Cryptography 2, Assignment 2: Certificate-Based and Certificateless Encryption

Rens van der Heijden
University of Twente
TU/e student number 0762495
r.w.vanderheijden@student.utwente.nl

July 2, 2011

Abstract

This paper discusses two new approaches to public key encryption. Both approaches are based on identity based encryption and avoid the inherent key escrow in this system, while still reducing requirements on key exchange. Compared amongst themselves, one scheme exchanges some efficiency for fine-grained revocation, while the other provides high bandwidth efficiency at the cost of some security or convenience. Both schemes will be compared to each other and the more well-known identity based encryption mechanism, as well as a standard public key infrastructure.

1 Introduction

In recent years, many new cryptographic schemes based on the original identity based encryption ideas from Shamir [5] and Boneh-Franklin [2] have been developed to overcome the limitation of both. This paper will review some of the challenges associated with standard public key encryption (PKE), based on a public key infrastructure (PKI). In addition, the challenges associated with an alternative approach, identity based encryption (IBE), will be discussed.

These discussions will be focused on the current challenges, opening the way for a discussion of two new schemes that are designed to have the benefits of both a PKI and IBE. The discussion of these will be centered around key distribution and key escrow, the two main problems associated with the above schemes. The two discussed solutions are Certificate-Based Encryption (CBE) [4] and Certificateless Encryption (CL-PKE) [1] will be explained, discussed and compared. Both of these schemes can be implemented in a way very similar to the Boneh-Franklin IBE scheme; this scheme, and a small piece of the mathematical background, pairings, will also be explained.

The remainder of this paper is arranged as follows: in section 2, the challenges of PKI and IBE will be discussed. Section 3 will provide the necessary background on pairings, trust and implementation of identity based systems. Each of these topics is briefly discussed and applied in sections 4 and 5, where the new schemes will be introduced. These sections will also contain an example implementation from their respective papers and some extensions on them. Finally, in section 6 the schemes will be compared to each other and to PKI and IBE, followed by a conclusion in section 7.

2 PKI and IBE

In this paper, different approaches to public key cryptography are discussed; this section reflects on the current PKI model and IBE. IBE was proposed in the 70s [5] and first implemented quite recently [2]. Confusingly, IBE is an instance of public key cryptography, but the latter term is often associated with a PKI (even though a PKI is not strictly required to perform public key cryptography).

Public key cryptography is based on the use of pairs of keys, rather than a single key as with symmetric key cryptography. Of each pair, the public key is published and used for encrypting data, while the private key is stored and used for decryption. This reduces the problem of key exchange from exchanging something very

secret to exchanging a public piece of information. However, note that without any security, an attacker can typically intercept and replace messages; thus, he is able to change the public key that a sender encrypts data with! This type of attack is a new problem for public key cryptography; publishing the public key without allowing the attacker to exchange it for his own.

2.1 PKI

In a PKI, the problem of public key exchange is solved by involving a trusted third party (TTP). This party provides some proof of validity for the certificate, usually in the form of a certificate. However, to create and trust a certificate, there is usually also public key cryptography involved, so the problem seems recursive. The typical solution is to trust some TTPs, and delegate trust to them, such that one knows sufficient TTPs to have a certificate for the destination one is sending a message to.

There are two models for this trust; the web of trust model and the hierachical model. In the web of trust model, the level of trust can be specified by the user on any other entity with a private key for signing; typically this is established through a secure channel (eg. a meeting in person). On the other hand, in the hierachical model, trust is absolute and centered around a number of large entites called Certificate Authorities (CAs), which sign certificates for many users. Since the average computer user is not interested in setting up for secure communication, the hierachical model is typically used on the internet; this paper will thus also focus on this model.

There are a number of issues associated with both of the above models of a trusted third party; these include cost, both computational and financial, the required trust in another party, but also technical problems. The technical problems are most importantly revocation and certificate size; the impact of a compromised master key (the private key the TTP uses to certify public keys) is also a relevant issue. The issue of revocation is typically solved by Certificate Revocation Lists (CRLs), which are nothing more than a list of certificates that were revoked before expiry and a signature of the CA.

2.2 IBE

Identity based encryption is another approach to public key encryption. The idea is that obtaining the public key of a user may be problematic; thus, instead of obtaining the public key, it would be quite useful if it could be computed from widely available information such as an email address. On a very high level, this is exactly what IBE does. A TTP takes a string, computes a keypair for the user and presents the key to the user. Anyone can repeat the computation for the public key, but obtaining the private key of the user requires the private key of the TTP, similar to the problems for the hierarchial model.

Notice however that this scheme has a significant disavantage; the TTP computes and thus knows the private key of the user (this is called key escrow)! This means that the user must completely trust the TTP in order for this scheme to be secure at all. In addition, notice that this key escrow means that IBE is inherently unsuitable for the purpose of providing non-repudiation. In recent years, there have been incidents concerning the private keys of smaller CAs. A compromise on that scale would be devastating for the security of all the users of this CA, since all messages that were ever sent to these users[1] can now be read by the attacker.

Another disadvantage is that the private key needs to be sent to the user over a secure channel. However, this is less problematic than some papers make it appear, as a secure channel is already necessary to provide a secure way of registering key material at the CA. Typically, such a registration entails the transmission of sensitive personal information, such as social security number for identification and credit card number for payment. Admittedly there are ways around this requirement for a secure channel[2], but it is desirable in order to ensure the correctness and trustworthyness of certificates.

Finally it should be noted that (instant) revocation still requires either a CRL- or OSCP-like mechanism to be provided by the TTP, which should thus still be an online entity with a signficant amount of bandwidth. This is unfortunate, because one of the ideas was to reduce the

[1] Assuming they were sent with the public key corresponding to the private key generated by the CA, of course

[2] There are CAs that give out free certificates; for example, GeoTrust gives out certificates based on an email address for 30-day trials.

load on the TTP. The typical approach to solve this, is to append a year or date to the identity string. While this solves the need for revocation without overhead on the side of the sender, this solution requires frequent updates of private keys and allows no fix if a key is compromised by accident (other than not providing a new key). Note that revocation does not protect against a compromised secret master key, unlike with a certificate authority (assuming no key escrow is in place). This is because the private keys necessary to read messages can be generated at will by the attacker, once he obtains the master key.

2.3 Trust and the Attacker

The above effects of a compromised master key rely on the fact that the TTP is given more information. Thus, the need arises to express the amount of trust from the users to a TTP; this also expresses part of the attacker model for this type of schemes. It turns out that indeed, the IBE scheme inherently requires more trust from the TTP. This requirement may be migitated at the cost of additional overhead and exchange of key material, but this is exactly what IBE attempts to avoid and is thus not a viable solution.

The solutions presented in this paper, CBE and CL-PKE, both decrease the required level of trust back to the level on which normal PKI operates. This is proved by their respective authors by extending the attacker model; in both cases, the attacker may exchange public keys and obtain private keys corresponding to arbitrary public keys, assuming they are not already registered. To model this attacker, both schemes design a game where the attacker may either play the role of an honest but curious CA or a hostile attacker that has no control over the master key. This model is exactly the one used for PKI; without this restriction, if the CA would be the attacker in a PKI, he may forge arbitrary certificates and the attacker would always win.

Note that this level of trust is realistic; in order for the attacker to gain something, he must use the certificates or key material he obtains. However, the attack will be detected shortly after this point, showing that the CA does not have the proper security measures in place. This will lead to grave legal consequences, as well as reducing the trust of any entity in the CA to zero. The point at which such an attack is worth this risk is the point at which is it more secure for the user to run a private CA; such attacks are far too complex and expensive to be executed against a typical user.

3 Background

This section will discuss the high-level mathematical background required to understand why the implementations of IBE and the two new schemes work, as well as the Boneh-Franklin scheme for IBE.

3.1 Elliptic Curves and Pairings

Elliptic curves are equations over a two-dimensional space, typically of the form $y^2 = x^3 + ax + b$, though other representations exist, such as Montgomery curves and Twisted Edwards curves. For this paper, what is relevant is that we can define groups over such a curve by choosing a finite field $\mathbb{F}$ over which the curve is defined and computing the points $P = (x, y)$ for which the equation applies. A point P can be used to generate an additive group $\langle P \rangle = \mathbb{G}_1$, given a generator P; this group can be used for cryptographic purposes is the curve is well-chosen (eg. discrete logarithm or diffie-hellman).

It is also possible to define a pairing on elliptic curves. Here, a pairing is defined as a bilinear mapping $e : \mathbb{G}_1 X \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $\mathbb{G}_2$ is a multiplicative group. There are a number of properties that must be satisified in order to define e and use it for cryptographic purposes. The most important property is bilinearity; given a generator of $\mathbb{G}_1$, P, we can write $e(a \cdot P, b \cdot P) = e(P, P)^{ab} = e(b \cdot P, a \cdot P) = e(ab \cdot P, P)$. Another required property is non-degeneracy, which basically means the mapping is non-trivial; $e(P, P) \neq 1$. Finally, it is required that e is computable in polynomial time, such that it may be used for encryption. This paper will not be concerned with details, such as the type of pairing and the other properties that must hold, which require an significant in-depth understanding of the underlying mathematics.

3.2 Boneh-Franklin

With this basic understanding of pairings, one can now understand the Boneh-Franklin scheme, which is an implementation of IBE [2]. As both of the new schemes, the Boneh-Franklin

scheme consists of a basic scheme and a full scheme. The basic scheme is introduced to understand the scheme, while the full scheme is a slight modification of it to make it resistant against adaptive chosen ciphertext attacks.

In both schemes, a TTP will run a single setup and repeat key generation as often as needed. A user requesting a key will obtain it over a secure channel to the TTP, similar to the X509 if a key is generated by a Key Generation Center (KGC). Note that the user can no longer generate his own key, unless he runs a TTP for his own keys; this would essentially reduce the system back to the original problem, since every user would be his own TTP[3] and thus require distribution of the public keys of these self-managed TTPs.

3.2.1 Basic scheme

This scheme provides an overview of how the system works. It is composed of four steps; setup, key generation, encryption and decryption.

Setup: First, choose groups $\mathbb{G}_1$ and $\mathbb{G}_2$, a map $e : \mathbb{G}_1 \text{ X } \mathbb{G}_1 \rightarrow \mathbb{G}_2$,a random generator $P \in \mathbb{G}_1$ and hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \rightarrow 0,1^n$ for some n. Then generate the public key of the TTP, using a random secret $s \in \mathbb{Z}_q^*$: $P_{pub} = sP$; s will be the master key.

Key generation: To generate a key pair from an identity string $ID \in \{0,1\}^*$, compute $Q_{ID} = H_1(ID)$ and $d_{ID} = sQ_{ID}$. Q_{ID} is the public key, which can be computed from ID at any time, because H_1 is a hash function; d_{ID} is the private key which can**only** be generated by the TTP. Note that ID may be an arbitrary bit string; it is up to the TTP to decide whether or not to assign some ID to some user. If this string is appended with a date, as suggested in the original paper, it should not be possible for an attacker to obtain certificates that have not yet been issued. To be able to send the "messages to the future", the TTP should also prevent the legitimate user from obtaining a keypair with a date in the future.

[3]The term Trusted Third Party is a little strange in this context; the idea is that the user fullfills the role of the TTP himself.

Encrypt: To send a message M to ID, first compute the public key $Q_{ID} = H_1(ID)$, then choose random $r \in \mathbb{Z}_q^*$ and compute the ciphertext as follows: $C = \langle rP, M \oplus H_2(g_{ID}^r)\rangle$, where $g_{ID} = e(Q_{ID}, P_{pub})$. The idea here is that, based on the random oracle model, $H_2(g_{ID}^r)$ generates a random bit string of length n, so the right half of the ciphertext does not reveal M.

Decrypt: After receiving $C = \langle U, V\rangle$, one can obtain the message from V by computing $V \oplus H_2(e(d_{ID}, U))$. This works, because $H_2(g_{ID}^r) = e(Q_{ID}, P_{pub})^r = e(Q_{ID}, sP)^r = e(sQ_{ID}, rP) = e(d_{ID}, U)$, so M is obtained from the above $\oplus$ operation, assuming that C was a properly encrypted ciphertext.

3.2.2 Full scheme

A technique due to Fujisaki-Okamoto [3] is applied to obtain a secure system against chosen ciphertext attacks, resulting in the full scheme. Given some constraints that will not be discussed here, the technique is applicable and can be expressed as follows: $E_2(M) = \langle E_1(\sigma; H_3(\sigma, M)), H_4(\sigma) \oplus M\rangle$, where E_1 is the basic scheme and E_2 is the full scheme.

The implementation of this rule is slightly more complicated. The basic scheme is extended as follows (key generation remains the same):

Setup: Also choose hash functions $H_3 : \{0,1\}^n \text{X} \{0,1\}^n \rightarrow \mathbb{Z}_q^*$ and $H_4 : \{0,1\}^n \rightarrow \{0,1\}^n$.

Encryption: As before, compute $Q_{ID} = H_1(ID)$; however, instead of generating random r, generate random $\sigma \in \{0,1\}^n$ and compute $r = H_3(\sigma, M)$ (which is a random r, assuming the random oracle model). Then the ciphertext is: $C = \langle rP, \sigma \oplus H_2(g_{ID}^r), M \oplus H_4(\sigma)\rangle$, where $g_{ID} = e(Q_{ID}, P_{pub})$. Basically, instead of hiding the message directly, a random bit string is used to encode the message M and this bit string is then encrypted.

Decryption: Given ciphertext $C = \langle U, V, W\rangle$ and decryption key d_{ID}, we can obtain the message by decrypting the random bit string σ (using $\sigma = V \oplus H_2(e(d_{ID}, U))$) and then obtaining $M = W \oplus H_4(\sigma)$. In addition, one must verify

that $U = H_3(\sigma, M)P$. This works for the same reasons as the basic scheme.

4 Certificate-based encryption

In this section, the certificate-based encryption scheme, due to Gentry [4], is discussed in two parts; a high level overview, similar to the one given in section 2 for IBE, and an example implementation similar to the Boneh-Franklin scheme from section 3.

4.1 Overview

The goal of the CBE method is to take away the need for queries by senders to CAs in PKE. The reason for these queries are typically to check whether the certificate of the client has been revoked. If revoked, the private key of the client could be compromised, so the sender should not encrypt the data with the key in the revoked certificate. However, such queries cost a significant amount of resources to execute, especially for large CAs, as well as opening them up to Denial of Service (DoS) attacks[4]. The CBE method must thus prevent receivers from decrypting data if their certification has been revoked. This may seem counter-intuitive, but notice that if a private key is compromised, the receiver will probably be an attacker and not the intended receiver. This is achieved by combining a standard PKI with IBE; the receiver creates his own keypair, but the certificate is created using the public component of this keypair along with the receiver's information. The sender can now compute the necessary key material from the public information of bob and the CA.

One important disadvantage of this scheme is that it looses the property of a public key that can be computed from just the name of a user. However, it is not the case that the attacker can exchange the public key in the public information of a receiver by an arbitrary public key; if this happens, the decryption will simply fail (for both the attacker and the receiver). This is a consequence of the fact that all public info (more than just a public key) is used as input for the certification and encryption procedures. An attacker cannot gain a certificate for this combination, as the identity is verified by the CA before a certificate is provided; if an attacker manager to exchange this information in the public information on the end of the sender, the sender should observe the mismatch between the public data and the intended receiver. This is also the case for current systems; the identity is usually an email address or domain name.

[4]Recently, activistic attacks on all kinds of authorities have increased; it is conceivable that this will also happen to CAs

4.2 Implementation

CBE can be implemented in a way similar to IBE, using pairings. This implementation is discussed in the remainder of this section; as with Boneh-Franklin, the discussion is split into a basic and a full scheme, where the full scheme is created from the basic one using the Fujisaka-Okamoto method.

4.2.1 Basic scheme

This scheme provides an overview of how the system works. It is composed of four steps; setup, key generation, encryption and decryption.

Setup: This phase is exactly the same as the one for Boneh-Franklin and generates groups $\mathbb{G}_1, \mathbb{G}_2$, a map $e : \mathbb{G}_1 \text{ X } \mathbb{G}_1 \to \mathbb{G}_2$,a random generator $P \in \mathbb{G}_1$, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \to 0, 1^n$ for some n, a master secret $s \in \mathbb{Z}_q^*$ and the CA public key $P_{pub} = sP$.

Certification: The user has a keypair $(s_B, s_B P)$ and sends $s_B P$ to the CA along with the other public info necessary for identification (together referred to as ID from here on). The CA will check this information and compute $P_B = H_1(sP, i, ID)$ for validity period i and $Cert_B = sP_B$; $Cert_B$ is then returned to the user. The user then signs his info and adds the certificate as follows: $P'_B = H_1(ID)$ computes $S_B = Cert_B + s_B P'_B$; recall that signing is simply encrypting with a private key; s_B in this case. Recall that $\mathbb{G}_1$ is an additive group and that the Diffie-Hellmann Problem is hard for properly chosen elliptic curve groups, so this is a proper signature.

Encrypt: To send a message M to ID, first compute the public key material $P'_B = H_1(ID)$

and $P_B = H_1(P_{pub}, i, ID)$, then choose random $r \in \mathbb{Z}_q^*$ and compute the ciphertext as follows: $C = \langle rP, M \oplus H_2(g_{ID}^r)\rangle$, where $g_{ID} = e(P_{pub}, P_B)e(PK_B, P'_B)$ and $PK_B = s_B P$. Notice that the latter component of g_{ID}, PK_B, P'_B can be stored, because the public key and public info remain the same for a long period of time (unless B frequently changes his public key), while the first component changes every period.

Decrypt: After receiving $C = \langle U, V\rangle$, one can obtain the message from V by computing $V \oplus H_2(e(U, S_B))$. To see that this works, we need to show that the input for the hash function is the same. This takes some more work than for Boneh-Franklin; $g_{ID}^r = (e(P_{pub}, P_B)e(PK_B, P'_B))^r = (e(sP, P_B)e(s_B P, P'_B))^r = e(rP, P_B)^s e(rP, P'_B)^{s_B} = e(rP, sP_B + s_B P'_B) = e(rP, sP_B + s_B P'_B) = e(U, S_B)$

4.2.2 Full scheme

The basic scheme is extended as follows:

Setup: Also choose a secure symmetric encryption scheme E and hash functions $H_3 : \{0,1\}^n \mathrm{X} \{0,1\}^n \rightarrow \mathbb{Z}_q^*$ and $H_4 : \{0,1\}^n \rightarrow \{0,1\}^n$. As will become clear later, allowing a choice for E is simply more general; $E_k(M) = k \oplus M$ in Boneh-Franklin.

Encryption: As before, compute $P'_B = H_1(ID)$ and $P_B = H_1(P_{pub}, i, ID)$; however, instead of generating random r, generate random $\sigma \in \{0,1\}^n$ and compute $r = H_3(\sigma, M)$, just like in Boneh-Franklin. Then the ciphertext is: $C = \langle rP, \sigma \oplus H_2(g_{ID}^r), E_{H_4(\sigma)}(M)\rangle$, where $g_{ID} = e(Q_{ID}, P_{pub})$. Again, the ciphertext is similar to that of Boneh-Franklin.

Decryption: Given ciphertext $C = \langle U, V, W\rangle$ and decryption key d_{ID}, we can obtain the message by decrypting the random bit string σ (using $\sigma = V \oplus H_2(e(U, S_B))$) and then obtaining $M = E_{H_4(\sigma)}(W)$. Just like Boneh-Franklin, one must also verify that $U = H_3(\sigma, M)P$.

Note that for both basic and full schemes, efficiency in bandwidth and computational cost is similar to the equivalent Boneh-Franklin implementation of IBE; the actual saving comes from the fact that the keys are more secure in the CBE scheme.

4.3 Extensions

Note that in this scheme, the CA is required to recompute every active and valid certificate, even when no revocation occurs. This is problematic from a buisness perspective; the paper introduces an approach to reduce the cost using subset covers. To achieve this, the certification phase now provides a long-lived certificate; this is provided to the sender together with the public key of the receiver. The CA builds a binary tree of depth m containing all of its (up to 2^m) clients at the leaves; the location is included in the certificate as an $m$bit serial number. All tree nodes map to a corresponding ID (and thus keypair); for encryption, the sender now encrypts his message $m + 1$ times, with the keys being those associated with the ancestor nodes of the receiver in the tree. Note the sender can obtain these keys from the serial number and the time period, as with the regular schemes.

The receiver is provided a reconfirmation certificate for every period i. These reconfirmation certificates are key material for one of the nodes between that of each non-revoked client and the root of the tree. The reconfirmation ceritificates are chosen such that only non-revoked clients are children of the chosen certificates. The receiver can decrypt this message with only one decryption step, because the sender has encrypted with all keys between root and the receiver.

This paper will not discuss the implementation of this improvement; however, note that this trick will be costly for users of large CAs. In addition, certificate size and encryption cost are important concerns that are left here; they are now no longer constant, but $O(log(N))$, where N is the amount of clients of the CA. In practice, this would mean $m = 28$ for large CAs[5].

Another improvement that is discussed introduces an additional costs for receiver, in order to gain an additional improvement The above process is done for each time period i, but considering only the revoked nodes in the previos period. Notice that the above costs more as the amount of revoked users increases; thus, this trick will further reduce costs for the CA. The cost is that the receiver has to increment and keep his certificate and that he must perform $m - 1$ additional multiplications for each decryption.

Note that neither improvement is strictly in

[5]this covers N=250 million mentioned in the original paper

the interest of the CA. These improvements increase costs and will thus not likely be accepted by the users. In addition, in time- or resource-constrained environments these mechanisms are not feasable.

5 Certificateless encryption

This section will discuss certificateless encryption, due to Al-Riyami & Paterson [1], in a structure similar to the discussion of CBE in section 4.

5.1 Overview

The goal of CL-PKE is to provide public key derivation from public information, like in IBE, without introducing the key escrow problem. In this scheme, the TTP generates partial private keys D_A, derived from the identity ID_A, which are provided to the user, who can then compute the actual private key S_A using some additional secrets. He can generate the associated public key P_A at any time, given the parameters of the system, which includes the public key of the TTP. A user can generate multiple (S_A, P_A) pairs from the same D_A, by varying the additional secrets used, as long as the same secrets are used for both components of the pair. A sender encrypts messages based on P_A and ID_A; both of which are publically available. Again, if ID_A does not correspond with the destination of the message, the sender can observe this. The public key P_A is published without additional security; the attacker may change P_A to an arbitrary key, but cannot obtain the secret this way.

5.2 Implementation

Just like for IBE and CBE, there is a basic and a full scheme. Both are shown here in a similar structure.

5.2.1 Basic scheme

Setup: This phase is exactly the same as the one for Boneh-Franklin and generates groups $\mathbb{G}_1, \mathbb{G}_2$, a map $e : \mathbb{G}_1 \text{ X } \mathbb{G}_1 \to \mathbb{G}_2$,a random generator $P \in \mathbb{G}_1$, hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1^*$ and $H_2 : \mathbb{G}_2 \to 0,1^n$ for some n, a master secret $s \in \mathbb{Z}_q^*$ and the CA public key $P_{pub} = sP$.

Partial key generation: Given identity ID_A, compute $sH_1(ID_A)$ and send it to the user; he may check it by checking $e(D_A, P) = e(H_1(ID_A), P_{pub})$.

Create secret: Select a random $x_A \in \mathbb{Z}_q^*$. This secret is used to bind public and private keys together; a user can generate multiple secrets to generate multiple key pairs from the same partial private key.

Create private key: Compute the private component of the key pair for x_A as follows; $S_A = x_A D_A$.

Create public key: Compute the public component of the key pair for x_A as follows; $P_A = \langle X_A, Y_A \rangle = \langle x_A P, x_A P_{pub} \rangle$.

Encryption: Given ID_A, $\langle X_A, Y_A \rangle$ and message M; check that P_A is a valid public key by checking $e(X_A, P_{pub}) = e(Y_A, P)$. Subsequently, compute $Q_A = H_1(ID_A)$, choose a random $r \in \mathbb{Z}_q^*$ and create ciphertext $C = \langle rP, M \oplus H_2(e(Q_A, Y_A)^r) \rangle$. Notice this is again nearly the same procedure as that of Boneh-Franklin.

Decryption: Given $C = \langle U, V \rangle$ and the private key S_A, one can obtain the plaintext by computing $V \oplus H_2(e(S_A, U))$. Decryption works on well-formed ciphertexts, because $e(S_A, U) = e(x_A D_A, rP) = e(x_A s H_1(ID_A), rP) = e(H_1(ID_A), x_A sP)^r = e(H_1(ID_A), x_A sP)^r = e(H_1(ID_A), Y_A)^r$.

5.2.2 Full scheme

Just like the other two schemes, Basic CL-PKE is extended to obtain resistance against chosen ciphertext attacks. The algorithms not mentioned here are the same as in the basic scheme.

Setup: Also choose hash functions $H_3 : \{0,1\}^n \text{X} \{0,1\}^n \to \mathbb{Z}_q^*$ and $H_4 : \{0,1\}^n \to \{0,1\}^n$.

Encryption: Given ID_A, $\langle X_A, Y_A \rangle$ and message M; check that P_A is a valid public key by checking $e(X_A, P_{pub}) = e(Y_A, P)$. As before, compute $Q_A = H_1(ID_A)$; however, instead of generating random r, generate random $\sigma \in \{0,1\}^n$ and compute $r = H_3(\sigma, M)$, just

like in Boneh-Franklin. Then the ciphertext is $C = \langle rP, \sigma \oplus H_2(e(Q_A, Y_A)^r), E_{H_4(\sigma)}(M) \rangle$.

Decryption: Given ciphertext $C = \langle U, V, W \rangle$ and private key S_A, we can obtain the message by decrypting the random bit string σ (using $\sigma = V \oplus H_2(e(S_A, U))$) and then obtaining $M = E_{H_4(\sigma)}(W)$. Just like Boneh-Franklin, one must also verify that $U = H_3(\sigma, M)P$.

5.3 Improvements

In some cases, it might be desirable to reduce the required trust in the TTP. In the above, the TTP can still generate several D_A for a specific identity; this allows him to generate several keys for a user without his knowledge. There is a simple patch that fixes this problem, but restricts the user by allowing him to create only one public key. This patch requires the user to fix the secret x_A before he obtains the partial private key by requiring that $D_A = sH_1(ID_A||P_A)$. The private key is still the same, as is the functionality of the system, except for the option of generating multiple x_A for multiple keys. The required trust is now the same as normal PKI.

A signature scheme can be created from the basic CL-PKE scheme; the concept is the same as a regular signature scheme, but the implementation is slightly different. This paper will not discuss the details, but only present the resulting implementation using pairings. All functions are the same, except encrypt and decrypt (replaced by sign and verify) and the setup, which now only provides one hash function $H : \{0,1\}^* \text{ X } \mathbb{G}_2 \rightarrow \mathbb{Z}_q^*$.

Sign: Given message M and key S_A, select random $a \in \mathbb{Z}_q^*$, compute $r = e(P, P)^a$ and $v = H(M, r)$. Then the signature is $\langle vS_A + aP, v \rangle$.

Verify: Given the signature $\langle vS_A + aP, v \rangle$, message M, identity ID_A and key $\langle X_A, Y_A \rangle$, check that the public key is valid by checking $e(X_A, P_{pub}) = e(Y_A, P)$. Then, compute $r = e(U, P)e(H(ID_A), -Y_A)^v$ and check that $v = H(M, r)$. If it holds, the signature is valid.

6 Comparison

In this section, both of the above schemes are compared with respect to efficiency, both in computation and in bandwidth, as well as properties like security and usability. Compared to traditional IBE, both schemes offer more security; as discussed briefly in section 2.3, both CBE and CL-PKE are better in this respect, offering additional security by reducing the required amount of trust. For CL-PKE, the required level of trust is reduced to the level of a PKI using an additional extension. For CBE, on the other hand, the required amount of trust is fixed to that of a PKI from the beginning. This shows a weakness in the CL-PKE system; in the standard schemes, some information may be leaked when D_A is revealed to a third party.

With respect to bandwidth efficiency during communication, each basic and full implementation has roughly the same performance as the boneh-franklin scheme. In CBE, the main gain is that only the holder of the certificate needs to communicate with the TTP in order to obtain a fresh certificate for period i. On the other hand, in CL-PKE, the sender must obtain a public key and the identity of the receiver; however, this need not be obtained from the TTP, nor sent over a secure channel. Notice that while this is convenient, it also exposes CL-PKE to denial of service attacks, so an actual implementation should seriously consider using a secure channel if appropriate key material is available.

For the TTP, the computational load in CBE is quite heavy; it needs to compute many certificates in each time period i, even when revocation is rare. Much of this cost can be avoided at the cost of additional load on the users and bandwidth. However, as noted in the discussion of these extensions, the costs for the users seen excessive, and there is no incentive for these users to accept these costs. In CL-PKE, the TTP does not have to do anything after the keys have been generated.

However, revocation in CL-PKE is based on the same principle as that of IBE; to revoke a certificate is to wait for its expiry. This will result in very short expiration periods, or a very high security risk, reducing the gain in efficiency for CL-PKE. Nevertheless, CL-PKE could have applications in resource-constrained environments, especially because certificates are no longer required. On the other hand, CBE proves powerful and fine-grained revocation capabilities; typically, when reconfirmation certificates are used, the refresh period will be short. While these reconfirmation certificates are small, the cost for this powerful revocation

mechanism may not outweight the bandwidth requirements and fast expiry of the certificates if not refreshed.

Note that for general usage, a PKI will probably remain the core system for many years to come, as many investments have been made by CAs that will not spend additional money if there is no serious gain. Both schemes would require users to transfer to different software; secure implementation of cryptography is another critical stage that must be passed before practical application is possible.

7 Conclusion

This paper has described and discussed two novel approaches to public key cryptography. Both approaches are designed to avoid the revocation problems in standard PKI while also avoiding the key escrow problem from IBE. However, CBE requires significant amounts of computational effort from the TTP in order to function. On the other hand, CL-PKE does away with the standard approach to cryptography by discarding certificates completely. While this saves bandwidth, there is the danger of a Denial of Service attack by arbitrarily exchanging certificates. These Denial of Service attacks may be negated by other mechanisms; CL-PKE is thus an interesting scheme for resource-constrained environments. For environments where fine-grained revocation is necessary, CBE provides an excellent alternative, at the cost of resource consumption. However, for the foreseeable future, PKIs will be the main system for providing public key encryption.

References

[1] S. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. Cryptology ePrint Archive, Report 2003/126, 2003.

[2] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.

[3] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO*, pages 537–554, 1999.

[4] C. Gentry. Certificate-based encryption and the certificate revocation problem. In E. Biham, editor, *Advances in Cryptology EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 272–293. Springer Berlin / Heidelberg, 2003.

[5] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 1984*, pages 47–53. Springer, 1984.