Duらのブロックチェーンベース複数キーワード公開鍵検索可能認証暗号システムに対する安全性評価

西野 陽大[1,a)]　江村 恵太[1,2]

概要：Du ら (Security and Communication Networks 2022) はブルームフィルタとブロックチェーンを用いた公開鍵検索可能認証暗号方式を提案した．ブルームフィルタは暗号化されたキーワードの検索に，ブロックチェーンは検索結果が正しいことを保証するために利用されている．本論文では，Du らの方式では暗号文からキーワードの情報が漏れていることを示す．ブルームフィルタが暗号文内に直接含まれていることに着目した．ブルームフィルタの偽陽性が発生しない確率で攻撃が成功することを示すとともに，偽陽性の発生確率を検証し提案攻撃の成功確率が十分に高いことも示す．また Du らの安全性モデルと想定利用状況について精査し，我々の攻撃の妥当性についても議論した．さらに，過去に作成された暗号文であれば，検索結果が異なっていても検証に通ること，すなわちブロックチェーンを用いた完全性検証が不十分であることも示した．

キーワード：公開鍵検索可能認証暗号，ブロックチェーン，ブルームフィルタ

Security Analysis on a Blockchain-based Public-Key Authenticated Encryption Scheme with Multi-Keyword Search

Hinata Nishino[1,a)]　Keita Emura[1,2]

Abstract: Du et al. (Security and Communication Networks 2022) proposed a public key authenticated encryption with keyword search scheme that employs bloom filters and blockchain. In this paper, we show that information of keyword is revealed from ciphertexts in the Du et al.'s scheme. The success probability of our attack is the same as the probability of false positive. We confirm that the success probability is sufficiently large under a reasonable setting. We also pointed out their blockchain-based integrity check is not sufficient by showing a case that a ciphertext passes the integrity check even the ciphertext is not a search result.

Keywords: Public-Key Authenticated Encryption with Keyword Search, Blockchain, Bloom Filter

1. はじめに

1.1 研究背景

クラウドコンピューティングでは，クラウドサーバからデータが漏洩する危険性を考慮して，クラウドサーバに保存するデータは暗号化しておくことが望ましい．クラウドサーバに暗号化データを保存する場合，データの秘匿性は担保される一方で，データの検索を阻害する．そのため，ユーザはクラウドサーバから所望のデータを得ることが困難になる．

この問題を解決するため，暗号化によりデータの内容を秘匿しながら，検索したいキーワードを含むデータを抽出可能な暗号方式である公開鍵検索可能暗号 (PEKS: Public-Key Encryption with Keyword Search) [4] が提案されている．PEKS では，受信者の公開鍵を用いてキーワードを暗号化し，検索時には受信者の秘密鍵を用いてキーワードに対応

[1] 金沢大学 Kanazawa University
[2] 産業技術総合研究所 National Institute of Advanced Industrial Science and Technology (AIST)
[a)] hinata828@stu.kanazawa-u.ac.jp

するトラップドアを作成する．検索に用いられたキーワードと，トラップドアに対応するキーワードが一致するかをテストアルゴリズムで検証する．

PEKS の問題点として，キーワード推測攻撃 (KGA: Keyword Guessing Attacks) が可能であることが挙げられる．つまり，トラップドアを持つ攻撃者を想定した場合，任意のキーワードの暗号文を生成しテストアルゴリズムを実行することで，トラップドアと選択したキーワードが対応しているかを判定することが出来る．すなわち，受信者が何を検索したのかという情報が漏洩する．プライバシーの観点から，KGA 耐性を持つことが望ましい．しかし，任意のキーワードに対して誰でも暗号化が可能な PEKS では，原理的に KGA 耐性を持たせることが不可能である．KGA 耐性を持つ方式として，公開鍵検索可能認証暗号 (PAEKS: Public-Key Authenticated Encryption with Keyword Search) [15,17] が提案されている．PAEKS では暗号化時に送信者の秘密鍵を用いることで，攻撃者による任意のキーワードに対する暗号文の取得を防いでいる．

1.2 ブロックチェーンを用いた PAEKS

検索可能暗号とブロックチェーンを組み合わせる試みは数多く行われている [2,5–14,16,18,19,22,26–28,30–39]．その中で，本論文では Du ら [9] による，ブロックチェーンを利用し検索結果の完全性を考慮した PAEKS 方式 (BB-PAEKS: Blockchain Based Public-Key Authenticated Encryption with Keyword Search) に着目する．彼らは，KGA に耐性を持つこと，ペアリングベースの PAEKS 方式と比較して計算コストを大幅に削減し高い実用性を持つこと，ブロックチェーンの改ざん耐性に基づき検索結果が正しいことを保証すること (彼らは検索結果の完全性 (Integrity) と表記)，の 3 つを方式の利点として挙げている．

1.3 研究動機

Du らの方式は，ブルームフィルタ [3] を用いて検索機能を実現している．ここで我々は，暗号文にもトラップドアにも直接ブルームフィルタが含まれている点に着目する．ブルームフィルタは検索キーワードのハッシュ値から計算されるため，同じキーワードに対しては同じブルームフィルタが生成される．そのため，ブルームフィルタからキーワードの情報が漏れていると考えられる．また，暗号文の一部のハッシュ値をブロックチェーンに保存，検索結果である暗号文から生成されるハッシュ値と同じかどうかを確認することで，検索結果が正しいことを保証すると主張している．[*1] しかし検索結果によらずハッシュ値が計算可能であることから，検索結果の完全性を保証するものではないと考えられる．

[*1] 具体的に，“*our scheme supports the integrity verification of the results. It ensures the reliability of search results by using the tamper proof characteristics of blockchain.*” と述べている．

1.4 本論文の貢献

本論文では，Du らの方式に対し，暗号文からキーワードの情報が漏洩していることを示す．前述の通り，暗号文に直接ブルームフィルタが含まれている点に着目した．ブルームフィルターに含まれるキーワードと含まれないキーワードをチャレンジキーワードとすることで，キーワードの識別可能性を破る攻撃者を構成する．なお偽陽性が発生する場合に識別に失敗することに注意されたい．そこで偽陽性の発生確率を検証し，提案攻撃の成功確率が十分に高いことも示す．なお Du らの安全性モデルでは，1 つのキーワードは 1 回のみ暗号化可能であるという制約を設けている．したがって，我々の攻撃手法はモデル外のものであることに注意されたい．しかしながら，彼らは 1 つのキーワードが複数のファイルに含まれる状況を想定している．上記制約の状況下では，あるキーワードの暗号文を使いまわすこととなる．すなわち，トラップドアを用いた検索を行うことなく複数のファイルに同じキーワードが含まれるという情報が漏洩する．これらを鑑み，我々の攻撃モデルは妥当であると結論付ける．より詳細な議論については 7 章を参照されたい．また，ブロックチェーンを用いた完全性検証が不十分であることも指摘する．具体的に，検索結果が異なる場合でも，暗号文が過去に作成されていれば検証に通ることを示す．

2. 準備

2.1 ブルームフィルタ

ブルームフィルタ [3] は，あるデータが集合の中に存在するかを判定可能なデータ構造である．本論文では Du らの論文 [9] で用いられている記法を用いる．

$\mathsf{BF.Gen}(L, J)$: ブルームフィルタの生成アルゴリズムは，ブルームフィルタのサイズ $L \in \mathbb{N}$ とハッシュ関数の数 $J \in \mathbb{N}$ を入力とする．$[L] := \{0, L-1\}$ と定義する．メッセージ $\mathsf{Meg} \in \{0,1\}^*$ を入力，k を鍵とし，$i \in [L]$ を出力するハッシュ関数 h_j の族 $\mathcal{H} = \{h_j\}_{j\in[J]}$ を定義する．$i \in [L]$ に対し，ブルームフィルタ $\mathsf{BF} \in \{0,1\}^L$ の i 番目の要素を $\mathsf{BF}[i] \in \{0,1\}$ と表記する．全ての i に対して $\mathsf{BF}[i] = 0$ と初期化し，$\mathcal{H}$ と BF を出力する．

$\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF}, \mathsf{Meg})$: ブルームフィルタの更新アルゴリズムは，$\mathcal{H} = \{h_j\}_{j\in[J]}$, $\mathsf{BF} \in \{0,1\}^L$, メッセージ $\mathsf{Meg} \in \{0,1\}^*$ を入力とする．全ての $j \in [J]$ について $\mathsf{BF}[h_j(\mathsf{Meg}, k)] \leftarrow 1$ と更新した BF を出力する．

$\mathsf{BF.Check}(\mathcal{H}, \mathsf{BF}, \mathsf{Meg})$: ブルームフィルタの確認アルゴリズムは，$\mathcal{H} = \{h_j\}_{j\in[J]}$, BF, Meg を入力とする．全

ての $j \in [J]$ について $\mathsf{BF}[h_j(\mathsf{Meg}, k)] = 1$ が成り立つ場合に 1, 成り立たない場合に 0 を出力する.

ブルームフィルタでは, 集合内に存在しないデータを存在すると判定する, 偽陽性が発生する可能性がある. 偽陽性発生確率は, J をハッシュ関数の数, K をフィルタに挿入するメッセージの数として次の式で表される [23].

$$\Pr[\mathsf{FalsePositive}] = (1 - (1 - 1/L)^{JK})^J$$

2.2 転置インデックス

転置インデックスは, データ群に含まれるキーワードの位置情報を示す索引構造である. Du ら [9] はデータとキーワードの関係性を表すインデックス行列の例として, 下記を挙げている.

表 1 インデックス行列 [9]

ID	Data	Keyword
1	d_1	w_1, w_2, w_4, w_5
2	d_2	w_1, w_3
3	d_3	w_1, w_2, w_4
4	d_4	w_1, w_2, w_3, w_4, w_5
5	d_5	w_3, w_4

図 2 からも分かる通り, Du らは複数のデータに同じキーワードが含まれることを想定している. 例えば d_1 に含まれる w_1 は, d_2, d_3, d_4 にも含まれている.

各キーワードがどのドキュメントに含まれるのかを示したのものが転置インデックスである. 上記例の場合, 表 2 となる. ここでは Manning ら [21] の表記に従い, Dictionary, Postings という語を用いた.

表 2 転置インデックスの例

Dictionary	Postings
w_1	$1, 2, 3, 4$
w_2	$1, 3, 4$
w_3	$4, 5$
w_4	$1, 3, 4, 5$
w_5	$1, 4$

転置インデックスと検索可能暗号の併用. データを任意の暗号化方式, キーワードを検索可能暗号で暗号化する. 表 2 の Dictionary 列にはキーワードの暗号文が保存される. 各データの暗号文は元データと同じ ID を持つ. 検索にヒットした場合, Postings 列に従い対応する ID に保存された暗号化データを受信者に返す.

3. BB-PAEKS の定義

本章では BB-PAEKS 方式のシンタックスと, 本論文で着目する CI 安全性 (CI: Ciphertext Indistinguishable) を紹介する.

定義 1 (BB-PAEKS 方式のシンタックス)

Setup: セットアップアルゴリズムは, セキュリティパラメータ λ を入力とし, 公開パラメータ pp を出力する.

$\mathsf{KeyGen}_{\mathsf{DS}}$: データ送信者の鍵生成アルゴリズムは, pp を入力とし, 公開鍵と秘密鍵のペア $(\mathsf{pk}_{\mathsf{DS}}, \mathsf{sk}_{\mathsf{DS}})$ を出力する.

$\mathsf{KeyGen}_{\mathsf{DR}}$: データ受信者の鍵生成アルゴリズムは, pp を入力とし, 公開鍵と秘密鍵のペア $(\mathsf{pk}_{\mathsf{DR}}, \mathsf{sk}_{\mathsf{DR}})$ を出力する.

PAEKS: 暗号化アルゴリズムは, $\mathsf{pk}_{\mathsf{DR}}$, $\mathsf{sk}_{\mathsf{DS}}$, キーワード w を入力とし, 暗号化キーワード C_w を出力する.

Trapdoor: トラップドア生成アルゴリズムは, $\mathsf{pk}_{\mathsf{DS}}$, $\mathsf{sk}_{\mathsf{DR}}$, キーワード w' を入力とし, キーワード w' に関するトラップドア $T_{w'}$ を出力する.

Search: 検索アルゴリズムは, C_w, T_w を入力とし, 0 か 1 を出力する.

Verify: 検索結果検証アルゴリズムは, 検索結果の集合 Rlt, チェックリスト Acc を入力とし, proof を出力する.

Dec: 復号アルゴリズムは, Rlt, $\mathsf{sk}_{\mathsf{DR}}$ を入力とし, キーワード w に関連する ID ind_w を出力する.

Update: 更新アルゴリズムは, w, k を入力とし, 更新後の暗号文 C_w を出力する.

なお Update アルゴリズムの入力である k は PAEKS アルゴリズムや Trapdoor アルゴリズム内で計算される値である. 実際の方式ではブルームフィルタで用いられる鍵に相当する.

BB-PAEKS 方式の流れを簡単に説明する. 登場人物はデータ送信者 (DS: Data Sender), データ受信者 (DR: Data Receiver), ストレージの提供者 (SP: Service Provider), ブロックチェーンの 4 つである. まず, DS はデータと検索キーワードを暗号化して SP に送付し, チェックリスト Acc をブロックチェーンに送付する. DR は検索キーワードのトラップドアを生成し, SP に送付する. SP はトラップドアを用いて, それに紐づいたキーワードを含むデータを検索する. SP は検索結果の暗号文を DR とブロックチェーンに送付する. ブロックチェーンは検索結果と DS から送付された Acc より proof を生成, proof を DR に送付する. なお, Du らの論文ではブロックチェーンにトラップドアを送付するという記述がある[*2]. ブロックチェーンが検索結果の証明を作成する際の処理にトラップドアを用いていないことから, DR からブロックチェーンへのトラップドア送付処理を省略する.

CI 安全性. 暗号文からキーワードに関する情報が漏れないことを定式化した CI 安全性を以下で定義する.

定義 2 (CI 安全性)

Initialization: チャレンジャー $\mathcal{C}$ は $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{pk}_{\mathsf{DS}}, \mathsf{sk}_{\mathsf{DS}}) \leftarrow \mathsf{KeyGen}_{\mathsf{DS}}(\mathsf{pp})$, $(\mathsf{pk}_{\mathsf{DR}}, \mathsf{sk}_{\mathsf{DR}}) \leftarrow$

[*2] 具体的に, "*The* DR *sends trapdoor* T_w *to the* SP *and Blockchain.*" と述べている.

$\mathsf{KeyGen_{DR}}(\mathsf{pp})$, を実行し, 攻撃者 $\mathcal{A}$ に $(\mathsf{pp}, \mathsf{pk_{DS}}, \mathsf{pk_{DR}})$ を送付する.

Phase 1: $\mathcal{A}$ はキーワード w を暗号化オラクル, トラップドアオラクルにクエリ, それぞれ暗号文, トラップドアを得る.

Challenge: $\mathcal{A}$ は暗号化オラクルとトラップドアオラクルに送付していないチャレンジキーワード w_0^*, w_1^* を宣言する. $\mathcal{C}$ はランダムに $b \in \{0,1\}$ を選択し, チャレンジ暗号文 $C_{w_{b^*}} \leftarrow \mathsf{PAEKS}(\mathsf{pk_{DR}}, \mathsf{sk_{DS}}, w_b^*)$ を $\mathcal{A}$ に送付する.

Phase 2: $\mathcal{A}$ はキーワード $w \notin \{w_0^*, w_0^1\}$ を暗号化オラクル, トラップドアオラクルにクエリ, それぞれ暗号文, トラップドアを得る.

Guess: $\mathcal{A}$ は $b' \in \{0,1\}$ を出力する. $b' = b$ の場合, $\mathcal{A}$ はゲームに勝利すると定義する.

$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CKA}}(\lambda) = |\Pr[b' = b] - \frac{1}{2}|$ を，$\mathcal{A}$ のアドバンテージとする. セキュリティパラメータに対し, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{IND\text{-}CKA}}(\lambda)$ が無視できる場合, BB-PAEKS 方式は選択キーワード攻撃に対し CI 安全であると定義する.

トラップドアからキーワードの情報が漏れないことを保証する TI 安全性 (TI: Trapdoor Indistinguishability) も定義されているが, 本論文では省略する. また, 検索結果の完全性検証は, Du らの論文内では定式化されていない. 以下, 彼らの完全性に関する記述を引用する. 下記記述を鑑み, Verify アルゴリズムは検索結果が正しい場合は true を, 誤った検索を行った場合は false を出力する想定であると仮定する.

> "*Our scheme supports the integrity verification of the results. It ensures the reliability of search results by using the tamper proof characteristics of blockchain.*"

4. BB-PAEKS 方式

具体的な構成を紹介する前に, 構成のアイデアを説明する. DS-DR 間で Diffie-Hellman (DH) 鍵共有

$$\mathsf{DH_{Key}} = \mathsf{pk_{DR}}^{\mathsf{sk_{DS}}} = \mathsf{pk_{DS}}^{\mathsf{sk_{DR}}} = g^{xy}$$

を (非対話で) 行う. 暗号文内ブルームフィルタ $\mathsf{BF_{DS}}$ とトラップドア内ブルームフィルタ $\mathsf{BF_{DR}}$ を生成する際に, $\mathsf{DH_{Key}}$ とキーワード w からハッシュ関数が選ばれる. 具体的に, $k = \mathcal{H}_1(\mathsf{DH_{Key}}, w)$ により $\mathcal{H} = \{h_j(k, w)\}$ を計算する. また, $\mathsf{BF_{DR}}$ の生成時にはキーワードに加えて複数のランダムなキーワード $\mathsf{num}_1, \ldots, \mathsf{num}_J$ をフィルタに挿入する (Du らはその理由を明記していないが, トラップドアからのキーワード情報の漏洩を防ぐ目的と思われる). 検索時には, $\mathsf{BF_{DS}}$ と $\mathsf{BF_{DR}}$ の内積を $\langle \mathsf{BF_{DS}}, \mathsf{BF_{DR}} \rangle$, $\mathsf{BF_{DS}}$ 内の 1 であるビット数を $|\mathsf{BF_{DS}}|$ として,

$$\langle \mathsf{BF_{DS}}, \mathsf{BF_{DR}} \rangle = |\mathsf{BF_{DS}}|$$

が成り立つ場合, 検索キーワードが $\mathsf{BF_{DS}}$ に含まれると判定する. 転置インデックス内のデータに紐づいた ID ind_w は, $\mathsf{pk_{DR}} = g^x$ を公開鍵とした ElGamal 暗号方式を用いて

$$C_1 = g^{r_1}, C_2 = g^{xr_1} \cdot \mathsf{ind}_w$$

と暗号化され SP に送付される. 同時に C_2 のハッシュ値 $C_3 = \mathcal{H}_2(C_2)$ がブロックチェーンに送付される. SP が検索結果を提出した際には, それに含まれる C_2 のハッシュ値と C_3 を比較して検索結果の完全性結果を行う. 具体的な方式は次の通りである.

$\mathsf{pp} \leftarrow \mathsf{Setup}(\lambda)$**:** 素数位数 p の巡回群 $\mathbb{G}_1$ を選ぶ. J をハッシュ関数の数, $j \in \{0, J-1\}$ として, 3 つのハッシュ関数 $\mathcal{H}_1 : \mathbb{G}_1 \times \{0,1\}^* \to {\mathbb{Z}_p}^*$, $\mathcal{H}_2 : \{0,1\}^* \to \{0,1\}^*$, $\mathcal{H}_3 : {\mathbb{Z}_p}^* \times \{0,1\}^* \to {\mathbb{Z}_p}^*$ と k を鍵としたハッシュ関数族 $\mathcal{H} = \{h_j(k, \mathsf{Meg})\}$ を定義する. $\mathbb{G}_1$ のランダムな生成元 g を選ぶ. 公開パラメータ $\mathsf{pp} = \{\mathbb{G}, p, g, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3, \mathcal{H}\}$ を出力する.

$(\mathsf{pk_{DS}}, \mathsf{sk_{DS}}) \leftarrow \mathsf{KeyGen_{DS}}(\mathsf{pp})$**:** DS は乱数 $y \in {\mathbb{Z}_p}^*$ を選び, $\mathsf{pk_{DS}} = g^y$, $\mathsf{sk_{DS}} = y$ を生成する.

$(\mathsf{pk_{DR}}, \mathsf{sk_{DR}}) \leftarrow \mathsf{KeyGen_{DR}}(\mathsf{pp})$**:** DR は乱数 $x \in {\mathbb{Z}_p}^*$ を選び, $\mathsf{pk_{DR}} = g^x$, $\mathsf{sk_{DR}} = x$ を生成する.

$(C_{w,1}, C_{w,2}) \leftarrow \mathsf{PAEKS}(\mathsf{sk_{DS}}, \mathsf{pk_{DR}}, w)$**:** DS はバージョン情報 $\mathsf{VerInfo} \in {\mathbb{Z}_p}^*$ を選ぶ. $\mathsf{BF.Gen}(L, J)$ を実行し, DS 側のブルームフィルタ $\mathsf{BF_{DS}}$ を初期化する. $\mathsf{DH_{Key}} = g^{xy} = \mathsf{pk_{DR}}^{\mathsf{sk_{DS}}} = \mathsf{pk_{DS}}^{\mathsf{sk_{DR}}}$ と鍵 $k = \mathcal{H}_1(\mathsf{DH_{Key}}, w)$ を生成する. k とキーワード w を元に, $\mathcal{H} = \{h_j(k, w)\}$, $\mathsf{BF_{DS}}$, w を入力として $\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF_{DS}}, w)$ を実行する. 乱数 $r_1 \in {\mathbb{Z}_p}^*$ を選び, $\mathsf{ind}_w = \{id_i | w \in id_i\}$ に対して, $C_1 = g^{r_1}$, $C_2 = g^{xr_1} \cdot \mathsf{ind}_w$, $C_3 = \mathcal{H}_2(C_2)$ を計算する. $C_{w,1} = \{\mathsf{BF_{DS}}, C_1, C_2, \mathsf{VerInfo}\}$ を SP に, $C_{w,2} = \{C_3, \mathsf{VerInfo}\}$ をブロックチェーンに送付する.

SP は $C_{w,1}$ を暗号化キーワードの集合 EDB に, ブロックチェーンは $C_{w,2}$ を Acc に追加する.

$T_w \leftarrow \mathsf{Trapdoor}(\mathsf{sk_{DR}}, \mathsf{pk_{DS}}, w)$**:** DR は $\mathsf{BF.Gen}(L, J)$ を実行し, DR 側のブルームフィルタ $\mathsf{BF_{DR}}$ を初期化する. $\mathsf{DH_{Key}} = g^{xy} = \mathsf{pk_{DR}}^{\mathsf{sk_{DS}}} = \mathsf{pk_{DS}}^{\mathsf{sk_{DR}}}$ と $k = \mathcal{H}_1(\mathsf{DH_{Key}}, w)$ を計算する. $\mathcal{H} = \{h_j(k, w)\}$, $\mathsf{BF_{DS}}$, w より $\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF_{DR}}, w)$ を実行する. ランダムな J 個のキーワード $\mathsf{num}_1, \ldots, \mathsf{num}_J \in \{0,1\}^*$ を選び, すべての $\ell \in [1, J]$ について $\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF_{DR}}, \mathsf{num}_\ell)$ を実行する. SP に $T_w = \mathsf{BF_{DR}}$ を送付する.

$\{0,1\} \leftarrow \mathsf{Search}(C_{w,1}, T_w)$**:** SP は $\mathsf{BF_{DS}}$ と $\mathsf{BF_{DR}}$ の内積 $\langle \mathsf{BF_{DS}}, \mathsf{BF_{DR}} \rangle$ と $|\mathsf{BF_{DS}}|$ を計算し, $\langle \mathsf{BF_{DS}}, \mathsf{BF_{DR}} \rangle = |\mathsf{BF_{DS}}|$ が成り立つならば 1 を, そうでなければ 0 を出

力する. SP は $C_{w,1}$ を検索結果のリスト Rlt に追加し, これを DR とブロックチェーンに送付する.

$\mathsf{proof} \leftarrow \mathsf{Verify}(\mathsf{Rlt}, \mathsf{Acc})$**:** ブロックチェーンは, 送付された Rlt を用いて, 暗号化したファイル識別子の集合 B からそれぞれの結果のハッシュ値を得る. 全ての $C_{w,1} \in \mathsf{Rlt}$ について, C_2 のハッシュ値 $\mathcal{H}_2(C_2)$ を計算し, Acc と比較する. 等しいなら検索結果は正しい, そうでなければ正しくないと判別して, 検証結果 $\mathsf{proof} \in \{\mathsf{true}, \mathsf{false}\}$ と Rlt を DR に送付する.

$\mathsf{ind}_w \leftarrow \mathsf{Dec}(\mathsf{Rlt}, \mathsf{sk}_{\mathsf{DR}})$**:** DR は $\mathsf{ind}_w = C_2/C_1^x$ を計算する.

$(C_{w,1}, C_{w,2}) \leftarrow \mathsf{Update}(w, k)$**:** DR はバージョン情報 $\mathsf{VerInfo}^* \in \mathbb{Z}_p{}^*$ を選び, $\mathsf{BF.Gen}(L, J)$ を実行する. $\mathsf{DH}_{\mathsf{Key}} = g^{xy} = \mathsf{pk}_{\mathsf{DR}}{}^{\mathsf{sk}_{\mathsf{DS}}} = \mathsf{pk}_{\mathsf{DS}}{}^{\mathsf{sk}_{\mathsf{DR}}}$, $k_1 = \mathcal{H}_1(\mathsf{DH}_{\mathsf{Key}}, w)$, $k_2 = \mathcal{H}_3(k_1, w)$ を計算する. $\mathcal{H} = \{h_j(k_2, w)\}$, $\mathsf{BF}_{\mathsf{DS}}$, w より $\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF}_{\mathsf{DS}}, w)$ を実行する. $\mathsf{ind}_w = \{id_i | w \in id_i\}$ に対して, $C_1 = g^{r_1}$, $C_2 = g^{xr_1} \cdot \mathsf{ind}_w$, $C_3 = \mathcal{H}_2(C_2)$ を計算し, $C_{w,1} = \{\mathsf{BF}_{\mathsf{DS}}, C_1, C_2, \mathsf{VerInfo}^*\}$ を SP に, $C_{w,2} = \{C_3, \mathsf{VerInfo}^*\}$ をブロックチェーンに送付する. SP は $C_{w,1}$ を暗号化キーワードの集合 EDB に, ブロックチェーンは $C_{w,2}$ を Acc に追加する.

5. 提案攻撃

5.1 暗号文に対する攻撃

提案攻撃では, DS-DR 間による DH 鍵とキーワードから暗号文に含まれるブルームフィルタ $\mathsf{BF}_{\mathsf{DS}}$ が一意的に決まることを利用する. Algorithm 1 に攻撃アルゴリズムを示す. 流れを簡単に説明する. $\mathcal{A}$ はチャレンジキーワード w_0^* を暗号化オラクルにクエリし, w_0^* の暗号文 $C_{w_0^*,1} = \{\mathsf{BF}_{\mathsf{DS}}{}', C_1', C_2', \mathsf{VerInfo}'\}$ を入手する. チャレンジ暗号文 $C_{w_b^*,1}$ 内の $\mathsf{BF}_{\mathsf{DS}}$ と $C_{w_0^*,1}$ 内の $\mathsf{BF}_{\mathsf{DS}}{}'$ を比較し, 一致するならば $b' = 0$, そうでなければ $b' = 1$ を出力する.

Algorithm 1 Du らの方式に対する提案攻撃

入力 $C_{w_b^*}, w_0^*$
出力 b'
  $C_{w_0^*} \leftarrow$ EncryptionQuery w_0^*
  Extract $\mathsf{BF}_{\mathsf{DS}}{}'$ from $C_{w_0^*}$
  Extract $\mathsf{BF}_{\mathsf{DS}}$ from $C_{w_b^*}$
  if $\mathsf{BF}_{\mathsf{DS}} == \mathsf{BF}_{\mathsf{DS}}{}'$ **then**
    return $b' = 0$
  else
    return $b' = 1$
  end if

偽陽性が発生した場合, $C_{w_0^*}$ と $C_{w_1^*}$ 内のブルームフィルタが一致する. すなわち, 攻撃成功確率は偽陽性が発生しない確率に等しい. そこで Du らが参照している [29] の論文を参照し, 偽陽性発生確率を評価する. フィルタに挿入するキーワードの数 K と目標とする偽陽性の発生確率 $\Pr[\mathsf{FalsePositive}]$ より, フィルタサイズ L は

$$L = -K \times \ln(\Pr[\mathsf{FalsePositive}])/\ln(2)^2$$

で表される. BB-PAEKS 方式では, 暗号化時に新規にブルームフィルタを生成するため, $K = 1$ である. 以下, 128 ビットセキュリティを考慮し, $\Pr[\mathsf{FalsePositive}] = 2^{-128}$ を (ひとまずの) 目標値とする.[*3] これらの値を用いると, フィルタサイズは $L = 185$ となる. 導かれた L に対して, 偽陽性発生確率が $\Pr[\mathsf{FalsePositive}]$ を達成する最適なハッシュ関数の数 J は, 次の式で求められる.

$$J = \lceil L \times \ln(2)/K \rceil$$

$L = 185$ に対して $J = 129$ が最適なハッシュ関数の数となる. なお 6 章の評価にて, $J(J+1) < L$ を満たす必要がある. 本論文ではこの条件下での J の最大値 $J = 13$ を用いる. 表 3 に J を最適値 129 から 13 と変更した場合の $\Pr[\mathsf{FalsePositive}]$ の劣化について記載する.

表 3 偽陽性発生確率評価

ハッシュ関数の数 J	$\Pr[\mathsf{FalsePositive}]$
13	$6.69 \times 10^{-16} \approx 2^{-50}$
129	$3.19 \times 10^{-39} \approx 2^{-128}$

$J = 13$ でも論文 [29] における $\Pr[\mathsf{FalsePositive}] = 10^{-4} \approx 2^{-13}$ と比較して非常に小さい値 $\Pr[\mathsf{FalsePositive}] = 10^{-16} \approx 2^{-50}$ を実現していることに注意されたい. よって, 提案攻撃は十分に高い確率で成功すると結論付ける.

5.2 完全性検証に対する考察

Acc は DS が作成した暗号文の C_2' のハッシュ値を保存しているに過ぎない. 前述の通り, Verify アルゴリズムは検索結果が正しい場合は true を, 誤った検索を行った場合は false を出力する想定であると仮定する. ここで SP が検索にヒットしない暗号文を返したと仮定する. この場合でも Verify アルゴリズムは true を出力する. Verify アルゴリズムが false を出力するのは, DS が SP に送付した暗号文をブロックチェーンには送付しなかった場合のみであり, SP の検索結果に依存しない. これらのことから, Du らの方式は完全性を満たさない.

6. 提案攻撃への対策と対策方式への攻撃

6.1 単純な対策

提案攻撃は, DS-DR 間による DH 鍵とキーワードから暗号文に含まれるブルームフィルタ $\mathsf{BF}_{\mathsf{DS}}$ が一意的に決まるこ

[*3] なおこの偽陽性発生確率目標値は実用上非常に小さい. 例えば論文 [29] では $\Pr[\mathsf{FalsePositive}] = 10^{-4} \approx 2^{-13}$ とセキュリティパラメータに対して非常に大きな値を想定している.

とを利用している．ここで Du らの方式では，トラップドアに含まれるブルームフィルタ $\mathsf{BF}_{\mathsf{DR}}$ をランダムなキーワード $\mathsf{num}_1, \ldots, \mathsf{num}_J$ を用いて更新していることに着目する．すなわち，全ての $\ell \in [1, J]$ について $\mathsf{BF.Upd}(\mathcal{H}, \mathsf{BF}_{\mathsf{DR}}, \mathsf{num}_\ell)$ を実行し，$\mathsf{BF}_{\mathsf{DR}}$ を得る．すなわち $\mathsf{BF}_{\mathsf{DR}}$ は確率的に決定する．ここで $\mathsf{BF}_{\mathsf{DR}}$ の代わりに $\mathsf{BF}_{\mathsf{DS}}$ をランダムなキーワードで更新する場合，同じキーワードの暗号文でも $\mathsf{BF}_{\mathsf{DS}}$ が別の値となる．そのため，提案攻撃を直接適用することはできない．しかし，変更後の方式 (以下，修正方式) についても，暗号文からキーワードの情報が漏れていることを示す．$\mathsf{BF.Upd}$ のアルゴリズムより，更新前の $\mathsf{BF}_{\mathsf{DS}}$ で 1 であるビットは，更新後にも必ず 1 となることに着目する．つまり，同じキーワードを暗号化した複数の暗号文を受け取ったとき，全ての暗号文で 1 が立っているビットを 1，一度でも 0 になったビットを 0 とすることで，確率的に更新前の $\mathsf{BF}_{\mathsf{DS}}$ を復元することが可能となる．Algorithm 2 に修正方式に対する攻撃アルゴリズムを示す．フィルタサイズを L，ハッシュ関数の数を J，暗号文オラクルへのクエリ回数を q_{enc}，チャレンジキーワードを w_0^* とする．

Algorithm 2 修正方式に対する攻撃

入力 $L, J, q_{\mathsf{enc}}, w_0^*$
出力 $\mathsf{BF}_{\mathsf{DS}}$

```
BF_DS ← 1 for i = 0 to L − 1
counter ← 0
while |BF_DS| > J do
    counter = counter + 1
    if counter > q_enc then
        abort
    end if
    C_{w_0^*} ← Encryption Query w_0^*
    Extract queryBF_DS from C_{w_0^*}
    for i = 0 to L − 1 do
        if queryBF_DS[i] == 0 then
            BF_DS[i] ← 0
        end if
    end for
end while

return BF_DS
```

6.2 確率評価

以下，更新前 $\mathsf{BF}_{\mathsf{DS}}$ の復元確率を評価する．評価にあたり，Du らが引用している論文 [23] と同じ仮定を用いる．具体的に，ブルームフィルタのハッシュ関数は一様ランダムな出力を返すと仮定する．すなわち，$\mathsf{BF.Gen}$ で生成された BF に $\mathsf{BF.Upd}$ で任意のキーワードを 1 つ入力した場合，BF 内で 1 となるビット数は J となる．また，フィルタに挿入するキーワードの個数 K に対して，$L \geq K \times J$ ならば，各キーワードの挿入時に 1 となるビットは重複しない．ランダムなキーワード $\mathsf{num}_1, \ldots, \mathsf{num}_J$ を用いた更新処理により，フィルタは元のキーワードを含めて $J(J+1)$ ビットが 1 となるため，$J(J+1) < L$ を満たすと仮定する．w_0^* のみが含まれる $\mathsf{BF}_{\mathsf{DS}}$ (上記アルゴリズムにて復元したい $\mathsf{BF}_{\mathsf{DS}}$) について，$J$ 箇所のビットは 1，$L-J$ 箇所のビットは 0 である．復元が失敗するということは，元々 0 である $L-J$ 箇所のビットのうち，少なくとも 1 つが 1 となることである．真の $\mathsf{BF}_{\mathsf{DS}}$ で 0 である $L-J$ ビットから 1 となる J ビットを J 回選ぶため，この確率は以下で表される．

$$\Pr[1 \leftarrow 0] = J^2/(L-J)$$

アルゴリズム内では，q_{enc} 個の $\mathsf{queryBF}_{\mathsf{DS}}$ を元に復元を行う．真の $\mathsf{BF}_{\mathsf{DS}}$ 内の 0 であるビットについて，q_{enc} 個の $\mathsf{queryBF}_{\mathsf{DS}}$ 全てで $1 \leftarrow 0$ の変更が行われている確率は以下で表される．

$$\Pr[\mathsf{fail}] = (J^2/(L-J))^{q_{\mathsf{enc}}}$$

提案攻撃では，1 回でも 0 を観測したビットは 0 として $\mathsf{BF}_{\mathsf{DS}}$ の復元を行う．そのため，q_{enc} 回のうち 1 回でも 0 であった場合，そのビットは正しく復元できる．したがって，$\mathsf{BF}_{\mathsf{DS}}$ の復元が成功する確率は以下で表される．

$$\Pr[\mathsf{success}] = 1 - (J^2/(L-J))^{q_{\mathsf{enc}}}$$

$\Pr[\mathsf{success}]$ が (セキュリティパラメータに対し) 無視できない値なら，我々の攻撃は成功すると定義する．以下，[29] より式を引用し，$\Pr[\mathsf{success}]$ が現実的なクエリ回数において 1 に近い値であることを示す．5.1 節での議論より，$J = 13$ の場合について確率を評価する．表 5 に攻撃成功確率を示す．

表 4 $\Pr[\mathsf{success}]$ の評価 ($J = 13$)

q_{enc}	$\Pr[\mathsf{success}]$
1	0.017
2	0.035
2^2	0.068
2^3	0.131
2^4	0.245
2^5	0.431
2^6	0.676
2^7	0.895
2^8	0.989
2^9	1.000

攻撃に要する時間について，1 回の暗号文作成に 1 秒かかるという非常に非効率な実装を考慮したとしても，2^9 個の暗号文作成は 8 分程度で完了する．このことから，我々の攻撃は十分に効率的であると結論付ける．

次に 1 回の暗号文クエリ時における攻撃成功確率 0.017 について評価する．以下 Oxford 英語辞典 (the second edition of the 20-volume)[*4]からキーワードを選ぶと仮定する．この

[*4] https://wordcounter.io/blog/how-many-words-are-in-the-english-language

辞典には 171,476 の単語が掲載されており, $2^{18} = 262,144$ で単語数をカバーできる．$\mathsf{BF}_{\mathsf{DS}}$ に含まれるキーワードを無作為に推測する場合, 攻撃成功確率は $1/2^{18}$ で抑えられる．この確率と比較して, 0.017 は十分に大きい, この観点から, 我々の攻撃は高い確率で成功すると評価する．

7. 議論

7.1 安全性モデル外攻撃の妥当性

提案攻撃において, 攻撃者はチャレンジキーワードを暗号化オラクルにクエリする．しかし Du らの安全性モデルではチャレンジキーワードを暗号化オラクルへクエリすることは禁止されている．これは 1 つのキーワードが 1 回のみ暗号化される状況を想定した安全性モデルであることに注意されたい．一方で, 表 1 に紹介した通り, Du らはあるキーワードが複数のデータに含まれる場合を想定している．ここでもし各キーワードが一度しか暗号化されない場合, 表 2 の Keyword 列には, 同じ暗号文が複数回保存されることとなる．すなわち, データ $d_1, \ldots, d_4$ に同じキーワードが含まれるという情報が検索前に漏洩することになる．なお検索可能である以上, ある検索クエリに対して返答された暗号文に対応するデータには同じキーワードが含まれるという情報が漏洩する．ここでは検索することなく, 暗号文テーブルを見るだけで同じキーワードが含まれるという情報が漏洩することを問題視している．以上より, チャレンジキーワードを暗号化オラクルにクエリ可能なモデルは妥当であると評価する．

7.2 完全性検証モデルの問題点

Du らは, SP が暗号文改ざんリスクのあるサービスを提供する場合などを想定して, 完全性検証が必要であると述べている[*5]．しかし前述の通り, Verify アルゴリズムが false を出力するのは, DS が SP には暗号文を渡し, ブロックチェーンには渡さなかった場合に限られる．なお検索結果であるデータの暗号文を復号することで, DR は SP の検索結果にキーワードが含まれているか否かを容易に確認可能であることに注意されたい．一方, 本来検索結果に含まれるべきデータ暗号文を SP が DR に返さなかった場合, その事実を DR は知る由がない．同様に, SP からの出力を基にブロックチェーンが証明を生成するモデルを採用する以上, ブロックチェーンでも検出できない．以上より, モデル自体に再考の余地があると結論付ける．

8. 結論

本論文では, Du らの方式は暗号文内ブルームフィルタからキーワードの情報が漏れていることを示した．提案攻撃は単純な変更を施すことで対策可能であるが, 変更後の方式に対して有効な攻撃手法も提案した．これらは Du らの安全性モデル外攻撃であるため, 攻撃環境の妥当性についても議論した．さらに完全性検証についても議論を行い, SP からの出力を基にブロックチェーンが証明を生成するモデルの再考が必要であると指摘した．

ブルームフィルタに対する攻撃手法は数多く報告されている．無視できない確率で偽陽性が発生する要素を出力する攻撃者を考える．ある値がブルームフィルタに含まれるかどうかの結果を見た上で, 次のクエリを決定可能な適応的攻撃者を想定した結果が知られている [20, 24, 25]．今回暗号化クエリに対してブルームフィルタの値を得られるモデルであるため, この適応的攻撃者に対する評価が適用できると考えられる．また機械学習を利用した攻撃モデル [1] も考案されている．昨今の機械学習の発展を鑑みると, このような攻撃モデルは自然であると言える．ブルームフィルタを用いた検索可能暗号の設計に際し, これら適応的攻撃者や機械学習の利用を想定した場合の評価を今後の課題とする．

謝辞 本研究は JSPS 科研費 JP21K11897, JP25H01106 の助成を受けたものです．

[*5] 具体的に, “*Third-party servers often benefit in practice from providing retrieval services with a risk that ciphertext data may be tampered with by malicious parties or that the server may fail. As a consequence, the integrity of search results is also a concern that cannot be overlooked.*” と述べている．

参考文献

[1] Ghada Almashaqbeh, Allison Bishop, and Hayder Tirmazi. Adversary resilient learned bloom filters. Cryptology ePrint Archive, Paper 2024/754, 2024.

[2] Mandira Banik and Sanjay Kumar. Blockchain-based public key encryption with keyword search for medical data sharing in cloud environment. *J. Inf. Secur. Appl.*, 78:103626, 2023.

[3] Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[4] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *EUROCRYPT*, pages 506–522, 2004.

[5] Partha Sarathi Chakraborty, Mangesh Shivaji Chandrawanshi, Puspesh Kumar, and Somanath Tripathy. BSMFS: blockchain assisted secure multi-keyword fuzzy search over encrypted data. In *IEEE Blockchain*, pages 216–221. IEEE, 2022.

[6] Lanxiang Chen, Wai-Kong Lee, Chin-Chen Chang, Kim-Kwang Raymond Choo, and Nan Zhang. Blockchain based searchable encryption for electronic health record sharing. *Future generation computer systems*, 95:420–429, 2019.

[7] Yang Chen, Chunlu Zhao, Jin Pan, and Yang Liu. Verifiable attribute-based multi-keyword ranked search scheme in blockchain. *The Computer Journal*, 68(5):510–519, 2025.

[8] Ningning Cui, Dong Wang, Jianxin Li, Huaijie Zhu, Xiaochun Yang, Jianliang Xu, Jie Cui, and Hong Zhong. Enabling efficient, verifiable, and secure conjunctive keyword search in hybrid-storage blockchains. *IEEE Trans. Knowl. Data Eng.*, 36(6):2445–2460, 2024.

[9] Haorui Du, Jianhua Chen, Fei Lin, Cong Peng, and Debiao He. A lightweight blockchain-based public-key authenticated encryption with multi-keyword search for cloud computing. *Security and Communication Networks*, 2022(1):2309834, 2022.

[10] Sheng Gao, Yuqi Chen, Jianming Zhu, Zhiyuan Sui, Rui Zhang, and Xindi Ma. BPMS: blockchain-based privacy-preserving multi-keyword search in multi-owner setting. *IEEE Trans. Cloud Comput.*, 11(3):2260–2272, 2023.

[11] Hongmu Han, Zhongpeng Wang, Zhigang Xu, Xinhua Dong, and Wenlong Tian. Enhancing iot security and efficiency: A blockchain-assisted multi-keyword searchable encryption scheme. *IEEE Access*, 12:148677–148692, 2024.

[12] Min Han and Peng Xu. Efficient and verifiable keyword search over public-key ciphertexts based on blockchain. *J. Inf. Secur. Appl.*, 89:103924, 2025.

[13] Ruiwei Hou, Fucai Zhou, Qiang Wang, Zi Jiao, Jintong Sun, and Zongye Zhang. Revokable blockchain-enabled ranked multi-keyword attribute-based searchable encryption scheme with mobile edge computing for vehicular. *IEEE Trans. Netw. Serv. Manag.*, 22(3):2764–2779, 2025.

[14] Shengshan Hu, Chengjun Cai, Qian Wang, Cong Wang, Xiangyang Luo, and Kui Ren. Searching an encrypted cloud meets blockchain: A decentralized, reliable and fair realization. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 792–800. IEEE, 2018.

[15] Qiong Huang and Hongbo Li. An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks. *Information Sciences*, 403:1–14, 2017.

[16] Shan Jiang, Jiannong Cao, Julie A. McCann, Yanni Yang, Yang Liu, Xiaoqing Wang, and Yuming Deng. Privacy-preserving and efficient multi-keyword search over encrypted data on blockchain. In *IEEE Blockchain*, pages 405–410. IEEE, 2019.

[17] Qinyi Li and Xavier Boyen. Public-key authenticated encryption with keyword search made easy. *IACR Commun. Cryptol.*, 1(2):16, 2024.

[18] Yihuai Liang, Yan Li, and Byeong-Seok Shin. Dynamic authenticated keyword search in hybrid-storage blockchain. *Future Gener. Comput. Syst.*, 155:53–65, 2024.

[19] Zheli Liu, Tong Li, Ping Li, Chunfu Jia, and Jin Li. Verifiable searchable encryption with aggregate keys for data sharing system. *Future Generation Computer Systems*, 78:778–788, 2018.

[20] Chen Lotan and Moni Naor. Adversarially robust bloom filters: Monotonicity and betting. *IACR Communications in Cryptology*, 2(1):24, 2025.

[21] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[22] Ying Miao, Keke Gai, Jing Yu, Yu-an Tan, Liehuang Zhu, and Weizhi Meng. Blockchain-empowered keyword searchable provable data possession for large similar data. *IEEE Trans. Inf. Forensics Secur.*, 20:1374–1389, 2025.

[23] James K. Mullin. A second look at bloom filters. *Commun. ACM*, 26(8):570–571, 1983.

[24] Moni Naor and Noa Oved. Bet-or-Pass: Adversarially robust bloom filters. In *TCC*, pages 777–808, 2022.

[25] Moni Naor and Eylon Yogev. Bloom filters in adversarial environments. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO Part II*, pages 565–584, 2015.

[26] Zheng Yao Ng and Iftekhar Salam. Blockchain-based multi-keyword search on encrypted COVID-19 contact tracing data. In *ISPEC*, pages 75–92, 2022.

[27] Fanfan Shen, Lin Shi, Jun Zhang, Chao Xu, Yong Chen, and Yanxiang He. BMSE: blockchain-based multi-keyword searchable encryption for electronic medical records. *Comput. Stand. Interfaces*, 89:103824, 2024.

[28] Jin Sun, Nana Song, Lu Wang, Kexin Ye, and Mengna Kang. A blockchain-based multi-keyword rank search scheme for B+ tree inverted index. *Comput. Stand. Interfaces*, 93:103968, 2025.

[29] Shi-Feng Sun, Ron Steinfeld, Shangqi Lai, Xingliang Yuan, Amin Sakzad, Joseph K Liu, Surya Nepal, and Dawu Gu. Practical non-interactive searchable encryption with forward and backward privacy. In *Usenix Network and Distributed System Security Symposium 2021*. The Internet Society, 2021.

[30] Haiqin Wu, Boris Düdder, Liangmin Wang, Zhenfu Cao, Jun Zhou, and Xia Feng. Survey on secure keyword search over outsourced data: From cloud to blockchain-assisted architecture. *ACM Comput. Surv.*, 56(3):63:1–63:40, 2024.

[31] Xixi Yan, Suwei Feng, Yongli Tang, Pei Yin, and Dazhi Deng. Blockchain-based verifiable and dynamic multi-keyword ranked searchable encryption scheme in cloud computing. *J. Inf. Secur. Appl.*, 71:103353, 2022.

[32] Xiaodong Yang, Jiaqi Wang, Wanting Xi, Tian Tian, and Caifen Wang. A blockchain-based keyword search scheme with dual authorization for electronic health record sharing. *J. Inf. Secur. Appl.*, 66:103154, 2022.

[33] Hongtao Yu, Suhui Liu, Liquan Chen, and Yuan Gao. Blockchain-enabled one-to-many searchable encryption supporting designated server and multi-keywords for Cloud-IoMT. *J. Syst. Archit.*, 149:103103, 2024.

[34] Duo Zhang, Shangping Wang, Qian Zhang, and Yaling Zhang. Attribute based conjunctive keywords search with verifiability and fair payment using blockchain. *IEEE Trans. Serv. Comput.*, 16(6):4168–4182, 2023.

[35] Kai Zhang, Xinyi Hu, Jian Zhao, Lifei Wei, and Jianting Ning. Blockchain-based revocable key-aggregate searchable encryption for group data sharing in cloud-assisted industrial IoT. *IEEE Internet Things J.*, 12(11):16899–16911, 2025.

[36] Leyou Zhang, Runze Tian, Qing Wu, and Fatemeh Rezaeibagha. BA-AMST: A blockchain-assisted anonymous and multi-keyword searchable-traceable data sharing system in industrial internet of things. *Engineering Science and Technology, an International Journal*, 66:102055, 2025.

[37] Lizhe Zhang, Wei Luo, Jiahao Li, Zhijun Wu, and Ruiqi Li. Blockchain-assisted keyword search scheme for SWIM service based on improved CSC-Cuckoo filter. *Int. J. Comput. Intell. Syst.*, 17(1):254, 2024.

[38] Yinghui Zhang, Robert H Deng, Jiangang Shu, Kan Yang, and Dong Zheng. TKSE: Trustworthy keyword search over encrypted data with two-side verifiability via blockchain. *IEEE Access*, 6:31077–31087, 2018.

[39] Yuan Zhang, Chunxiang Xu, Jianbing Ni, Hongwei Li, and Xuemin Sherman Shen. Blockchain-assisted public-key encryption with keyword search against keyword guessing attacks for cloud storage. *IEEE Trans. Cloud Comput.*, 9(4):1335–1348, 2021.