Public Key Encryption with Keyword Search based on K-Resilient IBE

Dalia Khader [1]

University of Bath, Department of Computer Science,
ddk20@bath.ac.uk

Abstract. An encrypted email is sent from Bob to Alice. A gateway wants to check whether a certain keyword exists in an email or not for some reason (e.g. routing). Nevertheless Alice does not want the email to be decrypted by anyone except her including the gateway itself. This is a scenario where public key encryption with keyword search (PEKS) is needed. In this paper we construct a new scheme (KR-PEKS) the K-Resilient Public Key Encryption with Keyword Search. The new scheme is secure under a chosen keyword attack without the random oracle. The ability of constructing a Public Key Encryption with Keyword Search from an Identity Based Encryption was used in the construction of the KR-PEKS. The security of the new scheme was proved by showing that the used IBE has a notion of key privacy. The scheme was then modified in two different ways in order to fulfill each of the following: the first modification was done to enable multiple keyword search and the other was done to remove the need of secure channels.

1 Introduction

Bob wants to send Alice confidential emails and in order to ensure that no one except her can read it, he encrypts the emails before sending them so that Alice, and her alone, will possess the capability of decrypting it. Consider the scenario where Alice would like to download only the urgent emails to her mobile, leaving the rest to check later from her computer. Alice requires access to the email server (gateway) so that she can search her emails for the keyword "urgent" prior to downloading them. The server should facilitate the search for this keyword without being able to decrypt Alice's private emails.

This scenario was first introduced in Boneh et al.'s paper [3]. They presented a general scheme called PEKS where Alice gives trapdoors for the words she wants the gateway to search for. The trapdoors come in the form of some kind of data that is used to test the existence of keywords within an email without revealing any other information(Section 3).

In [3] the authors constructed several schemes based on different security models but these schemes either had some limitation on the number of keywords to search for or were not secure enough (ie. were proven secure in random oracle). In [1] the authors pointed out two important features that were not covered in [3]. The first one was the ability to search for multiple keywords. The second characteristic, put forwards in Section 3.1 of this paper, was elimination of the requirement of secure channels, for sending trapdoors. These two new features issues are explained in details in the paper and new schemes to facilitate them will be introduced in this report paper(Section 3.6 and 3.7).

The ability of constructing IBE from PEKS was explored in [3]. IBE is a public key encryption where the public key is a direct product of the identity of the user [11] [4]. Building a PEKS from an IBE needs the latter to have some extra properties such as the notion of key privacy. Key privacy is a security property that implies that an adversary should not be able to guess which ID from a set of ID's was used in encrypting some email(Section 3.3).

Heng and Kurosawain [9] proposed a scheme called K-resilient IBE, this scheme does not lead to breach of privacy (Appendix A). The basic K-resilient IBE was used to construct a PEKS that is fully secure (Section 3.4).

The next Section entails some important concepts that form the foundation for our work. Following that, Section 3 explains the PEKS scheme, its security notions, the relation with IBE, and last but not least the construction of a new PEKS scheme using the K-Resilient IBE was introduced. The last Section in this paper concludes the results of our work.

2 Preliminaries

In this Section we will go through some definitions that will be used further in this document.

2.1 Decisional Diffie-Hellman

The Decisional Diffie Hellman (DDH) Problem [6] is the ability to distinguish between $\langle g, g^a, g^b, g^{ab}\rangle$ and $\langle g, g^a, g^b, T\rangle$ where $a, b, c \in Z_q$,$g \in G_q$, G_q is a group of prime order q, and T is a random element that belongs to G_q.

The quadruple $\langle g, g^a, g^b, g^{ab}\rangle$ is called the real quadruple and the quadruple $\langle g, g^a, g^b, T\rangle$ is called the random quadruple. So if we have an adversary D that takes $X, Y, T \in G_q$ and returns a bit $d \in \{0, 1\}$, consider the following two experiments.

For both experiments $X \leftarrow g^x$; $Y \leftarrow g^y$

```
•$Exp^{ddh-real}_{G_q,D}$
  $T \leftarrow g^{xy}$
  $d \leftarrow D(q, g, X, Y, T)$
  Return $d$
```

```
•$Exp^{ddh-rand}_{G_q,D}$
  A random $T \in G_q$
  $d \leftarrow D(q, g, X, Y, T)$
  Return $d$
```

The advantage of the adversary D in solving the Diffie-Hellman is defined as follows:-

$$Adv^{ddh}_{G_q,D} = Pr[Exp^{ddh-real}_{G_q,D} = 1] - Pr[Exp^{ddh-rand}_{G_q,D} = 1]$$

If this advantage is negligible for any adversary D then we say DDH is hard to solve. The DDH was used in PEKS Section 3.4.

2.2 Hash Functions

A family of hash functions $\mathcal{H} = (G; H)$ is defined by two algorithms [8]. G is a probabilistic generator algorithm that takes the security parameter k as input and returns a key K. H is a deterministic evaluation algorithm that takes the key K and a string $M \in \{0,1\}^*$ and returns a string $H_k(M) \in \{0,1\}^{k-1}$.

Definition Let $\mathcal{H} = (G; H)$ be a family of hash functions and let C be an adversary [8]. We consider the following experiment:
Experiment $Exp^{cr}_{\mathcal{H},C}(k)$
$K \leftarrow G(k); (x_0; x_1) \leftarrow C(K)$
if $((x_0 \neq x_1) \wedge (H_K(x_0) = H_K(x_1)))$ then return 1 else return 0.
We define the advantage of C via $Adv^{cr}_{\mathcal{H},C}(k) = Pr[Exp^{cr}_{\mathcal{H},C}(k) = 1]$:
The family of hash functions $\mathcal{H}$ is collision-resistant if the advantage of C is negligible for every algorithm C whose time-complexity is polynomial in k.

3 Public Key Encryption with Keyword Search

An encrypted email is sent from Bob to Alice [3]. The gateway wants to check whether a certain keyword exists in an email or not for some reason (for example routing). Nevertheless Alice does not want the email to be decrypted by anyone except her, not even the gateway itself. This is a scenario where public key encryption with keyword search (PEKS) is needed. PEKS encrypts the keywords in a different manner than the rest of the email. The gateway is given "trapdoors" corresponding to particular keywords. Using the PEKS of a word and trapdoor of a keyword, the gateway can test whether the encrypted word is the particular keyword or not.

General scheme According to [3] a PEKS consists of four algorithms as described below :

- `KeyGen(`s`)`: Take a security parameter s and generate two keys a public key A_{pub} and private key A_{priv}
- `PEKS(`A_{pub}, W`)`: It produces a searchable encryption for a keyword W using a public key A_{pub}
- `Trapdoor(`A_{priv}, W`)`: Produce a trapdoor for a certain word using the private key.
- `Test(`A_{pub}, S, T_w`)`: Given the public key A_{pub}, some searchable encryption S where $S = PEKS(A_{pub}, W')$, and the trapdoor T_w to a keyword W. Determine whether or not the word we are looking for W and the word encrypted W' are equal.

So Bob sends Alice through the gateway the following:

$$[E(A_{pub}, M), PEKS(A_{pub}, W_1), PEKS(A_{pub}, W_2), ..., PEKS(A_{pub}, W_m)]$$

where $PEKS(A_{pub}, W_i)$ is a searchable encryption of the keywords and $E(A_{pub}, M)$ is a standard public key encryption of the rest of the message M.

3.1 Security Notions Related to PEKS

Security under a Chosen Keyword Attack (CKA) For a PEKS to be considered secure we need to guarantee that no information about a keyword is revealed unless the trapdoor of that word is available [3]. To define security against an active adversary A we use the following game between A and challenger.

- `CKA-Setup`: The challenger runs the key generation algorithm and gives the A_{pub} to adversary A and keeps A_{priv} to itself.
- `CKA-Phase 1`: A asks the challenger for trapdoors corresponding to keywords of its choice.
- `CKA-Challenge`: The adversary decides when phase 1 ends. Then it chooses two words W_0, W_1 to be challenged on. The two words should not be among those for which A obtained a trapdoor in phase 1. The challenger picks a random bit $b \in \{0, 1\}$ and gives attacker. $C = PEKS(A_{pub}, W_b)$.
- `CKA-Phase 2`: A asks for more trapdoors like in phase 1 for any word of its choice except for the W_0, W_1.
- `CKA-Guess`: A outputs its guess of b' and if $b' = b$ that means A guessed the encrypted message and the adversary wins.

We say that the scheme is secure against a chosen keyword attack (CKA) if A has a low advantage of guessing the right word being encrypted.

Secure Channels: In the PEKS scheme in [3] there is a need to have a secure channel between Alice and the server, so that an eavesdropper (Eve) can not get hold of the trapdoors sent. No one but the server should be capable of testing emails for certain keywords. This is one of the drawback that the authors of [1] tried to solve by generating a public and a private key that belong to the server. The PEKS algorithm was modified to encrypt keywords using both Alice's and the server's public key, while the testing algorithm needs the server's private key as an input. In this way the scheme is secure channel free (SCF-PEKS) because Eve can not obtain the server's private key, therefore can not test.
The SCF-PEKS is said to be IND-SCF-CKA secure when it ensures that the server that has obtained the trapdoors for given keywords cannot tell a PEKS ciphertext is the result of encrypting which keyword, and an outsider adversary that did not get the server's private key cannot distinguish the PEKS ciphertexts, even if it gets all the trapdoors for the keywords that it queries

3.2 Handling Multiple Keywords

Multiple Keyword search in a PEKS is the capability of searching for more than one word either disjunctively or conjunctively. In PEKS [3] the only way to do this is to search for each word separately and then do the disjunctive or conjunctive operations on the result of the testing algorithm. This technique is impractical when it comes to a large number of keywords in one conjunctive search request, because every email is searched for every single keyword. In [7] a new scheme was suggested for conjunctive search called PECK. The scheme substitutes the PEKS algorithm with a PECK algorithm that encrypts a query of keywords. The testing is done with a trapdoor for each query instead of each word. So Bob sends Alice the following:

$$[E(A_{pub}, M), PECK(A_{pub}, (W_1, W_2, ..., W_m))]$$

We say that the scheme is secure against a chosen keyword attack (CKA) if an adversary has a low advantage in guessing the right query of keywords being encrypted.

3.3 The Strong Relation between IBE and PEKS

In [3] the authors showed how the algorithms used in IBE can be used for constructing a PEKS. They showed how with the four algorithms in an IBE Setup, Extract, Encrypt, and Decrypt [11] [4] could be used for to achieve the purpose of a PEKS scheme. So if keyword was used in place of an ID in the IBE scheme. The Setup algorithm will be equivalent to the KeyGen algorithm in a PEKS. Extracting the private key in the IBE will be in replace of generating trapdoors for keywords in PEKS scheme. Now if the Encryption algorithm in the IBE was used to encrypt some zero string of a certain length, the result will be a PEKS ciphertext that could later be tested by using the Decryption algorithm in an IBE and checking whether the result is the same string of zeros. The problem is that the ciphertext could expose the public key (W) used to create it. So we need to derive a notion of key privacy [2] for IBE to ensure that the PEKS is secure under a chosen keyword attack. Key privacy is a security notion first introduced in [2]. If an adversary can not guess which ID of a set of ID's in an IBE scheme was used in encrypting a particular ciphertext then that IBE scheme does not lead to breach the privacy of its keys. This could be under chosen plaintext attack(IK-CPA) or chosen ciphertext attack(IK-CCA).

3.4 Construction of a PEKS From the K-Resilient IBE (KRPEKS)

Since the K-resilient IBE scheme suggested in [9] is said to be IK-CCA secure(Appendix A) it was tempting to construct a PEKS using that scheme. As any other PEKS scheme there are four algorithms, summarized in the following.

- `KRPEKS-KeyGen`
  Step 1: Choose a group G of order q and two generators g_1, g_2
  Step 2: Choose 6 random k degree polynomials where the polynomials are
  chosen over Z_q
  $P_1(x) = d_0 + d_1x + d_2x^2 + ... + d_kx^k$; $P_2(x) = d'_0 + d'_1x + d'_2x^2 + ... + d'_kx^k$
  $F_1(x) = a_0 + a_1x + a_2x^2 + ... + a_kx^k$; $F_2(x) = a'_0 + a'_1x + a'_2x^2 + ... + a'_kx^k$
  $h_1(x) = b_0 + b_1x + b_2x^2 + ... + b_kx^k$; $h_2(x) = b'_0 + b'_1x + b'_2x^2 + ... + b'_kx^k$
  Step 3: For $0 \leq t \leq k$; Compute $A_t = g_1^{a_t} g_2^{a'_t}, B_t = g_1^{b_t} g_2^{b'_t}, D_t = g_1^{d_t} g_2^{d'_t}$

```
    Step 4: Choose a random collision resistant hash function $H$ (Section 2.2)
    Step 5: Choose a random collision resistant hash function $H'$ (Section 2.2)
    Step 6: Assign $A_{priv} = \langle F_1, F_2, h_1, h_2, P_1, P_2 \rangle$
            $A_{pub} = \langle g_1, g_2, A_0, ..., A_k, B_0, ..., B_k, D_0, ..., D_k, H, H' \rangle$
– KRPEKS
    Step 1: Choose a random $r_1 \in Z_q$
    Step 2: Compute $u_1 = g_1^{r_1}$ ; $u_2 = g_2^{r_1}$
    Step 3: Calculate for each keyword $w$
            $A_w \leftarrow \Pi_{t=0}^{k} A_t^{w^t}$ ; $B_w \leftarrow \Pi_{t=0}^{k} B_t^{w^t}$ ; $D_w \leftarrow \Pi_{t=0}^{k} D_t^{w^t}$
    Step 4: $s \leftarrow D_w^{r_1}$
    Step 5: Using the -exclusive or- operation calculate $e \leftarrow (0^k) \otimes H'(s)$
    Step 6: $\alpha \leftarrow H(u_1, u_2, e)$
    Step 7: $v_w \leftarrow (A_w)^{r_1}.(B_w)^{r_1 \alpha}$
    Step 8: $C \leftarrow \langle u_1, u_2, e, v_w \rangle$
– KRPEKS-Trapdoor
    Run Extract of the IBE and the output is the trapdoor
    $T_w = \langle F_1(w), F_2(w), h_1(w), h_2(w), P_1(w), P_2(w) \rangle$.
– KRPEKS-Test
    Step 1: $\alpha \leftarrow H(u_1, u_2, e)$
    Step 2: Test if $v_w \neq (u_1)^{F_1(w)+h_1(w)\alpha}.(u_2)^{F_2(w)+h_2(w)\alpha}$
            then Halt else go to Step 3.
    Step 3: $s \leftarrow (u_1)^{P_1(w)}.(u_2)^{P_2(w)}$
    Step 4: $m \leftarrow e \otimes H'(s)$
    Step 5: If the resulting plaintext is a $0^k$ conclude
            that $C$ is an encryption of $w$.
```

The security of this scheme relies on DDH and the collision resistant hash function as shown in the next Section 3.5.

3.5 Security of KRPEKS against CKA

The K-Resilient IBE scheme [9] is an identity based encryption that is based on the Decisional Diffie Hellman problem (DDH). The security of such scheme is based on the difficulty of solving DDH and whether the hash functions used are collision resistant or not. In (Appendix A), the following theorem was proved.

Theorem Let G be a group of prime order q. If DDH is hard in G then KRIBE is said to be IK-CCA secure. So for any adversary A attacking the anonymity of KRIBE under a chosen ciphertext attack and making in total a $q_d(.)$ decryption oracle queries, there exist a distinguisher D_A for DDH and an adversary C attacking the collision resistance of H such that

$$Adv_{KRIBE,A}^{IK-CCA}(K) \leq 2Adv_{G,D_A}^{DDH}(K) + 2Adv_{H,C}^{CR}(K) + (q_d(K) + 2)/(2^{k-3})$$

Since KRIBE is IK-CCA secure(Appendix A). Therefore if an adversary knows two IDs ID_0, ID_1 and is given a ciphertext encrypted using one of the

IDs. The adversary would not to be able to guess which one was used unless the DDH is not hard or the hash function is not collision resistant. In this Section we show that since the PEKS was built from the KRIBE and KRIBE has key privacy notions then PEKS should logically be proved to be secure under a CKA. Now if we compare both schemes the KRIBE in [9] and the KRPEKS we would notice that the key generation algorithm in the latter is the same as the setup in the former. The trapdoor is created the same way a secret key is created for an ID in the IBE scheme but instead of the IDs we have words. The encryption of IBE is equivalent to it for the PEKS but instead of a message, a k length zero string 0^k is encrypted. The testing of the existence of a keyword is done by using the same decryption algorithm of the IBE and checking whether the result is equal to 0^k. In other words if an adversary can not tell the difference between which ID was used to encrypt a given ciphertext. Then the same adversary would not know which word was used in creating ciphertext $C = PEKS(A_{pub}, W_b)$. The only difference between the two security notions is that instead of having a decryption oracle in proving IK-CCA in KRIBE we have a trapdoor oracle in the new PEKS. So someone can conclude that the advantage of guessing the right word depends on the DDH problem, the collision resistance of the hash function and Q_t where Q_t is the maximum number of trapdoor queries issued by the adversary.

3.6 Constructing K-Resilient SCF-PEKS scheme

In [1] the authors constructed a SCF-PEKS using the same methodology used in the PEKS in [3]. In this Section we will try to build a SCF-PEKS using the KR-PEKS described in 3.4.

- $SCF - KRPEKS - CPG$ (Common Parameter Generator) :
  Step 1: Choose a group G and two generators g_1, g_2
  Step 2: Choose random k
  Step 3: Choose a random collision resistant hash function H (Section 2.2).
  Step 4: Calculate the common parameter $cp = \langle G, g_1, g_2, H, k \rangle$
- $SCF - KRPEKS - SKG(cp)$ (Server Key Generator) :
  Step 1: Choose 6 random k degree polynomials, chosen over Z_q
  $P_1(x) = d_0 + d_1x + d_2x^2 + ... + d_kx^k \ ; P_2(x) = d'_0 + d'_1x + d'_2x^2 + ... + d'_kx^k$
  $F_1(x) = a_0 + a_1x + a_2x^2 + ... + a_kx^k \ ; F_2(x) = a'_0 + a'_1x + a'_2x^2 + ... + a'_kx^k$
  $h_1(x) = b_0 + b_1x + b_2x^2 + ... + b_kx^k \ ; h_2(x) = b'_0 + b'_1x + b'_2x^2 + ... + b'_kx^k$
  Step 2: For $0 \leq t \leq K$; Compute $A_t = g_1^{a_t} g_2^{a'_t}, B_t = g_1^{b_t} g_2^{b'_t}, D_t = g_1^{d_t} g_2^{d'_t}$
  Step 3: Assign $A_{priv_s} = \langle F_1, F_2, h_1, h_2, P_1, P_2 \rangle$
  $A_{pub_s} = \langle g_1, g_2, A_0, ..., A_k, B_0, ..., B_k, D_0, ..., D_k, H \rangle$
- $SCF - KRPEKS - RKG(cp)$ (Reciever Key Generator) :
  Step 1: Choose 6 random k degree polynomials, chosen over Z_q
  $\hat{P}_1(x) = \hat{d_0} + \hat{d_1}x + \hat{d_2}x^2 + ... + \hat{d_k}x^k \ ; \hat{P}_2(x) = \hat{d'_0} + \hat{d'_1}x + \hat{d'_2}x^2 + ... + \hat{d'_k}x^k$
  $\hat{F}_1(x) = \hat{a_0} + \hat{a_1}x + \hat{a_2}x^2 + ... + \hat{a_k}x^k \ ; \hat{F}_2(x) = \hat{a'_0} + \hat{a'_1}x + \hat{a'_2}x^2 + ... + \hat{a'_k}x^k$
  $\hat{h_1}(x) = \hat{b_0} + \hat{b_1}x + \hat{b_2}x^2 + ... + \hat{b_k}x^k \ ; \hat{h_2}(x) = \hat{b'_0} + \hat{b'_1}x + \hat{b'_2}x^2 + ... + \hat{b'_k}x^k$

  Step 2: For $0 \leq t \leq K$; Compute $\hat{A}_t = g_1^{\hat{a}_t} g_2^{\hat{a}'_t}, \hat{B}_t = g_1^{\hat{b}_t} g_2^{\hat{b}'_t}, \hat{D}_t = g_1^{\hat{d}_t} g_2^{\hat{d}'_t}$
  Step 3: Choose a random collision resistant hash function H' (Section 2.2).
  Step 4: Assign $A_{priv_r} = \langle \hat{F_1}, \hat{F_2}, \hat{h_1}, \hat{h_2}, \hat{P_1}, \hat{P_2} \rangle$
  $A_{pub_r} = \langle g_1, g_2, \hat{A}_0, ..., \hat{A}_k, \hat{B}_0, ..., \hat{B}_k, \hat{D}_0, ..., \hat{D}_k, H, H' \rangle$

- $SCF - KRPEKS$
  Step 1: Choose a random $r_1 \in Z_q$
  Step 2: Compute $u_1 = g_1^{r_1}$; $u_2 = g_2^{r_1}$
  Step 3: Calculate $A_w \leftarrow \Pi_{t=0}^{k} A_t^{w^t}$; $B_w \leftarrow \Pi_{t=0}^{k} B_t^{w^t}$; $D_w \leftarrow \Pi_{t=0}^{k} D_t^{w^t}$
  $\hat{A_w} \leftarrow \Pi_{t=0}^{k} \hat{A}_t{}^{w^t}$; $\hat{B_w} \leftarrow \Pi_{t=0}^{k} \hat{B}_t{}^{w^t}$; $\hat{D_w} \leftarrow \Pi_{t=0}^{k} \hat{D}_t{}^{w^t}$
  Step 4: $s \leftarrow D_w^{r_1} \hat{D_w^{r_1}}$
  Step 5: $e \leftarrow (0^k) \otimes H'(s)$
  Step 6: $\alpha \leftarrow H(u_1, u_2, e)$
  Step 7: $v_w \leftarrow ((A_w)(\hat{A_w}))^{r_1}.((B_w)(\hat{B_w}))^{r_1\alpha}$
  Step 8: $C \leftarrow \langle u_1, u_2, e, v_w \rangle$
- $SCF - KRPEKS - TG$ (Trapdoor Generator) :
  Calculate: $T_w = \langle \hat{F_1}(W), \hat{F_2}(W), \hat{h_1}(W), \hat{h_2}(W), \hat{P_1}(W), \hat{P_2}(W) \rangle$
- $SCF - KRPEKS - T$ (Testing Algorithm) :
  Step 1: $\alpha \leftarrow H(u_1, u_2, e)$
  Step 2: Test if $v_w \neq (u_1)^{F_1(w)+h_1(w)\alpha+\hat{F_1}(w)+\hat{h_1}(w)\alpha}.(u_2)^{F_2(w)+h_2(w)\alpha+\hat{F_2}(w)+\hat{h_2}(w)\alpha}$
  then halt else go to next step
  Step 3: $s \leftarrow (u_1)^{P_1(w)+P_1\hat{(w)}}.(u_2)^{P_2(w)+\hat{P_2}(w)}$
  Step 4: $m \leftarrow e \otimes H'(s)$
  Step 5: If the resulting plaintext is a 0^k conclude that C is the encryption of w.

Notice that the testing part can not be done except by the server. Therefore, the trapdoor could be sent via public channels.

3.7 Constructing a K-resilient PECK

In [7] [10] the authors constructed a PECK by adopting ideas from the PEKS in [3]. In this paper we will try to build a PECK scheme by adopting ideas from the KR-PEKS. The four algorithms that form this scheme are described as follows.

- `KRPECK-KeyGen:`
  Step 1: Choose a group G of order q and two generators g_1, g_2
  Step 2: Choose 6 random k degree polynomials, chosen over Z_q
  $P_1(x) = d_0 + d_1x + d_2x^2 + ... + d_kx^k$; $P_2(x) = d'_0 + d'_1x + d'_2x^2 + ... + d'_kx^k$
  $F_1(x) = a_0 + a_1x + a_2x^2 + ... + a_kx^k$; $F_2(x) = a'_0 + a'_1x + a'_2x^2 + ... + a'_kx^k$
  $h_1(x) = b_0 + b_1x + b_2x^2 + ... + b_kx^k$; $h_2(x) = b'_0 + b'_1x + b'_2x^2 + ... + b'_kx^k$
  Step 2: For $0 \leq t \leq k$; Compute $A_t = g_1^{a_t} g_2^{a'_t}, B_t = g_1^{b_t} g_2^{b'_t}, D_t = g_1^{d_t} g_2^{d'_t}$
  Step 3: Choose two random numbers $s_0, s_1 \in Z_q$
  Step 4: Calculate $S = g_1^{s_0}.g_2^{s_1}$
  Step 5: Choose a random collision resistant hash function H (Section 2.2).

```
  Step 6: Calculate: $A_{priv} = \langle F_1, F_2, h_1, h_2, P_1, P_2, s_0, s_1 \rangle$
          $A_{pub} = \langle g_1, g_2, A_0, ..., A_k, B_0, ..., B_k, D_0, ..., D_k, S, H \rangle$
– KRPECK:
  Step 1: Choose a random $r_1 \in Z_q$
  Step 2: Compute $u_1 = g_1^{r_1}$ ; $u_2 = g_2^{r_1}$
  Step 3: Calculate for every $W_i$ where $1 \leq i \leq m$
          $A_{w_i} \leftarrow \Pi^k_{t=0} A_t^{w_i^t}$ ; $B_{w_i} \leftarrow \Pi^k_{t=0} B_t^{w_i^t}$ ; $D_{w_i} \leftarrow \Pi^k_{t=0} D_t^{w_i^t}$
  Step 4: Calculate $e_i$ where $1 \leq i \leq m$ and $e_i \leftarrow D_{w_i}^{r_1}$
  Step 5: Calculate $\alpha_i$ where $1 \leq i \leq m$ and $\alpha_i \leftarrow H(u_1, u_2, e_i)$
  Step 6: $v_{w_i} \leftarrow (A_{w_i})^{r_1}.(B_{w_i})^{r_1 \alpha_i}$
  Step 7: $C \leftarrow \langle u_1, u_2, e_1, ..., e_m, v_{w_1}, ..., v_{w_m}, S^{r_1} \rangle$
– KRPECK-Trapdoors:
  Step 1: Choose $\Omega_1, ..., \Omega_t$ where $t$ is the number of keywords you want to
  search for and $\Omega_i$ is the keyword in position $I_i$
  Step 2: $T_1 = P_1(\Omega_1) + P_1(\Omega_2) + ... + P_1(\Omega_t) + s_0$
  Step 3: $T_2 = P_2(\Omega_1) + P_2(\Omega_2) + ... + P_2(\Omega_t) + s_1$
  Step 4: For $1 \leq j \leq t$ Compute $\alpha_j \leftarrow H(u_1, u_2, e_{I_j})$
  Step 5: $T_3 = F_1(\Omega_1) + ... + F_1(\Omega_t) + h_1(\Omega_1)\alpha_1 + ... + h_1(\Omega_t)\alpha_t$
  Step 6: $T_4 = F_2(\Omega_1) + ... + F_2(\Omega_t) + h_2(\Omega_1)\alpha_1 + ... + h_2(\Omega_t)\alpha_t$
  Step 7: $T_Q = \langle T_1, T_2, T_3, T_4, I_1, ..., I_t \rangle$
– KRPECK-Test:
  Step 1: Test if $v_{w_{I_1}}.v_{w_{I_2}}...v_{w_{I_t}} \neq u_1^{T_3}.u_2^{T_4}$ then halt else do Step 2
  Step 2: If $S^{r_1}.e_{I_1}.e_{I_2}.e_{I_3}...e_{I_t} = u_1^{T_1}.u_2^{T_2}$
          then output "True" otherwise output false
```

If all the words exist, then definitely the condition of the if-statement in Step 2 in the testing algorithm will be true.

4 Conclusion

The main aim of this research was to have a PEKS that is secure under a standard model rather than the random oracle model only. To do so, the first step was finding an IBE scheme that has key privacy notions. Use of IBE with Weil pairing to build a PEKS was demonstrated in [4] and the scheme was secure under a chosen keyword attack but under the random oracle only. The IBE suggested by Boneh and Boyenin in [5] also was not useful in constructing a PEKS as shown in [1]. It was tempting to try to prove the K-resilient IBE(Appendix A) to have a notion of key privacy because it was shown in [2] that the Cramer-Shoup encryption is secure. The KRIBE adopted a lot of techniques from this encryption scheme and KRIBE was proved to be secure(Appendix A).

The new PEKS scheme was then used to construct a public key encryption with conjunctive keyword search and a public key encryption that does not need a secure channel.

However, the new PEKS scheme still has some drawbacks because of the limitations of the KRIBE scheme itself, where the number of malicious users is

restricted to some value K. That is the number of trapdoors generated in the PEKS is limited to at most K. Nevertheless, that is not a serious problem where we could use a reasonably large K for email searching applications.
An additional concern lies in the basic formulation of the PEKS system. The idea of the sender, Bob having the sole power to decide which words to consider as keywords for the recipient, Alice, may not be as convenient in reality. In fact, Alice should have all influence on the email sorting and one solution would be for her to cache a set of criteria in form of queries. In that way, the emails categorized as 'urgent' would have greater possibility of being what she considers imperative to read.

Acknowledgement I would like to express my sincere gratitude to my supervisor Dr John Malone Lee for the immeasurable amount of support, and constructive advices. His efforts on my behalf have helped me to become a better scholar and writer. I would also like to thank Dr Bradford and Dr O'Neill in the University of Bath for encouraging me to finish this work. I am especially grateful to my parents for their love was the real strength behind all my work.

References

1. J. Baek, R. Naini, and W. Susilo. Public key encryption with keyword search revisited. Cryptology ePrint Archive, Report 2005/191, 2005. http://eprint.iacr.org/.
2. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key cryptography. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer-Verlag, 2001.
3. D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano. Public-key encryption with keyword search. In C. Cachin, editor, Proceedings of Eurocrypt 2004, 2004. citeseer.ist.psu.edu/boneh04public.html.
4. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
5. Dan Boneh and Xavier Boyen. Short signatures without random oracles. Cryptology ePrint Archive, Report 2004/171, 2004. http://eprint.iacr.org/.
6. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology - CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer-Verlag, 1998.
7. J. Cha D. Park and P. Lee. Searchable keyword-based encryption. Cryptology ePrint Archive, Report 2005/367, 2005. http://eprint.iacr.org/.
8. R. Hayashi and K. Tanaka. Elgamal and cramer shoup variants with anonymity using different groups extended abstract. C-200, 2004. http://www.is.titech.ac.jp/research/research-report/C/.
9. S. Heng and K. Kurosawa. K-resilient identity-based encryption in the standard model. In *Topics in Cryptology CT-RSA 2004*, volume 2964, pages 67–80. Springer-Verlag, 2004.
10. D. Park, K. Kim, and P. Lee. Public key encryption with conjunctive field keyword search. *Lecture Notes in Computer Science*, 3325:73–86, 2005.

11. A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology - CRYPTO '84*, volume 0193 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.

A K-Resilient IBE scheme in the Standard Model is IK-CCA Secure

In [2] the authors proved Cramer-Shoup to be IK-CCA secure. In this paper the same idea of the proof is used in showing that the K-Resilient IBE (KRIBE) scheme [9] is also IK-CCA secure. The main goal is to try to relate the probability of being able to distinguish between IDs to the ability to solve the DDH and the collision resistance property of hash functions used in KRIBE scheme.

Theorem Let G be a prime order group generator. If DDH is hard in G then KRIBE is said to be IK-CCA secure. So for any adversary A attacking the anonymity of KRIBE under a chosen ciphertext attack and making in total decryption oracle queries, there exist a distinguisher D_A for DDH and an adversary C attacking the collision resistance of H such that

$$Adv^{IK-CCA}_{KRIBE,A}(K) \leq 2Adv^{DDH}_{G,D_A}(K) + 2Adv^{CR}_{H,C}(K) + (q_d(K)+2)/(2^{k-3})$$

The Theorem aboves follows from the next two claims :

Claim 1 : For any K

$$Pr[Exp^{DDH-Real}_{G,D_A}(K) = 1] = 1/2 + (1/2)(Adv^{IK-CCA}_{KRIBE,A}(K))$$

Claim 2 : For every K

$$Pr[Exp^{DDH-Rand}_{G,D_A}(K) = 1] \leq 1/2 + (q_d(K)+2)/(2^{K-2}) + Adv^{CR}_{H,C}(K)$$

Proving Claim 1 In claim 1 the goal is to show that under some distinguisher D_A the view of the adversary A is exactly like in the real decisional Diffie-Hellman experiment. The first step in the proof is to build that distinguisher as shown bellow.

Adversary $D_A(q, g, X, Y, T)$

$K \longleftarrow^{R} GH(k)$

$g_1 \leftarrow 1$; $g_2 \leftarrow X$; $u_{1,0} \leftarrow Y$; $u_{2,0} \leftarrow T$

$w_0 \longleftarrow^{R} Z_q$; $w_1 \longleftarrow^{R} Z_q$

$u_{1,1} \leftarrow Y^{W_0}.g_1^{W_1}$; $u_{2,1} \leftarrow T^{W_0}.g_2^{W_1}$

Choose Six Random Polynomials as the following; Where the polynomials are chosen over Z_q

$P_1(x) = d_0 + d_1x + d_2x^2 + ... + d_jx^j$

$P_2(x) = d'_0 + d'_1x + d'_2x^2 + ... + d'_jx^j$

$F_1(x) = a_0 + a_1x + a_2x^2 + ... + a_jx^j$

$F_2(x) = a'_0 + a'_1x + a'_2x^2 + ... + a'_jx^j$

$h_1(x) = b_0 + b_1x + b_2x^2 + ... + b_jx^j$

$h_2(x) = b'_0 + b'_1x + b'_2x^2 + ... + b'_jx^j$

After Choosing the random polynomials calculate

$A_t = g_1^{a_t}.g_2^{a'_t}$; $B_t = g_1^{b_t}.g_2^{b'_t}$; $D_t = g_1^{d_t}.g_2^{d'_t}$

Choose two random IDs from a set of IDs

ID_0 ; $ID_1 \in \{0,1\}^*$

$sk_0 \leftarrow (P_1(ID_0); P_2(ID_0); h_1(ID_0); h_2(ID_0); F_1(ID_0); F_2(ID_0))$

$sk_1 \leftarrow (P_1(ID_1); P_2(ID_1); h_1(ID_1); h_2(ID_1); F_1(ID_1); F_2(ID_1))$

$Params \leftarrow (g_1; g_2; A_1; ...; A_j; B_1; ...; B_j; D_1; ...; D_j; K)$

$PK_0 \leftarrow ID_0$

$PK_1 \leftarrow ID_1$

$b \leftarrow^R \{0,1\}$

Run Adversary A

$(M, s) \leftarrow A(find, PK_0, PK_1, Params)$

$e \leftarrow (u_{1,b})^{P_1(ID_b)}.(u_{2,b})^{P_2(ID_b)}.M$

$\alpha \leftarrow \epsilon H_k(u_{1,b}, u_{2,b}, e)$

$v \leftarrow (u_{1,b})^{F_1(ID_b)+h_1(ID_b)\alpha}(u_{2,b})^{F_2(ID_b)+h_2(ID_b)\alpha}$

$d \leftarrow A(guess, u_{1,b}, u_{2,b}, e, v, s)$

If b=d then return 1 else return 0

The replying to A's decryption oracle is as follows

A makes a query C'to one of the oracles D_{sk_0} , D_{sk_1} where C'is (u'_1, u'_2, e', v')

$\alpha' \leftarrow \epsilon H(u'_1, u'_2, e')$

If $u_1'^{F_1(ID_i)+h_1(ID_i)\alpha}.u_2'^{F_2(ID_i)+h_2(ID_i)\alpha} = v'$

$m \leftarrow e'/(u'_1)^{P_1(ID_i)}.(u'_2)^{P_2(ID_i)}$

else m is invalid

Now after constructing an adversary we need to show the similarity between it and the KRIBE scheme and that will lead us to claim 1. The input of D_A is (q, g, g^{r_1}, g^{r_2}, $g^{r_1r_2}$) which is equivalent to (q, $g_1, g_2, g_1^{r_2}, g_2^{r_2}$)
The challenge triple and the computed triple are valid Diffie-Hellman.

- Challenge Triple $(g_2, u_{1,0}, u_{2,0})$
  $g_2 = g^{r_1}, u_{1,0} = g^{r_2}, u_{2,0} = g^{r_1r_2}$

- Computed Triple $(g_2, u_{1,1}, u_{2,1})$
  $g_2 = g^{r_1}, u_{1,1} = g^{w_1 + r_2 w_0}, u_{2,0} = g^{(w_1 + r_2 w_0) r_1}$

The challenge triple are independently distributed because they are exactly like the actual experiment and the computed triple are independently distributed because w_0, w_1 are random from Z_q. Now show the similarity between the D_A and the KRIBE scheme above.

- Both the D_A and the KRIBE scheme create six random polynomials of degree j
- A_t, B_t and D_t are calculated the same way
- The D_A chooses two IDs for challenging and two secret keys are extracted the same way it's done in the KRIBE by the PKG.
- Params are calculated in the same matter in both schemes
- Calculating e in the D_A and KRIBE gives same result as shown :
  Let pos be a variable that stands for power of something so

  if $u_{1,b} = g_1^{pos}$ then $u_{2,b} = g_2^{pos}$ hence

  $e \leftarrow u_{1,b}^{P_1(ID_b)}.u_{2,b}^{P_2(ID_b)}.M$

  $e \leftarrow (g_1^{pos})^{P_1(ID_b)}.(g_2^{pos})^{P_2(ID_b)}.M$

  $e \leftarrow (g_1^{pos})^{d_0 + d_1 ID_b + d_2 ID_b^2 + ... + d_j ID_b^j}.(g_2^{pos})^{d'_0 + d'_1 ID_b + d'_2 ID_b^2 + ... + d'_j ID_b^j}.M$

  $e \leftarrow ((g_1^{d_0} g_2^{d'_0}).(g_1^{d_1 ID_b} g_2^{d_1 ID_b}).(g_1^{d_2 ID_b^2} g_2^{d'_2 ID_b^2})...(g_1^{d_j ID_b^j} g_2^{d'_j ID_b^j}))^{pos}.M$

  $e \leftarrow (D_0.D_1^{ID_b}.D_2^{ID_b^2}...D_j^{ID_b^j})^{pos}.M$

  $e \leftarrow (\Pi_{t=0}^{k} D_t^{w^t})^{pos}.M$

- Calculating α in both schemes are in the same matter
- Calculating v gives same result in both schemes as shown in the following
  $v \leftarrow (u_{1,b})^{F_1(ID_b) + h_1(ID_b)\alpha}.(u_{2,b})^{F_2(ID_b) + h_2(ID_b)\alpha}$

  $v \leftarrow (g_1^{pos})^{F_1(ID_b) + h_1(ID_b)\alpha}.(g_2^{pos})^{F_2(ID_b) + h_2(ID_b)\alpha}$

  $v \leftarrow (g_1^{pos})^{(a_0 + ... + a_j ID_b^j) + (b_0 + ... + b_j ID_b^j)\alpha}.(g_2^{pos})^{(a'_0 + ... + a'_j ID_b^j) + (b'_0 + ... + b'_j ID_b^j)\alpha}$

  $v \leftarrow (((g_1^{a_0}).(g_2^{a'_0})...(g_1^{a_j ID_b^j}).(g_2^{a'_j ID_b^j}))((g_1^{b_0}).(g_2^{b'_0})...(g_1^{b_j ID_b^j}).(g_2^{b'_j ID_b^j}))^{\alpha})^{pos}$

  $v_{ID} \leftarrow (A_{ID})^{pos}.(B_{ID})^{pos\alpha}$
- Responding to the queries in the decryption oracle is done exactly like the decryption in the KRIBE

Now that it is obvious that the view of the adversary A is exactly like in the real experiment some one could conclude that the advantage of the adversary guessing the right bit whether it is equal to zero or one is equal to the probability of DDH-Real as shown bellow.

$Pr[Exp_{G,D_A}^{ddh-real}(k) = 1] = (1/2)(Pr[Exp_{KRIBE,A}^{IK-CCA-1}]) + (1/2)(1 - Pr[Exp_{KRIBE,A}^{IK-CCA-0}])$

$Pr[Exp_{G,D_A}^{ddh-real}(k) = 1] = (1/2) + (1/2)Adv_{KRIBE,A}^{IK-CCA}(k)$

Proving Claim 2 When the adversary A makes a query c=$\langle u_1', u_2', e', v' \rangle$ to the decryption oracle . The ciphertext is invalid if $log_{g_1} u_1' = log_{g_2} u_2'$. The reason why the ciphertext is invalid is it will be as if g_1 is raised to a random number to create u_1' and g_2 is raised to a different random number to create u_2'.

Events associated with D_A

- NR is true if $r_2 = r_1$ or $g_2 = 1$
- INV is true when the adversary A submits an invalid ciphertext to the decryption oracle and the respond is not "invalid"

To proof Claim 2 there are three other claims that will be introduced.

Claim 3 :

$$Pr[NR] \leq 1/2^{k-2}$$

Claim 4 :

$$Pr[Exp_{G,D_A}^{ddh-real}(K) = 1|b = 0 \wedge \neg NR \wedge \neg INV] = 1/2$$
$$Pr[Exp_{G,D_A}^{ddh-real}(K) = 1|b = 1 \wedge \neg NR \wedge \neg INV] = 1/2$$

Claim 5 : There exist a polynomial-time adversary C such that for any k

$$Pr[INV|\neg NR] \leq (q_d(k)/2_{k-2}) + Adv_{H,C}^{cr}(k)$$

If the three above claims are true then using the mathmatical rule $Pr[x] \leq Pr[x|\neg Y] + Pr[Y]$ and by conditioning we get

$Pr[Exp_{G,D_A}^{DDH-Real} = 1]$

$= (1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 0] + (1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 1]$

$\leq (1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 0 \wedge \neg NR \wedge \neg INV]$

$+(1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 1 \wedge \neg NR \wedge \neg INV] + Pr[NR] + Pr[INV]$

$\leq (1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 0 \wedge \neg NR \wedge \neg INV]$

$+(1/2)Pr[Exp_{G,D_A}^{DDH-Real} = 1|b = 1 \wedge \neg NR \wedge \neg INV] + 2Pr[NR] + Pr[INV|\neg NR]$

$\leq (1/2) + (q_d(k))/(2^{k-2}) + Adv_{H,C}^{CR}(K)$

Proof Claim 3 Claim 3 is easy to proof since r_1 and r_2 are two random elements that belong to Z_q and $2^{k-1} \leq q \leq 2^k$. Therefore the claim is true.

Proof Claim 4 To proof claim 4 we define a sample S which has the random values in $Exp_{G,D_A}^{ddh-rand}(K)$. So an element in S will be denoted as follows :

$$\begin{aligned}\boldsymbol{s} = (&P_1(ID_0); P_2(ID_0); F_1(ID_0); F_2(ID_0); h_1(ID_0); h_2(ID_0);\\ &P_1(ID_1); P_2(ID_1); F_1(ID_1); F_2(ID_1); h_1(ID_1); h_2(ID_1))\end{aligned}$$

If VIEW is a function with S as a domain and associates to any $\boldsymbol{s} \in S$ the view of an adversary A in the $Exp_{G,D_A}^{ddh-rand}(K)$ when the random choices on that experiment are those given in $\boldsymbol{s}$. In other words assuming the adversary is deterministic the view is all the inputs in both stages and the answers to all its queries to the decryption oracle. The adversary's output is a deterministic function of its view. Now we need another Claim to continue with the proof.

Claim 6 : Fix a specific view $\hat{v}$ of the adversary A simulated by D_A . Assume that the events $\neg NR \wedge \neg INV$ is true for this view. Then $Pr[view = \hat{v}|b = 0] = Pr[view = \hat{v}|b = 1]$ From that Claim we can conclude that if $\neg NR \wedge \neg INV$ is true then A can guess b correctly with a probability 1/2. This results with Claim 4. So the rest of the proof in this Section will be to show that claim 6 holds.

Let the view be equal to the following

$\hat{V} = (\hat{g_1}, \hat{g_2}, \hat{ID_0}, \hat{ID_1}, \hat{A}_1, ..., \hat{A}_j, \hat{B}_1, ..., \hat{B}_j, \hat{D}_1, ..., \hat{D}_j, \hat{u}_1, \hat{u}_2, \hat{e}, \hat{v})$

For $i \in \{0, 1\}$ define an E_i as the set of all $\boldsymbol{s} \in S$ such that $\boldsymbol{s}$ gives rise to b=i and $view(\boldsymbol{s}) = \hat{V}$ and $\neg NR$ is true when the random choices in the experiment are $\boldsymbol{s}$. Then

$Pr[V = \hat{V} \wedge b = 0] = |E_0|/|S| = |E_0|/(2q^{19})$

Now formulas 1-13 are introduced to get the value of $|E_0|$

$$
\begin{align}
& b = 0 \tag{1}\\
& g_1 = \hat{g_1} \tag{2}\\
& g_2 = \hat{g_2} \tag{3}\\
& P_1(ID_0) + \hat{\omega} P_2(ID_0) = \widehat{\beta_{D,0}} + ID_0 \widehat{\beta_{D,1}} + ID_0^2 \widehat{\beta_{D,2}} + ... + ID_0^j \widehat{\beta_{D,j}} \tag{4}\\
& h_1(ID_0) + \hat{\omega} h_2(ID_0) = \widehat{\beta_{B,0}} + ID_0 \widehat{\beta_{B,1}} + ID_0^2 \widehat{\beta_{B,2}} + ... + ID_0^j \widehat{\beta_{B,j}} \tag{5}\\
& F_1(ID_0) + \hat{\omega} F_2(ID_0) = \widehat{\beta_{A,0}} + ID_0 \widehat{\beta_{A,1}} + ID_0^2 \widehat{\beta_{A,2}} + ... + ID_0^j \widehat{\beta_{A,j}} \tag{6}\\
& P_1(ID_1) + \hat{\omega} P_2(ID_1) = \widehat{\beta_{D,0}} + ID_1 \widehat{\beta_{D,1}} + ID_1^2 \widehat{\beta_{D,2}} + ... + ID_1^j \widehat{\beta_{D,j}} \tag{7}\\
& h_1(ID_1) + \hat{\omega} h_2(ID_1) = \widehat{\beta_{B,0}} + ID_1 \widehat{\beta_{B,1}} + ID_1^2 \widehat{\beta_{B,2}} + ... + ID_1^j \widehat{\beta_{B,j}} \tag{8}\\
& F_1(ID_1) + \hat{\omega} F_2(ID_1) = \widehat{\beta_{A,0}} + ID_1 \widehat{\beta_{A,1}} + ID_1^2 \widehat{\beta_{A,2}} + ... + ID_1^j \widehat{\beta_{A,j}} \tag{9}\\
& u_1 = \hat{u_1} \tag{10}\\
& u_2 = \hat{u_2} \tag{11}\\
& P_1(ID_0)\hat{r_1} + P_2(ID_0)\hat{\omega}\hat{r_2} = log_{g_1}(\hat{e}/M) \tag{12}\\
& \hat{r_1} F_1(ID_0) + \hat{r_1} \alpha h_1(ID_0) + \hat{r_2} \hat{\omega} F_2(ID_0) + \alpha \hat{r_2} \hat{\omega} h_2(ID_0) = log_{g_1} \hat{v} \tag{13}
\end{align}
$$

Note that $\hat{\omega} = log_{g_1} \hat{g_2}$, $\hat{r_1} = log_{\hat{g_1}} \widehat{u_{1,0}}$, $\hat{r_2} = log_{\hat{g_2}} \widehat{u_{2,0}}$, $\hat{\alpha} = \epsilon H_k(\widehat{u_{1,0}}, \widehat{u_{2,0}}, \hat{e})$ $\widehat{\beta_{A,j}} = log_{\hat{g_1}} \hat{A_j}$,$\widehat{\beta_{B,j}} = log_{\hat{g_1}} \hat{B_j}$, $\widehat{\beta_{D,j}} = log_{\hat{g_1}} \hat{D_j}$

The following matrices A,B,C, and D are used to construct a main big matrix to resemble the above functions. So A,B,C, and D are described as follows

$$\text{A}=\begin{vmatrix} 1 & \hat{\omega} & 0 & 0 \\ 0 & 0 & 1 & \hat{\omega} \\ r_1' & r_2'\hat{\omega} & \hat{r_1}\hat{\alpha} & \hat{r_2}\hat{\omega}\hat{\alpha} \end{vmatrix} \qquad \text{B}=\begin{vmatrix} 1 & \hat{\omega} \\ \hat{r_1} & \hat{\omega}\hat{r_2} \end{vmatrix}$$

$$\text{C}=\begin{vmatrix} 1 & \hat{\omega} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \hat{\omega} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \hat{\omega} \end{vmatrix} \qquad \text{d}=\begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Now the formulas from 1-13 could be written with matrices such that $F_{13\times 19} \times X_{19} = B_{13}$ as shown bellow.

$$\begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \text{A} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \text{B} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \text{C} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \text{D} & 0 & 0 \end{vmatrix} \times \begin{vmatrix} \text{b} \\ P_1(ID_0) \\ P_2(ID_0) \\ F_1(ID_0) \\ F_2(ID_0) \\ h_1(ID_0) \\ h_2(ID_0) \\ P_1(ID_1) \\ P_2(ID_1) \\ F_1(ID_1) \\ F_2(ID_1) \\ h_1(ID_1) \\ h_2(ID_1) \\ g_1 \\ g_2 \\ u_{1,0} \\ u_{2,0} \\ u_{1,1} \\ u_{2,1} \end{vmatrix} = \begin{vmatrix} 0 \\ \hat{\beta_{D,0}} + ID_0\hat{\beta_{D,1}} + ID_0^2\hat{\beta_{D,2}} + ... + ID_0^j\hat{\beta_{D,j}} \\ \hat{\beta_{B,0}} + ID_0\hat{\beta_{B,1}} + ID_0^2\hat{\beta_{B,2}} + ... + ID_0^j\hat{\beta_{B,j}} \\ log_{g_1}\hat{V} \\ \hat{\beta_{A,0}} + ID_0\hat{\beta_{A,1}} + ID_0^2\hat{\beta_{A,2}} + ... + ID_0^j\hat{\beta_{A,j}} \\ log_{g_1}(e/M) \\ \hat{g_1} \\ \hat{g_2} \\ \hat{u_{1,0}} \\ \hat{u_{2,0}} \end{vmatrix}$$

Now it is shown in [2] that F has a full rank [1]and therefore the number of solutions for this system,“and which is $|E_0|$”is q^{19-13} $=q^6$. $|E_1|$ is calculated the same way $|E_0|$ is calculated with changing formulas 12, 13 to 14, 15 respectively where 14 and 15 are shown bellow.

$$P_1(ID_1)\hat{r_1} + P_2(ID_1)\hat{\omega}\hat{r_2} = log_{g_1}(\hat{e}/M) \tag{14}$$

$$\hat{r_1}F_1(ID_1) + \hat{r_1}\alpha h_1(ID_1) + \hat{r_2}\hat{\omega}F_2(ID_1) + \alpha\hat{r_2}\hat{\omega}h_2(ID_1) = log_{g_1}\hat{v} \tag{15}$$

Now by symmetry of view and the systems equations corresponding to E_0 and E_1 with respect to the random bit b $|E_0| = |E_1|$ and therefore

[1] Full Rank is proved if each row in the matrix has a leading entry after appplying gaussian elimination

$$Pr[view = \hat{v}|b = 0] = Pr[view = \hat{v}|b = 1]$$

Proof Claim 5 Assume adversary A does submit an invalid ciphertext to D_{sk_i}. The submitted ciphertext must not equal the challenged ciphertext $(u'_1, u'_2, e', v') \neq (u_1, u_2, e, v)$ Consider the following events:

$A' : (u'_1, u'_2, e') = (u_{1,b}, u_{2,b}, e)$

$B' : (u'_1, u'_2, e') \neq (u_{1,b}, u_{2,b}, e); \alpha' = \alpha_b$

$C' : (u'_1, u'_2, e') \neq (u_{1,b}, u_{2,b}, e); \alpha' \neq \alpha_b$

Claim: There exist a polynomial time adversary C such that

$$\begin{aligned} Pr[INV|\neg NR] = {} & Pr[INV|A' \wedge \neg NR].Pr[A'] + Pr[INV|B' \wedge \neg NR].Pr[B'] \\ & +Pr[INV|C' \wedge \neg NR].Pr[C'] \\ \leq {} & 0 + Pr[B'] + Pr[INV|C' \wedge \neg NR] \\ \leq {} & 0 + Adv^{CR}_{H,C} + Pr[INV|C' \wedge \neg NR]........(a) \end{aligned}$$

1. What makes (a) true

- Now in case of A',$v' \neq v$ so the oracle will reject it.
- Now in case of B' we can construct the adversary C which attacks the collision resistance H .

2. What is $Pr[INV|C' \wedge \neg NR]$ value? A ciphertext (u'_1, u'_2, e', v') is accepted when submitting it to D_{sk_i} when

- $(u'_1)^{F_1(ID_0)+h_1(ID_0)\alpha'} + (u'_2)^{F_2(ID_0)+h_2(ID_0)\alpha'} = v'; For i = 0......(b)$
- $(u'_1)^{F_1(ID_1)+h_1(ID_1)\alpha'} + (u'_2)^{F_2(ID_1)+h_2(ID_1)\alpha'} = v'; For i = 1......(c)$

Consider two more events

- $Inv_{i,j}$ is the case when adversary A during the ith query submits an invalid ciphertext subject to the conditions of C' to the decryption oracle and does not get an invalid where $i \in \{1, ..., q_d\}$; $j \in 0, 1$
- E^{INV}_i is a set $\{\boldsymbol{s} : \boldsymbol{s} \in S$ and $\boldsymbol{s}$ gives rises to a corresponding equation b or c,$\neg NR\}$ and the conditions from C'.

We first consider a simulation of D_{sk_0}. To submit a ciphertext that would not be rejected the adversary should come up with the coefficients of equation b which is consistent with its view and which can contain a hidden bit b=0 or b=1 with an equal opportunity. So

$Pr[Inv_{1,0}, \neg NR] = (1/2)Pr[E^{Inv}_0|E_0] + (1/2)Pr[E^{Inv}_0|E_1]$

$\leq (Pr[E^{inv}_0 \wedge E_0])/(2Pr[E_0]) + Pr[E^{inv}_0 \wedge E_1])/(2Pr[E_1])$

$= (|E^{inv}_0 \wedge E_0|.|S|)/(2|E_0|.|S|) + (|E^{inv}_0 \wedge E_1|.|S|)/(2|E_1|.|S|)$

$= (|E^{inv}_0 \wedge E_0|)/(2q^6) + (|E^{inv}_0 \wedge E_1|)/(2q^6).....(d)$

So the number of solutions for equations 1-13 plus the equation b is equal to $|E^{Inv}_0 \wedge E_0|$ while $|E^{Inv}_0 \wedge E_1|$is the number of solutions for equations 1-11, 14, 15, and c. Assuming $\neg NR$ in both of them .

To solve the system and get $|E_0^{Inv} \wedge E_0|$ we will have to add equation b to the system. So we let $u_1' = g_1^{r'_1}$, $u_2' = g_2^{r'_2} = g_1^{\omega r'_2}$ And matrix A becomes as follows

$$\text{A}=\begin{vmatrix} 1 & \hat{\omega} & 0 & 0 \\ 0 & 0 & 1 & \hat{\omega} \\ \hat{r_1} & \hat{r_2}\hat{\omega} & \hat{r_1}\hat{\alpha} & \hat{r_2}\hat{\omega}\hat{\alpha} \\ {r_1}' & {r_2}'\hat{\omega} & {r_1}'\alpha' & r_2'\hat{\omega}\alpha' \end{vmatrix}$$

As shown in the paper [2] F is still a full rank that means that the number of solutions now is q^5

To solve the system and get $|E_0^{Inv} \wedge E_1|$ we will have to add equation c to the system. So we let So we let $u_1' = g_1^{r'_1}$, $u_2' = g_2^{r'_2} = g_1^{\omega r'_2}$

Now since the challenge ciphertext corresponds to pk_1

- Matrix A becomes as follows

$$\text{A}=\begin{vmatrix} 1 & \hat{\omega} & 0 & 0 \\ 0 & 0 & 1 & \hat{\omega} \\ {r_1}' & {r_2}'\hat{\omega} & {r_1}'\alpha' & r_2'\hat{\omega}\alpha' \end{vmatrix}$$

- Matrix C becomes as follows

$$\text{C}=\begin{vmatrix} 1 & \hat{\omega} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \hat{\omega} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \hat{\omega} \\ \hat{r_1} & \hat{r_2}\hat{\omega} & \hat{r_1}\hat{\alpha} & \hat{r_2}\hat{\omega}\hat{\alpha} & 0 & 0 \end{vmatrix}$$

As shown in the paper F still has a full rank that means that the number of solutions now is also q^5 Combining the results in equation "d"we get

$Pr[Inv_{1,0}|\neg NR] \leq (1/q)$

Now by symmetry $Pr[Inv_{1,0}|\neg NR] = Pr[Inv_{1,1}|\neg NR]$, each time an invalid ciphertext is rejected set of next possible decryption oracle queries at most by one. Therefore

$Pr[Inv|\neg NR \wedge C'] \leq \Sigma_{i=1}^{q_d(k)} Pr[Inv_{i,0}|\neg NR] \leq \Sigma_{i=1}^{q_d(k)}(1)/(q-i+1) \leq (q_d(k))/(q-q_d(k)+1) \leq (q_d(k))/(2^{k-2})$

So with this claim 5 is proved to be right.