出来る. p が定数なら t も定数と考えて良い.

xgcd は, 終了条件以外は拡張 Euclid 互除法 そのものであり, $O((\log p)^2)$ のアルゴリズムである. (9) のタイプの検証は PSEC[3] の復号時の検査等で用いられる.

名称: $\mathbf{xgcd}(e_1, p, t)$. (拡張 Euclid 互除法)
入力: 整数 e_1, 素数 p, 整数 t. (但し $0 < e_1 < p$)
出力: $e'_1 \equiv e'_0 e_1 \pmod p$ を満たす (e'_0, e'_1).
処理: Step 1:
$\overrightarrow{v_0} \equiv (t_0, r_0) \leftarrow (0, p)$
$\overrightarrow{v_1} \equiv (t_1, r_1) \leftarrow (1, e_1)$
Step 2: $r_1 > t$ の間以下を繰り返す
$q \leftarrow \lfloor r_0/r_1 \rfloor$
$\overrightarrow{v_2} \leftarrow \overrightarrow{v_0} - q \cdot \overrightarrow{v_1}$
$\overrightarrow{v_0} \leftarrow \overrightarrow{v_1}$
$\overrightarrow{v_1} \leftarrow \overrightarrow{v_2}$
Step 3: $(e'_0, e'_1) \leftarrow \overrightarrow{v_1}$ を出力

2.2.2 $k \geq 3$ の場合

$k \geq 3$ の場合, $O((\log p)^2)$ で (7) を実現するのは難しいが, 圧縮率 σ を抑えれば

$$\sigma \sim \frac{2^{k-1} - 1}{2^{k-1}} \tag{10}$$

の以下の方法が $O((\log p)^2)$ となる.

入力: 整数 $e_1, \ldots, e_{k-1}$, 素数 p. (但し $0 < e_i < p$)
出力: $e'_i \equiv e'_0 e_i \pmod p$ を満たす $(e'_0, \ldots, e'_{k-1})$.
記法: $\sim x$ は x 同程度の整数.
処理:
Step 1:
$(e'_0, e'_1, \ldots, e'_{k-1}) \leftarrow (1, e_1, \ldots, e_{k-1})$
$i \leftarrow 1$, $c \leftarrow\sim p$, $t \leftarrow 1$
Step 2: $i < k$ の間以下を繰り返す
$c \leftarrow\sim \sqrt{c}$, $t \leftarrow\sim tc$
$(f_0, f_1) \leftarrow \mathbf{xgcd}(e'_i, p, t)$
$\forall j \in \{0, \ldots, k-1\}$ に対して

$$e'_j \leftarrow \begin{cases} e'_j f_0 & \text{if} \quad 0 \leq j < i \\ f_1 & \text{if} \quad j = i \\ e'_j f_0 \bmod p & \text{if} \quad i < j < k \end{cases}$$

$i \leftarrow i + 1$
Step 3: $(e'_0, e'_1, \ldots, e'_{k-1})$ を出力

(6) 及び (10) より高速離散対数検証法が高速となる条件は

$$2^{k-1} - k < \frac{S\omega}{M} \tag{11}$$

$k = 3$ ならば (11) は

$$M < S\omega \tag{12}$$

となる. 乗法群や一般的な楕円曲線ならば通常 $M < 2S$ であり, ウインドウサイズ ω は $\omega \geq 2$ であるので, $k = 3$ でも乗法群や一般的な楕円曲線では高速離散対数検証法 [1] は有意に機能する. $k = 3$, 即ち

$$h \stackrel{?}{=} h_1^{e_1} h_2^{e_2}$$

の型の検証は Shnorr 認証や ElGamal 署名 等でしばしば用いられる.

3 検証法変換の安全性

今, 何らかのシステムに於いて, ある検証者が検証法 V_1 で検証されるべきステートメント $\mathcal{S}$ を検証法 V_2 で検証してしまうと仮定する.

この場合, 攻撃の対象は検証者であり, 攻撃者と成り得るのは, 悪意ある $\mathcal{S}$ を作成する署名者または任意の第三者となる. ここでは, 強い攻撃のモデルとして悪意ある署名者を仮定する.

即ち基本的な攻撃のモデルは, 攻撃者が $V_2(\mathcal{S}) = \mathbf{True}$ なる $\mathcal{S}$ を検証者に信じ込ませて, 後から $V_1(\mathcal{S}) = \mathbf{False}$ を示して $\mathcal{S}$ を覆すことである.

即ち, もし攻撃者が $V_1(\mathcal{S}) \neq V_2(\mathcal{S})$ なる $\mathcal{S}$ を高確率で生成できてしまう場合は, 検証者が V_1 を V_2 で置き換える行為に対する攻撃が成功したことになり, 検証法 V_2 による安全性の根拠は V_1 とは別途のものを用意しなくてはならない.

攻撃者が $V_1(\mathcal{S}) \neq V_2(\mathcal{S})$ なる $\mathcal{S}$ を高い確率では生成できない事が示せれば検証法 V_2 に関する安全性の根拠は V_1 に帰着される.

3.1 高速離散対数検証法の安全性

高速離散対数検証法のジェネリックなモデルでの安全性は明らかである. 何故なら $h, h_1, h_2, \ldots$ が最初から素数位数 p の巡回群に入っていると分かっているなら $e_0'^{-1} \bmod p$ の存在によって (2) と (3) の同値性が保証されるからである.

よく用いられる乗法群や楕円曲線を用いる場合には (3) を (2) に帰着させる為には, 巡回群の元として入力されたデータ $h, h_1, h_2, \ldots$ がシステムで定義した巡回群に本当に所属しているか否かの検証 (ドメイン検証) を行い, 設計通りの群に乗っている事を確かめねばならない.

しかしながら, 例えば冪乗演算を用いて $h^p \stackrel{?}{=} 1$ 等を検証していたのでは高速離散対数検証法は高速性という観点からは完全に意味を失う. 従って高速離散対数検証法を用いる為には, ドメイン検証を高速に実行できるような巡回群を採用しなくてはならない.

署名等で良く使われる巡回群のうち, 高速なドメイン検証が利用できるものは

1 : 素数位数の素体または OEF 楕円曲線
2 : 位数が 2 × 素数 の $\mathbf{GF}(2^m)$ 楕円曲線
3 : q が Sophie Germain 素数の時の Z_q^*

等がある. 楕円曲線で Weierstrass 標準形を用いるとすれば高速なドメイン検証はそれぞれ以下のようになる. いずれも $O((\log p)^2)$ である.

1 : 入力 $(x, y) \in \mathbf{GF}(q)^2$ に対し
$y^2 \stackrel{?}{=} x^3 + a_4 x + a_6$
2 : 入力 $(x, y) \in \mathbf{GF}(2^m)^2$ に対し
$y^2 + xy \stackrel{?}{=} x^3 + a_2 x^2 + a_6$ かつ
$T(x) \stackrel{?}{=} T(a_2)$. (但し $T(X)$ は X のトレース) [4]
3 : 入力 $h \in Z_q$ に対して
Legendre 記号 $(h/q) \stackrel{?}{=} 1$

拡大体の楕円曲線で Koblitz curve を用いるような場合は, (8) や (12) で M に比べて S が極端に小さい場合を考えれば良く, 高速離散対数検証法のメリットはたとえあったとしても, 極めて小さいものと考えられる.

Batch 検証では p, q を素数として $q = 2pr + 1$ で, r の全ての素因数が p より大きい場合の Z_q^* において Legendre 記号 (h/q) が高速なドメイン検証として利用出来た [5]. しかし, 高速離散対数検証法に於いては攻撃者が冪指数 e_0' を容易に予想できる為に, このテクニックは使えない. Legendre 記号が高速なドメイン検証として働くのは q が Sophie Germain 素数 ($q = 2p + 1$) の時だけである.

4 まとめ

本研究では署名, ゼロ知識証明等の検証コストを軽減する高速離散対数検証法 [1] の有意性について演算コストと安全性観点から検討し, 以下の結論を得た.

- 巡回群の演算コストが冪指数どうしの演算コストよりずっと大きい極限では, 高速離散対数検証法は, 等式に登場する巡回群の元の数 k に依らず高速である事を示した.
- 拡張 Euclid 互除法を用いた高速離散対数検証法が $k = 3$ で有意になる条件 (12) を明らかにした.
- 署名や認証に良く用いられる巡回群に於いては, 高速離散対数検証を元の検証に帰着させる為には高速なドメイン検証が必要で, それが可能な例をいくつか示した.
- Strict Batch 検証に利用できて高速離散対数検証に利用できない巡回群の例を示した.

参考文献

[1] Kobayashi, T., Hoshino, F.: Checking $Q = kP$. In: Proceedings of the 2002 Symposium on Cryptography and Information Security. (2002) 1091–1094

[2] A.K. Lenstra, H.W. Lenstra, L. Lovász: Factoring polynomials with rational coefficients. In: Math. Ann. 261. (1982) 515–534

[3] Shoup, V.: A proposal for an iso standard for public key encryption(version 2.1) (December 20, 2001) a draft for a forthcoming ISO standard on publickey encryption, (available at `http://shoup.net/papers/iso-2_1.pdf`).

[4] Seroussi, C.: Compact Representation of Elliptic Curve Points over $\mathbb{F}_{2^n}$ (April 1998) Research Manuscript, Hewlett-Packard Laboratories, (available at `http://www.hpl.hp.com/techreports/98/HPL-98-94R1.pdf`).

[5] Hoshino, F., Abe, M., Kobayashi, T.: Lenient/strict batch verification in several groups. In Davida, G.I., Frankel, Y., eds.: 4th 2001. Volume 2200 of Lecture Notes in Computer Science., Berlin; Heidelberg; New York, Springer-Verlag (2001) 81–94