

# Identity Based Encryption Using Multiple Trust Authorities in Ad Hoc Networks

Christie Shuster

May 6, 2004

## 1 Introduction

Ad hoc networks are becoming more and more popular as the use of mobile computing devices increases. While ad hoc networks have many advantageous properties, they also pose several security challenges which are unique to networks of this kind. One of the main challenges that ad hoc networks have to face is authentication. Authentication is difficult to achieve in ad hoc networks because some type of key distribution system is needed to develop the authentication mechanism. Public key cryptosystems are a possible solution to this challenge, but then a certain level of trust must be established before the keys may be used. In many cases this can be accomplished using a certificate authority or central trust authority. However, this brings about many contradictions to the properties of ad hoc networks.

This paper will present some of the security challenges and opportunities encountered when utilizing ad hoc networks. Security attributes will be discussed so that the security of a network may be evaluated. Following the discussion of the security attributes, identity based encryption schemes based on the Weil pairing, presented by Boneh and Franklin in [1], will be presented. Identity based encryption will then be enhanced using the idea of multiple trust authorities, which was presented originally by Chen et al in [2]. Finally, ad hoc networks using multiple trust authorities in an identity based encryption scheme will be discussed. Throughout the paper two examples (sections 2.3 and 2.4) will be used to illustrate the ideas of [1], [2], and their applications to ad hoc networks.

## 2 Ad Hoc Networks

In order to discuss identity based encryption schemes within ad hoc networks, one must first have a general understanding of these networks. This section will begin by defining an ad hoc network and the properties associated with ad hoc networking environments. Next, the security requirements of such a network will be discussed. After presenting the properties and the definition of ad hoc networks, an example application will be presented. These scenarios were constructed for the purpose of modeling identity based encryption schemes where secure ad hoc networking systems might be used.

## 2.1 Definitions

First consider some qualities of ad hoc networks. In order for a network to qualify as ‘ad hoc’, the users (also known as nodes) should be able to be dynamic. In other words, nodes are considered to enter and exit the network as needed for their purposes of communication. Because we require that the nodes be allowed to connect/disconnect indeterminately, an ad hoc network will not have a fixed topology. Recall in star topologies, for example, a central unit (often referred to as a central authority) exists through which much or all of the communication is routed. Central authorities should not exist in ad hoc networks because the property of being a central authority directly conflicts with the property of being dynamic.

There are many definitions of ad hoc networks. Now that the dynamic and decentralized properties of ad hoc networks have been introduced, here is a definition of ad hoc networks that suits the needs of this paper.

**Ad Hoc Networks** are networks in which no fixed topology resides; hosts are allowed to be as dynamic as they choose; and ideally, every node is of equal importance.

Notice that we will only achieve true decentralization in an ideal environment. Because of network layouts and transmission traffic patterns, some nodes will likely perform more transmissions than others based on their locations and/or past transmissions. Thus, we only require an adequate level of decentralization in order to qualify as an ad hoc network.

## 2.2 Security

Ad hoc networks present many security challenges and opportunities, particularly with regard to their decentralized and dynamic natures. Five security attributes will be used to evaluate the security of the systems mentioned in this paper. It is important to note however, that only those challenges and opportunities related to identity based encryption will be presented. For a more thorough discussion of security issues in ad hoc networks, the reader is encouraged to consult [4] and [5].

### 2.2.1 Security Attributes

In order to discuss security, a set of security attributes should be considered. These attributes, which are generally used to identify the strengths, weaknesses, and overall security of any given system, are confidentiality, integrity, availability, authentication, and non-repudiation. Following are short descriptions of these terms and explanations of how they can be used in evaluating security issues within ad hoc networking environments.

*Confidentiality* can be defined as the disclosure of information only to authorized persons or services. Both the information being transmitted and the routing data need to be kept confidential. Since routing data describes where an information packet comes from, goes to, and everywhere the packet has been between sender and receiver, much can be gleaned about who sent the packet or what information may be contained within the packet. As will be discussed in section 2.4, some situations require sender location be kept confidential in order to ensure the physical security of the node.

*Integrity* assures hosts that information packets will be delivered without corruption. Because these networking environments are decentralized, packets will be passed through multiple

nodes before arriving at the final destination. Hosts need to be confident that information being sent across the network is not being altered, corrupted, or terminally lost.

*Availability* is concerned with the durability of the network. Hosts need to be able to communicate with each other at any time. If services are unavailable due to denial of service attacks, important messages may not be delivered within the time needed. Some ad hoc networks are used during emergency situations or military operations, for example, when information needs to be passed quickly. If the network is unavailable, the costs could be catastrophic.

*Authentication* helps hosts determine if other hosts are who they claim to be. Since nodes may come and go as they please, it is possible for adversaries to act as former or new nodes. Some applications of ad hoc networks need certification that hosts have not been compromised or impersonated. Sections 2.3 and 2.4 give examples of different levels of needed authentication.

Finally, *non-repudiation* eliminates the possibility of nodes denying receipt of information. Non-repudiation is oftentimes used for the identification of compromised nodes present in ad hoc networks.

Confidentiality, integrity and authentication will be the security attributes of most interest to this paper. For more information on non-repudiation and availability see [5].

### 2.2.2 Security Challenges and Opportunities

In general, mobile hosts (easily movable computing devices) are not a requirement for implementing ad hoc networks. However, most current standard ad hoc networking applications use different types of mobile computing devices like notebooks, personal digital assistants (PDAs) and cellular telephones. This paper will focus mostly on what is sometimes referred to as mobile ad hoc networks (MANETs). Even if all of our nodes are not mobile, the mobile hosts generate some of the main security issues faced in this type of networking environment. Thus, without a loss of generality, we will assume for the remainder of this paper that we are working within a MANET environment.

First consider confidentiality. Confidentiality will be beneficial to the conference example of section 2.3, but will be essential in the military implementation example found in section 2.4. The dynamic nature of ad hoc networks applies well to maintaining confidentiality because every time information is sent from sender to receiver, the path taken could be different. As a result of the dynamic routing in MANETs, locating or identifying any given user would be difficult even if routing information is made available. Similarly, the decentralization of ad hoc networks makes it difficult to identify any one host unless that one host identifies him/herself. Because there is no central authority which can be compromised in ad hoc networks, this information cannot be made readily available to an adversary through the central authority.

Next, integrity is essential in any networking scheme. If persons are unable to know if the information they receive is correct or corrupt, there is no point in trying to communicate. Integrity can both be increased and decreased with the properties of ad hoc networks. The nature of decentralization in ad hoc networks provides the opportunity to improve integrity since any one message can be sent along several different paths. If one node corrupts or deletes a message, there is a chance that the message will succeed proper delivery on one of the other paths. In addition, if a similar message is received by the destination host multiple times from different nodes along different paths, the receiver can either determine the correct message or they can ask the sender to send the message again along different routes. Transmitting the

message multiple times along different routes will help the destination host know better what is being sent and will also help evaluate who is corrupting information. Using the decentralization property of ad hoc networks in this way increases the level of integrity that can be achieved. However, integrity can also be worsened by the dynamic nature of ad hoc networks. Because nodes can come and go as needed, information packets sent on pathways where nodes leave will be lost. Thus, redundancy and more network traffic are required to counteract this loss of integrity.

Finally, authentication is of much concern, particularly to the topics of this paper. The lack of a central authority makes authentication particularly difficult in ad hoc networks. In most public key cryptosystems, a central authority, known as the trusted key authority or the trusted authority, distributes the public-private key pairs. Clearly, this will not work in an ad hoc network scheme unless the nodes in the network also have contact with a trusted authority outside the ad hoc network. Symmetric key cryptosystems could be used in place of public key cryptosystems, but then one has to consider the fundamental problem of key distribution. How can one transmit a symmetric key over an ad hoc network unless they have prior secure communication? It quickly becomes clear that not having a centralized trusted key authority is problematic for implementing most common cryptosystems. This issue will be directly addressed in section 5.

Now that security challenges and opportunities have been presented, especially in relation to confidentiality, authentication, and integrity, we will look at two possible scenarios in which ad hoc networks would be beneficial. Please note, however, that these are fictional schemes which have been created for the purposes of illustrating key concepts within this paper.

### 2.3 Example: Conference Implementation

Suppose a group of cryptographers are travelling to a conference in Somewheresville, Iowa. Attending the conference are Alice, Bob, Carol and Dave, among many others. Evelyn and Frank are the hosts/coordinators of the event. The coordinators have set up an ad hoc network for all of the conference attendees. At registration, each paid conference attendee is given an identifier,

$$ID_{name} = \text{conference attendee's name}$$

tied to their identity which will be publicly known. Call the set of valid identifiers  $I$ . The attendees are also given a public-private key pair,

$$[Q_{ID_{name}}, S_{ID_{name}}]$$

to use when transmitting messages across the network. All conference persons not carrying a public-private key pair should not be trusted. Note that any persons, namely George and Harriet, entering the conference area can listen to traffic on the network and can claim they have an identifier  $ID_{name}$ . Since George and Harriet did not pay to attend the conference,  $ID_{Harriet}, ID_{George} \notin I$ .

In this example, conference attendees may come and go as needed without disrupting the flow of network traffic. No central or trusted authority, other than the original setup of the public-private key pairs, exists in the network. Notice also that all of the nodes are of relatively

equal importance, barring location and transmission history, while on the network. Because the dynamic and decentralized properties stated in section 2.1 are satisfied, this conference implementation fits our definition of an ad hoc network.

### 2.3.1 Security Issues for Conference Implementation

First, consider the confidentiality of the conference. We have two levels of confidentiality to consider: confidentiality within the conference attendees and confidentiality between the conference attendees and the outside world. Confidentiality from persons outside the conference is important because anyone can connect into an ad hoc network by virtue of being dynamic. The confidentiality from members of the outside world will thus make sure that any conference documents sent over the network will remain confidential to the attendees of the conference. Confidentiality between persons within the conference is beneficial because the attendees want to have the comfort of knowing their conversations will remain private while on the network.

Authentication is important because confidentiality can take place only after authentication has been confirmed. If Alice wants to discuss the most recent session on public key cryptosystems with Bob, she has to somehow prove to Bob that she is Alice. Otherwise, Harriet, a non-attendee, might be able to impersonate Alice and obtain conference details from Bob. Authentication will help Alice, Bob and the other attendees make sure that the conference materials stay confidential to the participants of the conference.

Finally, consider integrity. Suppose Alice wants to send Bob a presentation, but because of their physical location at the time the information will have to go through George to get to Bob. Integrity will be important since Alice wants to make sure that Bob is getting the presentation she sends him. Alice also wants to be confident that Bob is not receiving corrupted presentations from George. Integrity will help assure Alice that the proper presentation got to Bob, even if George decides to delete or alter the message transmission.

This is a fairly relaxed example of an ad hoc network scenario. Information exchange within conferences is not as highly susceptible to attack as our next example which is much more formal and strict in the implementation of the ad hoc network.

## 2.4 Military Network Implementation

Suppose we have mobile military units  $A, B, C, \dots$  out on force in some hostile territory. We want these units to have communication with each other as often as possible, if not constantly, but we do not want any information to be available about their location within the hostile territory. Should their locales be discovered, the lives of the unit members could be threatened. Throughout the military occupation, messages might be sent to certain members of the unit or the entire unit all together. One message could be sent to ‘id.commander.unitA.hostileterritory.navy.us’ and another message could be sent to ‘id.hostileterritory.navy.us’. Using identifiers given to all military personnel at the time of entrance, messages may or may not be read by different personnel. As will be seen in section 4.1, the contexts associated with an identity will be crucial for authentication.

Suppose we have the following setup: Commander Alice and Petty Officer Bob are in unit A working with the Navy, Technology Specialist Carol and Commander Dave are in unit B working with the Army, and Sergeant Evelyn and Petty Officer Frank are in unit C working with the Navy. Suppose also that we have adversaries George and Harriet. George is a student

studying to be a Technology Specialist in the navy. He is stationed at the Naval Academy of Sciences, but has not been authorized for service in the hostile territory. Harriet is a spy working for high officials within the government of the hostile territory.

A dynamic nature will be inevitable for the military environment as members of the units may come and go as needed, members may be moved in/out of units, and units may be added/deleted as necessary. Thus, we have fulfilled the requirement of dynamism of ad hoc networks. Notice that the decentralization property has also been satisfied. If there was one central authority, that unit would be a sure target and would cease existence quickly! Once again, because we have the properties of having a dynamic, decentralized network with no set infrastructure, we have an ad hoc networking environment.

### 2.4.1 Security Issues for Military Implementation

Now we must address the security concerns of this ad hoc network. First, consider confidentiality outside the military network. Confidentiality in any military operation is essential, particularly so when working within a hostile territory. Should the wrong information be discovered by the adversary, the lives of the military members in operation could be at stake. Thus, confidentiality outside the military network is crucial. How much of a problem is confidentiality between specific members of a military unit or even confidentiality between different military units or branches? Again, confidentiality is crucial. The armed forces have been using the idea of 'need to know' for many years. One reason for this might be the fewer people that know about something, the less likely they are to talk about it. Another potential reason for keeping information on a 'need to know' basis is to make sure that no one person can put together the big picture. Should one of the members of unit *A* be captured, that person would only have bits and pieces of information about the operations taking place within the hostile territory. One other type of confidentiality to consider that of the routing information. This information should remain confidential because the location of the military personnel could be exposed if the adversary could gain the routing information. If the adversary knows where some of the stationary units are located, and they see that the information is traversing the network in some manner, they could evaluate a relative location for the unit personnel, thus compromising the security of not only of the individuals, but also of their equipment. Therefore, keeping information confidential as much as possible in a military setting is essential.

As in the previous example, confidentiality can only take place after a person's identity has been established, i.e. authentication of some kind must be achieved. In order for Alice in unit *A* to read messages sent to her, she needs to be able to establish that she is indeed Alice from unit *A*. Otherwise, Harriet might be able to impersonate Alice and could then gather information to tell the hostile government about the operation plans of the United States. On the other hand, Alice needs to be kept from authenticating as Carol, because Alice has no 'need to know' basis for technology specialist information. If Alice was able to authenticate as Carol, Alice could perhaps gain information about the private key of Carol's unit, unit *B*, thus decreasing the separation of information between the military units. Hence, authentication is also of utmost importance.

Finally, consider integrity. Because authentication and confidentiality are so important in an example of this nature, integrity can be somewhat relaxed. While integrity is still important in this example, namely making sure that the units get the correct information at the correct

times, the military is willing to lessen the importance of the integrity in order to maintain higher levels of confidentiality and availability. For example, in order to keep confidentiality high, a slower algorithm might be used to compute the encryption of a certain message. During the transformation of the message into ciphertext, some information might have been corrupted or lost as a result of some computer malfunction. If Alice were to receive this message and decrypt it, only to notice that the message was still garbled, she would simply ask the sending party to try again. To a point, the military finds this type of integrity loss acceptable, because they are maintaining high confidentiality as a result. On the other hand, it could still be disastrous if a unit received the command ‘Fire’ at an inappropriate time or if they did not receive the command ‘Retreat’ because the adversary intercepted the message.

In conclusion, confidentiality and authentication are of extreme importance in this high security example. While integrity is not as important as confidentiality and authentication in this example, it is still necessary in order to make sure that all operations remain functional within the hostile territory. Now we will consider these examples specifically in relation to identity based encryption schemes, then in relation to identity based encryption schemes using multiple trust authorities.

### 3 Identity Based Encryption Schemes

Identity based encryption was introduced in 1984 by A. Shamir [7]. Shamir wanted to know if there was a way to make certificate management easier for e-mail systems. In other words, Shamir was interested in finding some way to encrypt messages using a public identifier, known to everyone without validation — such as an e-mail address — as the public key. This would eliminate the need to exchange public keys, keep key directories, and use the extensive services of a third party. In the case of Identity Based Encryption systems (unlike public key cryptosystems), the third party is only needed to establish individual private keys. After this is achieved, the third party is no longer needed. After several attempts to find such a usable encryption scheme, Boneh and Franklin published an implementable version of identity based encryption[1].

Identity based encryption (IBE) schemes are based on four operations: setup, extract, encrypt, and decrypt. In order to understand these operations in relation to the Boneh-Franklin IBE scheme, however, a brief introduction to bilinear maps is needed. Please note that the information contained within this section comes from [1].

#### 3.1 Bilinear Maps

The identity based encryption scheme developed by Boneh and Franklin is dependent on a bilinear map. Let  $\mathbb{G}_1, \mathbb{G}_2$  be cyclic groups with prime order  $q$ . We will write  $\mathbb{G}_1$  additively and  $\mathbb{G}_2$  multiplicatively. Consider a map  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$  with the following three properties.

1. For all  $P, Q \in \mathbb{G}_1$  and all  $a, b \in \mathbb{Z}$  the following hold:
  - $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$
  - $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$
  - $\hat{e}(P, Q + R) = \hat{e}(P, Q)\hat{e}(P, R)$ .

If these properties hold, the function  $\hat{e}$  is considered to have *bilinearity*.

2.  $\hat{e}$  must be *non-degenerate*, i.e. not all pairs in  $\mathbb{G}_1 \times \mathbb{G}_1$  will map to the identity  $1 \in \mathbb{G}_2$ .
3. There exists some algorithm which computes  $\hat{e}(P, Q)$  efficiently for any  $P, Q \in \mathbb{G}_1$ . If such an algorithm exists, then the map  $\hat{e}$  is said to be *computable*.

If a map  $\hat{e}$  has bilinearity, is non-degenerate, and is computable, it is said to be a *bilinear map*. Two of the most common bilinear maps used for encryption schemes are the Weil pairing and the Tate pairing. More information on the Weil and Tate pairings and their implementation in IBE schemes can be found in [1] and [3].

### 3.2 Identity Based Encryption Operations

As previously mentioned, the four operations needed to implement an identity based encryption scheme are setup, extract, encrypt, and decrypt. Following is a brief explanation of what each of these operations does and why it is needed.

*Setup* is completed by a trusted authority called the Private Key Generator (PKG). This is the trusted authority that generates public-private key pairs and checks the validity of these keys when requested by the general public. During setup, system parameters (information known to the public) and a master key (known only by the trust authority) are determined.

The system parameters  $\langle \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, n, H_1, H_2 \rangle$  that are needed for identity based encryption are defined as follows:

- $\mathbb{G}_1, \mathbb{G}_2$  are cyclic groups of prime order  $q$ .
- $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$  is a bilinear map.
- $P \in \mathbb{G}_1$ .
- $s \in \mathbb{Z}_q^*$  is randomly selected to compute  $P_{pub} = sP$ .
- $n$  is the maximum length of messages chosen to populate the message space  $\mathcal{M} = \{0, 1\}^n$ .
- $H_1 : \{0, 1\}^* \mapsto \mathbb{G}_1^\times$  is a cryptographic hash function such that  $\mathbb{G}_1^\times = \{x \in \mathbb{G}_1 | x \neq O\}$  where  $O$  is the identity element of the additive group  $\mathbb{G}_1$ .
- $H_2 : \mathbb{G}_2 \mapsto \{0, 1\}^n$  is a cryptographic hash function.

The overarching master key will be the randomly selected  $s \in \mathbb{Z}_q^*$ . This master key will be used to create all further private keys. Notice that this introduces the problem of key escrow. In section 4.1, key escrow in IBE schemes is addressed. Notice also that the system parameters used for setup are public information! Having the system parameters as public information does not weaken the cryptosystem because the Boneh-Franklin IBE scheme is based on the hardness of the Bilinear Diffie-Hellman problem.

**Bilinear Diffie-Hellman Problem:** Given  $\langle P, aP, bP, cP \rangle$  for some  $a, b, c \in \mathbb{Z}_q^*$  compute  $x \in \mathbb{G}_2$  where  $x = \hat{e}(P, P)^{abc}$ .

For more information on the Bilinear Diffie-Hellman problem and the proof that the Boneh-Franklin IBE scheme is based on the hardness of the BDH problem, see [1]. Once setup has been completed, the system parameters and the master key will be used to run the extract operation.

*Extract* is used to compute the private key associated with the identifier  $ID$ . Any user believed to be trustworthy by the PKG can obtain a private key with their identity  $ID$ . Extract is performed with  $ID \in \{0, 1\}^*$  as input. Two values will be the output of Extract. First,

$$H_1(ID) = Q_{ID} \in \mathbb{G}_1^\times$$

will be computable by anyone holding the string  $ID$ . Second, the private key

$$S_{ID} = sQ_{ID}$$

will be computed using the master key  $s$ . Notice that only the PKG can compute  $S_{ID}$ . Now that the identifier  $ID$  has a public-private key pair

$$[Q_{ID}, S_{ID}]$$

directly associated with it, encryption and decryption can occur.

*Encrypt* is the operation used to encode a message  $M \in \mathcal{M}$  using the public key identifier  $ID$ . First, Encrypt computes  $Q_{ID}$ . Next, a random value  $r \in \mathbb{Z}_q^*$  is chosen and the value

$$g_{ID} = \hat{e}(Q_{ID}, P_{pub}) = \hat{e}(Q_{ID}, sP)$$

is computed. Finally, the ciphertext  $C$  will be the pair

$$C = \langle rP, M \oplus H_2(g_{ID}^r) \rangle = \langle U, V \rangle.$$

Notice that  $H_1, H_2, P, P_{pub}, Q_{ID}$  are all public information. Thus, Encrypt takes an arbitrary message and computes the ciphertext using only public information and the public key  $ID$ .

*Decrypt* is used to decode the message  $M \in \mathcal{M}$  which is hidden inside the ciphertext  $C$ . In order to decrypt the message, one uses the private key  $S_{ID}$  to compute

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

Decrypt decodes the encrypted ciphertext output because the owner of the identifier  $ID$  can compute  $H_2(\hat{e}(S_{ID}, U))$  and thus, can compute

$$\begin{aligned} V \oplus H_2(\hat{e}(S_{ID}, U)) &= M \oplus H_2(g_{ID}^r) \oplus H_2(\hat{e}(S_{ID}, U)) && \text{(substitute } V) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, P_{pub})^r) \oplus H_2(\hat{e}(S_{ID}, U)) && \text{(substitute } g_{ID}) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, P_{pub})^r) \oplus H_2(\hat{e}(S_{ID}, rP)) && \text{(substitute } U) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, sP)^r) \oplus H_2(\hat{e}(S_{ID}, rP)) && \text{(substitute } P_{pub}) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, sP)^r) \oplus H_2(\hat{e}(sQ_{ID}, rP)) && \text{(substitute } S_{ID}) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, sP)^r) \oplus H_2(\hat{e}(sQ_{ID}, P)^r) && \text{(bilinearity of } \hat{e}) \\ &= M \oplus H_2(\hat{e}(Q_{ID}, P)^{rs}) \oplus H_2(\hat{e}(sQ_{ID}, P)^r) && \text{(bilinearity of } \hat{e}) \\ &= M \oplus H_2(\hat{e}(sQ_{ID}, P)^r) \oplus H_2(\hat{e}(sQ_{ID}, P)^r) && \text{(bilinearity of } \hat{e}) \\ &= M && \text{(properties of } \oplus). \end{aligned}$$

It is important to note the security of this IBE scheme. The overall scheme is based on the hardness of the Bilinear Diffie-Hellman assumption. In addition, the Boneh-Franklin IBE scheme is secure against both chosen ciphertext attacks and adaptive chosen ciphertext attacks. In both types of attacks there are two players: the challenger and the adversary. Following is a description of each attack as well as a brief discussion on why it is important that any IBE scheme be secure against both a chosen ciphertext attack and an adaptive chosen ciphertext attack.

In a chosen ciphertext attack, the challenger runs an ‘Extract’ query for a fixed identifier  $ID$ . The challenger now holds the fixed private key  $d$  and thus is able to run the ‘Decrypt’ operation on any plaintext. The target plaintext message  $P$ , chosen by the challenger, is run through the ‘Decrypt’ operation and the corresponding ciphertext  $C$  is sent to the adversary. At this point, the adversary sends any plaintext  $P_i \neq P$  to the challenger who returns the corresponding ciphertext  $C_i$ . The adversary is able to ask for as many plaintext-ciphertext pairs they choose as long as  $P_i \neq P$  for any  $i$ . The goal of the adversary is to guess the original plaintext  $P$  from knowledge of the original ciphertext  $C$  and the plaintext-ciphertext pairs  $\langle P_i, C_i \rangle$ . For this attack, there is one fixed identifier for which all of the plaintext-ciphertext pairs are encrypted and decrypted. In the next attack, however, the adversary has to deal with multiple identifiers being used to encrypt and decrypt the plaintext-ciphertext pairs.

In an adaptive chosen ciphertext attack, the challenger chooses an identifier  $ID$  and runs an ‘Extract’ query to get the private key  $d$ . The challenger also must choose some plaintext  $P$  to be encrypted using  $d$ . Once  $P$  and  $ID$  have been chosen, the adversary is given the corresponding  $C$ . The adversary’s goal is to find  $P$ . In order to find  $P$ , the adversary is allowed to run the following two queries as many times as they choose.

1. Extraction query: The adversary can pick an identifier  $ID_j \neq ID$  in which the challenger must return the private key  $d_j$ .
2. Decryption query: The adversary can pick any plaintext  $P_i \neq P$  and any identifier  $ID_j \neq ID$  to encrypt  $P_i$ . The challenger must return the corresponding ciphertext  $C_{ij}$ .

The adversary’s goal is to determine the original plaintext  $P$  corresponding with the original ciphertext  $C$ . Should the adversary choose  $P_i = P$ , the adversary has gained no advantage over guessing, as this is how they came about choosing  $P_i$ . Under the IBE scheme proposed by Boneh and Franklin, an adversary launching an adaptive chosen ciphertext attack has only an arbitrarily small advantage over simply guessing the plain text  $P$  associated with the ciphertext  $C$ .

This is a brief description of the Boneh-Franklin IBE scheme and its security. For more detailed information, one should consult [1].

## 4 Multiple Trust Authorities in IBE Schemes

Chen et al in [2] wanted to find some way of combining keys so that fewer encryptions and/or decryptions could be used to send multiple persons an encrypted message using the Boneh-Franklin IBE scheme. Using current technology, a person wanting to send a message using a public key cryptosystem to three people sends the message three different times, once using each different public key. The authors of [2] came up with the idea of conjoining and disjoining keys

to reduce the number of transmissions needed to send one message to multiple persons holding different keys. They also discovered that this technique could be used for adding contextual information to an identity. Using contextual information in a cryptosystem would be of great use, particularly in the military implementation used throughout this paper. Suppose, for example, we set up our military operation so that Commander Alice working with the Navy in unit  $A$  would have the identifier

*alice.commander.unita.hostileoperation.navy.us.*

Technology Specialist Carol in unit  $B$  working with the Army would have the identifier

*carol.techspecialist.unitb.hostileoperation.army.us.*

Since George is in the navy as a Technology Specialist at the Navy Academy of Sciences, his identifier would be

*george.student.navyacadsci.navy.us.*

All members having gone through basic training and who are still being paid by the military, would have similar identifiers based on their status, skill, location, etc. Harriet would not have an identifier because she never went through the entrance requirements of the military when the identifier would have been assigned.

Note that the only identifier received after finishing basic training is name.position.branch.us. As each individual completes their training and promotions, they receive the appropriate addendums to their names. This would be done using several PKGs (private key generators), one for each addition onto the name. In this case, each individual identifier is conjoined with other identifiers (george and student and navyacadsci and navy and us). In addition, one could use the disjoin explained shortly to send a message to either george.student.navyacadsci.navy.us ‘or’ alice.commander.unita.hostileoperation.navy.us. In this example, there would be a United States PKG, Navy PKG, Army PKG, and so on. Notice that the individual PKGs do not need to trust each other as they are completely separate from the other PKGs. Each of the identifiers may be used to identify who needs to read messages sent out to members of the hostile operations in the field. Conjoining and disjoining keys is done using the key calculus developed in [2], as seen in sections 4.1 through 4.5. Please note, the information contained in the rest of this section is directly related to information found in [2].

#### 4.1 Key Calculus

Chen et al developed what they called a ‘key calculus’ for adding keys together. To begin with, we will establish some notation, then the conjunction and disjunction of keys will be explored. Since multiple PKGs, known as trusted authorities, will be dealt with from this point on, every trust authority in use will have its own public-private key pair

$$R_{TA_i} = s_i P$$

where  $R_{TA_i}$  is the public key for the  $i$ th trust authority and  $s_i$  is the associated private key. The set of trust authorities with public-private key pairs will be denoted  $T$ . In the conference

example, there would only be one trust authority. In the military example, however, there can be an arbitrary number of trust authorities. We will assume there are  $k$  trust authorities, each of which is trusted by all persons using the IBE scheme. This number can change as necessary for the situation at hand. Every user/identifier in the system will also need a public-private key pair.

The private key for the person with identifier  $ID_j$  using the  $i$ th trust authority will now be

$$S_{ID_j,i} = s_i Q_{ID_j}$$

where  $s_i$  is the trust authority's private key,

$$Q_{ID_j} = H_1(ID_j)$$

where the identifier's public key is  $Q_{ID_j}$ . Notice again that the system's security is based upon the hardness of the Bilinear Diffie-Hellman problem. Also note that  $H_1(ID_j)$  is not private.  $H_1$  is public information and  $ID_j$  is public information, so any person can compute the value  $Q_{ID_j}$ . Let us denote the set  $I = \{ID_j\}$  as the set of identifiers which have an identity based public key associated with them.

Because each trust authority can supply any given identifier  $ID_j$  a different private key, every identifier can have as many as  $k$  private keys; one private key for each respective trust authority. For the moment, fix an identifier  $ID$ . Let

$$b = (b_1, b_2, \dots, b_k)$$

be a binary string. We will define

$$S_{ID,b} = \sum_{i=1}^k b_i S_{ID,TA_i}$$

as a *virtual trust authority*. This trust authority is virtual because it is a combination of the trust authorities available in the system - it does not actually exist. The user can choose the string  $b$  arbitrarily and use the result as their virtual trust authority. Notice there are  $2^k$  virtual trust authority options available to each user. Each virtual trust authority will also have a *virtual private key* defined by

$$s_b = \sum_{i=1}^k b_i s_i.$$

This virtual private key is not computable by anyone unless the proper trust authorities collude. Thus, this is useful for preventing key escrow as will be seen shortly.

Now suppose we have a fixed trust authority,  $TA$ , and multiple identifiers  $ID_j$  for  $1 \leq j \leq l$  where  $l$  is the number of total identifiers. Our trust authority has the public-private key pair

$$R_{TA} = sP$$

where  $R_{TA}$  is the public key and  $s$  is the private key. The public keys for each of the identifiers will be

$$Q_{ID_j} = H_1(ID_j)$$

while the private key will be

$$S_{ID_j} = sQ_{ID_j}.$$

Let us choose an arbitrary  $l$ -length binary string  $b'$ . We can now create a *virtual identifier*  $Q_{ID,b'}$  with private key

$$S_{ID,b'} = \sum_{j=1}^l b'_j S_{ID_j}$$

and public key

$$Q_{ID,b'} = \sum_{j=1}^l b'_j Q_{ID_j}.$$

When using virtual trust authorities, key escrow is difficult for the authorities to achieve. By using any one arbitrary virtual trust authority (there are  $2^k$  possibilities), the collusion of the exact trust authorities being added together is needed to determine the identifier's private key. Hence, key escrow should be of little concern when using multiple trust authorities. An important benefit of using virtual identifiers was mentioned briefly during the setup of the military example. Each trust authority can be used to create a context for every person in the military. The navy can add the .navy extension while the army can add the .army extension to the associated personnel by assigning them a private key. While in the field, each person would then have a virtual identifier combining all of the trust authorities of which they have been authorized. This is necessary in the military example because it will be used as part of the authentication mechanism for retrieving messages.

To provide an explicit example of the contextual identity, let Unit  $A$  be a trust authority with

$$R_{unita} = rP.$$

All persons with the name Alice in unit  $A$  will receive the same public-private key pair

$$[H_1(\text{Alice}), S_{alice,unita}]$$

where

$$S_{alice,unita} = rQ_{alice} = rH_1(\text{alice}).$$

There are also several other trust authorities that trust Alice, namely Commander, Navy, Hostile Operation, and United States where

$$R_{commander} = sP, R_{navy} = tP, R_{hostileoperation} = uP \text{ and } R_{us} = vP$$

respectively. The virtual trust authority for Alice would now be

$$\begin{aligned} R_{virtual} &= R_{commander} + R_{unita} + R_{hostileoperation} + R_{navy} + R_{us} \\ &= sP + rP + uP + tP + vP \end{aligned}$$

and the corresponding secret key would be  $s + r + u + t + v$ . No existing party knows this key because it is the combination of several trust authorities. However, a message can be sent to a person named Alice who is a Commander in the Navy working with the hostile operation's unit A for the United States using the public key  $Q_{alice,\{commander,unita,hostileoperation,navy,us\}}$ . Alice will be able to decrypt the message intended for her because she has the virtual private key

$$S_{alice,commander} + S_{alice,unita} + S_{alice,hostileoperation} + S_{alice,navy} + S_{us}.$$

Now we will look at the conjunction and disjunction of identifier based keys.

## 4.2 Naïve Conjunction of Keys

To discuss either the conjunction or disjunction of keys, we need to first define an admissible set of keys. A set  $\mathbb{S}$  is *admissible* if satisfies one of the following properties:

1.  $\mathbb{S} = \{(ID_i, TA_i)\}$  for  $1 \leq i \leq k$ ,  $ID_i \in I$ ,  $TA_i \in T$ .
2.  $\mathbb{S} = \mathbb{S}_1 \otimes \mathbb{S}_2$  where  $\mathbb{S}_1 \otimes \mathbb{S}_2 = \{(ID_i, TA_j) : (ID_i, TA) \in \mathbb{S}_1 \cup \mathbb{S}_2, (ID, TA_j) \in \mathbb{S}_1 \cup \mathbb{S}_2\}$ . In other words, every combination of identifier and trust authority is included in the set  $\mathbb{S}_1 \otimes \mathbb{S}_2$ .

We can define the set of identifiers and trust authorities in the set  $\mathbb{S}$  as  $ID(\mathbb{S})$  and  $TA(\mathbb{S})$  respectively. Notice that the virtual trust authority corresponding to  $\mathbb{S}$  will be

$$R_{\mathbb{S}} = \sum_{TA_i \in TA(\mathbb{S})} R_{TA_i}$$

and the virtual identity will be given by

$$Q_{\mathbb{S}} = \sum_{ID_j \in ID(\mathbb{S})} Q_{ID_j}.$$

Any message encrypted using the public keys  $R_{\mathbb{S}}$  and  $Q_{\mathbb{S}}$  may only be decrypted by persons holding all of the private keys in the set  $\mathbb{S}$ . In other words, the decryptor must know

$$S_{\mathbb{S}} = \sum_{ID_j \in ID(\mathbb{S})} \sum_{TA_i \in TA(\mathbb{S})} S_{ID_j, TA_i}.$$

This may better be illustrated using an example.

### 4.2.1 Example

Suppose in our conference example we had an additional set of trust authorities involved. For example, let each field of study being represented at the conference have its own trust authority. We might have the set of trust authorities

$$T = \{IBEschemes, hashfunctions, algebra, \dots\}.$$

Now we can send a message that is meant for Alice who studies IBE schemes and Bob who studies algebra using one key. In this case, we have

$$\mathbb{S}_1 = \{(alice, IBESchemes)\} \text{ and } \mathbb{S}_2 = \{(bob, algebra)\}$$

so it then follows that

$$\mathbb{S} = \mathbb{S}_1 \otimes \mathbb{S}_2 = \{(alice, IBESchemes), (alice, algebra), (bob, IBESchemes), (bob, algebra)\}.$$

The virtual trust authority and the virtual identifier would be

$$R = R_{IBEScheme} + R_{algebra} \text{ and } Q = Q_{alice} + Q_{bob}$$

respectively. In order to decrypt the ciphertext  $C = \langle U, V \rangle = \langle rP, M \oplus H_2(g_{ID}^r) \rangle$  to the plaintext message  $M$ , one would need to compute

$$\begin{aligned} V = M \oplus H_2(\hat{e}(R, rQ)) &= M \oplus H_2 \left[ \prod_{TA_i=IBEScheme, algebra} \prod_{ID_j=alice, bob} \hat{e}(R_{TA_i}, rQ_{ID_j}) \right] \\ &= M \oplus H_2 \left[ \hat{e}(R_{IBEScheme}, rQ_{alice}) * \hat{e}(R_{algebra}, rQ_{alice}) \right. \\ &\quad \left. * \hat{e}(R_{IBEScheme}, rQ_{bob}) * \hat{e}(R_{algebra}, rQ_{bob}) \right] \\ &= M \oplus H_2 \left[ \hat{e}(s_{IBEScheme}P, rQ_{alice}) * \hat{e}(s_{algebra}P, rQ_{alice}) \right. \\ &\quad \left. * \hat{e}(s_{IBEScheme}P, rQ_{bob}) * \hat{e}(s_{algebra}P, rQ_{bob}) \right] \\ &= M \oplus H_2 \left[ \hat{e}(rP, s_{IBEScheme}Q_{alice}) * \hat{e}(rP, s_{algebra}Q_{alice}) \right. \\ &\quad \left. * \hat{e}(rP, s_{IBEScheme}Q_{bob}) * \hat{e}(rP, s_{algebra}Q_{bob}) \right] \\ &= M \oplus H_2 \left[ \hat{e}(U, S_{alice, IBEScheme}) * \hat{e}(U, S_{alice, algebra}) \right. \\ &\quad \left. * \hat{e}(U, S_{bob, IBEScheme}) * \hat{e}(U, S_{bob, algebra}) \right] \\ &= M \oplus H_2 \left[ \hat{e}(U, \sum_{TA_i=TA_{IBEScheme}, TA_{algebra}} \sum_{ID_j=alice, bob} S_{ID_j, TA_i}) \right]. \end{aligned}$$

This shows that we need the secret key

$$\sum_{TA_i=TA_{IBEScheme}, TA_{algebra}} \sum_{ID_j=alice, bob} S_{ID_j, TA_i}$$

which is exactly the sum of all four secret keys over the set  $\mathbb{S}$ . Hence, all four keys would be needed in order to decrypt the message  $(U, V)$ . There are two main problems with using naïve conjunction, however. First, not one single person should have all of this information because

it combines four separate identities. Thus, decrypting the message requires the collusion of several individuals. Second, there might be two keys which do not actually exist. Using the previous example, who is to say that there indeed exists a person Alice who studies algebra or a person Bob who studies IBE schemes? Even so, Alice studying IBE schemes and Bob studying algebra would need Alice studying algebra and Bob studying IBE schemes to cooperate in order to read the message sent using the naïve conjunction between identifiers (*Alice, Bob*) and trust authorities (*IBEschemes, algebra*). In this case, both Alices and both Bobs would have to share their keys in order to read the message  $M$  encrypted to  $(U, V)$ . While this is clearly not the desired outcome, we have achieved one of our goals: the number of transmissions needed to contact four identities with four separate keys has been decreased to one transmission. A more generalized form of this naïve conjunction can be performed which is less restrictive.

### 4.3 General Conjunction of Keys

The general conjunction of keys is less restrictive because it works as one would expect conjunction to work. If a person sends a message to Alice studying IBE schemes and Bob studying algebra, it would be reasonable to expect that no other persons need be involved in the decryption of the message. To encrypt a message  $M$  using the conjunction of an arbitrary number of admissible sets  $k$ , one computes

$$V = M \oplus H_2\left(\prod_{i=1}^k \hat{e}(R_{TA_i}, rQ_{ID_i})\right).$$

Recall  $U = rP$ , so by sending  $(U, V)$ , decryption can be performed by evaluating

$$M = V \oplus H_2\left(\hat{e}\left(U, \sum_{i=1}^k S_{ID_i, TA_i}\right)\right).$$

The addition of  $k$  private keys is still needed, but it is now only needed by the persons within the admissible sets  $S_i$ .

Relating directly back to the example above, we see that only Alice studying IBE schemes and Bob studying algebra need to combine their private keys to decrypt the message  $M$  because

$$\begin{aligned} V &= M \oplus H_2\left(\prod_{i=\{alice, bob\}} \hat{e}(R_{TA_i}, rQ_{ID_i})\right) \\ &= M \oplus H_2\left(\hat{e}(R_{TA_{alice}}, rQ_{ID_{alice}}) * (\hat{e}(R_{TA_{bob}}, rQ_{ID_{bob}}))\right) \\ &= M \oplus H_2\left(\hat{e}(s_{IBEschemes}P, rQ_{ID_{alice}}) * (\hat{e}(s_{algebra}P, rQ_{ID_{bob}}))\right) \\ &= M \oplus H_2\left(\hat{e}(rP, s_{IBEschemes}Q_{ID_{alice}}) * (\hat{e}(rP, s_{algebra}Q_{ID_{bob}}))\right) \\ &= M \oplus H_2\left(\hat{e}(U, S_{alice}) * (\hat{e}(U, S_{bob}))\right) \\ &= M \oplus H_2\left(\hat{e}(U, S_{alice} + S_{bob})\right). \end{aligned}$$

Conjunction, while not useful in a general sense, could prove useful when dealing with a system which needs key revocation or contextual information. The use and benefits of contextual

information was discussed previously in the introduction of this section. Key revocation, on the other hand, can be achieved by adding an additional trust authority, i.e. conjoining a trusted time authority. To illustrate the functionality of the time authority in key revocation let us go back to the military implementation discussed in sections 2.4 and 4.1.

Suppose we let the time authority issue keys every hour on the hour. In order for messages to be sent to Alice, a Commander in unit *A* working with the U.S. Navy during the Hostile Operation, a time identifier must be conjoined to her identity. All messages sent over the network will be required to have a time stamp associated with them. Thus, we might have a message sent to the identifier

*alice.commander.unita.hostileoperation.navy.us.010120041400*

which will require Alice to hold a private key for January 01, 2004 at 1400 hours in order to read the message. Messages can be encrypted using the time context before the time authority distributes associated private keys. The key will be received on January 01, 2004 at 1400 hours, thus determining when Alice can read messages sent to her using the 1400 hour time encryption key. Using the time authority will also help make sure that Alice has been authenticated with the time trust authority within a certain time of reading the message sent with time key 1400 hours. In order for Alice to obtain the time authority's administered private key, Alice will need to check-in with the time authority and download the key. However, we are going to require that the time authority maintain a running real-time list of compromised/revoked keys. It will then be the time authority's responsibility to make sure that the appropriate time keys are not distributed to un-trusted personnel.

In the military implementation example, suppose that Alice will be promoted to Captain on April 4, 2004 at 0000 hours. Clearly the Commander trust authority will no longer trust Alice after this time because she will no longer be a Commander. However, since Alice will be a Captain, she will obtain a public-private key pair for the identity Alice with trust authority Captain. Thus, the time authority will send the time authority keys not to

*alice.commander.unita.hostileoperation.navy.us*

but instead will send them to

*alice.captain.unita.hostileoperation.navy.us*

so that Captain Alice has a new identifier

*alice.captain.unita.hostileoperation.navy.us.040420040000.*

Because Alice will never receive the private key generated by the time authority for the Commander identity, she will only be able to check messages encrypted up to the time her promotion is official. In other words, Alice will only be able to decrypt messages using time keys for the Commander identifier up to and not including the 040420040000 time key. It will be the responsibility of the trust authorities to decide when they expire/revoke the keys.

What is one of the major problems with this scheme? If the scheme is implemented in a network that is not an ad hoc network, it is easy to see that there is a single point of failure: the trusted time authority. Without the trusted time authority, none of the users will be able

to access the time associated private keys, and thus will not be able to read their messages. The enemy could easily target the time authority for denial of service attacks or physical destruction attacks. For this reason, it is a requirement that any of the trust authorities be able to act as the time trust authority. Because all of the trusted authorities are assumed to be trusted to begin with, we will allow any of them to act as the time trust authority whenever necessary. This duty is able to change from one trust authority to another as needed because all of the users will have public identification information for all of the trust authorities, and hence for the time trust authority being run from any of the trusted authority locations.

If there is a node which serves as a single point of failure at any one time (when this node goes down, it causes the entire system to go down), there is a node that is more important than all others on the network, i.e. we do not have an ad hoc network. We will assume that we only use this time trust authority at predefined intervals, and thus, the trust authority is not actually a node on the ad hoc network. We propose that all nodes checking in with the time trust authority will leave the ad hoc network, check in with the time trust authority, then return to the network. This same leave-return process can also be used when communication is needed between any of the nodes and the other trust authorities.

Because we have seen that conjunction is not realistic for making messages available to more than one person using one encryption (since collaboration between users is needed), we will consider the idea of disjoining keys together to allow for multiple persons to view a message sent via one single encryption.

#### 4.4 Naïve Disjunction of Keys

Suppose we want to send a message to Alice studying IBE schemes and Bob studying algebra. What we really want is for Alice *or* Bob to be able to read the message. Following are two examples of naïve disjunction, the first assuming that Alice and Bob are only using one trust authority, while the second illustrates two trust authorities.

##### 4.4.1 Example: One Trust Authority, Two Identifiers

Suppose we have only one trust authority having the identity,  $TA$ . Alice and Bob will be the two people Dave is trying to send message  $M$  to. The key disjunction can be set up as follows.

We let the trust authority have the public key

$$R_{TA} = sP$$

where  $s$  is the trust authority's private key. We let the two identifiers be

$$Q_{alice} = H_1(alice) \text{ and } Q_{bob} = H_1(bob).$$

with the associated private keys as before,

$$S_{alice,TA} = sQ_{alice} \text{ and } S_{bob,TA} = sQ_{bob}.$$

We can generate a virtual public key and an auxiliary key for 'Alice or Bob' by

$$Q_{virtual} = Q_{alice} + x_1Q_{bob} \text{ and } Q_{aux} = Q_{alice} + x_2Q_{bob}$$

where  $x_1, x_2 \in \mathbb{Z}_q^*$ . Notice that

$$\begin{aligned}
Q_{virtual} &= Q_{alice} + x_1 \left[ \frac{1}{x_2} (Q_{aux} - Q_{alice}) \right] = [Q_{aux} - x_2 Q_{bob}] + x_1 Q_{bob} \\
&= Q_{alice} + \frac{x_1}{x_2} Q_{aux} - \frac{x_1}{x_2} Q_{alice} = Q_{aux} + (x_1 - x_2) Q_{bob} \\
&= \frac{x_1}{x_2} Q_{aux} + \left(1 - \frac{x_1}{x_2}\right) Q_{alice} = Q_{aux} + \xi Q_{bob} \\
&= \alpha Q_{aux} + \beta Q_{alice}
\end{aligned}$$

where  $\alpha = \frac{x_1}{x_2}$ ,  $\beta = \frac{x_2 - x_1}{x_2}$  and  $\xi = x_1 - x_2$ . Encryption to Alice *or* Bob can be accomplished by computing

$$R_{virtual} = R_{TA}, U_1 = rP, U_2 = rQ_{aux}, U_3 = ? \text{ and } V = m \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})).$$

In this case,  $U_3$  is anything (denoted ?) because we only have one trust authority. One can see in the next example that when there are more trust authorities involved,  $U_3$  will be the auxiliary key of the combined trust authorities. The cryptogram sent to Alice and Bob will be  $(U_1, U_2, U_3, V)$ .

Alice will be able to decrypt the message by computing

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(R_{TA}, r(\alpha Q_{aux} + \beta Q_{alice})) \\
&= \hat{e}(R_{TA}, r\alpha Q_{aux}) * \hat{e}(R_{TA}, r\beta Q_{alice}) \\
&= \hat{e}(R_{TA}, \alpha U_2) * \hat{e}(sP, r\beta Q_{alice}) \\
&= \hat{e}(R_{TA}, \alpha U_2) * \hat{e}(rP, s\beta Q_{alice}) \\
&= \hat{e}(R_{TA}, \alpha U_2) * \hat{e}(U_1, \beta S_{alice,TA}).
\end{aligned}$$

Alice is able to compute the last value because  $R_{TA}$ ,  $\alpha$  and  $\beta$  are all public information,  $U_1, U_2$  were sent as part of the cryptogram, and  $S_{alice,TA}$  is her private key. Because Alice knows all of this information and since  $V$  is also contained within the cryptogram, she is able to decrypt the message by

$$\begin{aligned}
V &= M \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(R_{TA}, \alpha U_2) * \hat{e}(U_1, \beta S_{alice,TA})).
\end{aligned}$$

Thus, Alice is able to decrypt messages sent to the virtual key for Alice *or* Bob.

Bob will be able to decrypt the cryptogram  $(U_1, U_2, U_3, V)$  by computing

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(R_{TA}, r(Q_{aux} + \xi Q_{bob})) \\
&= \hat{e}(R_{TA}, rQ_{aux}) * \hat{e}(R_{TA}, r\xi Q_{bob}) \\
&= \hat{e}(R_{TA}, U_2) * \hat{e}(sP, r\xi Q_{bob}) \\
&= \hat{e}(R_{TA}, U_2) * \hat{e}(rP, s\xi Q_{bob}) \\
&= \hat{e}(R_{TA}, U_2) * \hat{e}(U_1, \xi S_{bob,TA}).
\end{aligned}$$

Now Bob is able to decipher the message  $M$  because  $R_{TA}$  and  $\xi$  are both public,  $U_1, U_2$  were sent as part of the cryptogram, and  $S_{bob,TA}$  is his private key. Because Bob knows all of this information, he is able to decrypt the cryptogram using the same process as Alice, only substituting his equation in for hers:

$$\begin{aligned}
V &= M \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(R_{TA}, U_2) * \hat{e}(U_1, \xi S_{bob,TA})).
\end{aligned}$$

We have now achieved sending one message to more than one person using only one cryptogram. However, if we use more than one trust authority, the naïve disjunction runs into difficulties because more persons than likely intended are able to decrypt the cryptogram. The following example illustrates this problem.

#### 4.4.2 Example: Two Trust Authorities, Two Identifiers

Now suppose that Alice is studying IBE schemes and Bob is studying algebra. We now have two trust authorities *IBEschemes*, *algebra* and two identifiers *alice*, *bob*. We can set up the naïve disjunction as follows.

Let the trust authorities have their respective public keys

$$R_{IBEschemes} = s_{IBEschemes}P \text{ and } R_{algebra} = s_{algebra}P.$$

Our two identifiers will be created as in the previous example,

$$Q_{alice} = H_1(alice) \text{ and } Q_{bob} = H_1(bob).$$

with the associated private keys

$$S_{alice,IBEschemes} = s_{IBEschemes}Q_{alice} \text{ and } S_{bob,algebra} = s_{algebra}Q_{bob}.$$

We can generate a virtual public key and an auxiliary key for ‘Alice or Bob’ by

$$Q_{virtual} = Q_{alice} + x_1Q_{bob} \text{ and } Q_{aux} = Q_{alice} + x_2Q_{bob}$$

where  $x_1, x_2 \in \mathbb{Z}_q^*$ . We can also generate a virtual public key and auxiliary key for the trust authority ‘IBE scheme or algebra’. These can be given by

$$R_{virtual} = R_{IBEschemes} + y_1R_{algebra} \text{ and } R_{aux} = R_{IBEschemes} + y_2R_{algebra}$$

for  $y_1, y_2 \in \mathbb{Z}_q^*$ . Notice that, as before,

$$Q_{virtual} = \alpha Q_{aux} + \beta Q_{alice} = Q_{aux} + \xi Q_{bob}$$

but also that

$$\begin{aligned} R_{virtual} &= R_{IBEschemes} + y_1 \left[ \frac{1}{y_2} (R_{aux} - R_{IBEschemes}) \right] = [R_{aux} - y_2 R_{algebra}] + y_1 R_{algebra} \\ &= R_{IBEschemes} + \frac{y_1}{y_2} R_{aux} - \frac{y_1}{y_2} R_{IBEschemes} = R_{aux} + (y_1 - y_2) R_{algebra} \\ &= \frac{y_1}{y_2} R_{aux} + \left(1 - \frac{y_1}{y_2}\right) R_{IBEschemes} = R_{aux} + \delta R_{algebra} \\ &= \lambda R_{aux} + \gamma R_{IBEschemes} \end{aligned}$$

where  $\lambda = \frac{y_1}{y_2}$ ,  $\gamma = \frac{y_2 - y_1}{y_2}$  and  $\delta = y_1 - y_2$ .

Encryption to ‘Alice or Bob’ can now be accomplished by computing

$$U_1 = rP, U_2 = rQ_{aux}, U_3 = rR_{aux} \text{ and } V = M \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})).$$

The cryptogram sent to Alice and Bob will be  $(U_1, U_2, U_3, V)$ . Alice studying IBE schemes will be able to decrypt the message by computing

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r(\alpha Q_{aux} + \beta Q_{alice})) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r\alpha Q_{aux}) \\
&\quad * \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r\beta Q_{alice}) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r\alpha Q_{aux}) \\
&\quad * \hat{e}(\lambda R_{aux}, r\beta Q_{alice}) * \hat{e}(\gamma R_{IBEschemes}, r\beta Q_{alice}) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r\alpha Q_{aux}) \\
&\quad * \hat{e}(r\lambda R_{aux}, \beta Q_{alice}) * \hat{e}(r\gamma R_{IBEschemes}, \beta Q_{alice}) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, \alpha U_2) * \hat{e}(\lambda U_3, \beta Q_{alice}) \\
&\quad * \hat{e}(r\gamma s_{IBEschemes} P, \beta Q_{alice}) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, \alpha U_2) * \hat{e}(\lambda U_3, \beta Q_{alice}) \\
&\quad * \hat{e}(r\gamma P, \beta s_{IBEschemes} Q_{alice}) \\
&= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, \alpha U_2) * \hat{e}(\lambda U_3, \beta Q_{alice}) \\
&\quad * \hat{e}(\gamma U_1, \beta S_{alice, IBEschemes}).
\end{aligned}$$

Alice is able to compute the last value because  $R_{aux}$ ,  $R_{IBEschemes}$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ , and  $\lambda$  are all public information,  $U_1, U_2$  and  $U_3$  were sent as part of the cryptogram, and  $S_{alice, IBEschemes}$  is her private key. Because Alice knows all of this information and since  $V$  is also contained within the cryptogram, she is able to decrypt the message by

$$\begin{aligned}
V &= M \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\
M &= V \oplus H_2(\hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, \alpha U_2) * \hat{e}(\lambda U_3, \beta Q_{alice}) * \hat{e}(\gamma U_1, \beta S_{alice, IBEschemes})).
\end{aligned}$$

Thus, Alice studying IBE schemes is able to decrypt messages sent to the virtual key  $Q_{virtual}$  for Alice or Bob.

Bob studying algebra will be able to decrypt the message by computing

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(R_{aux} + \delta R_{algebra}, r(Q_{aux} + \xi Q_{bob})) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, rQ_{aux}) * \hat{e}(R_{aux} + \delta R_{algebra}, r\xi Q_{bob}) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, rQ_{aux}) * \hat{e}(R_{aux}, r\xi Q_{bob}) * \hat{e}(\delta R_{algebra}, r\xi Q_{bob}) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, rQ_{aux}) * \hat{e}(R_{aux}, r\xi Q_{bob}) * \hat{e}(\delta R_{algebra}, r\xi Q_{bob}) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, U_2) * \hat{e}(rR_{aux}, \xi Q_{bob}) * \hat{e}(\delta s_{algebra} P, r\xi Q_{bob}) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, U_2) * \hat{e}(U_3, \xi Q_{bob}) * \hat{e}(\delta rP, s_{algebra} \xi Q_{bob}) \\
&= \hat{e}(R_{aux} + \delta R_{algebra}, U_2) * \hat{e}(U_3, \xi Q_{bob}) * \hat{e}(\delta U_1, \xi S_{bob, algebra})
\end{aligned}$$

Bob studying algebra is able to decrypt message  $M$  because  $R_{algebra}$ ,  $R_{aux}$ ,  $\delta$  and  $\xi$  are all public,  $U_1, U_2$  and  $U_3$  were sent as part of the cryptogram, and  $S_{bob, algebra}$  is his private key. Because Bob knows all of this information, he is able to decrypt the cryptogram using the same process

as Alice, only substituting in his equation for hers:

$$\begin{aligned} V &= M \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\ M &= V \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual})) \\ M &= V \oplus H_2(\hat{e}(R_{aux} + \delta R_{algebra}, U_2) * \hat{e}(U_3, \xi Q_{bob}) * \hat{e}(\delta U_1, \xi S_{bob, algebra})). \end{aligned}$$

Notice, however, that Alice studying algebra and Bob studying IBE schemes (neither which is an intended recipient) are also able to decrypt and read the message  $M$  contained within  $(U_1, U_2, U_3, V)$ . This can be seen by

$$\begin{aligned} \hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(R_{aux} + \delta R_{algebra}, r(\alpha Q_{aux} + \beta Q_{alice})) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, r\alpha Q_{aux}) * \hat{e}(R_{aux} + \delta R_{algebra}, r\beta Q_{alice}) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, r\alpha Q_{aux}) * \hat{e}(R_{aux}, r\beta Q_{alice}) \\ &\quad * \hat{e}(\delta R_{algebra}, r\beta Q_{alice}) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, \alpha U_2) * \hat{e}(rR_{aux}, \beta Q_{alice}) \\ &\quad * \hat{e}(\delta R_{algebra}, r\beta Q_{alice}) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, \alpha U_2) * \hat{e}(U_3, \beta Q_{alice}) * \hat{e}(\delta s_{algebra} P, r\beta Q_{alice}) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, \alpha U_2) * \hat{e}(U_3, \beta Q_{alice}) * \hat{e}(\delta r P, \beta s_{algebra} Q_{alice}) \\ &= \hat{e}(R_{aux} + \delta R_{algebra}, \alpha U_2) * \hat{e}(U_3, \beta Q_{alice}) * \hat{e}(\delta U_1, \beta S_{alice, algebra}) \end{aligned}$$

and

$$\begin{aligned} \hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r(Q_{aux} + \xi Q_{bob})) \\ &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, rQ_{aux}) \\ &\quad * \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, r\xi Q_{bob}) \\ &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, rQ_{aux}) * \hat{e}(\lambda R_{aux}, r\xi Q_{bob}) \\ &\quad * \hat{e}(\gamma R_{IBEschemes}, r\xi Q_{bob}) \\ &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, U_2) * \hat{e}(\lambda r R_{aux}, \xi Q_{bob}) \\ &\quad * \hat{e}(\gamma s_{IBEschemes} P, r\xi Q_{bob}) \\ &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, U_2) * \hat{e}(\lambda U_3, \xi Q_{bob}) \\ &\quad * \hat{e}(\gamma r P, \xi s_{IBEschemes} Q_{bob}) \\ &= \hat{e}(\lambda R_{aux} + \gamma R_{IBEschemes}, U_2) * \hat{e}(\lambda U_3, \xi Q_{bob}) \\ &\quad * \hat{e}(\gamma U_1, \xi S_{bob, IBEschemes}). \end{aligned}$$

The reason for this is that we have two additional private keys

$$S_{alice, algebra} = s_{algebra} Q_{alice} \text{ and } S_{bob, IBEschemes} = s_{IBEschemes} Q_{bob}$$

because we are building this disjunction from the two admissible sets

$$S_1 = \{alice, IBEschemes\} \text{ and } S_2 = \{bob, algebra\}.$$

Thus, we have the set

$$S = S_1 \otimes S_2 = \{\{alice, IBEschemes\}, \{alice, algebra\}, \{bob, IBEschemes\}, \{bob, algebra\}\}$$

of which to perform the operations on. As in example 4.2.1, these two additional people may or may not exist. If they do, we do not necessarily want them reading the message  $M$ . In cases such as the military implementation example, which is too large to implement here, it would be crucial that only the intended recipients be able to read the message  $M$ . This brings us to the general disjunction of keys.

## 4.5 General Disjunction of Keys

The greatest difference between the naïve disjunction and the general disjunction is the development of the virtual and auxiliary keys. In the naïve disjunction,  $R_{virtual}$  and  $R_{aux}$  were made up of only trust authorities.  $Q_{virtual}$  and  $Q_{aux}$  were made up of only identifiers. In the case of the general disjunction, one identifier-trust authority pair will make up each key. First, let us return to example 4.4.1 to see how the general disjunction will work with one trust authority and two identifiers.

### 4.5.1 Example: One Trust Authority, Two Identifiers

In order to send a message to ‘Alice or Bob’ using one trust authority, we will create the following keys:

$$\begin{aligned} R_{virtual} &= Q_{bob} + x_1 R_{TA} & Q_{virtual} &= Q_{alice} + y_1 R_{TA} \\ R_{aux} &= Q_{bob} + x_2 R_{TA} & Q_{aux} &= Q_{alice} + y_2 R_{TA} \end{aligned}$$

where  $x_1, x_2, y_1, y_2 \in \mathbb{Z}_q^*$  are publicly known. From the above keys, one can compute

$$\begin{aligned} R_{virtual} &= (R_{aux} - x_2 R_{TA}) + x_1 R_{TA} &= Q_{bob} + x_1 \left[ \frac{1}{x_2} (R_{aux} - Q_{bob}) \right] \\ &= R_{aux} + (x_1 - x_2) R_{TA} &= Q_{bob} + \frac{x_1}{x_2} R_{aux} - \frac{x_1}{x_2} Q_{bob} \\ &= R_{aux} + \alpha R_{TA} &= \frac{x_1}{x_2} R_{aux} + \left(1 - \frac{x_1}{x_2}\right) Q_{bob} \\ & &= \gamma R_{aux} + \beta Q_{bob} \\ \\ Q_{virtual} &= [Q_{aux} - y_2 R_{TA}] + y_1 R_{TA} &= Q_{alice} + y_1 \left[ \frac{1}{y_2} (Q_{aux} - Q_{alice}) \right] \\ &= Q_{aux} + (y_1 - y_2) R_{TA} &= Q_{alice} + \frac{y_1}{y_2} Q_{aux} - \frac{y_1}{y_2} Q_{alice} \\ &= Q_{aux} + \xi R_{TA} &= \frac{y_1}{y_2} Q_{aux} + \left(1 - \frac{y_1}{y_2}\right) Q_{alice} \\ & &= \epsilon Q_{aux} + \delta Q_{alice} \end{aligned}$$

where  $\alpha = x_1 - x_2$ ,  $\beta = \frac{x_2 - x_1}{x_2}$ ,  $\gamma = \frac{x_1}{x_2}$ ,  $\epsilon = \frac{y_1}{y_2}$ ,  $\delta = \frac{y_2 - y_1}{y_2}$  and  $\xi = y_1 - y_2$ .

Once again, we let

$$U_1 = rP, U_2 = rQ_{aux}, U_3 = rR_{aux} \text{ and } V = m \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual}))$$

for some random  $r \in \mathbb{Z}_q^*$  chosen by the sender. To decrypt the cryptogram  $(U_1, U_2, U_3, V)$  sent by the sender, Alice studying IBE schemes will compute

$$\begin{aligned} \hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((R_{aux} + \alpha R_{TA}), r(\epsilon Q_{aux} + \delta Q_{alice})) \\ &= \hat{e}(R_{aux}, r\epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{TA}, r\epsilon Q_{aux}) * \hat{e}(\alpha R_{TA}, r\delta Q_{alice}) \\ &= \hat{e}(R_{aux}, r\epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{TA}, \epsilon r Q_{aux}) * \hat{e}(\alpha sP, r\delta Q_{alice}) \\ &= \hat{e}(rR_{aux}, \epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{TA}, \epsilon U_2) * \hat{e}(\alpha rP, s\delta Q_{alice}) \\ &= \hat{e}(U_3, Q_{virtual}) * \hat{e}(\alpha R_{TA}, \epsilon U_2) * \hat{e}(\alpha U_1, \delta S_{alice, TA}) \end{aligned}$$

and Bob will compute

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((\gamma R_{aux} + \beta Q_{bob}), r(Q_{aux} + \xi R_{TA})) \\
&= \hat{e}(\gamma R_{aux}, rQ_{aux} + r\xi R_{TA}) * \hat{e}(\beta Q_{bob}, rQ_{aux}) * \hat{e}(\beta Q_{bob}, r\xi R_{TA}) \\
&= \hat{e}(\gamma R_{aux}, rQ_{aux} + r\xi R_{TA}) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta Q_{bob}, r\xi sP) \\
&= \hat{e}(\gamma R_{aux}, r(Q_{aux} + \xi R_{TA})) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(s\beta Q_{bob}, \xi rP) \\
&= \hat{e}(\gamma r R_{aux}, Q_{aux} + \xi R_{TA}) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta s Q_{bob}, \xi U_1) \\
&= \hat{e}(\gamma U_3, Q_{virtual}) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta S_{bob,TA}, \xi U_1).
\end{aligned}$$

Alice and Bob can then compute

$$M = V \oplus H_2(\hat{e}(R_{virtual}, rQ_{virtual}))$$

as before to find the message  $M$  since  $\alpha, \epsilon, \delta, \gamma, \beta, \xi, Q_{virtual}, Q_{alice}$  and  $Q_{bob}$  are all public information,  $U_1, U_2$ , and  $U_3$  are sent with the cryptogram, and  $S_{bob,TA}$  is Bob's private key and  $S_{alice,TA}$  is Alice's private key.

Once again, we have achieved sending one message to two people using only one key. The benefits of using the general disjunction over the naïve disjunction, however, do not readily appear in this example as we are only using one trust authority. Recall that we did not have any problems using one trust authority with the naïve disjunction because

$$\mathbb{S} = \mathbb{S}_1 \otimes \mathbb{S}_2 = \{\{alice, TA\}, \{bob, TA\}\}$$

where  $\mathbb{S}_1 = \{alice, TA\}$  and  $\mathbb{S}_2 = \{bob, TA\}$ . There are no new key pairs created by implementing the  $\otimes$  operation with one trust authority. However, in example 4.5.2 there was the problem of having undesirable key pairs created using the  $\otimes$  operation with multiple trust authorities. By implementing the general disjunction with two trust authorities and two identifiers, we can see the benefit that the general disjunction has over the naïve disjunction.

#### 4.5.2 Example: Two Trust Authorities, Two Identifiers

We will use the same setup as was given in example 4.5.2 for the trust authorities and identifiers. Again, we will have the set

$$\mathbb{S} = \mathbb{S}_1 \otimes \mathbb{S}_2 = \{\{alice, IBESchemes\}, \{alice, algebra\}, \{bob, IBESchemes\}, \{bob, algebra\}\}.$$

We will now generate the virtual and auxiliary keys as follows:

$$\begin{aligned}
R_{virtual} &= Q_{bob} + x_1 R_{IBESchemes} & Q_{virtual} &= Q_{alice} + y_1 R_{algebra} \\
R_{aux} &= Q_{bob} + x_2 R_{IBESchemes} & Q_{aux} &= Q_{alice} + y_2 R_{algebra}
\end{aligned}$$

where  $x_1, x_2, y_1, y_2 \in \mathbb{Z}_q^*$  are publicly known. It can be seen that

$$\begin{aligned}
R_{virtual} &= (R_{aux} - x_2 R_{IBESchemes}) + x_1 R_{IBESchemes} &= Q_{bob} + x_1 \left[ \frac{1}{x_2} (R_{aux} - Q_{bob}) \right] \\
&= R_{aux} + (x_1 - x_2) R_{IBESchemes} &= Q_{bob} + \frac{x_1}{x_2} R_{aux} - \frac{x_1}{x_2} Q_{bob} \\
&= R_{aux} + \alpha R_{IBESchemes} &= \frac{x_1}{x_2} R_{aux} + \left(1 - \frac{x_1}{x_2}\right) Q_{bob} \\
& &= \gamma R_{aux} + \beta Q_{bob}
\end{aligned}$$

and

$$\begin{aligned}
Q_{virtual} &= [Q_{aux} - y_2 R_{algebra}] + y_1 R_{algebra} &= Q_{alice} + y_1 \left[ \frac{1}{y_2} (Q_{aux} - Q_{alice}) \right] \\
&= Q_{aux} + (y_1 - y_2) R_{algebra} &= Q_{alice} + \frac{y_1}{y_2} Q_{aux} - \frac{y_1}{y_2} Q_{alice} \\
&= Q_{aux} + \xi R_{algebra} &= \frac{y_1}{y_2} Q_{aux} + \left(1 - \frac{y_1}{y_2}\right) Q_{alice} \\
& &= \epsilon Q_{aux} + \delta Q_{alice}
\end{aligned}$$

where  $\alpha = x_1 - x_2$ ,  $\beta = \frac{x_2 - x_1}{x_2}$ ,  $\gamma = \frac{x_1}{x_2}$ ,  $\epsilon = \frac{y_1}{y_2}$ ,  $\delta = \frac{y_2 - y_1}{y_2}$  and  $\xi = y_1 - y_2$ .

As before, the sender will choose an arbitrary value  $r \in \mathbb{Z}_q^*$  and will develop the cryptogram  $(U_1, U_2, U_3, V)$ . The cryptogram can now be evaluated by Alice studying IBE schemes because

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((R_{aux} + \alpha R_{IBEschemes}), r(\epsilon Q_{aux} + \delta Q_{alice})) \\
&= \hat{e}(R_{aux}, r\epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{IBEschemes}, r\epsilon Q_{aux}) \\
&\quad * \hat{e}(\alpha R_{IBEschemes}, r\delta Q_{alice}) \\
&= \hat{e}(R_{aux}, r\epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{IBEschemes}, \epsilon r Q_{aux}) \\
&\quad * \hat{e}(\alpha s_{IBEschemes} P, r\delta Q_{alice}) \\
&= \hat{e}(rR_{aux}, \epsilon Q_{aux} + r\delta Q_{alice}) * \hat{e}(\alpha R_{IBEschemes}, \epsilon U_2) \\
&\quad * \hat{e}(\alpha r P, s_{IBEschemes} \delta Q_{alice}) \\
&= \hat{e}(U_3, Q_{virtual}) * \hat{e}(\alpha R_{IBEschemes}, \epsilon U_2) * \hat{e}(\alpha U_1, \delta S_{alice, IBEschemes}).
\end{aligned}$$

Notice all the information needed is public information, sent with the cryptogram, or deals with Alice's private information. Bob studying algebra can also decrypt the cryptogram because

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((\gamma R_{aux} + \beta Q_{bob}), r(Q_{aux} + \xi R_{algebra})) \\
&= \hat{e}(\gamma R_{aux}, r(Q_{aux} + \xi R_{algebra})) * \hat{e}(\beta Q_{bob}, rQ_{aux}) * \hat{e}(\beta Q_{bob}, r\xi R_{algebra}) \\
&= \hat{e}(\gamma R_{aux}, r(Q_{aux} + \xi R_{algebra})) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta Q_{bob}, \xi s_{algebra} r P) \\
&= \hat{e}(\gamma r R_{aux}, Q_{aux} + \xi R_{algebra}) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta s_{algebra} Q_{bob}, \xi U_1) \\
&= \hat{e}(\gamma U_3, Q_{virtual}) * \hat{e}(\beta Q_{bob}, U_2) * \hat{e}(\beta S_{bob, algebra}, \xi U_1)
\end{aligned}$$

again contains only public information, information sent within the cryptogram, or Bob's private information.

Now Alice studying IBE schemes and Bob studying algebra can compute the message  $M$  as in example 4.5.1. The question that we are most interested in, however, is whether Alice studying algebra or Bob studying IBE schemes will be able to decrypt the cryptogram. The answer is no. Alice studying algebra or Bob studying IBE schemes would need to use one of the other combinations of  $R_{virtual}$  and  $Q_{virtual}$ . One of the remaining combinations gives

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((\alpha R_{IBEschemes} + R_{aux}), (\xi R_{algebra} + Q_{aux})) \\
&= \hat{e}(\alpha R_{IBEschemes}, r\xi R_{algebra}) * \hat{e}(\alpha R_{IBEschemes}, rQ_{aux}) \\
&\quad * \hat{e}(R_{aux}, r\xi R_{algebra}) * \hat{e}(R_{aux}, rQ_{aux}) \\
&= \hat{e}(\alpha R_{IBEschemes}, r\xi R_{algebra}) * \hat{e}(\alpha R_{IBEschemes}, U_2) \\
&\quad * \hat{e}(R_{aux}, r(\xi R_{algebra} + Q_{aux})) \\
&= \hat{e}(\alpha R_{IBEschemes}, r\xi R_{algebra}) * \hat{e}(\alpha R_{IBEschemes}, U_2) \\
&\quad * \hat{e}(rR_{aux}, \xi R_{algebra} + Q_{aux}) \\
&= \hat{e}(\alpha R_{IBEschemes}, r\xi R_{algebra}) * \hat{e}(\alpha R_{IBEschemes}, U_2) * \hat{e}(U_3, Q_{virtual})
\end{aligned}$$

in which Alice studying algebra or Bob studying IBE schemes would need to know  $rR_{algebra}$  or  $rR_{IBEschemes}$ . The other remaining combination gives

$$\begin{aligned}
\hat{e}(R_{virtual}, rQ_{virtual}) &= \hat{e}((\beta Q_{bob} + \gamma R_{aux}), r(\delta Q_{alice} + \epsilon Q_{aux})) \\
&= \hat{e}(\beta Q_{bob}, r\delta Q_{alice}) * \hat{e}(\beta Q_{bob}, r\epsilon Q_{aux}) \\
&\quad * \hat{e}(\gamma R_{aux}, r\delta Q_{alice}) * \hat{e}(\gamma R_{aux}, r\epsilon Q_{aux}) \\
&= \hat{e}(\beta Q_{bob}, r\delta Q_{alice}) * \hat{e}(\beta Q_{bob} + \gamma R_{aux}, r\epsilon Q_{aux}) * \hat{e}(\gamma R_{aux}, r\delta Q_{alice}) \\
&= \hat{e}(\beta Q_{bob}, r\delta Q_{alice}) * \hat{e}(R_{virtual}, r\epsilon Q_{aux}) * \hat{e}(r\gamma R_{aux}, \delta Q_{alice}) \\
&= \hat{e}(\beta Q_{bob}, r\delta Q_{alice}) * \hat{e}(R_{virtual}, \epsilon U_2) * \hat{e}(\gamma U_3, \delta Q_{alice}).
\end{aligned}$$

$rQ_{alice}$  or  $rQ_{bob}$  would be needed to make the appropriate calculations. Obviously these values can not be found because the random value  $r \in F_p^*$  is known only by the sender who has sent the cryptogram  $(U_1, U_2, U_3, V)$ .

In conclusion, only Alice studying IBE schemes and Bob studying algebra have the information needed to decrypt the message  $M$ . This is highly desirable in situations such as the military implementation example when confidentiality is of great value. The greatest drawback to using the general disjunction is that it works for no more than two trust authorities and two identifiers. The reason for being limited to two trust authorities and two identifiers can be seen in the setup of the virtual and auxiliary keys.

Recall section 3 when the IBE scheme was introduced. The scheme is based on a bilinear map

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2.$$

There can only be two inputs, each from  $\mathbb{G}_1$ , to use the bilinear map  $\hat{e}$ . Using the generalized disjunction, however, there are as many inputs as there are identifier-trust authority pairs. Each virtual key uses one identifier and their associated trust authority. Thus, in order to use IBE schemes, these keys must somehow be reduced to a pairing. Up to this point, using more than two identifiers has not been problematic because the identifiers were combined in  $Q_{virtual}$  and the trust authorities were combined in  $R_{virtual}$ . In order to have a virtual key for every identifier-authority pair, we no longer only have  $Q_{virtual}$  and  $R_{virtual}$ , we have as many keys as there are identifier-trust authority pairs. This is highly detrimental because the military implementation presented in section 2.4 uses a minimum of six trust authorities. To be limited to two identifier-trust authority pairs is a significant disadvantage.

Each of the conjunction and disjunction operations have been introduced, along with the advantages and disadvantages of each. We now wish to consider which operations would be beneficial for implementation in each of our two applications. This, as well as the security attributes discussed in section 2.2.1, will be addressed in the next section.

## 5 IBE Using Multiple Trust Authorities in Ad Hoc Networks

Ad hoc networks, IBE schemes, and IBE schemes using multiple trust authorities have been presented. How do they relate to each other and why is this important?

Using multiple trust authorities in ad hoc network environments is beneficial because it decentralizes the trust authority. However, in order for a trust authority to be considered trustworthy, it must somehow establish a level of trust with all members using it for communication.

In order for this decentralization to take place among nodes within the network, every node would have to be trusted and would have to act as a trust authority themselves. This is highly problematic because IBE schemes are used for the purpose of establishing trust between nodes! In the examples presented in section 4, the trust authorities were assumed trustworthy. In other words, decentralization of the trust authorities among the nodes within the ad hoc network is ideal, but not reasonable to achieve.

Consider, on the other hand, the ideas presented in section 4.3 on general conjunction, particularly the use of a trusted time authority. Suppose each node is required to contact the time authority upon entrance into the network. After the initial communication with a time authority, nodes will be required to check-in periodically, based on their time stamp, to make sure that the information they have on other participating nodes is accurate. Using this idea, a network is able to benefit from the use of IBE schemes, key revocation, and multiple trust authorities with only minor disruptions taking place when the nodes exit the network to perform their check-in.

Each of the security attributes of interest to this paper, confidentiality, integrity, and authentication, will be affected by the use of a trusted time authority. Authentication will be improved because secure IBE schemes are being used. Thus, only persons holding the proper set of keys should be able to read a message or sign a message as long as the keys have not been compromised. Key revocation will improve authentication as well because adversaries have smaller time windows of which to compromise keys. Should a key be compromised, only the key for the designated time period will be compromised. The adversary will have to start anew for the refreshed time stamped key. Finally, the use of multiple trust authorities increases confidentiality as well as integrity. Using multiple trust authorities makes it difficult for trust authorities to collude and escrow keys, thus improving the level of confidentiality within the system. The use of several trust authorities provides for stronger integrity as well. This occurs because a message can be sent using trust authorities  $TA_1, TA_2, \dots, TA_k$  or it can be sent using any combination of these  $k$  trust authorities. If the sender sends multiple messages using alternative combinations of trust authorities, the message will have a better chance of arriving at least once unchanged, thus improving the integrity of the message. Integrity can be further increased by using the integrity advantages of ad hoc networks presented in section 2.2.2. Using different combinations for different routes throughout the ad hoc network will increase the integrity of the message using the network's characteristics as well as the characteristics of multiple trust authorities in IBE schemes.

As mentioned previously, it would be necessary to have more than one trusted time authority. If there would be only one time authority, a single point of failure is introduced into the ad hoc network. While not directly affecting the network, having the only time authority out of commission would either create a denial-of-service attack or would reduce the level of confidentiality within the network. If nodes were able to return to the network without getting updated information, compromised nodes or adversarial nodes would be able to continue communication. It is indeed likely that the adversary affecting the compromised nodes or acting as an adversarial node has affected the time authority for the purpose of protecting their false identity on the network. Thus, the confidentiality of the network has been compromised because the adversary can now communicate with all persons on the network. On the other hand, if the nodes were not able to return to the network without key updates, an easy denial-of-service attack could be implemented. The nodes would not be able to communicate until the time

authority came back online, thus rendering the network out-of-service for the nodes trying to get key updates.

In this final example, we go through a full implementation using the topics contained within this paper. Included in the example will be the setup and implementation of a communication system using an IBE scheme with multiple trust authorities. We believe this IBE implementation would be beneficial to the settings described in section 2.4.

### 5.1 Example: Full Implementation - Military Example

The military implementation of section 2.4 lends itself well to showing the advantages and disadvantages for both conjunction and disjunction. First, consider the initialization of the communication system. One trust authority must be set up for each of the following participants: countries, military branches, assignment locations, unit/squads, positions, and any other contextual identifiers that may be necessary for the identification of personnel. Each trust authority will distribute private keys to the persons they trust. For example, Alice would obtain a private key from the commander, unita, hostileoperation, navy, and us trust authorities.

After the private keys have been distributed, the trust authorities are only needed for purposes of promotions, unit changes, etc. In the interest of key revocation, however, an additional trust authority might be established to distribute time keys. These time keys, as mentioned in section 4.3 would be used to make sure that old keys are no longer valid. One of the main drawbacks of implementing a time stamp authority is that one of Shamir's goals has been overlooked. Shamir wanted the trust authority to no longer be needed after the original set up of the private keys. Thus, it must be determined by the system designers which is more important: key revocation or eliminating long lasting trust authorities.

By eliminating long lasting trust authorities, users of the communication system can be more confident that key escrow will not be used without their knowledge. Because this implementation uses the long lasting trust authorities, key escrow could be of concern to the users of the system. However, if we consider the purpose of this particular system, which is to allow military personnel to communicate, key escrow is not an issue. This is a work related communication system in which no amount of privacy is given to the users of the system. The government makes it known to their users that their communications will be monitored for purposes of keeping important governmental information secure. Thus, the implementors of this communication system would have decided that the use of long lasting trust authorities trump the users' concern of key escrow.

#### 5.1.1 Conjunction

The use of a trusted time authority has been assumed, so generalized conjunction obviously occurs in this case. In addition, however, general conjunctions are being used for communication between personnel. For instance, for Bob to communicate with U.S. Navy Commander Alice in Unit A of the Hostile Operations force, he would have to send messages to  $S_{alice,us}$ ,  $S_{alice,navy}$ ,  $S_{alice,commander}$ ,  $S_{alice,unita}$ , and  $S_{alice,hostileoperation}$ . Thus, only Alice holding each and every one of these private keys should be able to read the message. Adding in the trusted time authority, only Alice holding each of these private keys and the time key for xx-xx-xxxx:xxxx will be able to read messages sent to her which are to be read at time xx-xx-xxxx:xxxx. Consequently, conjunction

would certainly be of use in military implementations of this nature. Notice, however, that we have not accomplished the main goal expressed by Chen et al in [2]. We are still using one encryption per user when sending messages through this communication system.

Conjunction will not be useful, in general, for the purposes of decreasing the number of encryptions per user. In some cases, conjunction would be useful for sending a message to 'Alice and Bob and Carol'. There are several movie scenes which would be perfect for this setting. For instance, suppose Alice, Bob, and Carol are only able to gain access to some super secret safe if they are all present and use their keys. While far fetched, it could be a useful conjunction operation. For the general application of sending messages to 'Alice and Bob and Carol', however, the collaboration of these individuals would be difficult to require. Alice, Bob, and Carol may be in different locations, may not work the same hours, etc. Thus, in a general sense, conjunction would not be useful for transmitting messages to multiple users.

### 5.1.2 Disjunction

Because disjunction does not require the collaboration of users for reading messages, it has more potential to be useful. However, recall the disadvantage of using the naïve conjunction. When sending a message to multiple persons, unintended recipients are introduced. Because the military implementation has such a high concern for confidentiality, disjunction operations would be severely limited to transmissions which only include all intended recipients. Remember, the unintended recipients are predictable, so it might be possible, via protocol, to require transmissions to only 'valid' groupings of users - groupings in which the users of an admissible set are the exact same as the users in the grouping. General disjunction could fix the problem of having unintended recipients but because of the limitation to two users and two trust authorities, not much is gained in terms of decreasing the number of transmissions needed to communicate messages securely to multiple persons.

Again, the conjunction and disjunction operations on keys in this example have advantages and disadvantages. Depending on the situation at hand, conjunction or disjunction of keys might be useful when they ordinarily would not. Thus, when implementing an IBE scheme using multiple trust authorities in ad hoc networks, the situations and overarching goals of the proposed communication system should be considered before allowing transmissions to occur.

## 6 Conclusion

Mathematics and information security have many interesting links between them. IBE schemes working inside ad hoc networks is one link that has great potential for scenarios ranging from heavy combat to cordial conferencing. The security attributes used to evaluate information security have been presented, and have been used to consider a conference application as well as a military implementation. IBE schemes were presented in general, then their use with multiple trust authorities was investigated. Finally, these concepts were brought together to consider how multiple trust authorities would function in ad hoc networks.

There are many applications in which ad hoc networks are beneficial. However, authentication within ad hoc networks is a difficult and arduous task to implement because of the requirements placed on an ad hoc network. We believe that this paper has given some plausible methods of implementing authentication in ad hoc networks as well as two examples. These

examples were intended to illustrate the functionality and security opportunities made available through the use of the ideas presented.

While several disadvantages loom over the idea of IBE schemes in ad hoc networks, many advantages can be seen as well. It will be left up to the designers of the ad hoc network and the project at hand to determine whether these ideas will work under specific circumstances.

## 7 Acknowledgements

I would like to thank my co-major professors Dr. Clifford Bergman of the Iowa State University Mathematics Department and Dr. Thomas Daniels of the Department of Electrical and Computer Engineering. Their support and enthusiasm over the last two years helped me to find a topic of interest which I could run and develop with. Many thanks also go out to Dr. Davidson for her guidance and support over the last two years, Mike Hochstein and Kristi Meyer for being patient and reading through my many many drafts, and my family for putting up with my moods throughout these last two years.

## 8 Bibliography

1. D. Boneh and M. Franklin. *Identity Based Encryption from the Weil pairing*. Advances in Cryptology - CRYPTO 2001, Springer-Verlag LNCS 2139, pp. 213-229, 2001.
2. L. Chen, K. Harrison, D. Soldera, and N.P. Smart. *Applications of Multiple Trust Authorities in Pairing Based Cryptosystems*. Infrastructure Security International Conference - InfraSec 2002, Springer-Verlag LNCS 2437, pp. 260-275, 2002.
3. M. Gagné. *Identity-Based Encryption: A Survey*. CryptoBytes, a technical newsletter of RSA laboratories, Vol. 6, No. 1, 2003.
4. Z. Haas and L. Zhou. *Securing Ad Hoc Networks*. IEEE Network, Vol. 13, No. 6, pp. 24-30, 1999.
5. V. Kärpijoki. *Security in Ad Hoc Networks*. 2000. [referred 19.3.2004]  
<[http://www.hut.fi/~vkarpijo/netsec00/netsec00\\_manet\\_sec.html](http://www.hut.fi/~vkarpijo/netsec00/netsec00_manet_sec.html)> [in HTML format].
6. A. Katz, A. Khalili, W. Arbaugh. *Toward Secure Key Distribution in Truly Ad-Hoc Networks*. Applications and the Internet Workshops, Proceedings, pp 342-346, 2003.
7. A. Shamir. *Identity-Based Cryptosystems and Signature Schemes*. Advances in Cryptology - CRYPTO 1984, pp. 47-53, Proceedings, 1984.