# INFORMATION AND SYSTEM SECURITY

FACULTY:

PROF.JAYASUDHA.M



SUBMITTED BY:

G.PAVAN KUMAR

16MIS1097

# Key-Policy Attribute-Based Encryption With Equality Test in Cloud Computing

**ABSTRACT:**

The privacy of users must be considered as the utmost priority in distributed networks. To protect the identities of users, attribute-based encryption (ABE) was presented by Sahai et al. ABE has been widely used in many scenarios, particularly in cloud computing. Public key encryption with equality test is concatenated with key-policy ABE (KP-ABE) to present KP-ABE with equality test (KP-ABEwET).The proposed scheme not only offers ﬁne-grained authorization of ciphertexts but also protects the identities of users. In contrast to ABE with keyword search, KP-ABEwET can test whether the ciphertexts encrypted by different public keys contain the same information. Moreover, the authorization process of the presented scheme is more ﬂexible than that of Ma et al.’s scheme. Furthermore, the proposed scheme achieves one-way against chosen-ciphertext attack based on the bilinear Difﬁe–Hellman (BDH) assumption. In addition, a new computational problem called the twin-decision BDH problem (tDBDH) is proposed in this paper. tDBDH is proved to be as hard as the decisional BDH problem. Finally, for the ﬁrst time, the security model of authorization is provided, and the security of authorization based on the tDBDH assumption is proven in the random oracle model.

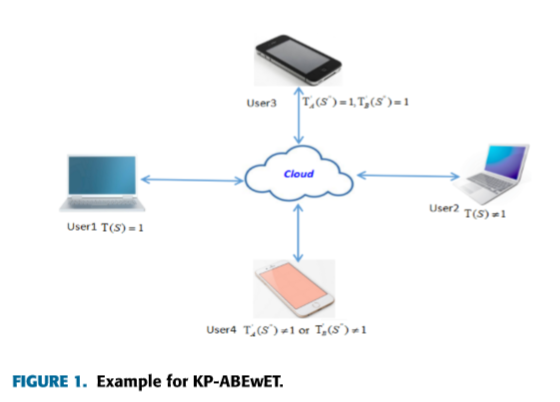
**INTRODUCTION:**

In the current network era, cloud service providers offer inﬁnite storage space and computing power for users to manage their data. To enjoy these services, individuals and organizations store their private data on cloud servers. However, in the case of security breaches, user’s private data stored in the cloud are no longer safe. When users outsource their data to cloud servers, they expect complete privacy of their data stored in the cloud. Protecting the privacy and data of users has remained a very crucial problem for cloud servers. To avoid any inconvenience, users store their private data in encrypted form. For ﬁne-grained sharing of encrypted data, Sahai and Waters presented attribute basedencryption(ABE).

ABE is a public key cryptosystem variant that allows users to access secret data based on their attributes. This cryptosystem enriches the ﬂexibility of the encryption policy and the description of users’ rights, and it changes from a one-one to one-many scenario during the encryption and decryption phases. Moreover, it hides the identities of the users in appropriate terms. In a subsequent work, Goyal et al. proposed key-policy attribute-based encryption (KP-ABE) in 2006.The underlying cryptosystem combines the secret key and the access structure. Bethencourt et al. proposed ciphertext-policy attribute-based encryption (CP-ABE) in 2007 [19], which combines the ciphertext and the access structure. Thereafter, numerous cryptographers presented many research works based on ABE [20]–[24]. Soon after its conceptualization, ABE reached prime importance in our daily.

Moreover,ABE is also being widely incorporated in cloud computing. However, if one wants to compare plaintexts corresponding to two ciphertexts, the secret key must be used to decrypt the two ciphertexts.

To overcome this problem, Yang et al presented a new cryptosystem called public key encryption with equality test (PKEwET) in 2010. His proposed system can test whether two ciphertexts contain the same plaintexts without decryption.



**PRELIMINARIES:**

In this part, we introduce some basic knowledge, including cryptographic assumptions, Shamir’s secret sharing scheme and access tree, that is employed in this paper.

**A. CRYPTOGRAPHIC ASSUMPTIONS:**

The following subsection presents the deﬁnitions of bilinear maps and the problem formulation.

Deﬁnition 1:

BilinearMaps:

LetG1 andG2 bemultiplicativegroupsofprimeorderq,e:G1×G1 →G2 beabilinear map, and g be a generator of G1.

Bilinear maps fulﬁll the following conditions:

(1) Bilinearity: ∀g1,g2 ∈ G1 and ∀a,b ∈ Zq, we havee (ga 1,gb 2)=e(g1,g2)ab.

(2) Non-degeneracy: e(g,g)6=1. (3) Computability: ∀g1,g2 ∈ G1, we can computee (g1,g2).

Deﬁnition 2:

Bilinear Difﬁe-Hellman (BDH) problem:

Let G1 and G2 be multiplicative groups of prime order q, e:G1×G1 →G2 beabilinearmap,andgbeageneratorof G1.

The BDH problem is that given a 4-tuple (g,ga,gb,gc), the aim is to compute e(g,g)abc, where a,b,c∈Zq.

Deﬁnition 3:

Decisional Bilinear Difﬁe-Hellman (DBDH) problem:

Let G1 and G2 be multiplicative groups of prime order q, e :G1 ×G1 →G2 be a bilinear map, and g be a generator ofG1. The DBDH problem is to distinguish betweenthedistributionsof5-tuples(g,ga,gb,gc,e(g,g)abc) and (g,ga,gb,gc,e(g,g)d), where a,b,c,d ∈Zq.

Deﬁnition 4:

Twin-Decision Bilinear Difﬁe-Hellman (tDBDH) problem:

Let G1 and G2 be multiplicative groups of prime order q, e : G1 ×G1 → G2 be a bilinear map, and g be a generator of G1. The tDBDH problem is to distinguish between the following two distributions: D0 = {(g,ga,gb,gc,gu,gv,e(g,g)abc,e(g,g)auv) : a,b,c,u,v $ ←−Z q} and D1 = {(g,ga,gb,gc,gu,gv,e(g,g)d,e(g,g)w) : a,b,c,d,u,v,w $ ←−Zq}.

In general, the tDBDH problem appears to be weaker than the DBDH problem. However, this problem is in fact as hard as the DBDH problem.

**Theorem 1:**

The tDBDH problem is as hard as the DBDH problem. Proof: ItisquiteclearthattDBDHDBDH.Next,we present the proof of DBDHtDBDH. To prove DBDH tDBDH, we suppose that there is an algorithmAthatcansolvethetDBDHprobleminpolynomial time. We construct an algorithm B as follows. B takes a 4-tuple (ga,gb,gc,e(g,g)d) as input, and its objective is to determine whether e(g,g)d =e(g,g)abc holds.

**B. SHAMIR’S SECRET SHARING SCHEME:**

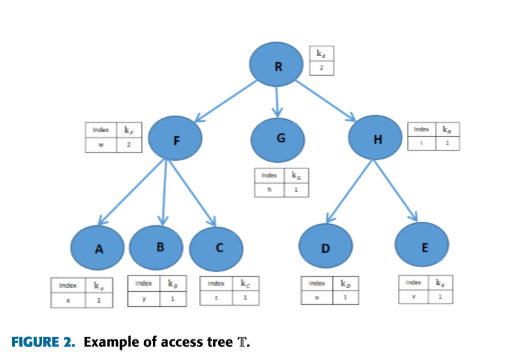
Shamir’s (t,n)-threshold secret sharing scheme is based on the Lagrange interpolation polynomial. A detailed introduction is described as follows: Given t distinct points (xi,f(xi)), where f(x) is a polynomial of degree less than t, f(x) is determined as follows:

f(x)= ∑t{πt{ (x−xj)/(xi−xj) }i=1}j =1,j!=i

Shamir’s scheme is deﬁned for a secret s ∈ Zp by settinga 0 =s and choosing a1,a2,··· ,at−1 ∈Zq. For all 1≤xi ≤q ,1 ≤ i ≤ n, the trusted party computes f(xi), where f(x) = Pt−1 k=0 akxk. The shares (xi,f(xi)) are distributed to n distinct parties. Since the secret is a constant term s = a0 = f(0), the secret can be recovered from any t shares (xi,f(xi)) as follows:

**C. ACCESS TREE :**

We suppose that T is an access tree composed of leaf nodes and non-leaf nodes (e.g., Fig. 2). Each leaf node represents an attribute, and each non-leaf node represents a threshold gate. Each threshold gate is represented by its children and the threshold value. Let numx be the number of children of a node x and kx be the threshold value of the node x;



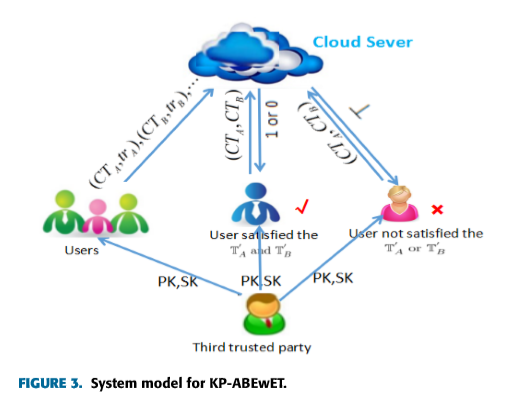
We suppose that the children of every node do have orders from 1 to num. Next, we deﬁne some new functions. The function parent(x) represents the parent of node x. The function att(x) is deﬁned as an attribute associated with the leaf node. The function index(x) returns the number associated with node x. Let r be the root of an access tree T, expressed as Tr. Tx refers to the subtree of T rooted at node x. Tx(S) = 1 means that the set of attributes S satisﬁes the tree Tx. Here, we use a recursive algorithm to compute Tx(S).

• If x is a non-leaf node, we compute Tx0(S) for all children x0 of x. If at least kx children return 1, then Tx(S) returns 1.

• If x is a leaf node, then Tx(S) returns 1 if att(x)∈S.

**PROBLEM FORMULATION:**

In this section,we present the system and the security model.



**SYSTEM MODEL:**

Fig.3:illustrates the system model of KPABEwET. The system has three participating entities: thecloudserver,the users and a trusted third party. The trusted third party generates public key pk and private key sk for users. The users encrypt and send their private data to the cloud server.If a user wants the cloud server to test the ciphertext,then the cloud server is authorized and gains a trapdoor.However,the cloud server can only test whether the two ciphertexts contain the same information and cannot decrypt them using the trapdoor.The legitimate users access data according to their attributes and can decrypt their ciphertexts or test the ciphertexts. If the legitimate users satisfy the access structure for the test, they can get the test results of the ciphertexts from the cloud server. If the legitimate users satisfy the access structure for the decryption, they can decrypt the ciphertexts.

An integrated KP-ABEwET scheme consists of six algorithms:

Setup,Encrypt,KeyGen,Trapdoor,Decrypt and Test. Here, we let M be plaintext space and C be ciphertext space.

(1)Setup(k):

It takes a security parameter k as input, and thenitoutputsthepublicparametersppandpk and the master key mk.

(2) Encrypt(M,pk,S,S0):

It takes a message M ∈ M, public key pk and two sets of attributes S,S0 as inputs, and then it outputs the ciphertext CT ∈C.

(3)KeyGen(T,T0,S,S0,pp,mk):

This algorithm takes as inputsthemasterkeymk,twoaccesstreesT,T0,andtwosets of attributes S,S0 that satisfy T(S) = 1 and T0(S0) = 1, and it subsequently outputs the private key sk.

(4)Trapdoor(S0,T0,mk):

It takes mk,T0 and S0 as inputs, and it outputs the trapdoor td.

(5) Decrypt(CT,sk,S,S0):

It takes as inputs a ciphertext CT ∈ C,S,S0 and the private key sk, and it outputs the message M if T(S) = 1 and T0(S0) = 1. Here, CT is encrypted using the sets S and S0.

(6) Test(CTA,CTB,tdA,tdB,S0):

Suppose that CTA is a ciphertext of the sets of attributes SA and S0 A and that CTB is a ciphertext of the sets of attributes SB and S0 B. This algorithm takes as inputs two ciphertexts CTA,CTB,the trapdoors tdA,tdB and the set S0 of attributes that satisfy T0A(S0)=1 and T0B(S0)=1, and then it outputs 1 if CTA and CTB contain the same message; otherwise, it returns 0.

**SECURITY ANALYSIS:**

1. SECURITY OF SCHEME:

The following subsection provides the security proof of the presented KP-ABEwET scheme.

Theorem 2:

Our proposed scheme is OW-CCA secure against the adversary who is authorized with a trapdoor based on the BDH assumption in the random oracle model.

Proof:

Suppose that A is the adversary that can break the presented KP-ABEwET scheme. Then, there is an algorithm C to solve the BDH problem with a non-negligible advantage.

**Init** Suppose that there is a universeU.Achooses a set of attributesS as his target.

Let Y1 = e(A,B) = e(g,g)ab,Y2 = e(g,g)y( y ∈ Zp). For i ∈ U, C sets Xi as follows: if i ∈ S, it chooses a random αi ∈ Zp and sets Xi=(xi = αi); otherwise, it chooses a random τi ∈ Zp and sets Xi = gbτi = Bτi. Then, C gives the public parameters pp = (X1,X2,··· ,X|U|,Y1,Y2,H1,H2,H3) to A. Here, H1 is a random oracle controlled byC, as described below.

**Phase 1**

A performs the following types of queries polynomially times.

• H1-query: A may issue queries to the random oracle H1. To respond to these queries, C maintains a list of tuples H1. Each element in the list is a tuple of the form (Sλ,δλ,ηλ). The list is initially empty. Responding to query (Sλ,δλ),C runs as follows:

If the query (Sλ,δλ) already appears in the H1 list in the form (Sλ,δλ,ηλ), thenC responds toAwith H1(Sλ,δλ)= ηλ.– Otherwise, C just takes ηλ ∈ G2, and then it responds to A with H1(Sλ,δλ)= . C adds the tuple (Sλ,δλ,ηλ) to the H1 list.

Key retrieve queries:A performs many queries for private keys for many access structures T, where S does not satisfy T.C sends sk toAas follows:

(1)C builds two algorithms: SatT and DNSatT, as follows:

SatT(Tx,S,vx):This algorithm constructs the polynomials for the nodes of an accesssub-tree with a satisﬁed root node when Tx(S) = 1. It takes as inputs a set of attributes S, an access tree Tx and a random number vx ∈ Zp, and it outputs a polynomial qx of degree dx for the root node x as follows: Let qx(0) = vx and randomly choose dx other points of the polynomial qx to construct qx. The algorithm constructs polynomials for each child node x0 of x by executing the algorithm SatT(Tx0,S,qx(index(x0))). DNSatT(Tx,S,gvx): This algorithm constructs the polynomialsforthenodeswhenTx(S)=0.Ittakesaset of attributesS, an access tree Tx and a random element gvx ∈G1,wherevx ∈Zp,and it outputs a polynomialqx of degree dx for the root node x as follows: Because Tx(S) = 0, the root node has less than dx satisﬁed children. Suppose that sx is the number of satisﬁed children of x, which implies that sx < dx. The algorithm chooses a random number vx0 ∈ Zp for each satisﬁed child x0 of x. Let qx(index(x0)) = vx0 and randomly chooseother dx−sx points ofthe polynomial qx to construct qx.

If the node x0 is a satisﬁed node, then it executes the algorithm SatT(Tx0,S,qx(index(x0))). Here, we know qx(index(x0)).

Otherwise, it runs the algorithm DNSatT(Tx0,S, gqx(index(x0))). Here, we know gqx(index(x0)). In the above algorithms, we know qx for each leaf node x clearly satisfyingTx; otherwise, we know gqx(0). Furthermore, qr(0)=a.

1. **SECURITY OF AUTHORIZATION:**

Finally, we provide the security proof of authorization.

Theorem3:

Our proposed scheme is T-CCA secure in terms of authorization against the adversary A2 based on the tDBDH assumption in the random oracle model.

Proof: Suppose thatA2 is the adversary that can break our cryptosystem. Then, there is an algorithmC to solve thetDBDH problem as follows.

The objective of algorithmC is to distinguish between the 7-tuples (A,B,C,E,F,D,G) = (ga,gb,gc,gu,gv,e(g,g)abc,e(g,g)ubv) and (A0,B0,C0,E0, F0,D0,G0) = (ga,gb,gc,gu,gv,e(g,g)d,e(g,g)w), where a,b,c,d,u,v,w∈Zp. Init Suppose that there is a universe U. A2 chooses two sets of attributesS andS0 as his target, where (S0∩S)=∅. Here,S is used for decryption , and S0 is used for the trapdoor. Setup Let Y1 = e(A,B) = e(g,g)ab,Y2 = e(E,B) =e (g,g)ub. For i,j∈U,C sets Xi as follows:

• If i ∈ S, it chooses a random αi ∈ Zp and sets Xi = gαi(xi = αi).

• Otherwise, it sets as follows: – If j ∈ S0, it chooses a random βj ∈ Zp and sets Xj =gβj(xj = βj);– Otherwise, it chooses a random τi ∈ Zp and sets Xi =gbτi =Bτi(xi =bτi). Subsequently, C provides the public parameters pp =( X1,X2,··· ,X|U|,Y1,Y2,H1,H2,H3) to A2. Here, H1 and H2 are random oracles controlled byC, as described below. Phase 1A2 performs the following types of queries polynomially times.

• H1-query:

A may issue queries to the random oracle H1. To respond to these queries, C maintains a list of tuples H1. Each element in the list is a tuple of the form (Sλ,δλ,ηλ). The list is initially empty. Responding to query (Sλ,δλ),C runs as follows: – If the query (Sλ,δλ) already appears in the H1 list in the form (Sλ,δλ,ηλ), thenC responds toAwith H1(Sλ,δλ)= ηλ.– Otherwise, C just takes ηλ ∈ G2, and C responds to A with H1(Sλ,δλ) = ηλ and adds the tuple (Sλ,δλ,ηλ) to the H1 list.

• H2-query: A may issue queries to the random oracle H2. To respond to these queries, C maintains a list of tuples H2. Each element in the list is a tuple of the form (S0 λ,θλ,µλ). The list is initially empty. Responding to query (S0 λ,θλ),C runs as follows: – If the query (S0 λ,θλ) already appears in the H2 list in the form (S0 λ,θλ,µλ), C responds to A2 withH 2(S0 λ,θλ)= µλ.– Otherwise, C just takes µλ ∈ G1, and C responds to A2 with H2(S0 λ,θλ) = µλ and adds the tuple( S0 λ,θλ,µλ) to the H2 list.

• Key retrieve queries: A2 performs many queries for private keys for many access structuresTandT0, where S and S0 do not satisfy T and T0, respectively. C sends sk toA2 as follows: (1) To generate secret key (Dx,Tt), C builds two algorithms: SatT and DNSatT. SatT(Tx,S,vx): This algorithm constructs the polynomials for the nodes when Tx(S) = 1. It takes a set of attributes S, an access tree Tx and a random number vx ∈ Zp, and it outputs a polynomial qx of degree dx for the root node x as follows:

Let qx(0) = vx, and randomly choose dx other points of the polynomial qx to construct qx. It constructs polynomials for each child node x0 of x by running the algorithm SatT(Tx0,S,qx(index(x0))).

DNSatT(Tx,S,gvx): This algorithm constructs the polynomials for the nodes when Tx(S) = 0. It takes a set of attributes W, an access tree Tx and a random element gvx ∈ G1, where vx ∈ Zp, and it outputs a polynomial qx of degree dx for the root node x as follows: Because Tx(S) = 0, the root node has less than dx satisﬁed children. Suppose that sx is the number of satisﬁed children of x, which implies that sx < dx. The algorithm chooses a random number vx0 ∈ Zp for each satisﬁed child x0 of x. Let qx(index(x0)) = vx0 and randomly chooseother dx−sx points ofthe polynomial qx to construct qx. If the node x0 is a satisﬁed node, then it runs the algorithm SatT(Tx0,S,qx(index(x0))). If the node x0 is not a satisﬁed node, then it executes the algorithm DNSatT(Tx0,S,gqx(index(x0))). In the above algorithms, we know qx for each leaf node x clearly satisfying Tx; otherwise, we know gqx(0). Furthermore, qr(0)=a. C constructs a polynomial Qx(·) = bqx(·) and sets y1 = Qr(0)=bqr(0)=ab. For T0, it obtains qt(·) for each node in T0 as follows. If the node t0 is a satisﬁed node, then it runs the algorithm SatT0(T0 t0,S0,qt(index(t0))). Here, we knowq t(index(t0)). Otherwise, it executes the algorithm DNSatT0(T0 x0,S0, gqt(index(t0))). Here, we know gqt(index(t0)). In the above algorithms, we know qt for each leaf node t clearly satisfying T0 t; otherwise, we know gqt(0). Furthermore, qr0(0)=u. C constructs a polynomial Qt(·) = bqt(·) and sets y2 = Qr0(0)=uqr0(0)=ub. Let i=att(x) and j=att(t). – If i=j, then it outputs⊥. – Otherwise, ∗ If i ∈ S, then Dx = gQx(0)/xi = gbqx(0)/αi = Bqx(0)/αi; ∗ Otherwise, Dx = gQx(0)/xi = gbqx(0)/bτi = gqx(0)/τi. – If j=i, then it outputs⊥. – Otherwise, ∗ If j ∈ S0, then Tt = gQt(0)/tj = gbqt(0)/βj = Bqt(0)/βj; ∗ Otherwise, Tt = gQt(0)/tj = gbqt(0)/bτj = gqt(0)/τj.

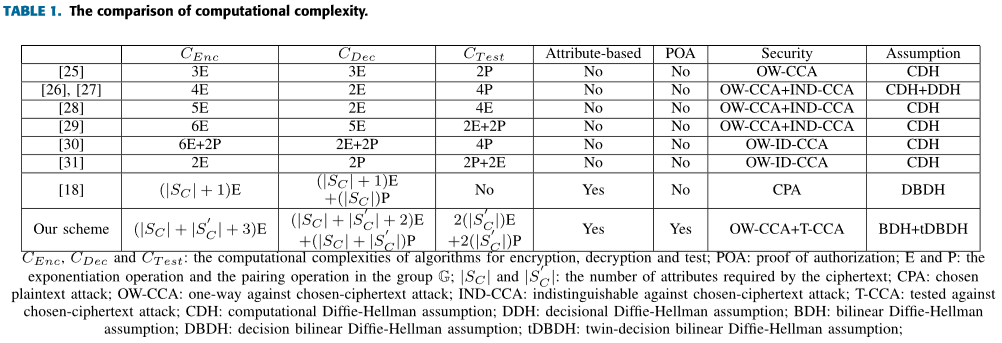
• Decryption queries: Suppose that the ciphertext CTλ = (Sλ,S0 λ,Cλ,1,Cλ,2,Cλ,3,Cλ,4,Cλ,5,Cλ,6), i=att(x).– If i / ∈ S, C generates a private key of Dx as above and calls the Decrypt algorithm with Dx and saves M and r1, then it continues as follows:

∗ If Cλ,1 = gr1 and Cλ,6 = H3(Mr1,Cλ,1,Cλ,2, Cλ,3,Cλ,4,Cλ,5) are established, thenC outputs M toA2.

∗ Otherwise,C outputs⊥toA2.– If i∈S,C proceeds as follows:

∗ If Sλ belongs to the H1 list in the form of (Sλ,δλ,ηλ), thenC executes as follows: a. M||r1 =Cλ,2⊕H1(Sλ,δλ) b. Checks whether Cλ,1 = gr1 and Cλ,6 = H3(Mr1,Cλ,1,Cλ,2,Cλ,3,Cλ,4,Cλ,5) are established. If yes, C outputs M to A2. Otherwise, C outputs⊥toA2.

∗ Otherwise, it outputs ⊥.



**PERFORMANCE EVALUATION:**

We theoretically analyze the asymptotic complexity of the proposed scheme and other PKEwET schemes in Table 1. We describe the computational complexity in terms of the exponentiation operation E and the pairing operation P. We denote the number of attributes required in the ciphertext by |SC| and |S0 C|. In Table 1, CEnc, CDec and CTest represent the encryption algorithms, decryption algorithms and test algorithms, respectively. POA represents the proof of authorization. From the second to the fourth columns, we present the computational complexities of CEnc, Cdec.

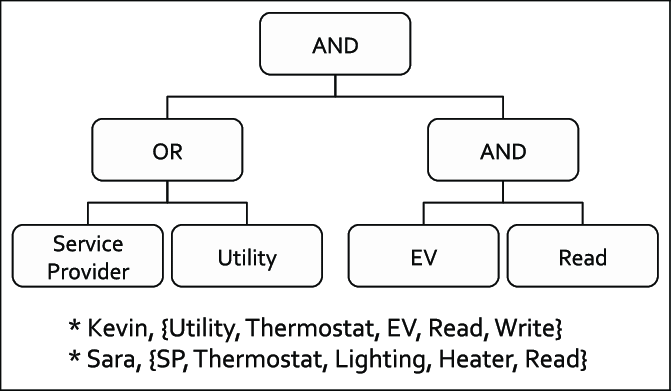
SOLVED EXAMPLES ON KP-ABE:

KEY POLICY ATTRIBUTE BASED ENCRYPTION:

* Cypher texts are labelled with a set of attibutes.
* Private keys are associated with structures that control which cipher text a user is able to decrypt it.

Example:

ACCESS POLICY OF A CLOUD SYSTEM:



**SETUP:**

U={a1= “Service Provider”,a2= “Utility”, a3= “EV”, a4= “READ”}

U: Key policy attributes set.

T:u→Zp

Produces t1,t2,t3,…..tn € Zp.

**MASTER KEY(MK):**

t1,t2,t3….tn€Zp

y€Zp

**PUBLIC KEY(Pk):**

T1=, T2=……,Tn=.

Y=.

**ENCRYPTION:**

E(M,U,Pk)=(,

M:MESSAGE.

U:ATTRIBUTE SET.

S=Y(ALL THE USERS).

**KEY GENERATION:**

Choose a polynomial qn for each Node.

|  |
| --- |
| Degree(qx)=k(x)-1 |

Degree(q1)=0.

Degree(q2)=1

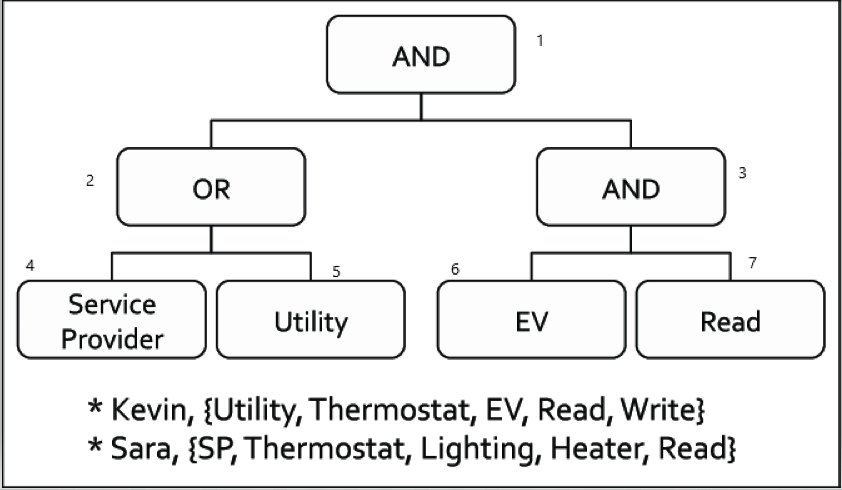
Degree(q3)=1

.

.

.

Degree(q8)=0.



The leaf NODE varies from q4 to q8.

|  |
| --- |
| D4= |
| D5=. |

These are the private keys of the “kevin” user which get checked by the user himself if these satisfy the KEY POLICY ATTRIBUTE SET the user can Decrypt the data.

**DECRYPTION:**

C=E(U,,).

Lets consider D4,D5:

E(D4,T “Utility” S)=e().

E(D5,T “EV” s)= e().

NOW some data structure algorithms are used to identify and compare the Key policy attibute set and the users public attribute set.

Status: “If the user’s private policy satisfies the Key policy attribute set then the User gets to decrypt the data”.

**ENCRYPTION AND DECRYPTION USING BLOWFISH ATTRIBUTE BASED ALGORITHM:**

Our Key-Policy Attribute based encryption is used in this blowfish algorithm as double “ENCRYPTION” feature for further security once the user is able to decrypt the kp-abe stage then the user’s private key is used as a Symmetric key for “ENCRYPTION” and “DECRYPTION”.

**CODE:**

import javax.swing.\*;

import java.security.SecureRandom;

import javax.crypto.Cipher;

import javax.crypto.KeyGenerator;

import javax.crypto.SecretKey;

import javax.crypto.spec.SecretKeySpec;

import java.util.Random ;

public class Blowfish {

byte[] skey = new byte[1000];

String skeyString;

static byte[] raw;

String inputMessage,encryptedData,decryptedMessage;

public Blowfish() {

try {

generateSymmetricKey();

inputMessage=JOptionPane.showInputDialog(null,"Enter message to encrypt");

byte[] ibyte = inputMessage.getBytes();

byte[] ebyte=encrypt(raw, ibyte);

String encryptedData = new String(ebyte);

System.out.println("Encrypted message "+encryptedData);

JOptionPane.showMessageDialog(null,"Encrypted Data "+"\n"+encryptedData);

byte[] dbyte= decrypt(raw,ebyte);

String decryptedMessage = new String(dbyte);

System.out.println("Decrypted message "+decryptedMessage);

JOptionPane.showMessageDialog(null,"Decrypted Data "+"\n"+decryptedMessage);

}

catch(Exception e) {

System.out.println(e);

}

}

void generateSymmetricKey() {

try {

Random r = new Random();

int num = r.nextInt(10000);

String knum = String.valueOf(num);

byte[] knumb = knum.getBytes();

skey=getRawKey(knumb);

skeyString = new String(skey);

System.out.println("Blowfish Symmetric key = "+skeyString);

}

catch(Exception e) {

System.out.println(e);

}

}

private static byte[] getRawKey(byte[] seed) throws Exception {

KeyGenerator kgen = KeyGenerator.getInstance("Blowfish");

SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");

sr.setSeed(seed);

kgen.init(128, sr); // 128, 256 and 448 bits may not be available

SecretKey skey = kgen.generateKey();

raw = skey.getEncoded();

return raw;

}

private static byte[] encrypt(byte[] raw, byte[] clear) throws Exception {

SecretKeySpec skeySpec = new SecretKeySpec(raw, "Blowfish");

Cipher cipher = Cipher.getInstance("Blowfish");

cipher.init(Cipher.ENCRYPT\_MODE, skeySpec);

byte[] encrypted = cipher.doFinal(clear);

return encrypted;

}

private static byte[] decrypt(byte[] raw, byte[] encrypted) throws Exception {

SecretKeySpec skeySpec = new SecretKeySpec(raw, "Blowfish");

Cipher cipher = Cipher.getInstance("Blowfish");

cipher.init(Cipher.DECRYPT\_MODE, skeySpec);

byte[] decrypted = cipher.doFinal(encrypted);

return decrypted;

}

public static void main(String args[]) {

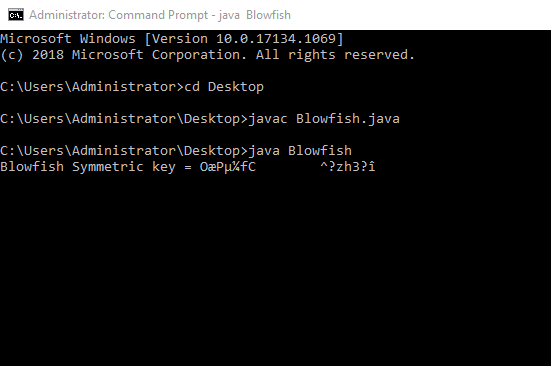
Blowfish bf = new Blowfish();

}

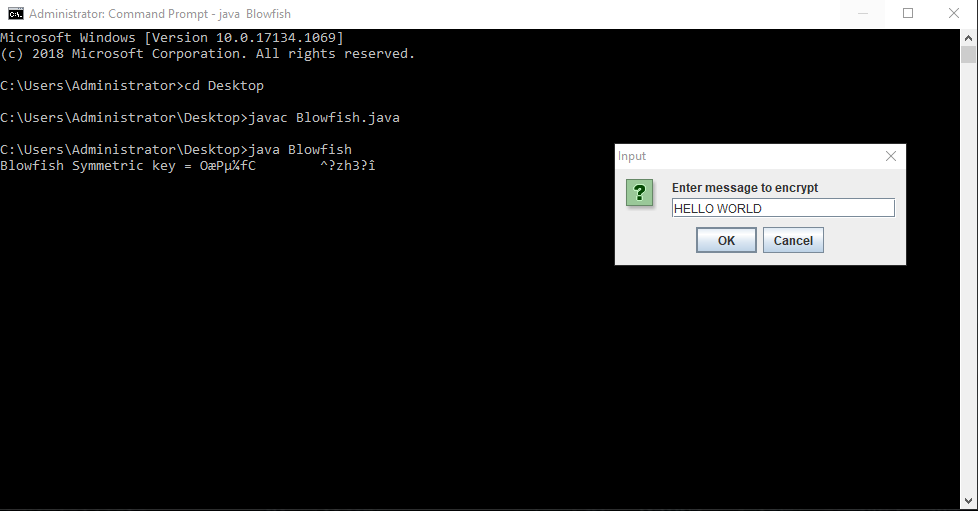
}

OUTPUT:

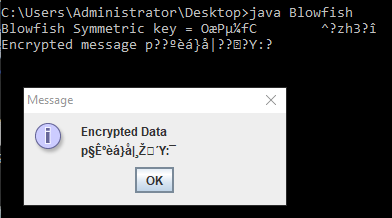
In the below figure once we compile the code it will generate a Symmetric key which is unique to the user and it is used to encrypt and also decrypt the data.



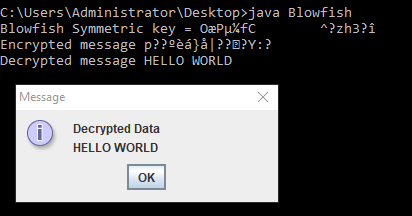
This the UI that takes the plain text as input and converts it to the Cyphertext.



This is the Cipher text after the Encryption is done.



It automatically decrypts the data which is the Cypher text to show the original Plain text



**CONCLUSION**

In this paper,a new crypto system called key-policy attribute based encryption with equality test (KP-ABEwET) is presented. To the best of our knowledge, KP-ABEwET is the ﬁrst attempt to combine the public key encryption supporting equality test with key-policy attribute-based encryption. The proposed scheme can be viewed as an extension of attribute based encryption with keyword search (ABEwKS) with the difference that it can test whether the ciphertexts contain the same information that were encrypted by different public keys. In contrast to previous schemes with equality test, the new scheme supports testing the ciphertexts with ﬁnegrained authorization and also hides the identity of the user. Moreover, the proposed scheme is one-way secure against chosen-ciphertext attack (OW-CCA) based on the bilinear Difﬁe-Hellman(BDH)problem.

Furthermore, a new computational problem called twin-decision bilinear Difﬁe-Hellman problem (tDBDH) is proposed and is proven to be as hard as the DBDH problem.Finally,the security model of authorization is presented, and the security of authorization based on the tDBDH assumption is proven in the random oracle model. To the best of our knowledge, this work is the ﬁrst to prove the security of authorization in such a scenario.

**REFERENCES:**

[1] A. Boldyreva, S. Fehr, and A. O’Neill, ‘‘On notions of security for deterministic encryption, and efﬁcient constructions without random oracles,’’ in Proc. Annu. Int. Cryptol. Conf., 2008, pp. 335–359.

[2] A.SahaiandB.Waters,‘‘Fuzzyidentity-basedencryption,’’inProc.Annu. Int. Conf. Theory Appl. Cryptograph. Techn., 2005, pp. 457–473.

[3] C. Wang, W. Li, Y. Li, and X. L. Xu, ‘‘A ciphertext-policy attribute-based encryption scheme supporting keyword search function,’’ in Proc. CSS, 2013, pp. 377–386.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, ‘‘Public key encryption with keyword search,’’ in Proc. Int. Conf. Theory Appl. Cryptograph. Techn., 2004, pp. 506–522.

[5] F. Han et al., ‘‘A general transformation from KP-ABE to searchable encryption,’’ Future Generat. Comput. Syst., vol. 30, pp. 107–115, Jan. 2014.