

Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Computing and Information Systems

School of Computing and Information Systems

12-2021

Broadcast authenticated encryption with keyword search

Xueqiao LIU

University of Wollongong

Kai HE

Guomin YANG

Singapore Management University, gmyang@smu.edu.sg

Willy SUSILO

Joseph TONIEN

See next page for additional authors

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research



Part of the [Information Security Commons](#)

Citation

LIU, Xueqiao; HE, Kai; YANG, Guomin; SUSILO, Willy; TONIEN, Joseph; and HUANG, Qiong. Broadcast authenticated encryption with keyword search. (2021). *Information Security and Privacy: 26th Australasian Conference, ACISP 2021, Virtual Conference, December 1-3: Proceedings*. 13083, 193-213. Available at: https://ink.library.smu.edu.sg/sis_research/7407








This Conference Proceeding Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

Author

Xueqiao LIU, Kai HE, Guomin YANG, Willy SUSILO, Joseph TONIEN, and Qiong HUANG



Broadcast Authenticated Encryption with Keyword Search

Xueqiao Liu¹ , Kai He² , Guomin Yang¹  , Willy Susilo¹ ,
Joseph Tonien¹ , and Qiong Huang³ 

¹ Institute of Cybersecurity and Cryptology, School of Computing and Information Technology, University of Wollongong, Wollongong 2522, Australia
{xl691,gyang,wsusilo,dong}@uow.edu.au

² School of Cyberspace Security, Dongguan University of Technology, Dongguan 523808, China

³ College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China
qhuang@scau.edu.cn

Abstract. The emergence of public-key encryption with keyword search (PEKS) has provided an elegant approach to enable keyword search over encrypted content. Due to its high computational complexity proportional to the number of intended receivers, the trivial way of deploying PEKS for data sharing with multiple receivers is impractical, which motivates the development of a new PEKS framework for broadcast mode. However, existing works suffer from either the vulnerability to keyword guessing attacks (KGA) or high computation and communication complexity. In this work, a new primitive for keyword search in broadcast mode, named broadcast authenticated encryption with keyword search (BAEKS), is introduced, in which the sender not only encrypts the keyword but also authenticates it, eliminating the threats of KGA. Moreover, on top of keyword privacy, we formalize the notion of *user anonymity* (or *key privacy*) for BAEKS, which echoes the notion of key privacy for public-key encryption introduced by Bellare et al. (ASIACRYPT'01). We present a practical BAEKS construction that achieves all the desirable features, including keyword privacy of both searchable ciphertext and trapdoor, KGA-resistance, receiver anonymity of both searchable ciphertext and trapdoor, and universal keyword set scalability. Moreover, the trapdoor of our scheme achieves constant computation and communication cost, making it more suitable for broadcast mode where trapdoors are generated by multiple receivers in the search operations. The security of our scheme is proved under the standard DBDH assumption.

K. He—This work is supported by National Natural Science Foundation of China (61902067), and the Foundation for Young Innovative Talents in Ordinary Universities of Guangdong (2018KQNCX255).

Q. Huang—This work is supported by National Natural Science Foundation of China (61872152), the Major Program of Guangdong Basic and Applied Research (2019B030302008) and the Science and Technology Program of Guangzhou (201902010081).

© Springer Nature Switzerland AG 2021

J. Baek and S. Ruj (Eds.): ACISP 2021, LNCS 13083, pp. 193–213, 2021.

https://doi.org/10.1007/978-3-030-90567-5_10

Keywords: Broadcast encryption · Multi-user · Public-key authenticated encryption with keyword search · Anonymity · Keyword guessing attack

1 Introduction

Public-key encryption with keyword search (PEKS) [4] was introduced by Boneh et al. to enable keyword search on encrypted content. However, in the textbook PEKS model, anyone can encrypt a keyword of interest and then use it to test a searching trapdoor, which is known as the keyword guessing attack (KGA) [9, 36]. To address the aforementioned problem, techniques such as public-key authenticated encryption with keyword search (PAEKS) [22], dual-server PEKS (DS-PEKS) [11], and server-aided public key encryption with keyword search (SA-PEKS) [10], were proposed to eliminate the threat. In PAEKS, in addition to encrypting the keyword, the sender authenticates it by taking the sender’s secret key as part of the input, thus preventing others from freely generating a ciphertext for testing.

While PEKS and PAEKS are designed for the single receiver setting, there are demands for allowing multiple receivers to perform keyword search in practice. For instance, due to the city lock-down caused by COVID-19, internet video-on-demand services have become popular. Without losing generality, we assume that a service provider is offering various videos that are stored in cloud storage for a paying viewer to watch at any time. The available videos can be labeled by the content type, such as “Animation”, “Sports”, “News”, and “Movie”, or the genre, such as “Comedy”, “Action”, and “Thriller”. If security and privacy are not a concern, a viewer can search the videos of interest by simply providing the searching keywords to the cloud server, which will perform the search and return the results to the user.

In the above application scenario, to protect the content of the videos uploaded by the service provider and the privacy of the search queries made by the viewers, a secure and practical searchable encryption scheme for multiple receivers is required. However, some prominent issues need to be addressed. On the service provider side, how to support multi-user accessing should be first considered. The trivial way is to share an identical key with every paying user, but it suffers from the key compromise issue. If any user is compromised or corrupted, the security of the entire system collapses and it is nearly impossible to trace the traitor. To avoid the risk of key compromise, public-key solutions for keyword search supporting multi-user access are more promising. The trivial way is to issue a separate PEKS (or PAEKS) key pair for each user and encrypt a video’s keyword under each user’s public key. Later, the user generates a trapdoor with her/his secret key, and the server tests the trapdoor with each video’s searchable ciphertext (encrypted keyword) to locate the matching ones without learning the keyword being located¹. However, such a trivial solu-

¹ The video content should also be encrypted, e.g., by using a standard mechanism such as broadcast encryption. We only focus on the searching phase in this paper.

tion is impractical for a large group of receivers due to the repetitive keyword encryption operations, massive storage overhead and a booming of transmission bandwidth. Thus, mitigating operation overhead, data redundancy, and communication cost turns to be the main challenge in deploying public-key based keyword search for multiple receivers.

Although PEKS and its variants considered the keyword privacy in a ciphertext and/or trapdoor, the identity privacy has been neglected in the prior research. Identity privacy means given a searchable ciphertext, the identity of the intended receiver is protected. In addition, for PEKS with multiple receivers, it is also desirable to protect the identity of the searching user who generates a searching trapdoor. As multiple nations and regions issued user privacy acts [18,35], the collection, storage, and analysis of any user information have been regulated, and user identity privacy plays a role as important as user data privacy. In traditional public-key encryption, a similar security notion named “key-privacy” or “anonymity”² has been formalized by Bellare et al. in [3], demanding that given a ciphertext, eavesdroppers should not be able to tell under which specific public key the given ciphertext is generated. In order to provide privacy protection for the users from all the angles, the key privacy should also be taken into consideration in PEKS (or PAEKS), i.e., a searchable ciphertext ought not to reveal the user identities of all intended/target receivers. On the other hand, different from the traditional public-key encryption in which only the ciphertext is exposed, in PEKS, the trapdoor is another potential spot of user identity exposure to the cloud server and other attackers. Back to the internet video-on-demand application, besides the security concern that no viewer would like parties other than the service provider to know whom a searchable ciphertext is prepared for, another practical privacy consideration is to conceal who is searching for the videos, i.e., the identity of a searching user should not be inferred from a searching trapdoor. We name such a key-privacy property regarding the trapdoor as “trapdoor anonymity”.

Taking the aforementioned internet video-on-demand service as an example, we summarize the desirable security and functionality features of a privacy-preserving keyword search scheme for multiple receivers as follows:

- supporting the multi-receiver setting;
- minimizing the online computation and communication overhead (trapdoor computation, trapdoor size, and testing);
- ensuring content confidentiality (searchable ciphertext semantic security);
- preserving search (trapdoor) privacy;
- allowing system expansion (scalable universal keyword set);
- maintaining recipient identity privacy for whom the searchable ciphertext is created (anonymity); and
- concealing user identity privacy from whom the trapdoor is submitted (trapdoor anonymity).

² The anonymity we discuss here only considers the application layer, hiding user identity using techniques on other layers such as IP address anonymization is beyond the scope of our work.

To the best of our knowledge, no existing PEKS (or PAEKS) scheme can satisfy all the above features. PAEKS [22] is not capable of supporting multiple receivers decrypting the same ciphertext. Similarly, searchable symmetric encryption (SSE) [34] is also not qualified because of the key management issue. The public-key primitive, broadcast encryption (BE) [6, 13, 16] seems suitable to be integrated with keyword search. Unfortunately, these schemes are not anonymous, exposing user identity information since the broadcast receiver set is taken as the input of the decryption algorithm. Its combination [31] with SSE realizes the multi-receiver setting and mitigates the key compromise but has unpromising communication performance for their multi-round interactions of token (trapdoor) generation and disallows universal keyword set expansion. The existing integrations [1, 23, 26] of BE and keyword search are unsatisfactory as well. Neither the content confidentiality nor the search (trapdoor) privacy is ensured by [1]. The test algorithm of [26] takes as input the set of intended receiver identities, not considering the security requirement of anonymity. Besides the public parameter size, the trapdoor size of [23] is also linear to the maximal number of receivers, resulting in large computational and communication overhead. It additionally suffers from limited expressive ability, i.e., a fixed universal keyword set. Moreover, their testing algorithm takes the broadcast receiver set as input, allowing the cloud server to access more sensitive information like all viewers' identities in the aforementioned scenario.

1.1 Contribution

Motivated by the broadcast scenario mentioned earlier, and the remaining unsolved challenges, we incorporate PAEKS with BE to present a new primitive called broadcast authenticated encryption with keyword search (or BAEKS, for short), followed by a concrete scheme. In particular, we provide a formal and comprehensive treatment for the user anonymity regarding both searchable ciphertext and trapdoor for BAEKS. Below we first outline the system architecture as Fig. 1, and then give a high-level description of our construction idea.

After setting up system parameters, KGC distributes a unique key pair (pk, sk) to each entity (sender or receiver). A sender S processes the underlying keyword w' of its document to generate the searchable ciphertext C , using its own secret key and all target receivers' public keys, and then uploads the document together with C to the cloud server. Any receiver R can compute the trapdoor T_w for the keyword w of interest with its own secret key and a sender's public key, and send T_w to the cloud server for a search query. The cloud server can test on C and T_w without knowing the receiver's identity, and the corresponding document will be returned if all the following hold: their underlying keywords are the same ($w' = w$), the trapdoor T_w is for querying the content from the sender S rather than other senders, and the receiver R is one of the target receivers of the searchable ciphertext C .

To prevent keyword guessing attack, the sender's secret key is taken as an input of the encryption algorithm to ensure parties other than the sender cannot manufacture the ciphertext. Assume there are t intended receivers, the sender

processes the sender's secret key, each intended receiver's public key, and the keyword to obtain a secret value V_i , and utilizes these t secret values as roots to construct a t -degree polynomial. Then the sender hides a randomly chosen secret element k in the polynomial and then includes the coefficients in the ciphertext. The remaining ciphertext components are calculated based on k . On the receiver side, the trapdoor generation algorithm takes the sender's public key, the receiver's secret key and the keyword of interest as the input and will get a trapdoor corresponding to the secret value V_i . On the cloud server side, the test algorithm takes the trapdoor and coefficients in the ciphertext to recover a value k' . Note that if the keyword is identical in the ciphertext and the trapdoor, then $k' = k$. With the help of k , the server can do further tests on the remaining ciphertext components to confirm whether the current ciphertext matches the trapdoor. However, the above construction has a security issue: two keyword ciphertexts can be linked if they have the same keyword and common receivers. To address the problem, we further randomize the polynomial in generating a keyword ciphertext to break the linkage.

Based on our above construction idea, we can see that neither the receiver nor the server requires the knowledge of intended receiver set in order to generate a trapdoor or perform a test, thus not impeding receiver anonymity, i.e., the current receiver needs not to recognize other intended receivers in order to search, and given ciphertext, the server learns nothing about intended receivers. Besides that the searchable ciphertext hides target receivers' identities (anonymity), a by-product is that the trapdoor hides the recipient identity in search (trapdoor anonymity), i.e., the server and other eavesdroppers cannot tell the recipient identity by observing the trapdoor, though they may be granted access to searchable ciphertexts (simulated by the ciphertext queries in security model). In addition, no predetermined universal keyword set is demanded and any keyword could be encrypted or searched, thereby maximizing the system scalability and flexibility. Moreover, the size and computational cost of public parameter and trapdoor are constant, which is more practical for the multi-receiver setting where a large number of trapdoors would be generated by different receivers.

1.2 Related Work

Broadcast encryption (BE) [15] was introduced in 1993. It is for broadcasting messages through public channel while keeping confidentiality. The message sender is to encrypt the message for a specified set of receivers so that only the intended receivers can access the message. BE outweighs the traditional point-to-point encryption in terms of that intended users are able to get the message by decrypting the same ciphertext. BE has been applied to content subscription and digital rights management in subsequent decades. The first fully collusion resistant scheme [6] was presented in 2005, where constant-size ciphertexts and private keys are obtained, but the size of public keys is still proportional to the maximal number of receivers. In 2007, the first identity-based broadcast encryption (IBBE) scheme [13] with constant-size ciphertexts and private keys was proposed by Delerabee, which is against adaptive chosen-ciphertext attacks

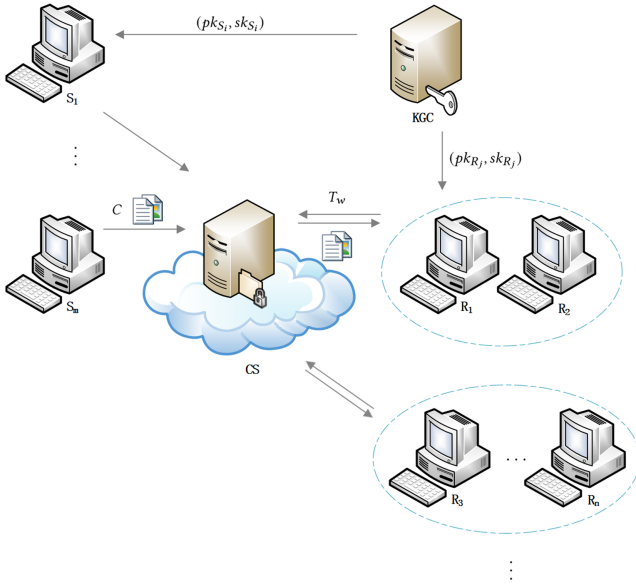


Fig. 1. BAEKS System Model. KGC: key generation center; CS: cloud server; S_i : a sender; R_j : a receiver.

(CCA) in the random oracle model. In 2009, Gentry and Waters first achieved the adaptive security in the standard model for IBBE [16]. In 2015, Kim et al. presented an adaptively CCA-secure IBBE scheme in the standard model [27] with a dual-system encryption technique. Researchers also worked on BE with special features such as user revocation [6, 32, 33] and constant-size ciphertexts and private keys [13, 16]. Anonymity is one of the desirable properties. With the digitization of each piece of information, identity is undoubtedly a kind of sensitive information. Conventional BE takes a receiver set as a part of ciphertext, exposing the identities of intended receivers. Anonymous BE schemes [2, 14, 20, 21, 30] were then constructed to tackle this problem.

Searchable encryption [34] is divided into two categories, searchable symmetric encryption (SSE) [12, 17, 24, 25, 28, 29] and PEKS [4]. Due to its intrinsic public-key characteristic, PEKS helps address the dilemma of key management and key abuse in the symmetric-key setting. However, PEKS encountered great challenges from KGA [9, 36] where adversaries can manufacture whatever ciphertexts of keywords of interest to test with a real trapdoor, learning the keywords being searched. One of the solutions to resisting such attacks is PAEKS [22]. PAEKS takes the sender’s secret key as input in addition to the receiver’s public key to ensure that no one else can forge a ciphertext for the test. There are also conceptions or applications such as certificateless PAEKS [19] derived from PAEKS. Another solution is to utilize the server-aided technique [10, 11], in which an assistant server is deployed to help resist KGA.

The idea of combining PEKS with BE is not new. In 2014, Ali et al. constructed a broadcast searchable encryption scheme [1] converted from Boneh et al.'s broadcast encryption [7]. Unfortunately, [1] is insecure against KGA. KGA can be launched on their scheme as follows. Anyone is able to manufacture a searchable ciphertext to test either their real searchable ciphertext or their real trapdoor, thereby unfolding the underlying keyword. In addition, it sounds quite unreasonable that their both trapdoor generation and test algorithm take the broadcast receiver set as input, which means anonymity is never guaranteed. In 2016, Kiayias et al. presented a broadcast keyword search scheme [26]. Unfortunately, the security models regarding anonymity were still not formalized in their work and their presented scheme's test algorithm still takes as input the set of intended receiver identities. In 2019, Jiang et al. introduced a primitive called identity-based broadcast encryption with keyword search [23] (IBEKS), combining PEKS with identity-based broadcast encryption to enable multiple intended receivers to search and decrypt the same ciphertext. Its searchable ciphertext generation takes the sender's secret key as input, preventing adversaries from manufacturing ciphertext to test real trapdoors. However, their trapdoor size and trapdoor computational complexity are linear to the number of the maximal number of receivers in the system. Moreover, the test algorithm requires the broadcast receiver set as input, which means the server needs to recognize all intended receivers before testing. A universal keyword set is chosen and predetermined in setup algorithm and keywords out of the set cannot be processed. Their security is proved on the intractability of Multi-Sequence of Exponents Decisional Diffie-Hellman Assumption (MSE-DDH). In conclusion, to the best of our knowledge, there has been no existing work addressed all the above problems simultaneously, including anonymity regarding both searchable ciphertext and trapdoor, defending KGA, and with universal keyword set scalability.

2 Preliminaries

2.1 Bilinear Map

Let $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear map, where \mathbb{G}, \mathbb{G}_T are two multiplicative cyclic groups of the same prime order p . It has the following properties [5]:

- Bilinearity: for any $a, b \in \mathbb{Z}_p, g, h \in \mathbb{G}, e(g^a, h^b) = e(g, h)^{ab}$.
- Non-degeneracy: for any generator $g \in \mathbb{G}, e(g, g) \neq 1$.
- Computability: for any $g, h \in \mathbb{G}, e(g, h)$ can be computed efficiently.

2.2 Decisional Bilinear Diffie Hellman Assumption

Given a generator $g \in \mathbb{G}$ and elements $g^a, g^b, g^c \in \mathbb{G}$ where $a, b, c \in \mathbb{Z}_p$ are randomly chosen elements, it is hard to distinguish $e(g, g)^{abc}$ from a random element $Z \in \mathbb{G}_T$ [8].

3 Syntax and Security Definitions

In this section, we first present the syntax and five algorithms of BAEKS. Then the formal security definitions of BAEKS including trapdoor privacy, ciphertext indistinguishability, trapdoor anonymity, and anonymity are presented.

3.1 Broadcast Authenticated Encryption with Keyword Search

- **Setup**(1^λ) $\rightarrow param$: Taking as input the security parameter 1^λ , it generates the public parameters $param$.
- **KeyGen**($param$) $\rightarrow (pk, sk)$: Taking as input the public parameter $param$, it generates a public/secret key pair (pk, sk) of an entity.
- **BAEKS**(w, sk_S, \mathcal{R}) $\rightarrow C$: Taking as input the keyword w , the sender's secret key sk_S and all intended receivers' public keys $\mathcal{R} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it generates the searchable ciphertext C .
- **Trapdoor**(w, pk_S, sk_{R_i}) $\rightarrow T_w$: Taking as input the keyword w , the sender's public key pk_S and the receiver's secret key sk_{R_i} , it generates trapdoor T_w .
- **Test**(T_w, C) $\rightarrow 1/0$: Taking as input a trapdoor T_w and a ciphertext C , it outputs 1 or 0.

Correctness. For any sender's keys $(pk_S, sk_S) \leftarrow \text{KeyGen}(param)$ and any receiver's keys $(pk_{R_i}, sk_{R_i}) \leftarrow \text{KeyGen}(param)$ for $R_i \in \mathcal{R}$, given a trapdoor $T_w \leftarrow \text{Trapdoor}(w, pk_S, sk_{R_i})$ generated by the receiver R_i of the broadcast set \mathcal{R} and a searchable ciphertext $C \leftarrow \text{BAEKS}(w, sk_S, \mathcal{R})$ generated by the sender S , the testing result must be $1 \leftarrow \text{Test}(T_w, C)$.

3.2 Security Models

Trapdoor Privacy. From intuition, the trapdoor should not reveal any sensitive information about its underlying keyword. Thus, we formulate a keyword distinguishing game to depict the security requirement for trapdoors given two trapdoors for distinct keywords from the same sender to the same receiver. To be noted, querying ciphertexts from the challenge sender and any receiver set containing the challenge receiver is prohibited to avoid trivial testing attacks.

1. **Setup:** Given the security parameter 1^λ , the challenger \mathcal{C} sends $param \leftarrow \text{Setup}(1^\lambda)$, the challenge sender's public key pk_S and the challenge receiver's public key pk_R to the adversary \mathcal{A} .
2. **Phase1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - Hash Queries: \mathcal{C} responds to hash queries with random numbers.
 - Ciphertext Queries: Given a keyword w , a receiver set's public keys $\tilde{\mathcal{R}} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it computes the ciphertext C with respect to sk_S and $\tilde{\mathcal{R}}$, and returns it to \mathcal{A} .
 - Trapdoor Queries: Given a keyword w , a sender's public key pk_S , it returns the trapdoor T_w with respect to sk_R and pk_S to \mathcal{A} .

3. **Challenge:** \mathcal{A} chooses two keywords w_0, w_1 such that (w_0, \mathcal{R}) and (w_1, \mathcal{R}) have not been queried for ciphertexts where $pk_R \in \mathcal{R}$, and (w_0, pk_S) and (w_1, pk_S) have not been queried for trapdoors, and sends them to \mathcal{C} . \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$, computes $T_{w_b} \leftarrow \text{Trapdoor}(w_b, pk_S, sk_R)$ and returns it to \mathcal{A} .
4. **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that neither (w_0, \mathcal{R}) nor (w_1, \mathcal{R}) can be queried for ciphertext where $pk_R \in \mathcal{R}$, and neither (w_0, pk_S) nor (w_1, pk_S) can be queried for trapdoor.
5. **Guess:** \mathcal{A} outputs a bit $b' \in \{0, 1\}$. It wins the game if $b' = b$.

We define the adversary \mathcal{A} 's advantage of successfully distinguishing the trapdoors of BAEKS as

$$Adv_{\mathcal{A}, BAEKS}^T(\lambda) = |Pr[b' = b] - \frac{1}{2}|.$$

Ciphertext Indistinguishability. Ciphertexts are required not to reveal any sensitive information about its underlying keyword as well. Thus, a keyword distinguishing game to set forth the security requirement for ciphertexts given two ciphertexts for different keywords from the same sender to the same broadcast receiver set. Here trapdoor queries from the challenge sender and any receiver of the challenge broadcast set should be refused to avoid trivial testing attacks.

1. **Setup:** Given the security parameter λ , the challenger \mathcal{C} sends $param \leftarrow \text{Setup}(\lambda)$, the challenge sender's public key pk_S and the challenge receiver set's public keys $\mathcal{R} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$ to the adversary \mathcal{A} .
2. **Phase1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - Hash Queries: \mathcal{C} responds to hash queries with random numbers.
 - Ciphertext Queries: Given a keyword w , a receiver set's public keys $\tilde{\mathcal{R}} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it returns the ciphertext C with respect to sk_S and $\tilde{\mathcal{R}}$ to \mathcal{A} .
 - Trapdoor Queries: Given a keyword w , a sender's public key \tilde{pk}_S , a chosen public key $pk_{R_i} \in \mathcal{R}$, it computes the trapdoor T_w with respect to sk_{R_i} and \tilde{pk}_S , returns it to \mathcal{A} .
3. **Challenge:** \mathcal{A} chooses two keywords w_0, w_1 such that (w_0, pk_S) and (w_1, pk_S) have not been queried for trapdoors, and sends them to \mathcal{C} . \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$, computes $C_b \leftarrow \text{BAEKS}(w_b, sk_S, \mathcal{R})$ and returns it to \mathcal{A} .
4. **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that neither (w_0, pk_S) nor (w_1, pk_S) can be queried for trapdoor.
5. **Guess:** \mathcal{A} outputs a bit $b' \in \{0, 1\}$. It wins the game if $b' = b$.

We define the adversary \mathcal{A} 's advantage of successfully distinguishing the ciphertexts of BAEKS as

$$Adv_{\mathcal{A}, BAEKS}^C(\lambda) = |Pr[b' = b] - \frac{1}{2}|.$$

Anonymity. Similar to anonymous broadcast encryption, ciphertexts are required not to reveal any sensitive information about their intended receivers. A broadcast receiver set distinguishing game describes the security requirement, in which adversary is to tell under which one of the two public key sets the challenge ciphertext for the identical keyword from the same sender is created. Here the two sets contain public keys of only one distinct receiver's public key pk_{R_0}/pk_{R_1} and $t - 1$ identical receivers' public keys. Trapdoor queries from the challenge sender and any of the two distinct receivers should not be responded to avoid trivial testing attacks.

1. **Setup:** Given the security parameter λ , the challenger \mathcal{C} sends $param \leftarrow \text{Setup}(\lambda)$, the challenge sender's public key pk_S and two different receiver set's public keys $\mathcal{R}_0 = \{pk_{R_0}, pk_{R_2}, \dots, pk_{R_t}\}, \mathcal{R}_1 = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$ of the same size to the adversary \mathcal{A} .
2. **Phase1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - Hash Queries: \mathcal{C} responds to hash queries with random numbers.
 - Ciphertext Queries: Given a keyword w , a receiver set's public keys $\tilde{\mathcal{R}} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it returns the ciphertext C with respect to sk_S and $\tilde{\mathcal{R}}$ to \mathcal{A} .
 - Trapdoor Queries: Given a keyword w , a sender's public key \tilde{pk}_S , a chosen public key from $\{pk_{R_0}, pk_{R_1}\}$, it computes the trapdoor T_w with respect to sk_{R_0} or sk_{R_1} , and \tilde{pk}_S , returns it to \mathcal{A} .
3. **Challenge:** \mathcal{A} chooses a keyword w^* such that (w^*, pk_S) has not been queried for trapdoors, and sends them to \mathcal{C} . \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$, computes $C_b \leftarrow \text{BAEKS}(w^*, sk_S, \mathcal{R}_b)$ and returns it to \mathcal{A} .
4. **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that (w^*, pk_S) cannot be queried for trapdoor.
5. **Guess:** \mathcal{A} outputs a bit $b' \in \{0, 1\}$. It wins the game if $b' = b$.

We define the adversary \mathcal{A} 's advantage of successfully breaking the anonymity of BAEKS as

$$Adv_{\mathcal{A}, \text{BAEKS}}^{\text{ANO}}(\lambda) = |Pr[b' = b] - \frac{1}{2}|.$$

Trapdoor Anonymity. While anonymity means that searchable ciphertext should not reveal intended recipients' identity, trapdoor anonymity implies that the trapdoor should not disclose any sensitive identity information about their maker, i.e., the receiver who is searching at present. Specifically, given two candidate receivers, the trapdoor fails to link the query to the user identity though testing on the current ciphertext can be utilized. A distinguishing game describes the security requirement for trapdoors for the identical keyword from the same sender to two distinct receivers. Of course, it should be restricted that both challenge receivers have the same inclusion relationship with the intended receiver set of the queried ciphertext C for the challenge keyword w^* , i.e., either $pk_{R_0}, pk_{R_1} \in \tilde{\mathcal{R}}$ or $pk_{R_0}, pk_{R_1} \notin \tilde{\mathcal{R}}$ in order to exclude the trivial testing attacks, i.e., distinguishing between the two receivers by running $\text{Test}(T_{w^*, b}, C) \rightarrow 1/0$.

1. **Setup:** Given the security parameter λ , the challenger \mathcal{C} sends $param \leftarrow \text{Setup}(\lambda)$, the challenge sender's public key pk_S and two different receivers' public keys pk_{R_0}, pk_{R_1} to the adversary \mathcal{A} .
2. **Phase1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - Hash Queries: \mathcal{C} responds to hash queries with random numbers.
 - Ciphertext Queries: Given a keyword w , a receiver set's public keys $\tilde{\mathcal{R}} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it returns the ciphertext C with respect to sk_S and $\tilde{\mathcal{R}}$ to \mathcal{A} .
 - Trapdoor Queries: Given a keyword w , a sender's public key \tilde{pk}_S , a chosen public key from $\{pk_{R_0}, pk_{R_1}\}$, it computes the trapdoor T_w with respect to sk_{R_0} or sk_{R_1} , and pk_S , returns it to \mathcal{A} .
3. **Challenge:** \mathcal{A} chooses a keyword w^* such that (w^*, pk_S) has not been queried for trapdoors, and (w^*, \mathcal{R}) has not been queried for ciphertexts where R_0, R_1 have different inclusion relationships with \mathcal{R} , and sends it to \mathcal{C} . \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$, computes $T_{w^*, b} \leftarrow \text{Trapdoor}(w^*, pk_S, sk_{R_b})$ and returns it to \mathcal{A} .
4. **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that neither (w^*, pk_S) can be queried for trapdoor, nor (w^*, \mathcal{R}) can be queried for ciphertexts where R_0, R_1 have different inclusion relationships with \mathcal{R} .
5. **Guess:** \mathcal{A} outputs a bit $b' \in \{0, 1\}$. It wins the game if $b' = b$.

We define the adversary \mathcal{A} 's advantage of successfully breaking the trapdoor anonymity of BAEKS as

$$Adv_{\mathcal{A}, \text{BAEKS}}^{T\text{-ANO}}(\lambda) = |\Pr[b' = b] - \frac{1}{2}|.$$

4 Broadcast Authenticated Encryption with Keyword Search

In this section, a concrete BAEKS scheme is proposed which has all the desired features as our expectation, followed by the correctness analysis.

4.1 Construction

- **Setup**(1^λ) $\rightarrow param$: Taking as input the security parameter 1^λ , it generates a bilinear map system $(p, \mathbb{G}, \mathbb{G}_T, e)$, where p is a prime s.t. $|p| = \lambda$, \mathbb{G} and \mathbb{G}_T are two cyclic groups with the same order p , e is a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. It picks random generators $g, u, v, z \in \mathbb{G}$, hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}, H_2 : \mathbb{G}_T \rightarrow \mathbb{Z}_p, H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_p$. The public parameters are $param = \{p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, z, H_1, H_2, H_3\}$.
- **KeyGen**($param$) $\rightarrow (pk, sk)$: Taking as input the public parameter $param$, it generates a random element $x \in \mathbb{Z}_p^*$, sets $sk = x, pk = g^x$ and outputs a public/secret key pair (pk, sk) .

- $\text{BAEKS}(w, sk_S, \mathcal{R}) \rightarrow C$: Taking as input the keyword w , the sender's secret key sk_S and all intended receivers' public keys $\mathcal{R} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, it chooses random elements $\tau, k, y \in \mathbb{Z}_p^*$. For $i = 1, 2, \dots, t$, computes $V_i = H_2(e(H_1(w)^{sk_S}, pk_{R_i}))$ and $f(x) = (x - y) \prod_{i \in \mathcal{R}} (x - V_i) + k = \sum_{j=0}^t a_j x^j + x^{t+1} \pmod{p}$, where a_j is the coefficient corresponding to x^j . It computes $A_j = g^{a_j}$ for $j = 0, 1, \dots, t$, $C_0 = g^k, h = H_3(C_0, A_0, A_1, \dots, A_t), C_1 = (u^h v^\tau z)^k$ and sets $C = (\tau, C_1, A_0, A_1, \dots, A_t)$.
- $\text{Trapdoor}(w, pk_S, sk_{R_i}) \rightarrow T_w$: Taking as input the keyword w , the sender's public key pk_S , and the receiver's secret key sk_{R_i} , it computes the trapdoor $T_w = H_2(e(H_1(w)^{sk_{R_i}}, pk_S))$.
- $\text{Test}(T_w, C) \rightarrow 1/0$: Taking as input a trapdoor T_w and a ciphertext $C = (\tau, C_1, A_0, A_1, \dots, A_t)$, it computes $C_0 = \prod_{j=0}^t A_j^{T_w^j} \cdot g^{T_w^{t+1}}, h = H_3(C_0, A_0, A_1, \dots, A_t)$. It outputs 1 if $e(C_1, g) = e(u^h v^\tau z, C_0)$; and 0 otherwise.

4.2 Correctness

Assume a trapdoor T_w and a searchable ciphertext $C = (\tau, C_1, A_0, A_1, \dots, A_t)$ are given to the server. Note that a trapdoor T_w generated by an intended receiver whose $pk_{R_i} \in \mathcal{R}$ is actually V_i that is used for constructing the searchable ciphertext:

$$V_i = e(H_1(w)^{sk_S}, pk_{R_i}) = e(H_1(w)^{sk_{R_i}}, pk_S) = T_w.$$

Then the server can recover the implied C'_0 using T_w as follows:

$$C'_0 = \prod_{j=0}^t A_0^{T_w^j} \cdot g^{T_w^{t+1}} = g^{\sum_{j=0}^t a_j T_w^j + T_w^{t+1}} = g^{f(T_w)} = g^{f(V_i)} = g^{k'}.$$

Obviously, the server can verify the searchable ciphertext C is the target one for the trapdoor T_w if the following equation holds:

$$e(C_1, g) = e((u^h v^\tau z)^k, g) = e(u^{h'} v^\tau z, g^{k'}) = e(u^{h'} v^\tau z, C'_0)$$

where $h' = H_3(C'_0, A_0, A_1, \dots, A_t)$.

Remark (Ciphertext Unlinkability). The random element $y \in \mathbb{Z}_p^*$ randomizes the searchable ciphertext C , specifically, the polynomial coefficients a_0, a_1, \dots, a_t , or A_0, A_1, \dots, A_t . Even in the case that two ciphertexts are encrypted for the same receiver set \mathcal{R} and the same keyword w , such randomization ensures the unlinkability for the two searchable ciphertexts.

5 Proof

In this section, we prove that our concrete scheme satisfies trapdoor privacy and ciphertext indistinguishability in accordance with our formulated security models. Due to space limitation, we only include theorems, and the proof of anonymity and trapdoor anonymity is detailed in the full version of this paper.

5.1 Trapdoor Privacy

Theorem 1. *If the adversary \mathcal{A} wins the trapdoor privacy game with advantage ϵ_T , then there exists a probabilistic polynomial time (PPT) adversary \mathcal{B} which can solve the DBDH problem with advantage*

$$\epsilon_{DBDH} \geq \epsilon_T \cdot \frac{2}{(q_T + q_C)e}$$

where q_T is the number of trapdoor queries and q_C is the number of ciphertext queries.

Proof. Assume that there is a PPT adversary \mathcal{A} which breaks the Trapdoor Privacy of our BAEKS scheme with a non-negligible advantage ϵ_C , then we can use it to construct another PPT algorithm \mathcal{B} to solve the DBDH problem.

- **Setup:** \mathcal{B} takes as input a DBDH problem instance, i.e. $(\mathbb{G}, \mathbb{G}_T, e, p, g, g^a, g^b, g^c, Z)$, where a, b, c are randomly chosen from \mathbb{Z}_p , and Z is either $e(g, g)^{abc}$ or a random element of G_T . Let β be a bit such that $\beta = 0$ if $Z = e(g, g)^{abc}$, and $\beta = 1$ if Z is random. \mathcal{B} randomly chooses generators $u, v, z \in_R \mathbb{G}$, hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}, H_2 : \mathbb{G}_T \rightarrow \mathbb{Z}_p, H_3 : \{0, 1\}^* \rightarrow \mathbb{Z}_p$ and sets $param = (p, \mathbb{G}, \mathbb{G}_T, e, g, u, v, z, H_1, H_2, H_3)$. \mathcal{B} sets $pk_S = g^a, pk_R = g^b$, sends $param$ and public keys to \mathcal{A} .
- **Phase 1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - H_1 Queries: \mathcal{B} maintains a list L_1 , which is initiated empty and contains tuples $\langle w, \cdot, \cdot \rangle$. Upon a query w_l , if the tuple $\langle w_l, d_l, h_{1,l} \rangle$ is already in L_1 , \mathcal{B} returns h_l ; otherwise, \mathcal{B} randomly chooses $d_l \in \mathbb{Z}_q^*$, tosses a coin γ_l such that $\Pr[\gamma_l = 0] = \delta$.
 1. If $\gamma_l = 0$, computes $h_{1,l} = g^{c \cdot d_l}$;
 2. otherwise, computes $h_{1,l} = g^{d_l}$. \mathcal{B} adds $\langle w_l, \gamma_l, d_l, h_{1,l} \rangle$ to L_1 and returns $h_{1,l}$.
 - H_2 Queries: \mathcal{B} maintains a list L_2 , which is initiated empty and contains tuples $\langle \alpha, \cdot \rangle$. Upon a query α , if the tuple $\langle \alpha, h_2 \rangle$ is already in L_2 , \mathcal{B} returns h_2 ; otherwise, \mathcal{B} randomly chooses $h_2 \in \mathbb{Z}_p^*$, adds $\langle \alpha, h_2 \rangle$ to L_2 and returns h_2 .
 - H_3 Queries: \mathcal{B} maintains a list L_3 , which is initiated empty and contains tuples $\langle \gamma, \cdot \rangle$. Upon a query γ , if the tuple $\langle \gamma, h_3 \rangle$ is already in L_3 , \mathcal{B} returns h_3 ; otherwise, \mathcal{B} randomly chooses $h_3 \in \mathbb{Z}_p^*$, adds $\langle \gamma, h_3 \rangle$ to L_3 and returns h_3 .
 - Ciphertext Queries: Given a keyword w_l , a receiver set's public keys $\tilde{\mathcal{R}} = \{pk_{R_1}, pk_{R_2}, \dots, pk_{R_t}\}$, \mathcal{B} first looks up L_1 to find the entry $\langle w_l, \gamma_l, d_l, h_{1,l} \rangle$.
 1. If $\gamma_l = 0$, aborts;
 2. otherwise, for each $pk_{R_i} \in \tilde{\mathcal{R}}$, computes $\alpha_i = e(g^a, pk_{R_i})^{d_l}$, looks up L_2 to find the entry $\langle \alpha_i, h_{2,i} \rangle$. If there is no such entry, randomly chooses $h_{2,i} \in \mathbb{Z}_p^*$, adds $\langle \alpha_i, h_{2,i} \rangle$ to L_2 , and sets $V_i = h_{2,i}$. \mathcal{B}

randomly picks $\tau, k, y \in_R \mathbb{Z}_p^*$, computes $f(x) = (x - y) \prod_{i \in \mathcal{R}} (x - V_i) + k = \sum_{j=0}^t a_j x^j + x^{t+1} \pmod{p}$, where a_j is the coefficient corresponding to x^j . It computes $A_j = g^{a_j}$ for $j = 0, 1, \dots, t$, $C_0 = g^k$, $h = H_3(C_0, A_0, A_1, \dots, A_t)$, $C_1 = (u^h v^\tau z)^k$ and sets $C = (\tau, C_1, A_0, A_1, \dots, A_t)$.

- **Trapdoor Queries:** Given a keyword w_l , a sender's public key \tilde{pk}_S , \mathcal{B} first looks up L_1 to find the entry $\langle w_l, \gamma_l, d_l, h_{1,l} \rangle$.
 1. If $\gamma_l = 0$, aborts;
 2. otherwise, computes $\alpha = e(g^b, \tilde{pk}_S)^{d_l}$, looks up to L_2 to find the entry $\langle \alpha, h_2 \rangle$. If there is no such entry, randomly chooses $h_2 \in \mathbb{Z}_p^*$, adds $\langle \alpha, h_2 \rangle$ to L_2 , and returns $T_w = h_2$.
- **Challenge:** \mathcal{A} chooses two distinct keywords w_0, w_1 such that (w_0, \mathcal{R}) and (w_1, \mathcal{R}) have not been queried for ciphertexts where $pk_R \in \mathcal{R}$, and (w_0, pk_S) and (w_1, pk_S) have not been queried for trapdoors, and sends them to \mathcal{B} . \mathcal{B} randomly chooses a bit $\beta \in \{0, 1\}$, looks up L_1 to find the entries $\langle w_0, \gamma_0, d_0, h_{1,0} \rangle$ and $\langle w_1, \gamma_1, d_1, h_{1,1} \rangle$,
 1. if $\gamma_0 = \gamma_1 = 1$, aborts;
 2. otherwise, computes $\alpha = Z^{d_\beta}$, looks up to L_2 to find the entry $\langle \alpha, h_2 \rangle$ and returns $T_w^* = h_2$ to \mathcal{A} .
- **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that neither (w_0, \mathcal{R}) nor (w_1, \mathcal{R}) can be queried for ciphertext where $pk_R \in \mathcal{R}$, and neither (w_0, pk_S) nor (w_1, pk_S) can be queried for trapdoor.
- **Guess:** \mathcal{A} outputs a bit β' . If $\beta' = \beta$, \mathcal{B} outputs 0, otherwise 1.

Here we use **abt** to denote the event that \mathcal{B} aborts in the game. There are two cases in which **abt** happens.

1. The event that $\gamma_l = 0$ in trapdoor and ciphertext queries. We denote it as **abt₁**. The probability that **abt₁** does not happen:

$$Pr[\neg \text{abt}_1] = (1 - \delta)^{q_T + q_C}$$

2. The event that $\gamma_0 = \gamma_1 = 1$ in challenge. We denote it as **abt₂**. The probability that **abt₂** does not happen:

$$Pr[\neg \text{abt}_2] = 1 - (1 - \delta)^2$$

Then the probability that \mathcal{B} does not abort is:

$$Pr[\neg \text{abt}] = Pr[\neg \text{abt}_1] \cdot Pr[\neg \text{abt}_2] = (1 - \delta)^{q_T + q_C} \cdot (1 - (1 - \delta)^2).$$

When $\delta = 1 - \sqrt{\frac{q_T + q_C}{q_T + q_C + 2}}$, the above probability takes the maximum, $Pr[\neg \text{abt}]$ approximately equals $\frac{2}{(q_T + q_C)e}$, which is non-negligible since q_T, q_C are polynomials and e is the natural logarithm base.

Thus, the probability that \mathcal{B} solves the DBDH problem is

$$\begin{aligned}
 \Pr[b' = b] &= \Pr[b' = b \wedge \text{abt}] + \Pr[b' = b \wedge \neg \text{abt}] \\
 &= \Pr[b' = b | \text{abt}] \cdot \Pr[\text{abt}] + \Pr[b' = b | \neg \text{abt}] \cdot \Pr[\neg \text{abt}] \\
 &= \frac{1}{2} \cdot (1 - \Pr[\neg \text{abt}]) + (\epsilon_T + \frac{1}{2}) \cdot \Pr[\neg \text{abt}] \\
 &= \frac{1}{2} + \epsilon_T \cdot \Pr[\neg \text{abt}]
 \end{aligned}$$

If ϵ_T and $\Pr[\neg \text{abt}]$ are non-negligible, so is

$$\epsilon_{DBDH} = |\Pr[b' = b] - \frac{1}{2}| \geq \epsilon_T \cdot \frac{2}{(q_T + q_C)e}.$$

5.2 Ciphertext Indistinguishability

Theorem 2. *If the adversary \mathcal{A} wins the ciphertext indistinguishability game with advantage ϵ_C , then there exists a PPT adversary \mathcal{B} which can solve the DBDH problem with advantage*

$$\epsilon_{DBDH} \geq \epsilon_C \cdot \frac{2}{(q_T + q_C)e}$$

where q_T is the number of trapdoor queries and q_C is the number of ciphertext queries.

Proof. Assume that there is a PPT adversary \mathcal{A} which breaks the Trapdoor Privacy of our BAEKS scheme with a non-negligible advantage ϵ_C , then we can use it to construct another PPT algorithm \mathcal{B} to solve the DBDH problem.

- **Setup:** Public parameter generation is same as Trapdoor Privacy game. \mathcal{B} sets $pk_S = g^a$, $\mathcal{R} = \{pk_{R_1^*}, pk_{R_2^*}, \dots, pk_{R_t^*}\} = \{g^{b \cdot r_1^*}, g^{b \cdot r_2^*}, \dots, g^{b \cdot r_t^*}\}$ where $r_i^* \in_R \mathbb{Z}_p^*$, and sends *param* and public keys to \mathcal{A} .
- **Phase 1:** \mathcal{A} is allowed to adaptively issue the following queries.
 - H_1, H_2, H_3 and Ciphertext Queries: same as Trapdoor Privacy game.
 - Trapdoor Queries: Given a keyword w_l , a sender's public key pk_S , a chosen public key $pk_{R_i^*} \in \mathcal{R}$, \mathcal{B} first looks up L_1 to find the entry $\langle w_l, \gamma_l, d_l, h_{1,l} \rangle$.
 1. If $\gamma_l = 0$, aborts;
 2. otherwise, computes $\alpha = e(g^b, pk_S)^{r_i^* \cdot d_l}$, looks up to L_2 to find the entry $\langle \alpha, h_2 \rangle$. If there is no such entry, randomly chooses $h_2 \in \mathbb{Z}_p^*$, adds $\langle \alpha, h_2 \rangle$ to L_2 , and returns $T_w = h_2$.
- **Challenge:** \mathcal{A} chooses two distinct keywords w_0, w_1 such that (w_0, pk_S) and (w_1, pk_S) have not been queried for trapdoors, and sends them to \mathcal{B} . \mathcal{B} randomly chooses a bit $\beta \in \{0, 1\}$, looks up L_1 to find the entries $\langle w_0, \gamma_0, d_0, h_{1,0} \rangle$ and $\langle w_1, \gamma_1, d_1, h_{1,1} \rangle$,
 1. if $\gamma_0 = \gamma_1 = 1$, aborts;

2. otherwise, for each $pk_{R_i^*} \in \mathcal{R}$ computes $\alpha_i = Z^{d_{\beta \cdot r_i^*}}$, looks up L_2 to find the entry $\langle \alpha_i, h_{2,i} \rangle$ and sets $V_i = h_{2,i}$. Randomly picks $\tau, k, y \in_R \mathbb{Z}_p^*$, computes $f(x) = (x - y) \prod_{i \in \mathcal{R}} (x - V_i) + k = \sum_{j=0}^t a_j x^j + x^{t+1} \pmod{p}$, where a_j is the coefficient corresponding to x^j . It computes $A_j = g^{a_j}$ for $j = 0, 1, \dots, t$, $C_0 = g^k, h = H_3(C_0, A_0, A_1, \dots, A_t), C_1 = (u^h v^\tau z)^k$ and sets $C = (\tau, C_1, A_0, A_1, \dots, A_t)$.
- **Phase2:** \mathcal{A} continues to issue queries as above, with restriction that neither (w_0, pk_S) nor (w_1, pk_S) can be queried for trapdoor.
 - **Guess:** \mathcal{A} outputs a bit β' . If $\beta' = \beta$, \mathcal{B} outputs 0, otherwise 1.

Here we use **abt** to denote the event that \mathcal{B} aborts in the game. There are two cases in which **abt** happens.

1. The event that $\gamma_l = 0$ in trapdoor and ciphertext queries. We denote it as **abt₁**. The probability that **abt₁** does not happen:

$$Pr[\neg \text{abt}_1] = (1 - \delta)^{q_T + q_C}$$

2. The event that $\gamma_0 = \gamma_1 = 1$ in challenge. We denote it as **abt₂**. The probability that **abt₂** does not happen:

$$Pr[\neg \text{abt}_2] = 1 - (1 - \delta)^2$$

Then the probability that \mathcal{B} does not abort is:

$$Pr[\neg \text{abt}] = Pr[\neg \text{abt}_1] \cdot Pr[\neg \text{abt}_2] = (1 - \delta)^{q_T + q_C} \cdot (1 - (1 - \delta)^2).$$

When $\delta = 1 - \sqrt{\frac{q_T + q_C}{q_T + q_C + 2}}$, the above probability takes the maximum, $Pr[\neg \text{abt}]$ approximately equals $\frac{2}{(q_T + q_C)e}$, which is non-negligible since q_T, q_C are polynomials and e is the natural logarithm base.

Thus, the probability that \mathcal{B} solves the DBDH problem is

$$\begin{aligned} Pr[b' = b] &= Pr[b' = b \wedge \text{abt}] + Pr[b' = b \wedge \neg \text{abt}] \\ &= Pr[b' = b | \text{abt}] \cdot Pr[\text{abt}] + Pr[b' = b | \neg \text{abt}] \cdot Pr[\neg \text{abt}] \\ &= \frac{1}{2} \cdot (1 - Pr[\neg \text{abt}]) + (\epsilon_C + \frac{1}{2}) \cdot Pr[\neg \text{abt}] \\ &= \frac{1}{2} + \epsilon_C \cdot Pr[\neg \text{abt}] \end{aligned}$$

If ϵ_C and $Pr[\neg \text{abt}]$ are non-negligible, so is

$$\epsilon_{DBDH} = |Pr[b' = b] - \frac{1}{2}| \geq \epsilon_C \cdot \frac{2}{(q_T + q_C)e}.$$

5.3 Anonymity and Trapdoor Anonymity

Theorem 3. *If the adversary \mathcal{A} wins the anonymity game with advantage ϵ_{ANO} , then there exists a PPT adversary \mathcal{B} which can solve the DBDH problem with advantage*

$$\epsilon_{DBDH} \geq \epsilon_{ANO} \cdot \frac{1}{(q_T + q_C + 1)e}$$

where q_T is the number of trapdoor queries and q_C is the number of ciphertext queries.

Theorem 4. *If the adversary \mathcal{A} wins the trapdoor anonymity game with advantage ϵ_{T-ANO} , then there exists a PPT adversary \mathcal{B} which can solve the DBDH problem with advantage*

$$\epsilon_{DBDH} \geq \epsilon_{T-ANO} \cdot \frac{1}{(q_T + q_C + 1)e}$$

where q_T is the number of trapdoor queries and q_C is the number of ciphertext queries.

6 Comparison with Existing Works

To the best of our knowledge, the IBEKS of [23] is the only existing multi-receiver keyword search scheme with KGA resistance before this work. A detailed functionality comparison between IBEKS [23] and our BAEKS is given in Table 1. Table 2 and Table 3 provide comparisons of computation cost and communication

Table 1. Functionality Comparison between [23] and Ours

	KGA resistance	Anonymity	Universal keyword set scalability	Assumption
[23]	✓	×	×	MSE-DDH
Ours	✓	✓	✓	DBDH

Table 2. Computation Cost Comparison

	Encrypt	Trapdoor	Test
[23]	$(2n + 4)G_e$	$(2n + 2)G_e + nG_p$	$2tG_e + 3G_p$
Ours	$(2t + 5)G_e + tG_p$	$G_e + G_p$	$(t + 4)G_e + 2G_p$

Table 3. Communication Complexity Comparison

	Public parameter size	Secret key size	Trapdoor size	Ciphertext size
[23]	$((2n + 1) + l(n + 2)) G $	$ Z_p + 2 G $	$(n + 1) G + n G_T $	$3 G $
Ours	$4 G $	$ Z_p $	$ Z_p $	$ Z_p + (t + 2) G $

overhead. \checkmark means “satisfy”, \times refers to “not satisfy”. n denotes the maximal number of receivers in the system, t denotes the number of intended broadcast receivers and l denotes the number of keywords of the universal keyword set. $|Z_p|$ refers to the element size of field \mathbb{Z}_p , $|G|$ refers to the element bit-length of group \mathbb{G} , and $|G_T|$ refers to the element bit-length of group \mathbb{G}_T . \mathbf{G}_e refers to exponentiation, \mathbf{G}_p refers to pairing.

As described in Table 1, both [23] and our scheme takes the sender’s secret key as input to authenticate the keyword when encrypting, hence they are immune to KGA. In terms of anonymity, [23] takes all the broadcast receiver identity information as the input of the test algorithm, while ours needs no such input and is proven to ensure anonymity as well as trapdoor anonymity. The universal keyword set is predetermined in setup algorithm and keywords out of the universal set cannot be encrypted and searched in [23], while there is no keyword limitation when encrypting or searching in ours. [23] is proved secure based on MSE-DDH, while our scheme is proved secure based on a simple and standard assumption DBDH.

Since calculation other than exponentiation and pairing are far less time-consuming, we merely evaluate and analyze the complexity of exponentiation and pairing. The computational complexity of [23]’s encryption is linear to the number of the maximal number of receivers $\mathcal{O}(n)$, so is that of their trapdoor generation. In contrast, our encryption computational complexity is only proportional to the number of intended broadcast receivers $\mathcal{O}(t)$, which is no greater than the maximal number of receivers. Our trapdoor generation complexity is constant $\mathcal{O}(1)$. In the comparison of test computation, even though both schemes’ cost is linear to the number of intended broadcast receivers $\mathcal{O}(t)$, our scheme’s actual cost is less than [23]. Details can be found in Table 2.

According to Table 3, in spite of the ciphertext size of [23] is constant $\mathcal{O}(1)$ and smaller than ours $\mathcal{O}(t)$, our performance on all the remaining sizes (public parameter size, secret key size, and trapdoor size) is better than theirs. Their public parameter size is not only linear to the maximal number of receivers in the system n but also proportional to the number of the universal keyword set size l , while ours is constant. Both schemes’ secret key size is constant but our specific complexity is smaller. Their trapdoor size grows with the number of the maximal number of receivers n , while ours remains unchanged.

In short, our scheme outperforms [23] on functionality, computation cost and communication complexity.

7 Conclusion

We first introduced a cryptographic primitive called broadcast authenticated encryption with keyword search that engages in authenticated keyword search in broadcast mode. The subsequent detailed scheme elegantly avoids the trapdoor size increasing with the number of broadcast receivers, requires no universal keyword set and is proved secure based on a simple and standard assumption. Moreover, its desirable properties, i.e., anonymity and trapdoor anonymity surpass the performance of existing constructions. Therefore, it accommodates the

demand for multi-user access, achieves competitive computational complexity and comprehensive security. We leave reducing the ciphertext size to a constant as an open problem and our future work.

References

1. Ali, M., Ali, H., Zhong, T., Li, F., Qin, Z., Abdelrahman, A.A.: Broadcast searchable keyword encryption. In: 2014 IEEE 17th International Conference on Computational Science and Engineering, pp. 1010–1016. IEEE (2014)
2. Barth, A., Boneh, D., Waters, B.: Privacy in encrypted content distribution using private broadcast encryption. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 52–64. Springer, Heidelberg (2006). https://doi.org/10.1007/11889663_4
3. Bellare, M., Boldyreva, A., Desai, A., Pointcheval, D.: Key-privacy in public-key encryption. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 566–582. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_33
4. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_30
5. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_13
6. Boneh, D., Gentry, C., Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 258–275. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_16
7. Boneh, D., Hamburg, M.: Generalized identity based and broadcast encryption schemes. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 455–470. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89255-7_28
8. Boyen, X.: The uber-assumption family. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing 2008. LNCS, vol. 5209, pp. 39–56. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85538-5_3
9. Byun, J.W., Rhee, H.S., Park, H.-A., Lee, D.H.: Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In: Jonker, W., Petković, M. (eds.) SDM 2006. LNCS, vol. 4165, pp. 75–83. Springer, Heidelberg (2006). https://doi.org/10.1007/11844662_6
10. Chen, R., et al.: Server-aided public key encryption with keyword search. *IEEE Trans. Inf. Forensics Secur.* **11**(12), 2833–2842 (2016)
11. Chen, R., Mu, Y., Yang, G., Guo, F., Wang, X.: Dual-server public-key encryption with keyword search for secure cloud storage. *IEEE Trans. Inf. Forensics Secur.* **11**(4), 789–798 (2015)
12. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. *J. Comput. Secur.* **19**(5), 895–934 (2011)
13. Delerablée, C.: Identity-based broadcast encryption with constant size ciphertexts and private keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76900-2_12

14. Fazio, N., Perera, I.M.: Outsider-anonymous broadcast encryption with sublinear ciphertexts. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 225–242. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_14
15. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_40
16. Gentry, C., Waters, B.: Adaptive security in broadcast encryption systems (with short ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_10
17. Goh, E.J., et al.: Secure indexes. IACR Cryptology ePrint Archive **2003**, 216 (2003)
18. Goldman, E.: An introduction to the California Consumer Privacy Act (CCPA). Santa Clara University, Legal Studies Research Paper (2020)
19. He, D., Ma, M., Zeadally, S., Kumar, N., Liang, K.: Certificateless public key authenticated encryption with keyword search for industrial internet of things. IEEE Trans. Ind. Inf. **14**(8), 3618–3627 (2017)
20. He, K., Weng, J., Au, M.H., Mao, Y., Deng, R.H.: Generic anonymous identity-based broadcast encryption with chosen-ciphertext security. In: Liu, J.K., Steinfeld, R. (eds.) ACISP 2016. LNCS, vol. 9723, pp. 207–222. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40367-0_13
21. He, K., Weng, J., Liu, J., Liu, J.K., Liu, W., Deng, R.H.: Anonymous identity-based broadcast encryption with chosen-ciphertext security. In: Chen, X., Wang, X., Huang, X. (eds.) Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2016, Xi'an, China, 30 May–3 June 2016, pp. 247–255. ACM (2016)
22. Huang, Q., Li, H.: An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks. Inf. Sci. **403**, 1–14 (2017)
23. Jiang, P., Guo, F., Mu, Y.: Efficient identity-based broadcast encryption with keyword search against insider attacks for database systems. Theor. Comput. Sci. **767**, 51–72 (2019)
24. Kamara, S., Papamanthou, C.: Parallel and dynamic searchable symmetric encryption. In: Sadeghi, A.-R. (ed.) FC 2013. LNCS, vol. 7859, pp. 258–274. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39884-1_22
25. Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, pp. 965–976 (2012)
26. Kiayias, A., Oksuz, O., Russell, A., Tang, Q., Wang, B.: Efficient encrypted keyword search for multi-user data sharing. In: Askoxylakis, I., Ioannidis, S., Katsikas, S., Meadows, C. (eds.) ESORICS 2016. LNCS, vol. 9878, pp. 173–195. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45744-4_9
27. Kim, J., Susilo, W., Au, M.H., Seberry, J.: Adaptively secure identity-based broadcast encryption with a constant-sized ciphertext. IEEE Trans. Inf. Forensics Secur. **10**(3), 679–693 (2015)
28. Kurosawa, K., Ohtaki, Y.: UC-secure searchable symmetric encryption. In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol. 7397, pp. 285–298. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32946-3_21
29. Kurosawa, K., Ohtaki, Y.: How to update documents *verifiably* in searchable symmetric encryption. In: Abdalla, M., Nita-Rotaru, C., Dahab, R. (eds.) CANS 2013. LNCS, vol. 8257, pp. 309–328. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-02937-5_17

30. Libert, B., Paterson, K.G., Quaglia, E.A.: Anonymous broadcast encryption: adaptive security and efficient constructions in the standard model. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 206–224. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30057-8_13
31. Liu, X., Yang, G., Mu, Y., Deng, R.: Multi-user verifiable searchable symmetric encryption for cloud storage. *IEEE Trans. Dependable Secure Comput.* (2018)
32. Lotspiech, J.B., Naor, D., Naor, S.: Method for broadcast encryption and key revocation of stateless receivers. US Patent 7,039,803, 2 May 2006
33. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_3
34. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: *Proceeding 2000 IEEE Symposium on Security and Privacy, S&P 2000*, pp. 44–55. IEEE (2000)
35. Voigt, P., Von dem Bussche, A.: *The EU General Data Protection Regulation (GDPR). A Practical Guide*, 1st edn. Springer, Cham (2017)
36. Yau, W.-C., Heng, S.-H., Goi, B.-M.: Off-line keyword guessing attacks on recent public key encryption with keyword search schemes. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) ATC 2008. LNCS, vol. 5060, pp. 100–105. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-69295-9_10