Vol.2016-IOT-32 No.32
2016/3/4

楕円曲線暗号解読における Dynamic DNS を用いた解読成功判定

三好 俊介[1,a)]　山井 成良[2,b)]　野上 保之[1,c)]

概要：
楕円曲線暗号は今日広く用いられている暗号方式である．楕円曲線暗号の効率的な解読方法として Pollard の Rho 法がある．この方法では与えられた楕円曲線上の有理点を次々に生成し，これまでに生成したものと同じ有理点が発見された場合に解読に成功したと判定するため，生成する膨大な有理点を記憶するためには膨大な記憶領域を必要とする．しかし，1 台の計算機で利用可能な記憶領域には限界があるため，生成した有理点は分散して記憶する必要がある．そこで我々は DNS を一種の大規模分散データベースとして使用し，Dynamic DNS 機能を用いて有理点を登録する方法を用いた．本稿ではその方法や性能について述べる．

キーワード：
楕円曲線暗号，Pollard の Rho 法，Dynamic DNS

Successful Attack Detection of Elliptic Curve Cryptography with Dynamic DNS

Shunsuke Miyoshi[1,a)]　Nariyoshi Yamai[2,b)]　Yasuyuki Nogami[1,c)]

Abstract:
Elliptic Curve Cryptography (ECC) is one of cryptographic schemes that has been widely used today. Pollard's Rho method is an efficient method to attack ECC. This method needs much memory pool to detect the same points (collision) in a huge group of rational points on the given elliptic curve. Since one computer has limitations in its memory pool, rational points should be saved in a distributed manner. We used DNS as a distributed database system and we registered rational points using Dynamic DNS fuction. In this paper, we show how to use DNS for attacking ECC and its performance.

Keywords: Elliptic Curve Cryptography, Pollard's Rho Method, Dynamic DNS

1. はじめに

楕円曲線暗号（Elliptic Curve Cryptography, ECC）は現在様々な場面で実用化されている暗号方式の 1 つである．この暗号は楕円曲線上の離散対数問題（Elliptic Curve Discrete Logarithm Problem, ECDLP）の計算量的困難性を安全性の根拠としている．楕円曲線暗号に対する解読（攻撃）法として Pollard の Rho 法 [1] がよく知られている．Rho 法では，与えられた楕円曲線上で有理点（rational

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University
3-1-1, Tsushima-naka, Kita, Okayama 700–8530, Japan
[2] 東京農工大学工学研究院
Institute of Engineering, Tokyo University of Agriculture and Technology
2-24-16, Nakacho, Koganei, Tokyo 184–8588, Japan
[a)] shunsuke.miyoshi@s.okayama-u.ac.jp
[b)] yamai@okayama-u.ac.jp
[c)] yasuyuki.nogami@okayama-u.ac.jp

point）を次々に生成してこれまでに生成された有理点と比較し，同じ有理点が発見された（これを「衝突」と呼ぶ）場合に，ECDLP が解けたことになる．その際，衝突までに生成した有理点を保持しておく必要があるが，楕円曲線暗号の鍵長が大きくなると保持する必要のある有理点の数が膨大になり，1 台の計算機で処理できなくなる可能性があった．

このような問題に対して，我々は DNS（Domain Name System）[2], [3] がスケーラビリティの高い一種の分散データベースである点に着目し，Dynamic DNS[4] を用いて分散する方法を検討した．しかし，DNS は登録可能なデータに制約があるなど，汎用のデータベースとは異なる点がある．また，攻撃速度の向上のためには様々な工夫が必要となる．本稿では DNS を楕円曲線暗号の攻撃に用いる際に得られた知見を明らかにし，またどの程度の性能が得られるかを示す．

なお，本稿では素体の標数および楕円曲線群の位数が 100 bit の楕円曲線暗号を攻撃目標として考察し，58 bit の楕円曲線暗号を攻撃するプログラムを実装してその性能を評価する．

2. 楕円曲線暗号と攻撃方法

本節では楕円曲線暗号および Pollard の Rho 法について説明する [5], [6], [7].

2.1 楕円曲線暗号

素数 p に対する素体 $\mathbb{F}_p$ 上で定義される楕円曲線 E を考える．一般的な楕円曲線の式は以下のように表される．

$$E : y^2 = x^3 + ax + b, \qquad a, b \in \mathbb{F}_p \tag{1}$$

無限遠点 $\mathcal{O}$ を含む E 上の有理点の集合は，有理点に関する加算 [*1] を定義することで可換群をなし，これを $E(\mathbb{F}_p)$ と表す．ここで $E(\mathbb{F}_p)$ の位数を r とする．

前節でも述べたが，楕円曲線暗号は，鍵長が大きくなると楕円曲線上の ECDLP を解くことが現実的に困難であることを安全性の根拠としている．ECDLP とは，$E(\mathbb{F}_p)$ 上の有理点 $P, Q = [s]P$ に対し，そのスカラ値 s を求める問題である．ただし，$[s]P$ とは P を s 個加算したものを意味する．楕円曲線暗号では，一般に P, Q を公開情報、スカラ値 s を秘密情報とする．

2.2 Pollard の Rho 法

一般の ECDLP を効率よく解く手法の一つとして，Pollard の Rho 法 [1] がよく知られている．例えば，素数位数 r の巡回群に属する，ある有理点を P とする．このとき $Q = [s]P$ となる 2 点 P, Q を用いて秘密情報 s を求めるとき，Rho 法では以下のような手順を行う．まず，$\mathbb{Z}_r$ を r 未満の非負整数の集合とし，a_i, b_i をランダムな $\mathbb{Z}_r$ の要素とする．そして，有理点 R_i を以下のように構成する．

$$R_i = [a_i]P + [b_i]Q \tag{2}$$

ここで，R_i を次々と生成していくと，有理点の数は有限 (r 個) であることから，$R_i = R_j$ $(i \neq j)$ となる衝突点 R_j が必ず存在する．このとき，それらの点を構成するスカラ値 a_i, b_i, a_j, b_j を用いて秘密情報 $s \equiv -(a_i - a_j)/(b_i - b_j) \pmod r$ を求めることができる．通常の Rho 法では，衝突点が生成されるまでに必要な有理点の数の期待値は誕生日のパラドックス（birthday paradox）より，$\sqrt{\pi r/2} \simeq 1.25\sqrt{r}$ で求められるが，群構造を利用した改良 [8] を行うことで $\sqrt{\pi r/12} \simeq 0.512\sqrt{r}$ まで減らすことが可能である．

Algorithm1 に Rho 法のアルゴリズム例を示す．ただし，このアルゴリズムでは（2）を用いて R_i とこれを生成した際の a_i, b_i を n 組保存したテーブルを作成しておき，このテーブルを用いて $i > n$ である R_i を R_{i-1} から求める方法を採用している．また，$\eta(R)$ は R により一意に決まる n 未満のランダムな非負整数を返す関数である．このような動作により，新しく生成される有理点 R_i は直前に生成された有理点 R_{i-1} を用いて一意に決まる．そのため，$R_i = R_j$ $(i \neq j)$ のとき，$R_{i+1} = R_{j+1}$ となる．これは一度衝突が発生すると，これ以降生成される有理点もすべて衝突することを意味する．

Algorithm 1: Pollard の Rho 法

Input: $P, Q(= [s]P) \in E(\mathbb{F}_p)$ $(s \in \mathbb{Z}_r)$

Output: s

```
for i = 0 to n − 1 do
    Two random elements in Z_r are assigned to a_i, b_i
    R_i ← [a_i]P + [b_i]Q
Two random elements in Z_r are assigned to a_n, b_n
R_n ← [a_n]P + [b_n]Q
for i = n + 1 to r − 1 do
    l ← η(R_{i−1})
    a_i ← a_{i−1} + a_l, b_i ← b_{i−1} + b_l, R_i ← R_{i−1} + R_l
    if R_i = R_j (0 ≤ j ≤ i) then
        go out this loop
s ← −(a_i − a_j)/(b_i − b_j) (mod r)
```

[*1] 厳密ではないが，楕円曲線 E 上の 2 点 A, B の加算は直線 AB と E のもう一つ交点の，y 座標の符号を反転したものと定義される．

2.3 Rho 法の高速化

まず，並列化により Rho 法の高速化を図る方法について述べる．複数の計算機を用いる場合，Algorithm1 の 1–3 行目で生成されるテーブルと 7 行目の関数 η を各計算機で共通に用い，4, 5 行目で生成される有理点を各計算機で異なるようにすれば，各計算機で生成された有理点はすべて衝突判定に利用することができる．したがって同じ性能を有する M 台の計算機を用いると，M 倍の高速化が可能である．その際，生成された有理点は M 台の計算機で共有する必要がある．Rho 法の実装では（2）における有理点 R_i の x 座標データ [*2] とスカラ値 a_i, b_i を保持すればよい．

ここで，攻撃対象の r が 100 bit の数である場合，このままでは R_i を $\sqrt{\pi r/12} \simeq 2^{49}$ 個程度保持する必要がある．1 つの有理点情報を 100[B] と仮定すると，この分量は約 50[PB] の記憶容量に相当する．この分量を大幅に削減し，衝突判定に要する時間を短縮する方法として Distinguished Point Method が提案されている [6], [7]．この方法では，有理点に容易にチェック可能な特定の性質を定め，その性質に合致した特徴点（distinguished point）のみを保持し，特徴点同士で衝突判定を行う手法である．本稿での攻撃では，x 座標がある値 $\theta = 2^k$ で割り切れる点を特徴点とした．これにより保持される点の個数は $1/\theta$ に削減され，これに伴い衝突検出に要する時間も短縮できる．たとえば，攻撃対象の r を 100 bit，θ を 2^{20} とすると，有理点数は約 2^{29} 個，記憶容量は約 50[GB] に削減できる．しかし，θ を大きくし過ぎると特徴点の数が減りすぎ，特徴点同士では衝突が発生しない（特徴点でない有理点のみで衝突が発生する）可能性がある点に注意する必要がある．

3. Dynamic DNS を用いた衝突判定

本節では Rho 法の衝突点の検出に Dynamic DNS を用いる方法を説明する．

3.1 Dynamic DNS の利用

DNS は主にインターネット上のホスト名や電子メールの宛先となるドメイン名に対して，その IP アドレスやメールサーバとの対応付けを管理するために用いられている．また，逆に IP アドレスに対して，そのホスト名との対応付け（逆引き）を管理するためにも用いられている．以前の DNS ではこれらの対応付けは静的に管理され，DHCP で IP アドレスを動的に割り当てられるホストに関してはその名前と IP アドレスの対応付けを登録，管理することは困難であった．このような対応付けの動的な更新を可能にするための仕組みが Dynamic DNS である．

Dynamic DNS では更新メッセージをプライマリ DNS に送ることで資源レコードの動的な更新を行う．その際，資源レコードを更新するために必要な前提条件（prerequisite）を指定することができる．この前提条件が満たされない場合にはエラーとなる．指定可能な前提条件は以下の 5 通りである（名称は nsupdate コマンド中での指定方法に準じる）．

(1) nxdomain ドメイン名
　指定されたドメイン名が存在しない．
(2) yxdomain ドメイン名
　指定されたドメイン名が存在する．
(3) nxrrset ドメイン名 [クラス] タイプ
　指定されたドメイン名，クラス，およびタイプが一致する資源レコードが存在しない．
(4) yxrrset ドメイン名 [クラス] タイプ
　指定されたドメイン名，クラス，およびタイプが一致する資源レコードが存在する．
(5) yxrrset ドメイン名 [クラス] タイプ データ
　指定されたドメイン名，クラス，タイプおよびデータが一致する資源レコードが存在する．

Rho 法を用いた攻撃では，新しく生成した有理点がこれまでに生成した有理点と衝突すれば攻撃成功となるため，生成した有理点の x 座標をラベル（ドメイン名のうち「.」で区切られた部分）として用い，上記の前提条件のうち nxdomain を指定して登録すれば，有理点の衝突を同時に検証することが可能になる．すなわち，更新メッセージに対する応答メッセージ中の応答コード（RCODE）が NOERROR(0) であれば登録が正常に完了したことを意味し，YXDOMAIN(6) であれば衝突が発生したことを意味する．また，Dynamic DNS ではデータの更新処理のアトミック性が保証されているため，有理点の生成が並列化されたされた場合でも正しく衝突を検出できる．

Rho 法では 2.2 節で述べたように一度衝突が発生するとこれ以降に生成される有理点もすべて衝突するという特徴を持つ．したがって，有理点の登録や衝突検証が多少失敗しても攻撃成功までの時間はあまり変わらない．そこで，各計算機と DNS サーバとの通信は信頼性を保証しない代わりにオーバヘッドが小さい UDP を用いることにした．

3.2 有理点情報のエンコード

衝突判定では有理点 R_i の x 座標データをラベルとして用い，スカラ値 a_i, b_i をドメイン名に対応する TXT レコードデータとして DNS サーバに登録することにした．この場合，攻撃対象の r を 100 bit の数だとすると，R_i の x 座標，a_i, b_i の値も高々 100 bit となり，これらを 16 進の文字列として表す場合にはそれぞれ 25 文字で表現できることになる．しかし，この場合には 1 文字で 4 bit 分の情報しかなく，効率が悪い．そこで 1 文字で表せる情報量を増やす方法を検討した．

DNS のラベルは一般にドメイン名やホスト名の一部と

[*2] 群構造を利用した改良 [8] により，x 座標が一致すれば y 座標の一致を確認しなくても衝突と判定できる．

```
0&!\\j:]0/4        TXT    "!=9w]&5$_5,|g!xy8s_13"
0&\"tz\"‘\)?8      TXT    "4!{<=‘2l)8,hy}9:2rp18"
0&\$w&yzsz7        TXT    "[11\"0j0d\;,9m\"o%&#xa1"
0&’5su3p%7         TXT    "7j\;u=5ezx1,i{6%6m%564"
0&-^e&79t3         TXT    "]5i)i4y\\s5,\"/:u850sr8"
0&-d0^jbo5         TXT    "7^1c{}oui7,6x]#rp8w75"
```

図 1 Dynamic DNS で登録されたゾーンデータの例
Fig. 1 Example of zone data registered by Dynamic DNS.

して用いられるため，RFC1034[2] では英字で始まり英数字で終わる，英数字と “-” からなる文字列の使用が推奨されている．また，ドメイン名の比較は大文字と小文字の区別を行わないことが示されている．RFC2673[9] ではビット列でのラベル指定が認められていたが，RFC6891[10] では使用が禁止された．そこで今回作成した攻撃プログラムでは，効率と可読性を両立できるように，ASCII 文字のうち制御文字と英大文字を除いた 69 文字の中から 64 文字を選び，1 文字で 6 bit の情報を表せるようにした．これにより攻撃対象の r が 100 bit の数である場合，R_i の x 座標，a_i, b_i をそれぞれ 17 文字で表現できる．

例として，R_i の x 座標，a_i, b_i がそれぞれ 58 bit の場合における，Dynamic DNS で登録されたゾーンデータの例を**図 1** に示す．なお，それぞれの値は 10 文字 [*3]（60 ビット分）で表され，また a_i, b_i の区切りには ‘,’ が使われている．たとえば，一番上の資源レコードのラベル「0&!\\j:]0/4」は 0x013b034d93fa4a00 を表す．ここで，最初の文字の ‘0’ が最下位の 6 bit を表し，最後の文字の ‘4’ が最上位の 6 bit を表す．その理由については後述する．

3.3 登録データの集約

一般に Dynamic DNS を用いれば，正引きと逆引きの両方の情報を 1 つの更新メッセージで DNS サーバに登録することができる．このことから，1 つの更新メッセージで複数の有理点を登録することが可能であることがわかる．多数の有理点を 1 つの更新メッセージで登録すると，通信のオーバヘッドを減らす効果が期待できる．従来の DNS では UDP で扱えるパケットサイズの上限は 512 オクテットに制限されていたが，拡張機能 EDNS0[10] を用いれば最大 64[KB] まで拡大できる．ただし，最大パケットサイズにあまり大きな値を指定すると断片化や再構成のオーバヘッドが大きくなるため，最大値として 4,096 オクテットが推奨されている．この場合，攻撃対象の r が 100 bit の数だとすると 1 つの更新メッセージで約 50 個の有理点を登録，衝突検証できることになる．

[*3] エスケープ文字として使われる ‘\’ を除く．

3.4 複数 DNS サーバ使用時のゾーン分割

DNS サーバを複数台用意する場合，これらの間で有理点の空間を分割する必要がある．これを実現するには，R_i の x 座標を表現するラベルの途中に ‘.’ を挿入し，‘.’ の右側の文字列に対応する各ゾーンに対応する DNS サーバを決定すればよい．

たとえば，図 1 の場合，「0&!\\j:]0/4」に対して「0&!\\j:]0/.4」のようにラベルの最後の 1 文字の前に ‘.’ を挿入すると，‘.’ の右側の文字（列）の種類は 64 通り存在するため，64 ゾーンに分割して 64 台の DNS サーバで有理点データを管理することができる．たとえば DNS サーバの台数が 32 台の場合には，各 DNS サーバに 2 つのゾーンを管理させることもできる．このように DNS サーバの台数は柔軟に変更することが可能である．

図 1 の例では 2.3 節における θ を 2^9 としているため，有理点の x 座標データの下位 9 bit はすべて 0 である．このような場合，座標データの最下位 6 bit をラベルの最後の文字に対応付けると，ラベルの最後の文字だけでは座標空間を分割できないことになる．このような状況を事前に回避するため，3.2 節の最後で述べたように最上位 6 bit は最初の文字に，最下位 6 bit はラベルの最初の文字に対応するようにエンコードしている．

4. 性能評価実験

Dynamic DNS により有理点を登録する方法を用いた場合の Rho 法の性能を評価するため，楕円曲線暗号を攻撃するプログラムを実装し，更新メッセージ当たりの有理点数を変化させ，衝突が発生するまでの時間を測定した．本節ではその結果を示す．

4.1 実験環境およびパラメータ

性能評価実験は表 1 に示す計算機を用いて行った．この実験ではクライアント，サーバとも 1 台ずつであり，2.3 節で述べた並列化は行わなかった．攻撃プログラムは 2.2 節で説明した Rho 法において群構造を利用した改良を行ったものを用い，パラメータは表 2 に示す値を用いた．

4.2 実験結果と考察

1 つの更新メッセージに含まれる有理点数を変化させた場合において，実際に衝突が発生するまでの時間を測定した．その結果を図 2 に示す．なお，EDNS0 での最大パケット長の推奨値 4,096 オクテットに収まる範囲での，1 つの更新メッセージに含まれる有理点数の最大値は 71 であった．

この結果から，全体的な傾向として更新メッセージ当たりの有理点数が小さいと攻撃時間が大きくなり，更新メッセージ当たりの有理点数がある程度大きくなると，攻撃時間が 20 秒程度に落ち着くことがわかる．更新メッセージ

表 1　実験環境
Table 1　Experimental environment.

クライアント	OS	Windows 7 Professional(64bit)
	CPU	Intel Core i7-3770 (3.40GHz)
サーバ	OS	CentOS 6.7(64bit)
	CPU	Intel Core 2 Duo E7400(2.80GHz)
	DNS	BIND 9.8.2rc1-RedHat-9.8.2-0.37.rc1.el6_7.5

表 2　実験パラメータ
Table 2　Parameters used in the experiment.

項目	値
ECDLP サイズ [bit]	58
群位数 r	164,337,598,803,413,881
2.3 節における θ	2^{12}
衝突までに生成される有理点数の期待値	207,421,028

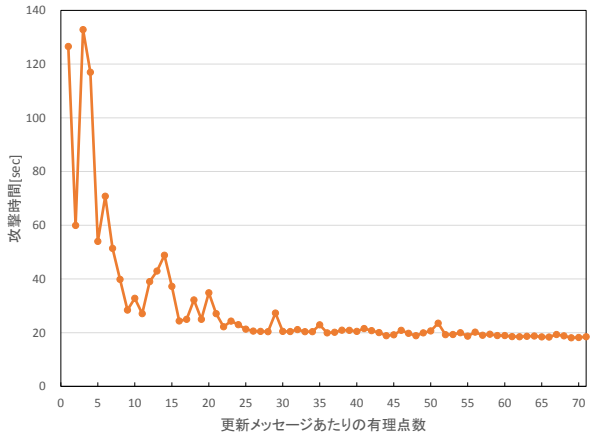

図 2　更新メッセージ当たりの有理点数と攻撃時間の関係
Fig. 2　Attack time against the number of rational points per update message.

当たりの有理点数が 29，35，51 の個所で小さなピークが見られたため，パケットサイズの MTU（Maximum Transfer Unit）超過により発生する断片化・再構成のオーバヘッドの可能性を疑ったが，3 か所でのパケットサイズはそれぞれ 1,666~1,682 オクテット，2,007~2,023 オクテット，2,913~2,934 オクテットであり，有理点数を 1 つ増やすとパケットサイズは 57 オクテット前後しか増加しないため，無関係であると思われる．Rho 法では本質的に衝突が発生するまでに生成する必要がある有理点数にばらつきが生じるため，その影響によるものと思われる．

5. まとめ

本稿では一般的な ECDLP を Pollard の Rho 法を用いて攻撃する際に生成される膨大な有理点情報を，Dynamic DNS の仕組みを用いて複数の DNS サーバで分散して効率的に登録，衝突検証を行う方法について述べた．また，各 1 台のクライアントと DNS サーバを用いて性能評価を行い，多数（25 程度以上）のデータを 1 つのパケットにまとめると効率的であることを示した．

今後の課題としては，複数の DNS サーバを用いた場合での性能評価が挙げられる．また，2.3 節で述べた Distinguished Point Method を用いた場合における，θ の値を変化させた場合の性能評価も挙げられる．しかし，本稿での考察の過程で θ を 2^{20} まで大きくすると，攻撃対象の r が 100 bit の場合には，記憶容量は約 50[GB] に削減できることが示されたため，この程度であれば広域分散データベースを導入する必要がないとも考えられる．しかし，現時点の ECC の攻撃成功記録と同じ 112 bit[11] を攻撃対象とする場合，θ が同じであれば必要な記憶容量は約 205[TB] と膨大になるため，有理点の Dynamic DNS による登録，衝突検証の有効性について認められる余地が残されていると考えている．

謝辞　本研究の成果の一部は「科学研究費補助金：基盤研究（B）（25280047）『楕円ペアリング暗号に対する共役有理点ノルムを用いた分散並列攻撃法の開発と実証実験』」の助成による．

参考文献

[1] J. Pollard, “Monte Carlo Methods for Index Computation (mod p),” Math. Comp, vol. 32, pp. 918–924, 1978.
[2] P. Mockapetris: DOMAIN NAMES - CONCEPTS AND FACILITIES, RFC1034, IETF, 1987.
[3] P. Mockapetris: DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION, RFC1035, IETF, 1987.
[4] P. Vixie (ed), S. Thomson, Y. Rekhter, J. Bound: Dynamic Updates in the Domain Name System (DNS UPDATE), RFC2136, IETF, 1997.
[5] 三好俊介，野上保之，山井成良：“楕円曲線暗号における DNS を用いた衝突判定”，電子情報通信学会第 36 回情報理論とその応用シンポジウム (SITA2013) 予稿集 (CD-ROM)，発表番号 1.4.1，pp.51–54，2013 年 11 月．
[6] 三好俊介，山井成良，野上保之：“BN 曲線上の ECDLP に対する Rho 法の DNS を用いた衝突検出の性能評価”，2014 年暗号と情報セキュリティシンポジウム (SCIS2014) 予稿集 (CD-ROM)，電子情報通信学会，発表番号 4F2-4，

2014 年 1 月.
[7] 三好俊介，野上保之，日下卓也，山井成良: “70 台程度の計算機を並列に用いた 94bit の ECDLP の解読”，2015 年暗号と情報セキュリティシンポジウム (SCIS2015) 講演論文集，2B4-1，電子情報通信学会，2015 年 1 月.
[8] 森佑樹，角力大地，野上保之，松嶋智子，上原聡: “Barreto-Naehrig 曲線上のある特殊な巡回群に対する Frobenius 写像を用いた ρ 法による攻撃の実装評価”, 電子情報通信学会第 34 回情報理論とその応用シンポジウム (SITA2011) 予稿集 (CD-ROM)，発表番号 6.3.1，pp.370–375，2011 年 11 月.
[9] M. Crawford: Binary Labels in the Domain Name System, RFC2673, IETF, 1999.
[10] J. Dames, M. Graff, P. Vixie: Extension Mechanisms for DNS (EDNS(0)), RFC6891, IETF, 2013.
[11] Ecole polytechnique fédérale de Lausann: “112-bit prime ECDLP solved — LACAL (online)”, available from ⟨http://lacal.epfl.ch/112bit_prime⟩ (2016.02.02).