検索可能暗号の安全性再考

菅 孝徳 †　　西出 隆志 ‡　　櫻井 幸一 ‡

† 九州大学大学院システム情報科学府
819-0395 福岡県福岡市西区元岡 744
suga@itslab.inf.kyushu-u.ac.jp

‡ 九州大学大学院システム情報科学研究院
819-0395 福岡県福岡市西区元岡 744
nishide@inf.kyushu-u.ac.jp, sakurai@csce.kyushu-u.ac.jp

あらまし データの暗号化と検索を両立可能な検索可能暗号と呼ばれる暗号がある．この検索可能暗号では，安全性モデルに基づき安全性が証明されているものの，ファイルの内容を表すキーワードを単語単位で暗号化する為，構成によって頻度分析により解読される危険性が存在する．この攻撃は，暗号文の出現頻度を測定し，推測される単語の出現頻度と比較することで平文を推測しようとするものである．本研究では，確定的暗号と確率的暗号のどちらにより構成されているか，また，対応する検索方式により頻度分析に対する耐性の違いについて検討する．更に，既存方式の頻度分析への耐性を検討し，検索可能暗号の頻度分析への耐性について考察を行う．

Security of Searchable Encryption Revisited

Takanori Suga†　　Takashi Nishide‡　　Kouichi Sakurai‡

†Graduate School of Information Science and Electrical Engineering, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, JAPAN
suga@itslab.inf.kyushu-u.ac.jp

‡Faculty of Information Science and Electrical Engineering, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, JAPAN
nishide@inf.kyushu-u.ac.jp, sakurai@csce.kyushu-u.ac.jp

Abstract There are encryption schemes called searchable encryption schemes. We can prove the security of these schemes based on a security model. However, we face the risk of the frequency analysis in some construction because these schemes encrypt the keywords per word in the file as well as itself. In this attack, an adversary measures the frequency of the ciphertext and guesses the plaintext by comparing the frequency of the plaintext with it. In this work, we consider this attack to the existing works and consider the resistance of the searchable encryption to this attack by considering the difference between the deterministic ones and the probabilistic ones, and the supported searches.

1 はじめに

1.1 研究背景

近年，計算機の演算能力や通信技術が向上し，クラウドコンピューティングの普及が急速に進んでいる．クラウドコンピューティングでは，データを外部に預託するだけではなく，処理もサーバに行わせる．クラウド上でセキュリティ上の事故が発生した場合，ユーザ側でクラウド事業者の管理体制を知ることが出来ない為，ユーザ側では対策を行うことが出来ない．その為，クラウドにデータを送信する前に暗号化を行うなどの対策を行う必要がある．しかし，通常の暗号を用いた場合，サーバが平文の内容を知らない為，サーバ上で検索を行うことが出来なくなってしまう．そこで，近年では暗号化されたデータを復号することなく検索を行うことを可能にした検索可能暗号と呼ばれる暗号が多数提案されている．既存の検索可能暗号に対するモデルでは，頻度分析に対する耐性が考慮されていない．攻撃者にとって，インデックス中のキーワードの確率分布が既知の時，キーワードに対するインデックスが確定的である場合には，インデックスの頻度分布とキーワードの頻度分布を比較することで，元のキーワードを推測することが出来る．本研究では，既存方式においてどのように行うことが出来るのか，また，どのような検索可能暗号を構築すれば安全か，という点に着目して検索可能暗号の頻度分析に対する安全性を再考する．

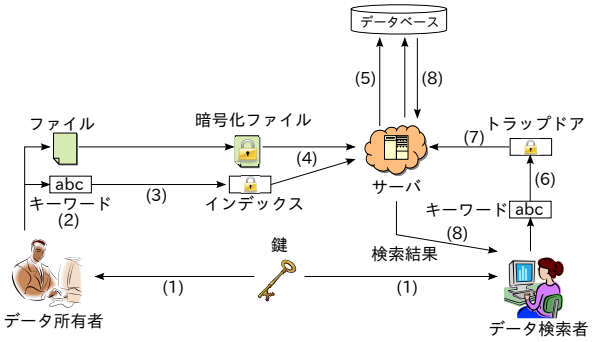

図 1　Flow of Searchable Encryption

2　既存研究

2.1　検索可能暗号

図 1 に 共通鍵ベースの検索可能暗号の概略を示す．

1. データの所有者と，検索を行うユーザの間で，事前に鍵の共有を行っておく．
2. データの所有者は，暗号化したデータの内容を表すキーワード $\mathrm{Kw_A}$ を指定する．このキーワードは複数指定することも出来る．
3. $\mathrm{Kw_A}$ のインデックス $\mathrm{Idx_A}$ を計算する．ここで，インデックスはキーワードの暗号文に相当するデータのことである．
4. 暗号化したデータ C_A と $\mathrm{Idx_A}$ をサーバに送信する．
5. サーバは C_A と $\mathrm{Idx_A}$ をデータベースに登録する．
6. データの検索を行うユーザは，検索したいキーワード $\mathrm{Kw_B}$ のトラップドア $\mathrm{Td_B}$ を計算する．ここで，トラップドアは検索したいキーワードの暗号文に相当するデータのことである．
7. $\mathrm{Td_B}$ をサーバに送信する．
8. サーバは，データベースから $\mathrm{Td_B}$ にヒットするデータを検索し，検索を行ったユーザに結果を返却する．

検索可能暗号において，$\mathrm{Idx_A}, \mathrm{Td_B}$ は，$\mathrm{Kw_A}, \mathrm{Kw_B}$ に復元することができないように，暗号化されている．検索可能暗号では，このようにサーバにさえキーワードを知らせることなく，キーワードのプライバシーを保護したまま検索をすることが出来る．

公開鍵ベースの検索可能暗号については，鍵の共有をデータ検索者が生成した公開鍵をデータ所有者に渡すと読み替えればよい．

2.2　頻度分析

頻度分析は古典暗号，特に単一換字式暗号で問題とされていた攻撃手法である．単一換字式暗号では，平文の 1 文字の出現頻度と暗号文中の対応する 1 文字の出現頻度が等しくなる為，暗号文中の文字の出現頻度を測定し平文に想定される文字の出現頻度と比較することで暗号文を復号することが出来た．例えば，一般に英語の文章中では “e” が最も多くなると言われており，単一換字式暗号による暗号文で最も多く現れる文字が “e” になると推測できる．

現代暗号においては対策が進んでおり，問題とされることが少ないが，検索可能暗号においてはキーワードの取りうるパターンが限られており，キーワードとインデックスが一対一の関係にある場合，同様の問題が生じる恐れがある．

2.3　既存の検索可能暗号

2000 年に Song らにより，初めて実用的な共通鍵暗号方式の検索可能暗号が提案された [9]．その後，様々な共通鍵暗号方式の検索可能暗号が提案されている [1, 4, 5, 10]．近年では，Li らにより類似検索に対応する共通鍵暗号方式の検索可能暗号が提案されている [7]．また，我々はキーワード単位ではなく，文字単位での検索に対応する共通鍵暗号方式の検索可能暗号が提案している [12]．

2004 年に Boneh らにより公開鍵暗号方式の検索可能暗号の概念が提案された [3]．その後，様々な公開鍵暗号方式の検索可能暗号が提案されている．また，2011 年に Sedghi らによりワイルドカードに対応する公開鍵暗号方式の検索可能暗号が提案されている [8]．

また，後述するが，頻度分析への攻撃を考慮した検索可能暗号として，Bellare らの方式 [2] や伊藤らの方式 [13] などがある．

2.4　既存安全性モデル

既存の安全性モデルとしては，以下のようなものが存在する．

まず，Goh らにより提案された IND-CKA，IND2-CKA[6] がある．この安全性モデルは，共通鍵の検索可能暗号に対する安全性モデルであり，攻撃者に既知の 2 つのキーワードの集合から作られたインデックスを攻撃者が区別できるか，というゲームによって定義される．2 つのキーワードの集合のキーワード数が等しいと制限したものが IND-CKA であり，異なっても良いと制限を緩和し，攻撃者により有利にしたのが IND2-CKA である．

これらの安全性モデルにおいては，検索対象となるインデックスの安全性のみ保証しており，検索要求に用いられるトラップドアの安全性については保証されない．その為，長沼らは検索要求の秘匿性も保証する安全性モデルとして，公開鍵暗号を対象とする IND-CKQA 及び IND-CKQA$^\natural$ を提案している [11]．

これらの安全性モデルでは，頻度分析を考慮していない．より安全な検索可能暗号を考える上では頻度分析についても，考慮する方が望ましい．

3 確定的暗号と確率的暗号

同一の平文に対して，2 回暗号化を行うことを考える．ここで，それぞれの暗号文が必ず等しくなるとき，この方式を確定的暗号 (Deterministic encryption) と呼び，確率的に異なるとき，この方式を確率的暗号 (Probabilistic encryption) と呼ぶ．

まず，単純に確定的暗号を使って検索可能暗号を構成した場合に頻度分析攻撃が行われうることを示す．

共通鍵確定的暗号 $\Pi_{\mathrm{d}} = (\mathrm{Gen_d}, \mathrm{Enc_d}, \mathrm{Dec_d})$ を利用し，キーワードを $\mathrm{Enc_d}(keyword)$ のように暗号化し，検索するユーザも同様に暗号化し，それぞれの暗号文が一致するものをサーバで検索する，という検索を考える．

ここで，攻撃者がキーワードの出現頻度を知っていた場合，例えば，攻撃者がキーワードとして都道府県が利用されることを知っていて，“東京都” が一番多くなる，と推測した場合を考える．“東京都” が実際に最多となる場合は，$\mathrm{Enc_d}$(“東京都″) が最多となり，サーバに登録された暗号文の出現頻度を調べることで，$\mathrm{Enc_d}$(“東京都″) が “東京都” というキーワードに対する暗号文であると知ることが可能である．

この議論は各キーワード毎にインデックスを生成する場合にのみ適用可能である．複数のキーワードをまとめて一つのインデックスとして生成する場合には，適用できない．複数のキーワードをまとめた場合には，インデックスの頻度が用いたキーワードの頻度すべてに影響され，単独のキーワードの頻度からは推測できない為である．この場合には，別のアプローチで検討する必要がある．

ここで，既存の検索可能暗号について述べる．Li らは編集距離が一定以内のものがヒットするような類似検索を可能にする確定的暗号の検索可能暗号を提案している [7]．この方式では，類似検索に必要なキーワードを提案された変換規則に基づいて列挙し，擬似乱数関数の出力をデータを登録するユーザと検索するユーザの両者が送信することで，類似検索を実現している．擬似乱数関数とは，乱数関数と計算量的に区別が困難な関数のことであり，実用的には暗号学的ハッシュ関数を利用することが出来る．しかし，変換規則により列挙されるキーワードの組は一定であり，また擬似乱数関数の出力も同じ入力に対しては同じ出力となる．つまり，前述の頻度分析による攻撃が可能となる．

一方で確率的暗号を用いた場合には，同じ入力に対して異なる入力が得られることから，同様の構成を行った場合にも，後述のようにトラップドアを考慮しない限り，一般に頻度分析のリスクは存在しない．

しかし，確定的暗号を用いる方が，サーバでの検索時間の点で有利である．データサイズを n とするとき，確率的暗号を用いると全数探索を行う必要があり，$\mathcal{O}(n)$ の検索時間を要するのに対し，確定的暗号を用いる場合はその必要がなく，$\mathcal{O}(\log n)$ ですむと伊藤らは指摘している [13]．

確定的暗号による検索可能暗号

確定的暗号を用いて安全な検索可能暗号を構築しようとする試みとしては，Bellare らの方式がある [2]．

この方式は，公開鍵暗号方式の検索可能暗号であり，確定的暗号の下で可能な限り強い安全性の定義を主張している．伊藤らはこの方式がキーワード空間が十分に大きい，理想的な環境を仮定しており，実システムでは耐性を持たないと指摘しており，同音換字暗号の考え方を応用した，頻度分析に耐性を持つ方式を提案している [13]．

この方式は，各キーワードを出現頻度の大きさに応じた数の記号を割り当て，頻度をほぼ均一化するという手法をとっている．

例えば，キーワードとして，70% の確率で “dog” が出現し，30% の確率で “cat” が出現する場合は，“dog” に対し，“dog1,” $\cdots$, “dog7” の 7 個の記号を，“cat” に対し，“cat1,” $\cdots$, “cat3” の 3 個の記号を割り当て，暗号化の際には割り当てられた記号をランダムに選択し暗号化することで頻度を均一化する．

検索時には検索を行うユーザが割り当てられた全ての記号の暗号文をサーバに送信し，サーバはそのいずれかの暗号文と一致する検索結果を返却する．

これにより，各記号に対応する暗号文の出現確率はそれぞれ 1/10 に均一化され，頻度分析を行うことができなくなる，という主張である．

この方式は，一度も検索が行われていない，または，検索結果を攻撃者が入手できないという条件下では，筆者らの主張通り，頻度分析に耐性を持つが，十分に検索が行われた状態かつこの結果を攻撃者が入手できるとすると，頻度分析への耐性を失ってしまう．この条件には，攻撃者がクラウドの事業者である場合などが該当する．

前述の例では，“dog” の検索が行わる際にサーバに “dog1,” $\cdots$, “cat7” の全ての暗号文がサーバに送信される．これにより，サーバは “dog1,” $\cdots$, “dog7” の暗号文がすべて同じ，キーワードから生成されていることを知ることが出来る．

ここから，確定的暗号を用いて頻度分析に耐性をもたせる為に乱数化を行う場合，サーバに乱数化された暗号文が一致していることを知られないように注意して，構築しなければならないことが分かる．

4 検索方法による頻度分析のリスク

最も基本となる検索に完全一致検索と呼ばれる検索がある．この検索は，名前の通り，検索時に与えられたキーワードと完全に一致するキーワードを持つものを検索するものである．

この完全一致検索のみに対応する検索可能暗号は確定的

暗号と確率的暗号のどちらを使っている場合においても，誤りなく必ず正しい検索結果が得られる場合，検索が十分に行われた時点で頻度分析の可能性が発生する．

乱数化されていても，そうでなくても，また，どのように乱数化されていたとしても，完全一致検索で得られる検索結果は，全て等しいキーワードであるからである．

検索結果として得られた組は全て同じキーワードの組であるから，この組を全て同じキーワードとして，出現回数を測定することで頻度分析が可能となる．

部分一致検索などその他の検索方式に対応する場合は，検索検索結果の組が一定とならず，完全一致検索のみに対応する検索可能暗号と比べ，頻度分析に耐性を持たせやすいと考えられる．しかし，完全一致検索がなされたことをサーバ側が知ることが出来る場合，同様に，完全一致検索が十分になされたとき，頻度分析を行うことが出来る．

例えば，Sedghi らの方式 [8] は，双線形写像を用いた確率的暗号として構築されており，同じキーワードに対するインデックスは異なるものとなる．また，“2011/10/**” で “2011/10/19” などをヒットさせるなど，特殊記号で任意の文字にヒットさせるワイルドカード検索を可能としている．また，しかし，この方式ではワイルドカードの特殊記号の位置の集合をそのままサーバに送信しており，この集合が空集合であるとき，完全一致検索がなされたと知ることが出来る．つまり，前述の通り，完全一致検索が十分になされたとき，この方式に対して頻度分析攻撃が可能になる．

5 既存方式への頻度分析による攻撃可能性

一部の既存方式に対して，前述の頻度分析による攻撃を試みる場合の攻撃可能性を考える．攻撃に用いる情報としてのトラップドアの有無と，その時の攻撃可能性を，表 1 に示す．

ここで，インデックスのみとは攻撃者が十分な量のインデックスを利用可能とし，トラップドアありとは攻撃者が十分な量のインデックスのみならず，十分な量のトラップドアとその検索結果も利用可能であるとする．

ここで Goh らの方式 [6] については，複数のキーワードを一括して 1 つのデータとしてサーバに登録しており，個別のキーワードに分割できないことから，インデックスのみで頻度を分析することは困難である．また，ブルームフィルタのパラメータによって大小は異なるが，検索結果には偽陽性が含まれることから，頻度分析を行った場合にも確定的には推測できず，結果が確率的になってしまう．

6 終わりに

本論文では，既存の検索可能暗号に対するモデルで考慮されていない頻度分析に対する耐性について，検索可能暗号を確定的暗号と確率的暗号のどちらで構築するかという点，及び，検索の種類として完全一致検索のみに対応するか，または完全一致検索を行ったことをサーバに知られてしまうような検索可能暗号を構築した場合に生じる問題について考察した．

インデックス中に複数のキーワードを統合している場合や，完全一致検索以外の機能を持ち，完全一致検索をした場合にそうであるとサーバに秘匿できる場合の安全性などについて検討を行うこと，また，より強い検索可能暗号を考える上で頻度分析に対する耐性を考慮した安全性モデルの検討などが今後の課題として挙げられる．

謝辞

本研究の一部は，日本学術振興会 科学研究費補助金 若手研究 B (23700021) による補助のもとで行われた．

参考文献

[1] F. Bao, R. H. Deng, X. Ding, and Y. Yang. Private query on encrypted data in multi-user settings. *ISPEC'08*, pp. 71–85, 2008.

[2] M. Bellare, A. Boldyreva, A. Deterministic and efficiently searchable encryption. *CRYPTO 2007*, pp. 535–552, 2007.

[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *EUROCRYPT 2004*, Vol. 3027 of *LNCS*, pp. 506–522. Springer, 2004.

[4] Y.-C. Chang and M. Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In *ACNS'05*, Vol. 3531 of *LNCS*, pp. 391–421. Springer, 2005.

[5] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. In *CCS' 06*, pp. 79–88. ACM, 2006.

[6] E.-J. Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. `http://eprint.iacr.org/2003/216/`.

[7] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy keyword search over encrypted data in cloud computing. In *INFOCOM'10*, pp. 1–5, March 2010.

[8] S. Sedghi, P. van Liesdonk, S. Nikova, P. Hartel, and W. Jonker. Searching keywords with wildcards on encrypted data. In *SCN 2010*, Vol. 6280 of *LNCS*, pp. 138–153. Springer, 2010.

表 1　既存方式の頻度分析による攻撃可能性

方式	安全性証明	インデックスのみ	トラップドアあり
Goh らの方式 [6]	あり	不可能	可能 *
Li らの方式 [7]	あり	可能	可能
Sedghi らの方式 [8]	あり	不可能	可能
Song らの方式 [9]	なし	可能	可能
伊藤らの方式 [13]	なし	不可能	可能
我々の方式 [12]	なし	可能	可能

表 2　インデックス生成元のキーワードと出現回数

キーワード	ハッシュ値	出現回数
ekpweles	a95f84f862081914d1c034617226ad4988388155aa7f9fbebbac87616a32cbea	99
didacticity	f3ed85edd29f89c65b1156dc2fb21d5d28ef7c31bfadc7cc50c77940b5cfa14d	95
prochronism	316e591a205c22e7f8141eb3995631a9980f4dd55a2cbc785f07cced1816d149	82
tahona	5069063bd9bf21ccef899707695268aed7e25631cd330364e2e2f3d0e9cba129	76
angiopressure	3ba84aed8f86601b99a8de396e3d27a0b97fc8ca55251e8b7c9eeecb70af3e4d	73
prosabbatical	526c4fab1cb7c366e57f2a554d98b5b92cbb3cc3f7b172fdf14e0028e056ae92	60
indestructible	d01b297b6bc892eceb98c519d08b953ad30d0bfd4fea8dd40173db33ce5d69a4	52
Cybil	49a0c8c76b0e4c87e31db76d28ebbf3fb39cc9329855c6800fa78ae4ee0e3759	49
larderellite	c4b15fec1310cdb90167a46c650c0439b8eb7e8d9263772f847cd3aec1ce3675	46
Maimonist	f14605f3c25d9a76d2081309c2a2d4dc0a7c0f51d7b56ef3802ea268480f301b	32

[9] D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *IEEE Security and Privacy 2000*, pp. 44 –55, 2000.

[10] B. Waters, B. R. Waters, D. Balfanz, D. Balfanz, G. Durfee, G. Durfee, D. K. Smetters, and D. K. Smetters. Building an encrypted and searchable audit log. In *NDSS'04*, 2004.

[11] 長沼, 吉野, 佐藤. ＤＢ向け検索可能暗号方式の検討 (1). 暗号と情報セキュリティシンポジウム 2011, 2011.

[12] 菅, 西出, 櫻井. ブルームフィルタを用いた検索自由度の高い検索可能暗号の設計と実装評価. 電子情報通信学会技術研究報告, 111(30):111–116, 2011.

[13] 伊藤, 服部, 松田, 坂井, 太田. 頻度分析耐性を持つ高速秘匿検索方式. 電子情報通信学会技術研究報告, 110(443):1–6, 2011.

付録 A　Li らの方式の実験的評価

Li らの方式について頻度分析攻撃が存在することを実験的に確認した.

この実験では，CentOS 5.6 に含まれる英単語の辞書からランダムな数のキーワードを選択し，各キーワードについて頻度に偏りが出るようにした上でランダムな回数だけ，HMAC-SHA256 により暗号学的ハッシュ関数の出力を計算した.

この実験では，Li らの方式を単純化したものであり，変換規則を適用していないなどの違いは存在するが，あるキーワードから生成される変換規則を適用したキーワードの組に対応する擬似乱数関数の組が一定であることから，あるキーワードからこの擬似乱数関数の組を出力する関数を考えると，頻度分析攻撃を考える上では等価である.

実験結果としては，自明ではあるが，元のキーワードの頻度と HMAC-SHA256 の出力の頻度は完全に一致し，頻度分析は可能であることが確認された.

具体的な実験結果を 1 つ，表 2 に示す．元になるキーワードとハッシュ値及びそれらの出現回数を表している．ここで，出現回数はキーワード，ハッシュ値ともに完全に一致している為，まとめて書いている．その他のキーワードや出現回数の組でも同様の結果が得られている.

付録 B　我々の方式の実験的評価

我々の方式 [12] は，キーワード単位ではなく文字とその位置の組としてブルームフィルタに，格納することで文字単位の検索を実現している．その為，ここではキーワード単位ではなく，文字とその位置の組に対する頻度分析の実験を行った．キーワードの字数が漏洩するという問題があった為，Goh らの方式 [6] と同様に，見かけ上は固定長となるようにブルームフィルタに対してランダムにビットを立てる操作を加えている.

文字の頻度からブルームフィルタのビットの頻度がどの

表 3　擬似乱数の数を変化させた場合

	1	2	3	4	5	6	7	8	9	10
1	0.51	0.33	0.27	0.19	0.17	0.14	0.11	0.10	0.12	0.08
2		0.65	0.52	0.38	0.32	0.28	0.24	0.22	0.21	0.19
3			0.78	0.57	0.49	0.41	0.37	0.32	0.31	0.28
4				0.79	0.66	0.57	0.50	0.43	0.42	0.37
5					0.83	0.71	0.62	0.54	0.53	0.45
6						0.86	0.74	0.65	0.62	0.56
7							0.88	0.76	0.71	0.64
8								0.87	0.82	0.74
9									0.91	0.82
10										0.91

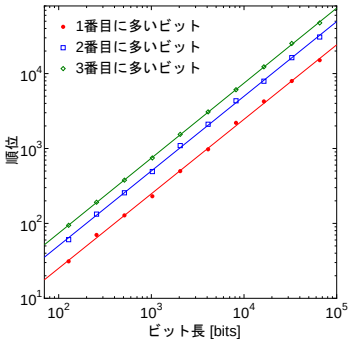

図 2　ビット長を変化させた時の頻度分析

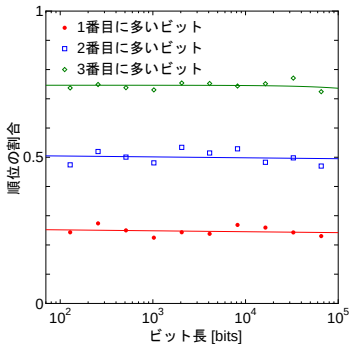

図 3　ビット長を変化させた時の頻度分析

程度現れるか測定を行った．CentOS 5.6 に含まれる辞書の全てのキーワードに対して，BuildIndex を実行することでこの実験を行った．鍵を変えながら 100 回行うことで頻度を測定している．

この辞書には，479,623 個のキーワードが含まれ，2 文字目の “e” が最頻出の位置と文字の組であった．仮に，本提案方式が頻度分析に対する脆弱性を持つのであれば，2 文字目の “e” に対応するビットが高い確率で最頻出またはそれに近い頻度で現れるはずである．

下記の実験結果から，辞書を変えた場合の挙動も今後調査すべきではあるものの，今回の実験では有意に頻度分析を行うことが出来ないことが確認できた．

B.0.1　ビット長を変化させた場合

ビット長を変化させた場合の，2 文字目の “e” に対応するビットの出現頻度を測定した．この結果を図 2, 図 3 に示す．図 2 は，100 回の試行において 2 文字目の “e” に対応するビットが平均して何番目に多いビットであるかを表しており，図 3 はこれが全ビット長の中どの程度の割合であるかを表している．両図において，1 位，2 位，3 位は 3 つのハッシュ関数から立てられるビットの頻度の順位を表している．

図 3 より，ビット長を変化させた場合，およそ 25%，50%，75% で推移していることが分かる．これは均等に 4 等分したときの境界に他ならない．

この結果は，ランダムに点を取ったときの挙動と等しい．例えば，1 から 100 の間でランダムに 3 つ選ぶという操作を繰り返すことを考える．この 3 つの数値のうち，1 番大きい値の平均は約 75 になり，2 番目に大きい値の平均は約 50 になり，3 番目に大きい値の平均は約 25 となる．

B.0.2　ハッシュ関数の数を変化させた場合

同様にハッシュ関数の数を変化させた場合の，2 文字目の “e” に対応するビットの出現頻度を測定した．表 3 は，100 回の試行において 2 文字目の “e” に対応するビットが平均して何番目に多いビットであるかを表しており，表 3 はこれが全ビット長の中どの程度の割合であるかを表している．

ハッシュ関数の数を変化させた場合にも，ビット長を変化させた場合と同様に均等に等分したときの境界に順位が現れることが分かる．