# A Performance Analysis of Identity-Based Encryption Schemes

Pengqi Cheng, Yan Gu, Zihong Lv, Jianfei Wang, Wenlei Zhu, Zhen Chen, Jiwei Huang

Tsinghua University, Beijing, 100084, China

**Abstract** We implemented four of the most common IBE schemes: Cocks IBE, Boneh-Franklin IBE, Authenticated IBE, Hierarchical IBE. For each algorithm in an IBE scheme, we recorded the execution time and space cost with different lengths of key. Then, we made a comparison among these IBE schemes and analyzed their characteristics.

**Keywords** Identity-Based Encryption (IBE), performance, time complexity, execution time, space cost

## 1 Introduction and Our Main Work

In 1984, Shamir[8] first gave the concept of Identity-Based Encryption (IBE). In an IBE scheme, when Alice sends an email to Bob, Alice will use Bob's email address, bob@bob.com for example, to encrypt it, without needing to get Bob's public key certificate before sending it. When Bob receives an email, he first authenticates himself to the Private Key Generator (PKG) and gets his private key. Then he uses this private key to decrypt the email. This process is largely different from existing secure email infrastructure in the following aspects:

- Senders can send an email without a public key, which can reduce a lot of process for the certificate management.
- There is no need for an online lookup for a sender to obtain the recipient's certificate.
- Only the PKG holds the private key for recipients, and it is able to refresh the recipients' private keys in every short time period.

After the problem was posed, many implementations are presented by computer scientists to fulfill the idea given by Shamir. In this paper, we chose four of the most wide-used IBE schemes and compared their performance. These four IBE schemes are: Cocks IBE[4], Boneh-Franklin IBE[3], Authenticated IBE[6], and Hierarchical IBE[5]. Hierarchical IBE is a little special, because it is a cascade of IBE schemes to form a tree-style hierarchical PKG. These IBE schemes have different performance on server, clients and the need of bandwidth. So our work is to implement all these algorithms and make a comparison and analysis.

## 2 Definitions

An identity-based encryption scheme is composed by the four following algorithms: Setup, Extract, Encrypt and Decrypt [3]:

1. **Setup**: gets a security parameter $k$, and then returns system parameters params and master-key. The system parameters are public, which include a finite message space $\mathcal{M}$ and a finite ciphertext space $\mathcal{C}$. The master-key is accessible only by the PKG.
2. **Extract**: based on params and master-key given by Setup, and an arbitrary $\mathsf{ID} \in \{0,1\}^*$, returns a private key $d$. Here ID will be used as a public key, and $d$ is the corresponding private key.
3. **Encrypt**: gets params, ID and message $M \in \mathcal{M}$, and then outputs ciphertext $C \in \mathcal{C}$.
4. **Decrypt**: gets params, $d$ and ciphertext $C \in \mathcal{C}$, and then outputs message $M \in \mathcal{M}$.

For correctness, these algorithms above must satisfy the standard consistency constraint, that is to say, if $d$ is the private key generated by Extract corresponding to public key ID, then

$$\forall M \in \mathcal{M} : \mathsf{Decrypt}(\mathsf{params}, \mathsf{Encrypt}(\mathsf{params}, \mathsf{ID}, M), d) = M$$

## 3 Typical IBE Schemes

We will discuss four typical IBE schemes in this paper. First, we will introduce them in this section.

### 3.1 Cocks IBE[4]

- Setup
  The PKG gets following inputs to generate a private key:
  1. an RSA module $n = pq$, where $p, q$ are two private prime numbers which satisfy $p \equiv q \equiv 3 \pmod 4$
  2. a message space $\mathcal{M} = \{-1, 1\}$ and a ciphertext space $\mathcal{C} = Z_n$
  3. a secure common hash function $f : \{0,1\}^* \to Z_n$
- Extract
  - Input: parameters generated by Setup and an arbitrary ID
  - Output: the private key $r$
  1. Generate $a$ which satisfies $(\frac{a}{p}) = 1$ with a deterministic procedure ID.
  2. Let $r = a^{\frac{n+5-p-q}{8}} \bmod n$ which satisfies $r^2 = \pm a \bmod n$.
- Encrypt
  - Input: parameters generated by Setup, ID of the sender and a message $M$
  - Output: corresponding ciphertext $C$

1. Select a random $t$ which satisfies $m = (\frac{t}{n})$, where $m$ is an arbitrary bit of $M$.
2. Let $c_1 = t + at^{-1} \bmod n$ and $c_2 = t - at^{-1} \bmod n$.
3. Send $s = \langle c_1, c_2 \rangle$ to the recipient.

– Decrypt
  - Input: ciphertext $C$ and the private key and parameters generated by the PKG
  - Output: original message $M$

1. Let $\alpha = c_1 + 2r$ if $r^2 = a$, otherwise $\alpha = c_2 + 2r$.
2. Return $m = (\frac{\alpha}{n})$.

### 3.2 Boneh-Franklin IBE[3]

– Setup
1. Get a security parameter $k$ and two groups of order $q$ (a generated prime number): $G_1$ and $G_2$, and an admissible bilinear map $\hat{e} : G_1 \times G_1 \to G_2$, and select a random generator $P \in G_1$.
2. Generate a random number $s \in Z_q^*$. Let $P_{pub} = sP$.
3. Select a hash function $H_1 : \{0,1\}^* \to G_1^*$. For any specified $n$, specify a hash function $H_2 : \{0,1\}^n \to G_2^n$.
4. Return $\mathsf{params} = \langle q, G_1, G_2, \hat{e}, n, P, P_{pub}, H_1, H_2 \rangle$ and $\mathsf{master\text{-}key} = s$.

– Extract
1. Get string $\mathsf{ID} \in \{0,1\}^*$.
2. Let $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in G_1^*$.
3. Return the private key $d_{\mathsf{ID}} = sQ_{\mathsf{ID}}$, where $s$ is the master key.

– Encrypt
1. Let $Q_{\mathsf{ID}} = H_1(\mathsf{ID}) \in G_1^*$.
2. Select a random number $r \in Z_q^*$.
3. Based on the message $M \in \mathcal{M}$, return the ciphertext $C$:

$$C = \langle rP, M \oplus H_2(g_{\mathsf{ID}}^r) \rangle, g_{\mathsf{ID}} = \hat{e}(Q_{\mathsf{ID}}, P_{pub}) \in G_2^*$$

– Decrypt
Input the ciphertext $C = \langle U, V \rangle \in \mathcal{C}$ encrypted with $\mathsf{ID}$. Return the message with the private key $d_{\mathsf{ID}} \in G_1^*$:

$$M = V \oplus H_2(\hat{e}(d_{\mathsf{ID}}, U))$$

### 3.3 Authenticated IBE[6]

– Setup
1. Get a security parameter $k$ and then generate a prime number $q$, two groups of order $q$: $G_1$ and $G_2$, and an admissible bilinear map $\hat{e} : G_1 \times G_1 \to G_2$. Select a random generator $P \in G_1$.
2. Generate a random number $s \in Z_q^*$. Let $P_{pub} = sP$.

3. The PKG selects a random generator $g \in G_1$ and hash function $H_1 : F_q \times G_2 \to \{0,1\}^n, H_2 : \{0,1\}^* \to G_1, H_3 : \{0,1\}^* \times \{0,1\}^* \to F_q, H_4 : \{0,1\}^n \to \{0,1\}^n$.
4. Return $\mathsf{params} = \langle q, G_1, G_2, g, g^s, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle$ and $\mathsf{master\text{-}key} = s$.

– Extract
The PKG calculates the private key of user $\mathsf{ID}_A$: $d_A = H_2(\mathsf{ID}_A)^s$.
– Authenticated Encrypt
User $A(\mathsf{ID}_A)$ uses another user $B(\mathsf{ID}_B)$'s private key $d_A$ to encrypt the message $M \in \{0,1\}^*$:
1. Select a random number $r \in R\{0,1\}^n$.
2. Let $c_1 = H_3(r, M)$ and $c_2 = e(d_A, H_2(\mathsf{ID}_B))$.
3. Return the ciphertext $C = \langle r \oplus H_1(c_1, c_2), E_{H_4(r)}(M) \rangle$.

– Authenticated Decrypt
User B uses A's ID $\mathsf{ID}_A$, his private key $d_B$ and $\mathsf{params}$ to decipher the ciphertext $\langle U, V, W \rangle$.
1. Let $c_2 = e(H_2(\mathsf{ID}_A), d_B)$.
2. Let $r = V \oplus H_1(U, c_2)$.
3. Let $M = D_{H_4(r)}(W)$.
4. Compare $U$ and $H_3(r, M)$.
5. If $U \neq H_3(r, M)$, discard the ciphertext, otherwise return the message $M$.

### 3.4 Hierarchical IBE[5]

The Gentry-Silverberg hierarchical IBE scheme is composed by cascading Boneh-Franklin IBE schemes. In this scheme, every user has an n-tuple $\mathsf{ID}$ in the hierarchy tree. The n-tuple $\mathsf{ID}$ is composed by the $\mathsf{ID}$s of the user itself and its ancestors. All users in the $i$-th level are denoted by $Level_i$. Thus, the root of the hierarchy tree, $Level_0$, is the PKG.

HIBE is composed by the five following algorithms:

– Root Setup
1. Based on a security parameter $k$, generate a big prime $q$ using IG (BDH Parameter Generator).
2. Use $q$ to generate two fields $G_1$ and $G_2$, which satisfy the bilinear map $\hat{e} : G_1 \times G_1 \to G_2$.
3. Pick an arbitrary element $P_0$ in $G_1$, and then pick a random number $S_0 \in Z/qZ$ as the $\mathsf{master\text{-}key}$. Calculate the system parameter $Q_0 = S_0 P_0$.
4. Generate two hash functions $H_1 : \{0,1\}^* \to G_1, H_2 : G_2 \to \{0,1\}^n$.

– Lower-level Setup
For each user $E_t \in Level_t$, specify a random number $s_t \in Z/qZ$.
– Extract

1. For each user $E_t$ with $\mathsf{ID} = \langle \mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_t \rangle$, its father calculates $P_t = H_1(\mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_t) \in G_1$, where $s_0$ is the identity of $G_1$.
2. Return the private key $S_t = S_{t-1} + s_{t-1}P_t = \sum_{i=1}^{t} s_{i-1}P_i$ of $E_t$ and parameter $Q_i = s_i P_0$.

– Encrypt
1. For a message $M$ and $\mathsf{ID} = \langle \mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_t \rangle$, calculate:

$$P_i = H_1(\mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_i) \in G_1$$

2. For any $r \in Z/qZ$, return the ciphertext:

$$C = \langle rP_0, rP_2, \cdots, rP_t, M \oplus H_2(g^r) \rangle, g = e(Q_0, P_1) \in G_2$$

– Decrypt
For ciphertext $C = \langle U_0, U_2, \cdots, U_t, V \rangle$ and $\mathsf{ID} = \langle \mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{id}_t \rangle$, return the message:

$$M = V \oplus H_2 \left( \frac{\hat{e}(U_0, S_t)}{\prod_{i=2}^{t} \hat{e}(Q_{i-1}, U_i)} \right)$$

## 4 Performance Testing

### 4.1 Implementation

Based on the characters of these IBE schemes, we use the same framework for these different cases, except for Cocks IBE scheme. Therefore, we used Stanford IBE Secure Email[1] as the framework for those three schemes, and then modified some kernel functions. In this way, sources for different IBE schemes have the most amount of similar codes, which will decrease error caused by implementation. For Cocks IBE scheme, due to the reason that it is largely different from others, we directly finished its codes.

In summary, we have the final versions of four IBE schemes: Cocks IBE (Cocks), Boneh-Franklin IBE (BF), Authenticated IBE (AIBE), and Hierarchical IBE (HIBE).

### 4.2 Testing Method

Since an IBE scheme has four algorithms, we focus on the performance of each algorithm in order to guarantee that algorithms for different IBE schemes are running in the same condition. For the reason that the initializing process Setup is run only once for a certain IBE system, it is less valuable to test its performance so we just ignore it. Therefore, our goal is to test performance of the three algorithms Extract, Encrypt and Decrypt. For each algorithm of a certain IBE scheme, we record the execution time under different parameters. Obviously, different IBE schemes have different performance, and we will analyze the reasons that make such differences. In order to decrypt the ciphertext correctly, these schemes will add some additional information about parameters. The additional cost of space and bandwidth of such information will be discussed in Section 5.

### 4.3 Environment

| | |
|---|---|
| Processor | Pentium Dual T2330 @ 1.60GHz |
| Memory | 2GB RAM |
| OS | Arch Linux 2.6.38 |
| Compiler | GCC 4.4.4 |
| Timing | use command *time*, with accuracy of 1ms |
| File I/O | work in memory, so execution time can be ignored |

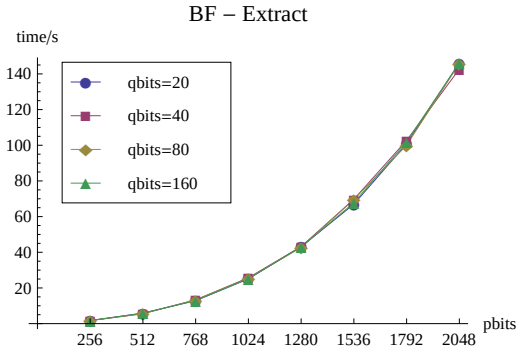
## 5 Results and Analysis

### 5.1 Cocks IBE

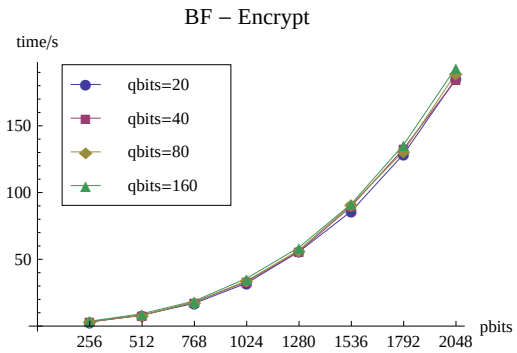
| Pbits | 256 | 512 |
|---|---|---|
| Extraction Time (1000 Keys) | 0.870s | 2.903s |
| Encryption Time | 5.710s | 12.636s |
| Decryption Time | 2.497s | 5.830s |
| Plain Text Length | 2KB | 2KB |
| ciphertext Length | 4126KB | 8224KB |

Compared to other schemes in this paper, Cocks IBE is much easier. However, our test shows its performance is quite low. Even if the master key length is only 256 bits, the encryption/decryption speed is below 1KB/s, which cannot be acceptable in most cases. Moreover, the size of ciphertext is thousands times that of plain text. This conclusion is obvious, since for each bit of the message, the Encrypt algorithm returns two $P$-bit numbers. Therefore, considering both the time and space cost, Cocks IBE is not applicable for real environments.

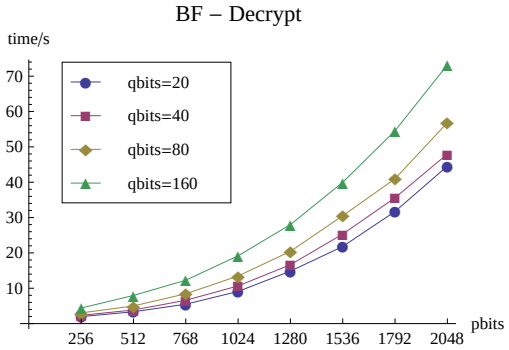
### 5.2 Boneh-Franklin IBE



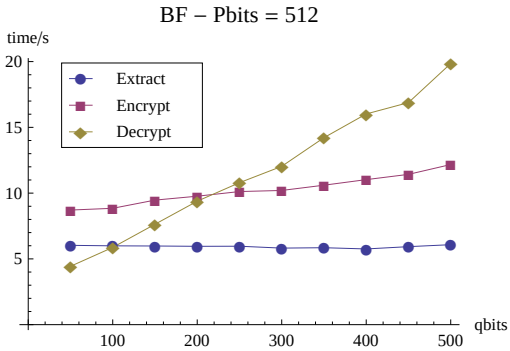

Extract 1000 Keys



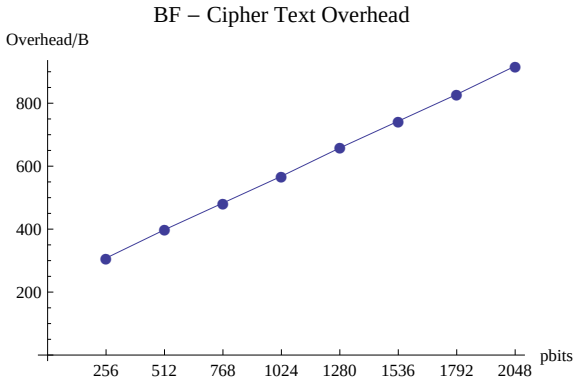

Encrypt 10MB




Decrypt 10MB



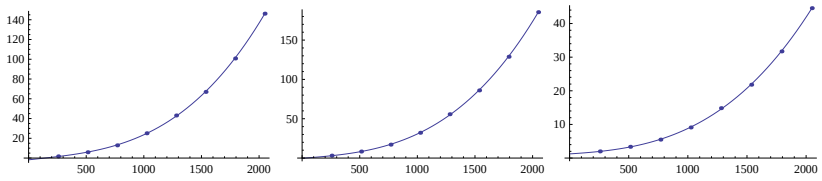

Qbits 1000 Keys / 10MB




ciphertext Overhead

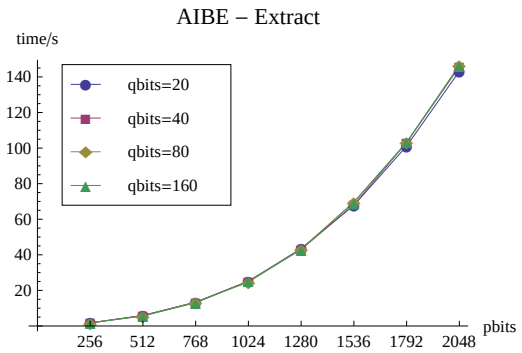
From these figures above, we get:

1. The performance of BF IBE is enough for middle-sized applications.
   Using typical values $\log p = 512, \log q = 160$, we have:
   Extraction speed: $1000/5.863 \approx 170.6$Key/s
   Encryption speed: $10/9.253 \approx 1.08$MB/s
   Decryption speed: $10/7.959 \approx 1.26$MB/s
2. The extraction, encryption and decryption times are nearly cubic with $\log p$, that is mainly because the calculation of $H_1$. Below are the results of cubic polynomial fitting:

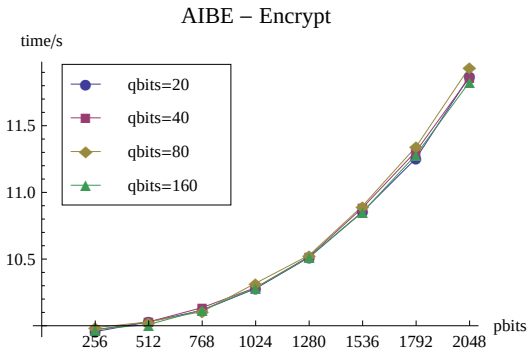



3. If $\log p$ holds constant, Extract is almost unaffected, and Encrypt and Decrypt are nearly linear with $\log q$. However, $\log p$ affects the time of Encrypt and Decrypt much more significantly. This result of performance is made by $H_2$ in this scheme.
4. Extra space cost is linear with $\log p$ and has no relationship with $\log q$, because in the ciphertext, BF IBE needs to store $rP$. Actually, added space less than 1KB is ignorable for large messages. But for small messages, it requires relatively more space and network bandwidth. Since the framework we used outputs base64 ciphertext, we can easily decrease the space cost by one fourth in practice.

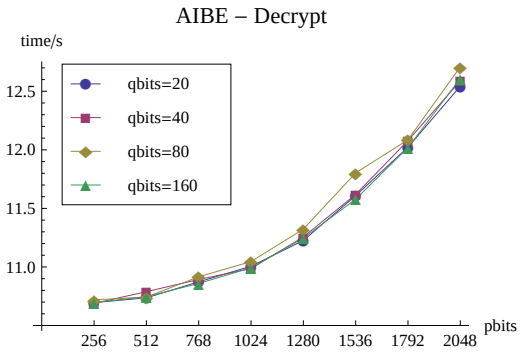
### 5.3 Authenticated IBE



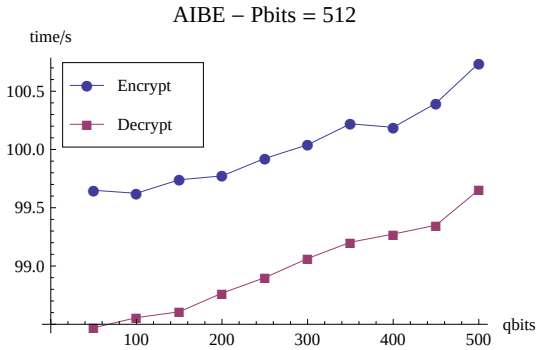

Extract 1000 Keys



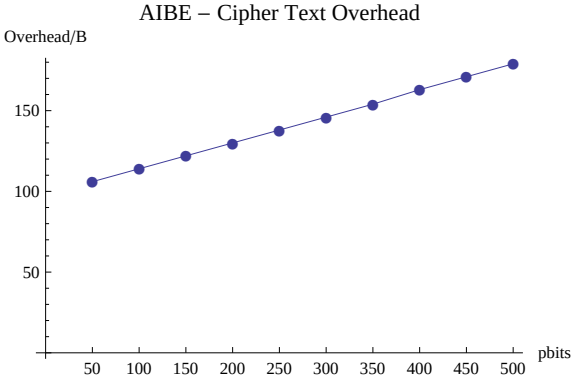

Encrypt 10 × 10MB




Decrypt 10 × 10MB



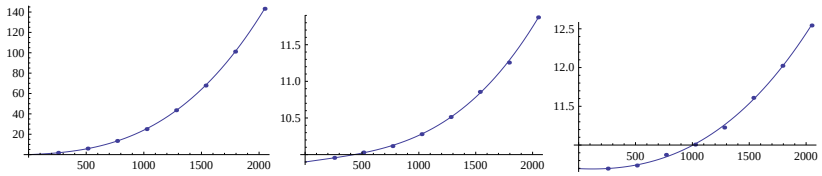

Qbits 1000 Keys / 100 × 10MB




ciphertext Overhead

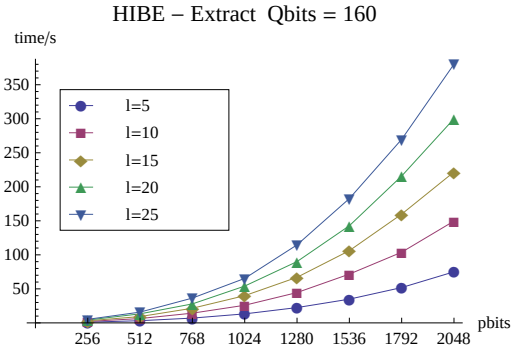
Here are our conclusions:

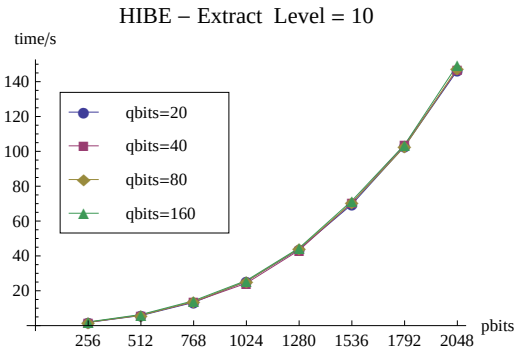
1. AIBE is a little faster than BF, so it is also applicable.
   As the same with BF, let $\log p = 512, \log q = 160$:
   Since the two Extract algorithms are the same, their speeds are not quite different: $1000/5.64 \approx 177.3$Key/s
   Encryption speed: $100/10.009 \approx 9.99$MB/s
   Decryption speed: $100/10.746 \approx 9.30$MB/s
   Because AIBE does not need to calculate the expoential $r$ as in BF, it has a higher encryption/decryption speed, about 10 times that of BF.
2. Like BF, determined by hash functions, the running times of Extract, Encrypt and Decrypt are cubic with $\log p$. Here are the fitting results:



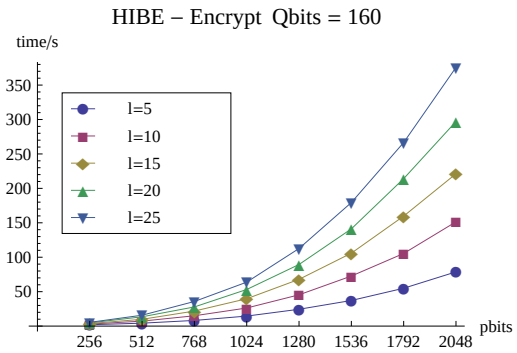

3. If $\log p$ remains unchanged, the relationship between running times and $\log q$ is just like that of BF.
4. Extra space cost is from $E_{H_4(r)}(M)$ in the ciphertext, which is determined by $\log q$ rather than $\log p$. It is also less than 1KB, which may only affect small files.

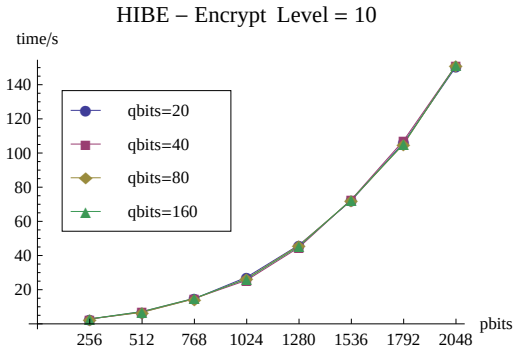
### 5.4 Hierarchical IBE



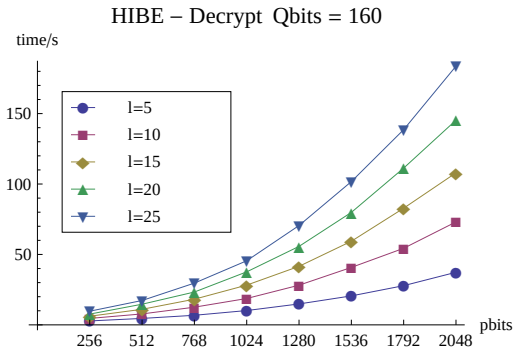



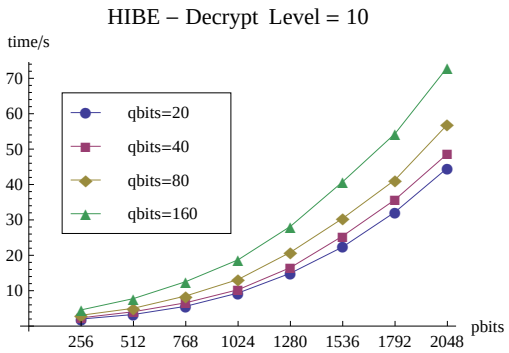

Extract 100 Keys



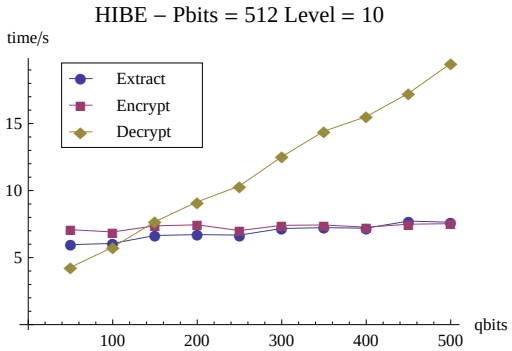




Encrypt 10MB

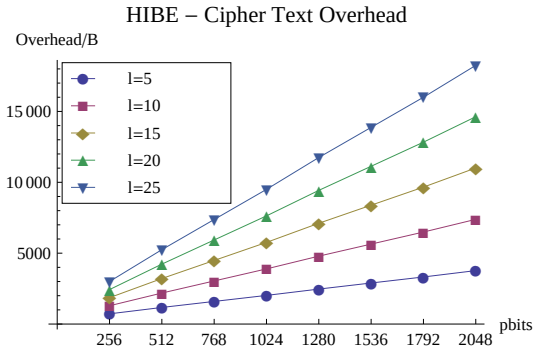





Decrypt 10MB




Qbits 100 Keys / 10MB



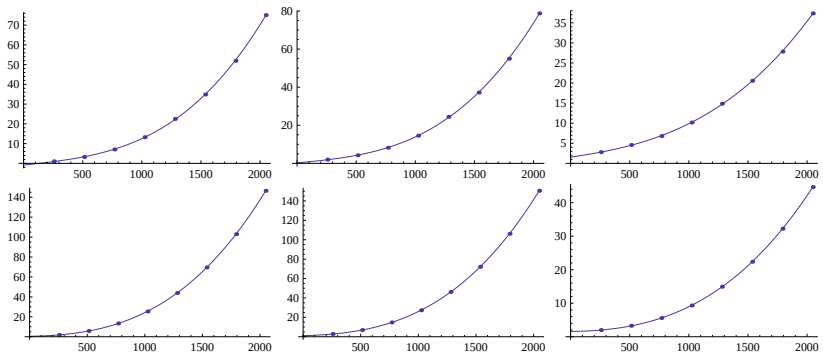

ciphertext Overhead

Here are our conclusions:

1. HIBE is slower than BF.
   Let $\log p = 512, \log q = 160, l = 10$, where $l$ is the number of levels.
   Extraction speed: $100/6.37 \approx 15.7$Key/s
   Encryption speed: $10/7.21 \approx 1.38$MB/s
   Decryption speed: $10/7.769 \approx 1.28$MB/s

Added length of the ciphertext: 2187B, larger than other schemes
The speed is about one tenth that of BF, approximately the same as $1/l$.

2. The relationship between the time/space cost of HIBE and the two parameters $\log p$ and $\log q$ is like that of BF. Here are the fitting results:



3. The time/space cost of HIBE is nearly linear to the number of levels.

## 6 Summary

As a new encryption mechanism different from traditional PKI-based schemes, IBE is free from key distribution and certificate management. Moreover, some typical IBE schemes have enough performance for applications in practice. Here we list the characteristics of all the schemes in this paper:

- **Cocks**
  Cocks is the simplest and slowest scheme. As one bit in message is translated into one big number, its time and space cost is unbearable.
- **BF**
  BF is the commonest implementation in practice. It has a good balance between performance and security.
- **AIBE**
  AIBE can verify the identity of the sender in the decryption step, since the reccipent can decrypt the message correctly only if the sender uses his private key to encrypt it. As it does not contain the exponential calculation in BF, it is a little faster, with the cost of security.
- **HIBE**
  HIBE is a very complicated implementation. The running time is linear with the number of levels. In practice, it is rare to use many levels.

In addition, all these schemes will store some extra information for decryption, which may add much cost while transferring a large amount of small files.

Currently, most IBE schemes take much use of elliptic curve encryption, so they cannot reach a very satisfying performance. Thus, it is critical to improve the performance in the future. Since the $O(\log^3 p)$ time cost of elliptic curve

hash functions is the bottleneck of all the scheme, to make a breakthrough in the performance of IBE schemes, except optimizing such hash functions, the only way is to use totally different encryption/decryption algorithms in other transformation fields. We expect to see more creative improvement in this area.

## References


1. http://crypto.stanford.edu/ibe/.
2. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *Proceedings of Eurocrypt 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer-Verlag, 2004.
3. Dan Boneh and Matthew Franklin. Identity-based encryption from the Weil pairing. In *Proceedings of Crypto 2001*, pages 213–229. Springer-Verlag, 2001.
4. Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *IMA International Conference*, pages 360–363. Springer-Verlag, 2001.
5. Craig Gentry and Alice Silverberg. Hierarchical ID-based cryptography. In *Proceedings of Asiacrypt 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566, 2002.
6. Ben Lynn. Authenticated identity-based encryption. *Cryptology ePrint Archive*, 2002.
7. Amit Sahai, Brent Waters, and Ronald Cramer. Fuzzy identity-based encryption. In *Advances in Cryptology C EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, 2005.
8. Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology - Crypto 1984*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, 1984.
9. Brent Waters and Ronald Cramer. Efficient identity-based encryption without random oracles. In *Advances in Cryptology C EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, 2005.