IDENTITY BASED ENCRYPTION PROTOCOL FOR PRIVACY AND AUTHENTICATION IN WIRELESS NETWORKS

A Thesis

by

CLIFTON J. MULKEY

Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

Texas A&M University - Corpus Christi

Corpus Christi, Texas

May 2011

Major Subject: Computer Science
ABSTRACT

Wireless networks are inherently insecure due to the fact that information on the network can be passively retrieved by an eavesdropper using off-the-shelf network hardware and free software applications. The most common solution for this vulnerability is the Wireless Protected Access (WPA) protocol. This protocol provides data encryption and access control for wireless networks. However, the WPA protocol contains drawbacks in its authentication mechanisms that can cause inconveniences for end users and performance degradation for the network. Furthermore, many of the authentication methods used with WPA are not efficient for small and resource-constrained wireless devices. This work presents the design of a new wireless security protocol for privacy and authentication using efficient Identity Based Encryption (IBE) techniques. This protocol eliminates the need for a central authentication server for enterprise networks, as well as provides the new feature of privacy without authentication for public wireless networks. This work also puts forth an analysis and validation of the new protocol, including security strength, storage overhead, communication overhead, and computational efficiency.
ACKNOWLEDGMENTS

I would like to thank Dr. Dulal Kar, Dr. Ahmed Mahdy, and Dr. Ajay Katangur for their guidance and support throughout my thesis work and my time at TAMUCC.

A special thanks goes to my wife, Lauren, for her constant love and support that always encourages me in my work.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Contribution</td>
<td>3</td>
</tr>
<tr>
<td>2 LITERATURE REVIEW</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Identity Based Encryption</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Elliptic Curve Cryptography</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Pairing Based Cryptography for IBE</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Recent Developments in IBE for Wireless Sensor Networks</td>
<td>10</td>
</tr>
<tr>
<td>2.5 IEEE 802.11 Wireless Ethernet</td>
<td>12</td>
</tr>
<tr>
<td>2.5.1 Functionality</td>
<td>12</td>
</tr>
<tr>
<td>2.5.2 Important Frame Types</td>
<td>13</td>
</tr>
<tr>
<td>2.5.3 802.11 Frame Format</td>
<td>15</td>
</tr>
<tr>
<td>2.6 Wireless Protected Access</td>
<td>16</td>
</tr>
<tr>
<td>2.6.1 Privacy</td>
<td>16</td>
</tr>
<tr>
<td>2.6.2 Authentication</td>
<td>19</td>
</tr>
<tr>
<td>3 PROTOCOL DESIGN</td>
<td>21</td>
</tr>
<tr>
<td>3.1 Preliminaries</td>
<td>21</td>
</tr>
<tr>
<td>3.2 Functionality Without Authentication</td>
<td>24</td>
</tr>
<tr>
<td>3.3 Functionality With Authentication</td>
<td>26</td>
</tr>
<tr>
<td>3.3.1 Initial Connection</td>
<td>27</td>
</tr>
<tr>
<td>3.3.2 Subsequent Connections</td>
<td>30</td>
</tr>
<tr>
<td>3.4 Key Management and Changes</td>
<td>33</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>3.4.1 New Clients</td>
<td>33</td>
</tr>
<tr>
<td>3.4.2 IBE Parameter Changes</td>
<td>34</td>
</tr>
<tr>
<td>3.5 Operation on Multiple Access Point Networks</td>
<td>35</td>
</tr>
<tr>
<td>3.6 Frame Format</td>
<td>37</td>
</tr>
<tr>
<td>4 PROTOCOL ANALYSIS</td>
<td>40</td>
</tr>
<tr>
<td>4.1 Security</td>
<td>40</td>
</tr>
<tr>
<td>4.1.1 Protocol Without Authentication</td>
<td>41</td>
</tr>
<tr>
<td>4.1.2 Protocol With Authentication</td>
<td>42</td>
</tr>
<tr>
<td>4.1.3 WPA-CCMP Secure Channel</td>
<td>46</td>
</tr>
<tr>
<td>4.2 Storage Overhead</td>
<td>46</td>
</tr>
<tr>
<td>4.2.1 Access Point</td>
<td>47</td>
</tr>
<tr>
<td>4.2.2 Client</td>
<td>49</td>
</tr>
<tr>
<td>4.3 Communication Overhead</td>
<td>50</td>
</tr>
<tr>
<td>4.3.1 Protocol Without Authentication</td>
<td>50</td>
</tr>
<tr>
<td>4.3.2 Protocol With Authentication</td>
<td>51</td>
</tr>
<tr>
<td>4.4 Computation Overhead</td>
<td>51</td>
</tr>
<tr>
<td>4.4.1 Benchmarks</td>
<td>52</td>
</tr>
<tr>
<td>4.4.2 Parameter Generation</td>
<td>55</td>
</tr>
<tr>
<td>4.4.3 Protocol Without Authentication</td>
<td>56</td>
</tr>
<tr>
<td>4.4.4 Protocol With Authentication</td>
<td>57</td>
</tr>
<tr>
<td>4.4.5 Summary</td>
<td>58</td>
</tr>
<tr>
<td>5 CONCLUSION</td>
<td>61</td>
</tr>
</tbody>
</table>

REFERENCES | 63 |
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>AP parameter storage requirements</td>
<td>48</td>
</tr>
<tr>
<td>II</td>
<td>Client parameter storage requirements</td>
<td>50</td>
</tr>
<tr>
<td>III</td>
<td>IBE Computation Benchmarks at 3.0 GHz</td>
<td>53</td>
</tr>
<tr>
<td>IV</td>
<td>IBE Computation Benchmarks at 656 MHz</td>
<td>53</td>
</tr>
<tr>
<td>V</td>
<td>Upper Bounds for Computing EC scalar multiplication and exponentiation over $F_{3^{509 \times 6}}$</td>
<td>55</td>
</tr>
<tr>
<td>VI</td>
<td>Summary of client computation time bounds</td>
<td>59</td>
</tr>
<tr>
<td>VII</td>
<td>Summary of AP computation time bounds</td>
<td>59</td>
</tr>
<tr>
<td>VIII</td>
<td>Summary of RSA computation times (3072 bit modulus)</td>
<td>60</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IEEE 802.11 Wireless Ethernet Frame Format</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>WPA-TKIP Frame Format [23]</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>WPA-CCMP Frame Format [23]</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>Sequence diagram of protocol without authentication</td>
<td>27</td>
</tr>
<tr>
<td>5</td>
<td>Sequence Diagram for first-time client connection and authentication</td>
<td>28</td>
</tr>
<tr>
<td>6</td>
<td>Sequence Diagram for subsequent client connection and authentication</td>
<td>31</td>
</tr>
<tr>
<td>7</td>
<td>IBE Protocol Frame Layout</td>
<td>37</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Background

From large corporations to street-corner coffee shops, wireless networks provide an important means of connectivity in today’s world. However, security in wireless communications is hindered by the fact that wireless media is by nature a broadcast media. Since wireless signals are carried on radio waves, any receiver within range can potentially intercept a message passed between a wireless access point and a client. In fact, free tools exist that allow users to listen to traffic in promiscuous mode, capturing any data that happens to come to the wireless network card. One such tool is Wireshark, powered by the pcap library [26]. This graphical tool allows even novice users to capture and analyze traffic on a wireless network. The tool even provides routines for reconstructing messages, files, or images from common application protocols. Furthermore, this type of network sniffing is passive, and cannot be detected by intrusion detection systems that may be operating on the network.

Without encryption, a wireless eavesdropper can capture potentially sensitive information from other clients on the wireless network. Although some traffic is protected using application or session layer security protocols, there are still many plaintext messages that are sent across the wireless media. For example, email, instant message programs, and HTTP often send messages in plain text which can be captured easily. This inherent security risk on wireless networks is often mitigated today by using the Wireless Protected Access (WPA) protocol [8]. WPA and its second revision, WPA2, both operate in the same manner. WPA2 provides a higher
security level by using the Advanced Encryption Standard (AES) symmetric cipher, whereas WPA uses the RC4 cipher [15]. The WPA protocol includes two operating modes, WPA Pre-Shared Key (WPA-PSK) and WPA Extensible Authentication Protocol (WPA-EAP). Using WPA-PSK, a pre-shared key is required to gain access to the network, so that only authenticated users can connect to the access point. In addition, a session key is established between each client and the access point upon connection, so that there is a separate encrypted channel for each client [15]. Though WPA-PSK has greatly improved wireless security, it still may not be a good solution for all scenarios. Since the WPA-PSK key is a symmetric key, it must be distributed to all users of the network. This may be infeasible for very large wireless networks. In addition, this creates a problem for networks that expect to have a large number of temporary users. If a client needs to use the network for one hour, he will have to be provided with the key. When he leaves, the key for the network should probably be changed to avoid unauthorized sharing with other users. This need to change keys reiterates the previous problem of key distribution. Furthermore, any user that obtains the WPA-PSK network key may still be able to sniff traffic from other users on the network. Though a separate session key is established for each client in the network, this key is derived from the symmetric network key during the initial connection handshake. If this handshake is captured by a wireless sniffer with the network key, it may be possible to calculate the session key [15]. WPA-EAP provides a much stronger authentication method and does not require a pre-shared key among the access point and all clients. However, WPA-EAP requires a central authentication server with a persistent database with all authorized user credentials [15]. For this reason, it may not be suitable for small or single access point wireless networks. The reliance on a central server can also create a bottleneck or single point of failure for the network. In addition, WPA-EAP uses higher level public key protocols that may
require certificate storage and exchange, as well as expensive encryption. Finally, for some wireless networks, such as in a coffee shop or airport, it may be desirable to have traffic confidentiality but not client authentication.

1.2 Contribution

The threat of passive information sniffing has been eliminated on modern wired networks by switching technology. On a switched network, each link between a host and the network switch is physically private. Self-learning switches keep track of the MAC address of the host attached to each port [12]. Using this information, frames on the network are only forwarded between the sending and receiving host. No other host on the network is able to sniff information because the network traffic is physically isolated between the host and the switch. The end result is that passive eavesdropping is prevented by the network without the need for any password or special configuration at the hosts. One should note that eavesdropping is still possible on switched networks by using an ARP spoofing attack [1]. The defense against this type of active attack is beyond the scope of this thesis. However, active attacks such as ARP spoofing can often be detected by modern intrusion detection systems [1].

The characteristics of switched wired networks motivate a solution for preventing passive information sniffing on wireless networks. Using cryptographic techniques, it is possible to bring the traffic isolation properties of switched networks to wireless media. This solution involves providing a separate encrypted channel between each host and the wireless access point. This solution also avoids the downfalls of WPA, which either requires a shared key and the possibility of session key capturing, or a separate authentication server with a complex configuration. This solution can also go beyond the privacy characteristics of a switched network to provide client and access point authentication if required by the network. Furthermore, clients will be able to
authenticate seamlessly between multiple access points on larger wireless networks.

To achieve this solution, a novel protocol using a form of asymmetric cryptography known as Identity Based Encryption (IBE) is proposed. Asymmetric cryptography is needed to avoid the requirement for shared keys. More specifically, IBE is used because it eliminates the need for public key storage and certificate infrastructure. IBE can be used to securely share symmetric keys, which can later be used by AES symmetric encryption to establish secure channels. IBE key exchange methods can also be formulated in such a way to authenticate both the access point and the client. Furthermore, IBE uses Elliptic Curve Cryptography (ECC) which provides high cryptographic strength with a smaller amount of computation and smaller key size in comparison to traditional public key methods [2, 11]. This provides for a smaller communication, computation, and storage overhead for the protocol, and opens the door for the protocol to be used on smaller devices. Because of these characteristics of IBE, this protocol can be managed by individual access points without the need for a central server. This allows the work of network authentication to be distributed and removes a single point of failure. Further details of IBE and ECC are discussed in Chapter 2.

The overall contribution of this work is the design of such a protocol for wireless network privacy and authentication. This protocol design includes the case for networks requiring privacy and authentication and for networks without a need for authentication. The design also considers key management and authentication for both single and multiple access point networks. Finally, this design also specifies the mathematics and cryptographic techniques necessary to achieve the protocol functionality, as well as the communication steps and link-layer frame specifications needed for practical operation.

This work also consists of an analysis and validation of security strength and
overheads of the new protocol. The security analysis discusses the cryptographic strength of the protocol in relation to NIST standards [19]. It also defines several possible attacks to the protocol and proofs of resistance to these attacks. For the analysis of protocol overheads, this work considers the storage, communication, and computation overheads that are associated with clients and wireless access points participating in the protocol.

The remainder of the paper is structured as follows. Chapter 2 provides a review of previous research and background information in fields relating to IBE and wireless network security. Chapter 3 gives the design and details of the new wireless security protocol contributed by this work. Chapter 4 contains the security and overhead analysis of the new protocol. Finally, Chapter 5 is the conclusion of this work and gives some topics that may be considered for future research in the area.
CHAPTER 2

LITERATURE REVIEW

2.1 Identity Based Encryption

The concept of identity based encryption was first introduced by Shamir in 1984 [22]. IBE is a form of asymmetric or public key cryptography. However, instead of using pseudo-random public keys derived through mathematical functions, common identity strings are used. The example proposed by Shamir involves using email addresses as public keys. In our case, we will use MAC addresses as public keys since the wireless security protocol will operate on the link layer. Another significant difference between IBE and traditional public key systems is that clients cannot generate their own private keys. Instead, an IBE system requires some authoritative server with a master secret key. This server is the only one that can generate and distribute private keys based on identity strings. This requirement stems from the fact that public keys are not mathematically derived. If clients had the ability to generate the private key for their own identity string, they could also generate the private key for any other client’s identity string. It should be noted, however, that the infrastructure required for an IBE key generating authority is still minimal compared to the requirements of the certificate system used in traditional public key cryptosystems [16]. In the case of a wireless network, the function of IBE key generation and authentication could be performed by the access point itself.

The following simplified example shows the process in which secure and authenticated communication could be achieved with IBE. Suppose Alice and Bob have already obtained the IBE private key corresponding to their identities from the key generation server. Now if Alice wants to send a message to Bob, she first signs the
message using her private key. She then obtains Bob’s host name and encrypts the signed message using Bob’s name as the public key. She sends the encrypted message to Bob, along with her own identity. When Bob receives the message, he can first decrypt it using his own private key. He can then verify the message using Alice’s identity. Though the basic process is similar to traditional public key cryptosystems, it can be seen that less overhead is required because there is no need to communicate public keys. At the time of its introduction there were no mathematical systems that could provide the properties required for the functioning of an IBE cryptosystem [22]. It was the invention of elliptic curve cryptography and pairing functions on elliptic curves that later made IBE possible.

2.2 Elliptic Curve Cryptography

Traditional public key algorithms rely on mathematical properties of prime-order groups and modular arithmetic. These systems gain their security from the difficulty of either the discrete logarithm or integer factorization problems [21]. For the purpose of this research, we focus on the Discrete Logarithm Problem (DLP). The definition of the DLP is to determine the scalar \( e \) given \( P \) and \( R \), where \( R = P^e \). In order for a public key system to be secure, the DLP must be infeasible, which means the prime-order groups in operation must be significantly large. In fact, a 1024 bit key size must be used in order to achieve the security strength of an 80 bit symmetric key [19]. This requirement for large numbers causes traditional public key cryptographic operations to be relatively slow compared to symmetric schemes.

Instead of operating on prime-order groups, Elliptic Curve Cryptography (ECC) systems involve operations on elliptic curves defined over finite fields. Commonly used fields are prime fields \( F_p \) and binary fields \( F_{2^m} \). The two dimensional elliptic curve is specified by a publicly known equation. Points on the elliptic curve form an
abelian group under point addition [14]. For a further definition of point addition, see [2]. The foundational operation used in ECC is scalar point multiplication. Point multiplication is defined as repeated point addition of a point on the curve, \( P = (x, y) \). The security of ECC schemes is based on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP). The definition of the ECDLP is to determine \( e \) given curve points \( P \) and \( R \), where \( R = eP \). Any point \( P \) on the elliptic curve forms a cyclic subgroup \( \{ P, 2P, 3P, \ldots \} \), so the order of the subgroup generated by \( P \) must be sufficiently large to prevent brute force guessing [14]. Since the ECDLP is similar to the DLP, many traditional public key protocols can be transformed to ECC protocols. One of these protocols is the Diffie-Hellman protocol, which is known as the Elliptic Curve Diffie-Hellman (ECDH) when transformed to the ECC domain [14, 21]. The major advantage of ECC over traditional public key methods is that it provides much more security per bit. NIST standards state that a 160 bit ECC system can provide equivalent security to a 1024 traditional public key system [19]. As a result, ECC cryptosystems require less communication and computation overhead than traditional public key methods. For this reason, ECC has been heavily developed for use in small and embedded devices such as sensor nodes in wireless sensor networks. More importantly for IBE, ECC has properties that enable the calculation of pairing functions. These functions provide the necessary mappings to build an IBE system.

2.3 Pairing Based Cryptography for IBE

Pairing based cryptography (PBC) is the mathematical mechanism that makes modern IBE implementations possible. The first practical IBE cryptosystem using PBC was developed by Boneh and Franklin in 2003 [6]. The Boneh and Franklin scheme uses a pairing function known as the Weil pairing. This pairing performs the
mapping \(G_1 \times G_1 \rightarrow G_2\), where \(G_1\) is a additive group of elliptic curve points, and \(G_2\) is a finite field. In general terms, a pairing function maps two elliptic curve points to an element in a finite field, known as the extension field. This function is usually represented by \(e\), with \(e(P, R) = k\), where \(P\) and \(R\) are points on an elliptic curve and \(k\) is an element of the extension field. In order for PBC and IBE to work, the pairing function \(e\) must have the bilinear property: \(e(sP, tR) = e(P, R)^{st}\), where \(s\) and \(t\) are scalars.

The following example from [11] shows how this property may be used to share an encrypted message using IBE. First assume you have a trusted party holding a secret \(s\). The trusted party computes \(R = sP\) and publishes \(R\) and \(P\), which are points on an elliptic curve. Also assume that each party has already retrieved its private key \(d\) from the trusted server. The private key is \(sQ_x\), where \(Q_x\) is an elliptic curve point representing that party’s ID. Suppose Alice wants to generate an encryption key for sending data to Bob. She first determines \(Q_{Bob}\) using Bob’s identity string and some known hash-to-point function. Next, she generates an elliptic curve point \(U = kP\), where \(k\) is a random scalar. Finally, Alice generates a key using \(e(kR, Q_{Bob})\) and sends this along with \(U\) to Bob. Assuming Bob has already retrieved his private key \(sQ_{Bob}\) from the trusted server, he can now determine the key generated by Alice calculating \(e(U, sQ_{Bob})\). The following equation shows how Bob can retrieve the shared key using the bilinearity property.

\[
e(U, sQ_{Bob}) = e(kP, sQ_{Bob}) = e(P, Q_{Bob})^{ks} = e(kSP, Q_{Bob}) = e(kR, Q_{Bob})
\]  

(2.1)

Using similar methods, IBE and pairing functions can also be used to achieve authentication using signatures.

The security of pairing based IBE systems relies on the difficulty of either the Bilinear Diffie Hellman Problem (BDHP) or the Gap Diffie-Hellman Problem (GDHP)
The definition of the BDPH is to find \( e(P, P)^{abc} \) given \( \{P, aP, bP, cP\} \) where \( P \) is a point on an elliptic curve and \( a, b, \) and \( c \) are scalars. The definition of the GDHP is to find \( abP \) given \( \{P, aP, bP\} \). One or both of these problems should be infeasible in order for an IBE cryptosystem to be secure. In addition, the pairing function in an IBE system has the side effect of allowing the ECDLP in \( G_1 \) to be converted to a DLP in \( G_2 \). For this reason, both of these problems must also be sufficiently hard to maintain security. The order of the extension field \( (G_2) \) is \( q^k \), where \( q \) is the order of the field on which the elliptic curve is defined. The value \( k \) is known as the embedding degree and is a property of the elliptic curve. It is desirable to maximize the embedding degree in order to achieve a difficult DLP in \( G_2 \) while keeping the order of \( G_1 \) relatively small. For example, the authors of [3] detail the construction of pairing friendly elliptic curves over \( F_p \), where \( p \) is a 256 bit prime. These curves are specially constructed to have an embedding degree of 12. This creates an extension field of order \( 256 \times 12 = 3072 \), which meets the DLP hardness standards to achieve 128 bit security. Besides the Weil pairing used in the Boneh and Franklin scheme, another pairing function known as the Tate pairing can also be used. The Tate pairing is often more computationally efficient than the Weil pairing [18]. Many other speed improvements have to IBE have also been made due to its popularity for use in small devices.

### 2.4 Recent Developments in IBE for Wireless Sensor Networks

Due to the small key sizes of ECC and the fact that IBE does not need a public key infrastructure to operate, IBE has become a popular candidate for encryption and authentication in Wireless Sensor Networks (WSNs) [17, 24, 27]. Since the computing devices involved in WSNs are severely limited in terms of computation power and energy, much research has been done in this field for further increasing the effi-
ciency of IBE. One of the significant advances that has been achieved in some recent implementations of IBE for WSNs is the reduced Tate ($\eta T$) pairing [11]. The $\eta T$ pairing algorithm operates over special types of elliptic curves, known as supersingular curves. Using this type of pairing, IBE has been implemented efficiently for small wireless devices. As shown in [24], one such implementation, known as Tiny-IBE, achieved a decryption operation on a 32-bit, 416MHz Imote2 in just over 14 ms. Furthermore, this operation only required 4.12 KB of RAM on the device. This implementation used a supersingular curve over the binary field $F_{2^{271}}$ with embedding degree 4. This provides roughly the same security level as 1024 bit RSA or an 80 bit symmetric algorithm [19]. Note that the decryption performance is chosen as a reference here because it is the only step involving a pairing calculation in this particular protocol. In addition, research in WSNs has provided efficient algorithms for necessary IBE helper functions. One such function is the hash-to-point function, which maps an identity string to a point on the specified elliptic curve. This function can be expensive to implement but has been simplified in recent research [25]. The IBE scheme outlined in [24] even leverages properties of supersingular curves to eliminate the need for hash-to-point operations and reduce the number of pairing operations necessary. Due to the efficiency achieved for IBE on WSNs, it is likely that IBE can also provide great performance on the more powerful devices, such as access points and laptop PCs, that are involved in a traditional wireless LAN. Furthermore, this shows that IBE security for wireless LANs can also be efficient enough for use with the growing number of mobile and hand-held devices.
2.5 IEEE 802.11 Wireless Ethernet

2.5.1 Functionality

The IEEE 802.11 Wireless Ethernet protocol is the standard for today’s wireless LAN technology. The standard has had many different revisions (denoted by lowercase letters), but all revisions provide the same basic functionality [23]. In some ways, the wireless Ethernet protocol is similar to the wired Ethernet protocol. The basic transmission unit in wireless Ethernet is a frame. Addressing is done using MAC addresses attached to each physical device. Error checking is done with the calculation and verification of a Cyclic Redundancy Check (CRC) code that is attached as a trailer to each frame.

One of the main differences that wireless Ethernet has from the standard wired Ethernet protocol is collision avoidance [23]. Since frame collisions can happen very easily in the wireless medium, wireless Ethernet uses a protocol in which the medium is verified to be clear by the sender before sending. Further details on this functionality will be discussed in Section 2.5.2. Wireless LAN (WLAN) networks can operate in two modes, managed and ad-hoc. Ad-hoc mode is used for peer-to-peer and mobile wireless networks. The more common managed mode takes on a client server model with a fixed wireless Access Point (AP) and multiple wireless clients. In this type of WLAN, all wireless traffic is relayed through the AP before going to any other destination. The AP along with its clients from a Base Service Set (BSS) and the MAC address of the AP is known as the BSSID. The topics discussed in this work deal solely with managed mode WLANs.

Wireless LANs can take on many different topographies. A simple home or small office wireless network usually only consist of one AP or BSS connected to a wired backbone. All wireless clients connect to this single AP for access to the Internet.
or other networks. In a larger setting, many APs may be connected to the same wired backbone forming a system known as an Extended Service Set (ESS). In this sense, the ESS is a collection of base service stations that is seen by the user as a single wireless network. The ESSID of the wireless network is usually a name that identifies the entire system of access points. An example of this a university campus network. Though many different access points exist in different buildings throughout the campus, a wireless user only sees and connects to a particular ESSID, regardless of what specific BSS the client is actually a member of. With larger networks such as this, the problem of authentication becomes more complicated.

2.5.2 Important Frame Types

The 802.11 protocol includes specification for many different frame types. Only the frame types that are important to the understanding of this work are discussed here. The wireless Ethernet frame types can be divided into three major categories: control, data, and management [23].

Control frames help to provide the collision avoidance function of wireless Ethernet. The first type is Request To Send (RTS). This frame is sent by a wireless client to the AP before the client wants to transmit data. The client will not transmit any data frames until it received a Clear To Send (CTS) frame addressed from the AP to itself. The CTS frame is also used as a signal to other clients on the network that they are not allowed to transmit any frames. Once a client is finished sending a data frame, it waits for an acknowledgement (ACK) frame from the AP, signaling the successful receipt of the frame. If the ACK is not seen by the sending client, the frame will be retransmitted after a certain timeout. The ACK frame also acts as a signal to other clients on the network that they can now make a request to send. No frames are ever sent on the network without the use of these control frames. In this
way, collision avoidance is enforced on the wireless medium.

Data frames are simply used to carry payloads consisting of higher level protocols. These are the simplest frame types, but it should be noted that data frames can, in some cases, be combined with control frames, such as DATA + ACK.

Management frames are the most relevant to this work. These frames are used set up communication channels between an AP and a wireless client. The following list gives a summary of important management frame types:

**Beacon** – Beacon frames are broadcast periodically by the AP to let clients know of its existence. These frames usually include the ESSID of the network and the BSSID of the access point.

**Probe Request and Response** – A probe request can be sent by a client or an AP to get more detailed information about the device. This frame is also used by a client to locate an AP that is not broadcasting Beacon frames. A probe response is sent to answer with the information requested in the probe request.

**Association Request and Response** – An association request is sent by a client to an AP to request a position on the wireless network. The association request usually includes the capabilities of the client such as available security protocols or authentication mechanisms. The AP can either accept the client as a part of the network with an association response, or reject the client with a Disassociation frame.

**Authentication and De-authentication** – In a wireless network where clients must be authenticated, the association request and response is followed by a series of authentication exchanges. Authentication frames are used to determine whether the client is authorized to join the network and to set up a secure
communication channel. De-authentication frames can be sent by the AP or a client to signal the end of a secure session.

**Disassociation** – A disassociation frame can be sent either by the access point or a client to terminate an association with the AP. It should be noted, however, that a client can leave the network without sending a disassociation frame.

### 2.5.3 802.11 Frame Format

The following diagram shows the specified format for an 802.11 wireless Ethernet frame [23]:

![IEEE 802.11 Wireless Ethernet Frame Format](image)

**FC** – Frame Control Field – This field contains many different subfields that give protocol control information about each frame. The most important parts of this field are the type and subtype subfields. The binary values in these fields are used to specify the frame types described above. Another notable subfield is a 1 bit flag indicating whether the frame payload is encrypted.

**DCI** – Duration/Connection ID – This field may either be used for the time the wireless channel will be allocated for data transmission or for a connection ID tag.

**SC** – Sequence Control – This field is used to number the frames sent between a client and the AP. It also contains a subfield that can be used for fragmentation and reassembly of frames.
**Addresses** – Wireless frames can contain different number of addresses depending on the context of the wireless network. In a managed network, the wireless frames usually contain three MAC addresses: the sender, the receiver, and the AP (BSSID).

**Payload** – This is the data portion of the frame that carries all information from higher-layer protocols. The 802.11 standards specify that frames can carry a payload up to 2312 bytes. However, different wireless hardware may have a smaller Maximum Transmission Unit (MTU). Wired Ethernet has an MTU of 1500 bytes [12]. For this reason, it may be desirable to use the same MTU for wireless Ethernet to prevent fragmentation at the external wired interface of the AP.

**FCS** – Frame Check Sequence – This field holds the 32 bit CRC value that is used to detect errors in the frame. If errors are found, the frame is simply dropped, causing the sender to resend the frame after a specified timeout.

### 2.6 Wireless Protected Access

The WPA addition to wireless Ethernet protocol was originally specified in the IEEE 802.11i revision [8]. WPA was designed to overcome serious security vulnerabilities found in the previous Wired Equivalent Privacy (WEP) standard, as well as add advanced authentication methods.

#### 2.6.1 Privacy

The privacy specifications for the WPA protocol can be divided into two revisions. This first is WPA Temporal Key Integrity Protocol (WPA-TKIP) [8]. WPA-TKIP was designed to be fully compatible with existing WEP hardware. For this
reason, TKIP uses the same RC4 stream encryption algorithm as WEP. This algorithm encrypts data using a 104 bit key and a 24 bit initialization vector. WPA-TKIP overcomes the cryptographic weakness of WEP by using an Initialization Vector (IV) and an Extended Initialization Vector (EIV) field to create a stronger initialization vector for the cipher algorithm. By mixing the first 2 bytes of the IV field and all 4 bytes of the EIV field, a 48 TKIP sequence number (TSC) is created and is different for each frame. This 48 bit TSC could theoretically be used over 100 years without overlapping at the maximum transmission rate of wireless Ethernet [8]. Along with the TSC, TKIP uses a temporal key for encrypting frames. The temporal key is derived from the master network key in WPA-PSK mode, or is established by other means in WPA-EAP. Each frame in a WPA-TKIP session is encrypted with a different key. The 128 bit packet key, (24 bit initialization vector plus 104 bit RC4 key) is derived from the 48 TSC, the temporal key, and the sender’s MAC address. The following figure shows the format of a WPA-TKIP wireless frame:

![Fig. 2. WPA-TKIP Frame Format](image)

**IV and EIV** – The IV and EIV are plaintext fields that are used to derive the initialization vector for packet encryption and decryption. These fields are also used as a sequence identifier to prevent replay attacks

**Data** – This is the encrypted payload of the frame. Similar to the general 802.11 protocol, the specifications allow up to 2312 bytes of data to be sent in the data field. However, the size of the field in practical scenarios is likely smaller.
MIC – Message Integrity Code – This encrypted field holds a hash value of the plaintext payload plus the source and destination addresses. This is used to ensure that frames are not tampered with en route. The hash value is calculated with a secure, one-way algorithm known as Michael [8].

ICV – Integrity Check Vector – This field holds an encrypted version of the CRC checksum calculated over the frame. This value was used for frame integrity in the WEP protocol, but was found to be insecure. Since TKIP uses the MIC field for frame integrity, this field is only present for backward compatibility.

The second mode of privacy specified by the 802.11i standard is WPA Counter Mode CBC Protocol (WPA-CCMP or WPA2) [23]. This form of WPA is more computationally intensive and is not compatible with older WEP based hardware. However, the protocol is much more cryptographically secure as it uses the 128 bit AES algorithm in Cipher Block Chaining (CBC) mode. As seen in figure 3, the CCMP frame format is similar to the TKIP format. The IV and EIV are again used as a 48 bit initialization vector to seed the AES algorithm. The 128 bit AES session key is derived in a similar manner to that in TKIP. The MIC field is also calculated using the Michael hash function as in TKIP. The major difference in the CCMP frame format is the omission of the ICV field, which is no longer needed since backward compatibility is not a goal of WPA-CCMP.

<table>
<thead>
<tr>
<th>Bytes</th>
<th>≤ 30</th>
<th>4</th>
<th>4</th>
<th>≤ 2312</th>
<th>8</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>802.11 Header</td>
<td>IV</td>
<td>EIV</td>
<td>Payload</td>
<td>MIC</td>
<td>FCS</td>
</tr>
</tbody>
</table>

Fig. 3. WPA-CCMP Frame Format [23]
2.6.2 Authentication

WPA provides two different modes of authentication that can be used for wireless networks [8]. The two modes are known as Pre Shared Key (WPA-PSK) and Extensible Authentication Protocol (WPA-EAP). It should also be noted that WPA does not provide privacy without authentication. At least one of these authentication modes is used on any WPA network.

The simpler and more commonly used of the two is PSK mode. In PSK mode, network authentication is controlled by a master key. Any client that wishes to join the network is required to have this master key. As mentioned above, this master key is used to derive the session key used to encrypt frames between the access point and the client. This authentication mode is convenient for home and small office networks with a limited number of users. No central authentication server is required and a single access point can enforce access to the network. However, WPA-PSK does not scale well because of the need for every client to have knowledge of the same master password.

For larger networks involving multiple access points, WPA-EAP can be used. WPA-EAP requires a central authentication server with a database of credentials for authorized users. EAP uses upper layer protocols to create a secure connection to the authentication server. In some EAP systems, digitally signed certificates are used to verify the access point, authentication server, or clients. WPA-EAP can be further divided in to four major types [8]:

**EAP-TLS** – EAP with Transport Layer Security uses the same type of certificate based infrastructure that is used on the Internet. Clients and authentication servers are authenticated using public/private keys and signed identification certificates.
**EAP-TTLS** – EAP with Tunneled TLS uses the same type of authentication that is common when logging in to secure websites. An encrypted connection is established using TLS methods, then the client provides a user name and password to the server to authenticate.

**PEAP** – Protected EAP functions similar to EAP-TTLS. It uses a password-based system for network authentication.

**LEAP** – Lightweight EAP is a proprietary system developed by Cisco. It uses a challenge-response based authentication mechanism.

Each client of an EAP network has its own set of credentials, so the system scales much better than PSK mode. Key management is also effective because user credentials can easily be added, removed, or modified by the system administrator on the authentication server. However, the central authentication server can also be a single point of failure or a performance bottleneck for the wireless network.
CHAPTER 3

PROTOCOL DESIGN

3.1 Preliminaries

Before describing the design and functionality of the new protocol for wireless network security, the necessary preliminaries for defining and understanding the protocol will be discussed. As mentioned in Chapter 1, identity based encryption techniques will be used to provide authentication and key setup. Although many different IBE pairing methods exist, this protocol uses IBE over supersingular elliptic curves. Many definitions of supersingular exist for elliptic curves [9]. However, the specific property of supersingular curves that we are interested in for this work is the existence of efficient distortion maps. Pairing over many elliptic curves breaks down when the two points being paired are linearly dependent [9]. For this reason the second argument to the pairing function must be a point on some variation of the original elliptic curve over a larger finite field [10]. With supersingular curves, the distortion map gives an easy way to transform a point on an elliptic curve to a second point that is in a different cyclic subgroup, essentially removing linear dependence [9]. The result is that the pairing over supersingular curves can operate on two points from the same elliptic curve. This allows for simpler IBE protocols and can be used to eliminate the need for a hash-to-point function. The specific pairing algorithm used in this work is the $\eta T$ pairing, which is the fastest known pairing over supersingular curves [27].

To analyze this protocol, we use a supersingular curve defined over the ternary field $F_{3^{509}}$, given in [4]. The curve over this field is defined by $y^2 = x^3 - x + b$, where $b$ is between $-1$ and $1$. The modulus used for the field is the polynomial $x^{509} - x^{318} - x^{191} + x^{127} + 1$. This curve has an embedding degree of 6, which results in
the extension field $F_{3^{509} \times 6}$. The order of the curve and the size of the extension field is large enough to provide 128 bit security, or about the same security level as 3054 bit RSA [4, 19]. A curve at this security level is chosen in order to match the standards of WPA-CCMP and the NIST standards for the foreseeable future (beyond 2030) [19]. More curves are available that can provide this level of security. This protocol provides a method for choosing to use different curves within some pre-defined set that is known beforehand by clients and access points.

To ensure privacy on the network once a secure connection is established, the WPA-CCMP encryption standard and packet format will be used. This provides for 128 bit security and message integrity. The WPA-CCMP privacy standard is chosen because the protocol is widely accepted and supported by most modern devices.

The following list defines the specific parameters that are involved in the design of the protocol and give the notations that will be used for the remainder of this work:

$E$ – This is the notation for the elliptic curve being used. The specific curve used in this work is defined over $F_{3^{509}}$. Points in on the elliptic curve are denoted as being in the group $E(F_{3^{509}})$. For functionality of this protocol, $E$ will be an index referring to a pre-defined set of elliptic curve parameters that are known beforehand.

$e$ – This denotes the pairing function, which gives the following mapping:

$$E(F_{3^{509}}) \times E(F_{3^{509}}) \longrightarrow F_{3^{509} \times 6}$$ (3.1)

$s$ – This is the IBE master secret key that is known only to the access point. $s$ is a 128 bit integer.

$P$ – This is a random point on the elliptic curve. $P$ is chosen by the AP and is part of the public parameters of the IBE system.
$Q$ - This is another point on the elliptic curve and is calculated by $Q = sP$ on the AP. $Q$ is also a public parameter.

$g$ - This is calculated by $e(P, P)$ on the AP. $g$ is another public parameter of the system.

$H1$ - This is a hash function that converts a binary MAC address to a 128 bit integer, which is suitable for elliptic curve scalar multiplication. MD5 might be used for $H1$. However, a simpler and faster hash function could be used because the security of the protocol does not rely on $H1$ being one-way.

$H2$ - This is a hash function that converts an element of the extension field $(F_{3^{509 \times 6}})$ to a 128 bit integer. This hash should be one-way. MD5 could also be used in this case despite its weaknesses. Possible birthday attacks are not an issue for the usage of this hash.

**Public Keys** - The public key of each node involved in the protocol is a 128 bit integer calculated by $H1$(node MAC address). The MAC address for each node works well for the public key since MAC addresses are unique. MAC addresses are already required for communication at the link layer, so no overhead is incurred to share public keys. The public key for the access point will be denoted as $a$. The public key for each network client will be denoted as $c_i$.

**Private Keys** - The private key of each device is an elliptic curve point calculated by the AP as $\frac{1}{s+h}P$, where $h$ is the node public key. The value $\frac{1}{s+h}$ is calculated over the finite field $\mathbb{Z}_p$, where $p$ is a 128 bit or larger prime integer. The private key for the AP will be denoted as $A$, ($A = \frac{1}{s+a}P$). The private keys for each client will be denoted as $C_i$, ($C_i = \frac{1}{s+c_i}P$).
3.2 Functionality Without Authentication

One of the main features of this protocol is the option to provide privacy without authentication. As mentioned in Chapter 1, this feature could be used for public wireless networks to eliminate the security risk of passive traffic sniffing. In this protocol, an IBE method adapted from [20, 24] is used to establish a shared session key between each wireless client and the AP. This method is not as concise as some other IBE protocols, but it has the advantage that the client does not have to perform a costly pairing calculation for key setup. The client does not need to hold any IBE parameters or network secrets prior to connecting to the AP. Since the 802.11 wireless protocol already provides for the exchange of node information via probe requests, we will use this mechanism to retrieve IBE parameters from the AP. Once a session key has been established and shared, the remainder of the communication between the AP and the client is encrypted using the method of WPA-CCMP. The following steps describe the steps taken in the protocol to provide privacy without authentication.

1. The AP periodically sends a beacon frame as specified in 802.11 protocol to make clients aware of its existence and security requirements.

2. A client wishes to connect to the AP. If the client does not have the parameters for the AP cached, it sends a probe request to retrieve the IBE parameters for the network.

3. The AP responds with a probe response containing the necessary IBE parameters: \( \langle E, P, Q, g \rangle \)

4. The client now begins the connection process. It sends an association request to the AP, indicating that it wishes to connect using the supported unauthenticated privacy method.
5. The AP sends an association response to the client, indicating the client can go ahead with session key generation.

6. The client generates two random 128 bit integers, \( w \) and \( t \). \( t \) is the eventual session key. The client also calculates the AP public key, \( a = H1(\text{AP MAC address}) \). Note that the client already possesses the MAC address of the AP from the initial beacon frame.

7. The client then generates two more values: \( M1 = w(Q + aP) \) and \( M2 = t \oplus H2(g^w) \) (\( \oplus \) denotes the xor operation). \( M1 \) and \( M2 \) are sent to the AP in an authentication management frame.

8. The AP retrieves the session key \( t \) by calculating \( t = H2(e(A, M1)) \oplus M2 \).

9. AP sends an authentication successful message back to the client. Although the client is not actually authenticated in a cryptographic sense, authentication frames are used to signal that the client has permission to join the network with the secure channel.

10. The client and AP are now ready to commence normal data communication. The remainder of the communication is encrypted using the shared session key and WPA-CCMP.

Note that the AP is able to successfully retrieve \( t \) because of the bilinearity property if the IBE pairing function:
\[ e(A, M1) = e \left( \frac{1}{s + a} P, w(Q + aP) \right) \]
\[ = e(P, Q + aP)^{\frac{w}{s + a}} \]
\[ = e(P, (s + a)P)^{\frac{w}{s + a}} \]
\[ = e(P, P)^{w} = g^{w} \quad (3.2) \]

and

\[ H2(g^{w}) \oplus M2 = t \quad (3.3) \]

Figure 4 illustrates the protocol steps and the messages that must be sent between the AP and the wireless client. In this way, a client is able to generated and share a random session key with the AP without having any prior keys or parameters from the AP. This session key is then used to derive the values needed to encrypt packets with AES for WPA-CCMP privacy. Since each client generates its own random key per session, the data sent over the encrypted channel can only be read by that client and the AP. The security of this key establishment protocol without authentication is discussed further in Section 4.1.1.

### 3.3 Functionality With Authentication

If authentication is required by the network, it can also be achieved by the IBE protocol without the reliance on upper layer methods. It is undesirable to use a master network password that is shared among users as in WPA-PSK. Instead, we want each client authentication to be independent of other clients. Independent authentication is also a part of WPA-EAP. However, the proposed IBE protocol has the advantage of not requiring a user database and authentication server. To provide this functionality, IBE private keys will be generated for each network client and
connections will be authenticated using this private key. IBE has the feature that private keys have an inherent authenticity in that the AP generates the private keys using the IBE master secret. In contrast, clients generate their own private keys in traditional public key cryptography systems, resulting in the need for certificate verification by trusted outside authorities [16].

3.3.1 Initial Connection

Though IBE private keys provide an effective means for authentication, we must still have a way for each client to make an initial connection to the AP and securely retrieve its private key for the future. For this step we must involve a system administrator or some outside system to provide temporary user IDs and random passwords

Fig. 4. Sequence diagram of protocol without authentication
for the first connection by each client. These user ID and password pairs can be stored on the AP until the client connects and receives its private key for the first time. Neither the temporary passwords or the IBE private keys need to be stored permanently on the AP. Figure 5 shows the steps taken when a client connects and authenticates to an AP for the first time, after the temporary id/password pair has been set up and issued to the client.

Fig. 5. Sequence Diagram for first-time client connection and authentication
The following steps are involved in the first-time connection.

1. The client and AP establish a secure connection in the same way as the unauthenticated protocol above.

2. The client sends the temporary user ID to the AP.

3. The AP looks up the user ID and password hash in its table. It then sends the password hash to the client. This step is performed in order to authenticate the AP to the client.

4. The client authenticates the AP by calculating the hash of its password and comparing it to the hash received from the AP. If the hashes match, the client proceeds to send the temporary password to the AP. If the hashes do not match, the client sends a disassociation frame to terminate the connection.

5. The AP receives the temporary password from the client and calculates the hash. If the hash matches what is stored in its database, the AP proceeds. Otherwise, the AP sends a disassociation frame to terminate the connection.

6. The client and AP have now been authenticated. The AP calculates the new client private key: $C_i = \frac{1}{s+c_i}P$. The AP sends the private key to the client in an authentication success frame. The temporary user ID and password is also deleted from the database.

7. The client receives and saves its private key for future connections. The client also saves the public IBE parameters for the AP in order to avoid the need for another parameter probe request in the future. Normal data communication can now begin and is encrypted over the authenticated secure channel.
3.3.2 Subsequent Connections

Once the client has performed the initial connection and authentication, all subsequent connections will be authenticated using the IBE private key. Since the IBE private keys are generated using the master secret that is held by the AP, both the client and AP can be authenticated to each other. In this protocol, a key exchange similar to the Diffie-Hellman method is used [21]. Unlike a traditional Diffie-Hellman exchange, the properties of IBE ensure that keys can only be agreed upon between an authentic AP and an authentic client. Once the key setup steps are completed, a challenge and response mechanism is used to authenticate the client and allow it to connect. The security of this key exchange and authentication mechanism is studied further in Section 4.1.2. Figure 6 shows the protocol steps for client connection and authentication after the client has obtained its private key.

The steps involved in the protocol are as follows:

1. The AP periodically sends beacon frames to make clients aware of its existence.

2. The client sends an association request to the AP, indicating that it already possesses its private key.

3. The AP sends and association response indicating that the client can go ahead with authentication.

4. The client generates a random 128 bit number $p$ and calculates $p(Q + aP)$. The client sends $p(Q + aP)$ to the AP in an authentication request frame.

5. The AP receives the authentication request, generates a random 128 bit $r$, and calculates $r(Q + c_i P)$. The AP sends $r(Q + c_i P)$ back to the client.
Fig. 6. Sequence Diagram for subsequent client connection and authentication
6. The AP and the client simultaneously calculate the session key:

   The client calculates \( H_2(e(C_i, r(Q + c_i P))^p) \)

   The AP calculates \( H_2(e(A, p(Q + aP))^r) \)

Both the client and the AP are able to securely share the 128 bit session key because of the relation:

\[
e(C_i, r(Q + c_i P))^p = e \left( \frac{1}{s + c_i} P, r(sP + c_i P) \right)^p
= e(P, (sP + c_i P))^{r_{p_{c_i}}}
= e(P, P)^{rp} \quad (3.4)
\]

and

\[
e(A, p(Q + aP))^r = e \left( \frac{1}{s + a} P, p(sP + aP) \right)^r
= e(P, (sP + aP))^{r_{p_{a}}}
= e(P, P)^{rp} \quad (3.5)
\]

7. Once the session key is calculated, the AP sends a random data payload to the client as a challenge.

8. The client receives the random challenge, encrypts it using the session key and WPA-CCMP, and returns it to the AP.

9. The AP also performs encryption on the challenge data using the session key.
   It compares the result to the challenge response received from the client to determine if the client has arrived at the correct session key. If the client does have the correct session key, it is authenticated and the AP sends an
authentication success frame. Otherwise, the AP sends a disassociation frame to terminate the connection.

10. The client and AP can now commence normal data communication. All further traffic is encrypted using WPA-CCMP standards and the shared session key.

In this way, a client can achieve authentication while establishing a secure channel to the network. In this scenario, little or no user interaction is required on the client or AP side. The only requirement is that the client possesses the private IBE key that it was given during the initial connection. It should also be noted that no previous information about the client needs to be stored on the AP. All client information needed by the AP for this protocol can be calculated from the MAC address of the client.

3.4 Key Management and Changes

3.4.1 New Clients

As mentioned above, adding new clients to an authenticated network will require some form of initial interaction. In the simplest case, a system administrator could be charged with generating user ID and random password combinations for each new client. The user ID numbers may be random or may be an employee or student ID. For added security, the MAC address of the new client could also be obtained and stored along with the user ID/password. The AP could then check whether the temporary password is being used with the correct MAC before allowing first-time authentication. However, this would of course increase the difficulty of the initial setup. Temporary passwords should be stored as a SHA-224 hash in keeping with the NIST standard for 128 bit security [19]. The password hashes and IDs will be uploaded to the AP via some secure channel. Since the temporary passwords are no
longer needed after the first-time client connection and authentication, they can be automatically deleted to free up space on the AP. In this way, no central user database or authentication server is needed beyond the AP. For larger wireless networks, it may be desirable to have some kind of automated system for the setup and distribution of the initial temporary keys. The design of such a system is beyond the scope of this work, however it is important to note that this system would not have to be working at all times for the network to function.

3.4.2 IBE Parameter Changes

For security reasons, it will likely be desirable to generate new master keys and IBE parameters on a periodic basis. A change in either the master key or the public parameters will invalidate the private keys and make clients unable to authenticate. It would be infeasible to treat all clients as new users and redo the first-time authentication steps after each key change. In order to facilitate key changes and parameter changes more efficiently a key timestamp mechanism is added to the protocol. With this mechanism, the AP stores one or more sets of old parameters. For each old set, only \(\langle E, s, P \rangle\) need to be stored. During the initial authentication, the AP attaches a timestamp to the client private key. The client stores this timestamp along with its private key. Then, every time the client wishes to join the network, it sends its timestamp to the AP in the association request. The AP can use this timestamp to determine whether the requesting client possesses a private key generated with current parameters or with an old set. If the clients private key is current, the authentication steps proceed normally. If the timestamp shows that the private key was generated with an older set of parameters stored on the AP, the client will be allowed to authenticate using the old set of parameters. Once authentication is complete and a secure channel is established, the AP generates a new timestamp and client private
key and sends it to the client. The client must then removed its old private key and store the new one. The client must also request and store the new set of parameters from the AP. The lifetime of each set of IBE master key and parameters and the time period during which the old parameters are valid should be dictated by the company or network policy. If a client possesses a private key that is older than any set of parameters on the AP, it simply must be treated as a new user and go through the first-time authentication process.

3.5 Operation on Multiple Access Point Networks

Many wireless networks that use authentication will involve more than just a single access point. As discussed in the literature review, many access points can be joined together over a wired network to form an ESS. To the user, the ESS appears as a single wireless network even though multiple access points are involved. For this type of network, the authentication should behave the same from the user perspective regardless of the specific AP the client is associating with. In addition, clients should be able to migrate between different access points on the ESS as seamlessly as possible. WPA-EAP provides this functionality by storing all authentication information on a central server. When a client sends an authentication request to a particular AP, the AP contacts the central server over a secure wired channel to get the client authentication information. On a large network with many users moving on and off the network, the central server may become a bottleneck since it must handle authentication requests from all APs. Furthermore, the authentication server becomes a single point of failure for the network.

Due to the small storage requirements of IBE parameters, the protocol in this work can achieve multi-AP network functionality simply by replicating the IBE parameters over all access points. Here it is assumed that the access points are connected
by a secure wired backbone. A single AP or a system administrator can be tasked with generating IBE parameters and distributing them to all other access points across the wired backbone. In the same way, temporary user IDs and passwords for new clients can be replicated to all access points. In this case, the AP that performs the initial authentication for a new client must send a message to all other access points to delete the temporary password from their own new client database. With this setup, all access points will be able to perform their own authentication functions for clients that connect directly to them. This removes the bottleneck and the single point of failure created by the reliance on a central authentication server. Since all access points possess the same IBE parameters, client migration will still be able to function as normal. When a client moves from one AP to another in the same ESS, the same basic authentication steps are taken as in normal connections. The only difference is that reassociation management frames are used when connecting to the new access point. This causes the new AP to retrieve any frames that are buffered for the client at the old AP after authentication is successful. This functionality is already included as a part of the 802.11 wireless protocol [23]. In order for migration to be seamless for the user, the IBE authentication steps must be able to complete quickly. An analysis of the time involved in the IBE computations can be found in Section 4.4. An interesting feature that could be explored for extra security would be the use of a secure splitting scheme to store the master secret among all access points. This would make the network resistant to a compromised AP since the master secret is not known by any single AP. However, performance would be decreased because many or all access points would need to participate in order to authenticate each client.
3.6 Frame Format

All messages in key establishment and authentication will be passed using 802.11 management frames. For the development of the protocol, we assume a maximum transmission unit of 1500 bytes in keeping with the common Ethernet MTU [12]. The communication analysis in Section 4.3 shows that this payload size is more than sufficient for all protocol messages that must be passed. In order to facilitate this IBE protocol, a 4 bit control code is added at the beginning of the data payload. The management frame type along with the control code is used to determine the protocol message type. Figure 7 shows the diagram for the frame format. The following list shows control codes that can be used for each protocol message type. Of course these codes could be changed if needed depending on the implementation.

0000 IBE parameters request. This message occurs in a probe request frame.

0001 IBE parameters response. This message occurs in a probe response frame and contains the IBE public parameters for the AP.

0010 Association request with no authentication. This message is used for the operation of the protocol when authentication is not being used. The message is contained within an association request management frame.

0011 Association response with no authentication. This message is sent by the AP to the client to notify the client to generate a session key to be used in the
protocol without authentication. The message is contained within a association request management frame.

**0100** Host generated session key. This message is used to pass the session key from the client to the AP for the protocol without authentication. The message is contained in an authentication management frame.

**0101** First time authentication, user ID. This message is sent from the client to the AP and contains the temporary user ID used for first time authentication. The message is contained within an authentication frame.

**0110** First time authentication, password hash. This message is sent from the AP to the client and contains the hash of the password corresponding to the previously sent user ID. This message is used to authenticate the AP to the client and is contained within an authentication management frame.

**0111** First time authentication, temporary password. This message is sent from the client to the AP and contains the temporary password used for first time authentication. The message is contained within an authentication management frame.

**1000** Association request with authentication. This message is sent to the AP from a client to begin an authentication session for a client that already possesses its private key. The message is sent using an association request management frame and should contain the timestamp of the private key (see Section 3.4.2).

**1001** Authentication agreement. This message is sent from the AP to a client to accept the authentication request from the client and notify the client to go ahead with its calculations for the key generation. The message is contained in an association response management frame.
1010 Key agreement. This message is sent by both the client and the AP to communicate each half of the key setup and authentication information. The data sent in this message is $p(Q + aP)$ from the client and $r(Q + c_iP)$ from the AP. The message is contained in an authentication frame.

1011 Challenge message. This message is used to send the random data challenge from the AP to the client in order to test the client’s authenticity. The message is contained in a authentication management frame.

1100 Challenge response. This message is sent from the client to the AP and contains the encrypted challenge from the AP. The message is contained in the authentication management frame.

1101 Authentication successful. This frame is sent from the AP to the client at the end of connection/authentication whether or not client authentication is being used. This message serves as a notifier that all future communication should be encrypted with the established session key and WPA-CCMP. The message is contained in an authentication management frame.

Beside the 4 bit control code, a 4 bit pad is added to the frame in order to achieve byte alignment. These 4 bits can also be used for other protocol fields in the future if needed. This concludes the design specifications for the IBE protocol for wireless network security. The remaining portion of this paper discusses and analyzes the performance and security of this new protocol.
CHAPTER 4

PROTOCOL ANALYSIS

In order to validate the new IBE wireless security protocol designed in this work, this chapter provides an analysis of protocol performance. The cryptographic security of the protocol is covered first, considering both the authenticated and unauthenticated versions of the protocol as well as resistance to some possible attacks. Secondly, the overhead of the protocol is discussed. The overhead analysis consists of storage, communication, and computation overheads. The results of the overhead analysis show that this protocol is a viable solution for wireless security and is also suitable for hand-held and limited-resource wireless devices.

4.1 Security

It should first be noted that an attacker may have access to a portion of the network and protocol information due to the nature of wireless networks. To discuss possible attacks on the protocol, it is assumed that the attacker has obtained or is able to obtain all information that is passed across the network. For any wireless network, this includes the MAC address of the AP and the MAC addresses of all clients. It is also important to realize that an attacker can easily spoof the MAC address of any client. From the protocol designed here, the attacker can easily obtain this set of information: \( \langle E, P, Q, g, a, c_i \rangle \). However, these parameters are already considered public to the protocol and, as shown below, will not aid an attacker in capturing a session key or feigning authentication.
4.1.1 Protocol Without Authentication

For the functionality of the protocol without authentication, we are not concerned with the authenticity of either the client or the AP. However, we do wish to maintain privacy for each client connected to the network. To this end, the session key $t$ must be unavailable to any other party beside the AP and the respective client. In the communications necessary for this protocol, $M1 = w(Q+aP)$ and $M2 = t \oplus H2(g^w)$ can be captured from the network by an attacker. In addition, the attacker could easily calculate $(Q+aP)$. The exclusive or of $t$ with $H2(g^w)$ is essentially a one-time pad. In order for an attacker to obtain $t$, he must obtain $g^w$, or just $w$ since $g$ is already public. Now the attacker does know $w(Q+aP)$ and $(Q+aP)$, but $w$ cannot be obtained from these values due to the ECDLP. As mentioned in the literature review, this ECDLP could be converted to a DLP in the extension field $F_{3^{59 \times 6}}$. However, the size of the extension field still makes the DLP sufficiently infeasible to solve. $g^w$ could also be obtained by $e(A,w(Q+aP))$, but this would require access to $A = \frac{1}{s+a} P$. Although the attacker does know $a$ and $P$, he cannot obtain $A$ because the master secret $s$ is known only to the AP. Furthermore, $s$, $t$, and $w$ have $2^{128}$ possible values, making a brute force attack infeasible. In conclusion, an attacker cannot likely obtain $t$ from any information communicated across the network in this protocol.

Assuming that an access point can only allow one instance of a particular MAC address to be connected to the network at a time, it may be possible for an attacker to perform a denial of service attack against a particular client. As mentioned previously, an attacker can obtain the MAC of another client and log on to the unauthenticated network using the stolen MAC. This creates a denial of service because the legitimate client cannot log on or establish a session key with the AP as long as the attacker remains connected. However, privacy is not breached in this scenario due to the fact
that session keys are randomly generated by the client per association.

4.1.2 Protocol With Authentication

When discussing the security of this IBE protocol on networks where authentication is desired, we must consider both AP authentication and client authentication. Furthermore, both the initial authentication with temporary passwords and subsequent authentications using IBE private keys must be taken into account.

The network access point must be able to authenticate itself to the client in order to prevent an attacker from pretending to be an AP and establishing a fake secure channel with clients. This is especially true for the initial client connection and authentication because an unauthenticated AP could trick a client into sending its temporary initial password, thus giving an attacker a means to access the legitimate network. It is for this reason that the AP must first send the temporary password hash stored in its database to the client upon an initial connection. This is seen in step 3 of the first-time connection protocol described in Chapter 3. By doing this, the client can verify that the AP does possess the correct temporary password that was created by the system administrator or other outside system. Only an authentic AP on the network could possess this information, and the client will not send the temporary password to the AP unless this verification is successful. For subsequent connections, the AP is authenticated to the client by its possession of the IBE master secret $s$. Without $s$, an AP will not be able to generate its own private key or retrieve the session key calculated in the protocol steps for subsequent authentication. The reasons for this are shown below in the discussion of possible attacks. If the AP is not able to calculate the session key, no data from the client can be decrypted, so no information will be leaked.

Client authentication is also achieved on the first connection using the temporary,
first-time password. Once the client has determined that the AP is authentic, it will send the temporary password to the AP over the encrypted connection that has already been established. The AP can then compare the received password with its database to determine if the client is authentic. This of course relies on some outside secure method for distributing temporary passwords to first-time clients. Subsequent authentications require the client to possess a valid IBE private key and the MAC address corresponding to that key. The authentication of clients is validated by using the challenge-response mechanism described in the subsequent authentication protocol. Using this method, the AP can determine whether the client has been able to calculate the correct session key, thus proving the authentication of the client. Once again, the validity of an IBE private key is contingent on the master secret $s$.

Many possible attacks exist for which this IBE protocol must be able to resist. Below is a description of likely attacks to the network and how this protocol proves to be resilient to these attacks.

**Rogue AP** As mentioned previously, an attacker could insert an AP that claims to be a part of the legitimate ESS. In this case, the attacker can easily obtain the public parameters $⟨E, P, Q, g⟩$, and the IBE public keys corresponding to any MAC address. However, the rogue AP cannot obtain the master secret $s$, which is known only to authentic access points. Furthermore, the rogue AP does not have the database of temporary password hashes used for first-time authentication, so it will not be able to authenticate to itself to any first-time clients. For subsequent connection authentication, assume that the rogue AP possesses its own secret $k ≠ s$, with which it has generated its own private key $A_k$. During the key generation phase of subsequent authentication, the client
calculates:

\[ e(C_i, r(Q + c_iP))^p = e(P, P)^{rp(s+c_i)_{s+c_i}} \]  \hspace{1cm} (4.1)

However, the AP calculates:

\[ e(A_k, p(Q + aP))^r = e(P, P)^{rp(s+a)_{s+a}} \]  \hspace{1cm} (4.2)

Therefore, the rogue AP is unable to retrieve the correct session key. Even if the AP falsely tells the client that the authentication is successful, it will not be able to retrieve any data encrypted by the client with the session key.

**Authentic client private key used with wrong MAC address** If an attacker somehow manages to steal the private key, \( C_i \), from a client without obtaining the client’s MAC address, the IBE protocol will not allow the attacker to authenticate with the network. In this case, the MAC address used by the attacker will correspond to some public key \( c'_i = H_1(\text{attacker MAC address}) \). During session key generation, the AP will calculate:

\[ e(A, p(Q + aP))^r = e(P, P)^{rp(s+a)_{s+a}} \]  \hspace{1cm} (4.3)

However, the client calculates:

\[ e(C_i, r(Q + c'_iP))^p = e(P, P)^{rp(s+c'_i)_{s+c'_i}} \]  \hspace{1cm} (4.4)

which does not result in the correct session key. Consequently, the client will not be able to correctly respond to the challenge from the AP and will be disassociated from the network.

**MAC of an authentic client with an invalid private key** In order for an IBE private key to authenticate correctly with the network, it must have been generated with the correct master secret \( s \). Even if an attacker possesses the MAC
of an authentic client, it cannot connect to the network with an invalid private key. Assume that the attacker has generated its own IBE private key, $C'_i$, using the public parameters and some secret key $k \neq s$. During the session key generation phase, the AP calculates the same value shown in Equation 4.3. However, the client calculates:

$$e(C'_i, r(Q + c_i P))^p = e(P, P)^{rP(s+c_i)}$$

which does not result in the correct session key. Again, the client will not be able to successfully respond to the authentication challenge from the AP.

**Invalid MAC address and invalid private key** The proof of security in this case clearly follows from the previous two cases.

**Rogue/Hijacked Client** In some cases, an attacker may be able to capture a client that possesses an authentic IBE private key. Another scenario resulting in the same situation is an inside attack, where a previously authenticated user decides to attack the network using an authenticated client. In this case, the client will still be able to access the network, but will not be able to sniff traffic from any other clients of the network due to the randomly generated session keys. Furthermore, even an authentic client cannot obtain private keys from other clients without obtaining the master secret $s$. Even though an authentic client does possess $C_i, c_i$, and $P$, it will not be able to obtain $\frac{1}{s+c_i}$ or $s$ due to the ECDLP.

**Hijacked AP** In the event that an AP is physically hijacked by an attacker, it may be possible for the attacker to retrieve $s$ and any saved temporary password hashes, breaching the security of the network. The method for secure storage of secrets on the access point is beyond the scope of this paper. However, it is
likely that such a hijacking would be easily detected on a single AP network. On a multiple AP network, a secure splitting scheme could be used to store \( s \) and the temporary hashes among all the access points. If a splitting scheme is used, the compromise of an AP would not result in a security breach as the secrets cannot be obtained from any one AP.

4.1.3 WPA-CCMP Secure Channel

In order to provide security once the secure channel between the AP and client has been established, the widely accepted WPA-CCMP (WPA2) protocol is used. The 128 bit AES encryption algorithm in cipher block chaining mode used for this protocol provides a security level recommended by NIST through the 2030 [19]. Furthermore, the 48 bit IV used in encrypting each packet has a theoretical lifetime of 100 years at the maximum wireless transmission rate [8]. The use of this IV effectively provides a unique encryption key for every frame transmitted on the network. More details on WPA-CCMP can be found in Chapter 2. Further analysis of WPA security mechanisms can by found in [8, 13].

4.2 Storage Overhead

In order for this protocol to be carried out entirely on access points and clients without help from outside servers, it should require only a small amount of storage space. Here, we analyze the data storage requirements for this IBE protocol on both access points and clients. The storage requirements of the programming needed for this protocol is not analyzed in this work.
4.2.1 Access Point

We will first discuss the storage requirements for access points. Each access point must store the elliptic curve attributes for multiple different curves that are to be used in the protocol. As given by the index, $E$, in the protocol, there can be a total of 256 different elliptic curves stored and chosen from. However, there will likely be a much smaller number of stored curves in practical use. The following list shows the attributes that must be stored for each elliptic curve [7]:

- The elliptic curve equation
- $f(x)$ - The polynomial field modulus
- $a, b$ - Coefficients of the curve equation
- $G$ - The base generator point on the curve
- $n$ - The order of $G$
- $h$ - Cofactor of $n$. $n \times h = \text{total order of the curve}$
- $e$ - Embedding degree

The size of each of these attributes can vary depending on the type and security level of each elliptic curve. It is likely that the largest element of these attributes is $G$, which is an element of the base field and a point on the elliptic curve.

Once a particular curve is chosen the access point must also store IBE parameters for the operation of a specific instance of the protocol. Here we use storage sizes specific to the elliptic curve over $F_{2^{509}}$ used throughout this work. It is first important to discuss the storage of points on the elliptic curve. Since the curve is defined over a ternary field, each of the 509 ternary digits in the field are represented with 2 bits.
Therefore, an element of $F_{509}$ is stored with 1018 bits. Furthermore, a point on an elliptic curve consists of $(x, y)$, where $x$ and $y$ are both elements of the base field. To avoid the need for two field elements to be stored for each curve point, we can take advantage of the fact that an elliptic curve has two possible $y$ values for a given $x$ value. Using this information, we can simply store an extra bit that is used to choose between the two $y$ values [27]. Using this method, an elliptic curve point can be stored with $1018 + 1$ bits, which rounds to 128 bytes. $g$, which is an element of $F_{3509 \times 6}$ also requires 2 bits for each of its 3054 ternary digits. This results in a space requirement of 6108 bits which rounds to 764 bytes. Table I shows the parameters that must be stored on the AP.

In addition to the IBE parameters, the AP must also store the database of temporary user IDs and passwords for new clients. We will assume that each user ID is 10 characters long. Along with the SHA-224 hash used to store passwords, a single ID/password combination requires 38 bytes. However, since these combinations are removed from the database after first-time authentication, there should never be a considerably large number of ID/passwords stored on the AP.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$</td>
<td>1</td>
</tr>
<tr>
<td>$s$</td>
<td>16</td>
</tr>
<tr>
<td>$P, Q$</td>
<td>128 each</td>
</tr>
<tr>
<td>$g$</td>
<td>764</td>
</tr>
<tr>
<td>$A$</td>
<td>128</td>
</tr>
<tr>
<td>Total</td>
<td>1165</td>
</tr>
</tbody>
</table>

Table I. AP parameter storage requirements
All other values needed in the operation of the protocol can be calculated on the fly by the access point. However, it would improve speed to cache some parameters that are commonly used. The first value that should likely be cached is the IBE private key for the AP, $\frac{1}{s+a} P$. This is simply a point on the elliptic curve, which requires 128 bytes. Another value to aid in computation is $(Q + c_i P)$ corresponding to recent or frequent clients. This is also an elliptic curve point.

If storage is severely constrained, a method given in [27] can be used to further reduce the size of elliptic curve points. Consider a string of 5 ternary digits. These digits can represent the decimal values 0 through 242, which can also be represented by 8 bits. Therefore, every 5 ternary digits can be mapped to 8 bits. Using this method, the storage requirement for an elliptic curve point can be reduced from 128 bytes to 102 bytes. The parameter $g$ can also be reduced from 764 bytes to 611 bytes. This does, however, incur an extra computation overhead. For now, we assume that storage space is not limited to the point of justifying this compression method.

### 4.2.2 Client

As in the AP, the client must also store curve attributes for each available elliptic curve in the protocol. Beyond this, the client only needs to store its own private key, $C_i$, which is an elliptic curve point. All other values needed in the protocol can either be retrieved from the AP or calculated. However, to avoid excess communication, the client should also cache $\langle E, P, Q \rangle$. This is also necessary for functionality of the IBE parameter changing protocol described in Section 3.4.2. Furthermore, the client should store the point $(Q + aP)$ to reduce computation on future authentications. Table II gives a summary of parameters that need to be stored on the client.
Table II. Client parameter storage requirements

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td>128</td>
</tr>
<tr>
<td>$E$</td>
<td>1</td>
</tr>
<tr>
<td>$P, Q$</td>
<td>128 each</td>
</tr>
<tr>
<td>$(Q + aP)$</td>
<td>128</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>513</td>
</tr>
</tbody>
</table>

4.3 Communication Overhead

The IBE security protocol in this work has a relatively small communication overhead. Each protocol frame contains a control field that constitutes 1 byte, including padding. Only some of the protocol frames contain additional payloads, as described below. Since no upper layer protocols are used, there is no need for many frames to be transferred in order to establish upper layer sessions, such as TCP. The only point in the protocol in which a significant overhead is incurred is during connection setup.

The largest portion of data that must be transferred is the IBE public parameters, $(E, P, Q, g)$. As shown in the above storage analysis, this constitutes a total of 1021 bytes, which can easily fit in one frame. These parameters are only sent when requested by the client via a probe request frame. In addition, clients can cache the parameters to reduce communication during subsequent connections.

4.3.1 Protocol Without Authentication

After the exchange of public parameters, the protocol without authentication requires an exchange of four frames, as seen in figure 4. The only frame that contains a payload during this exchange is the transmission of $M1$ and $M2$. Since $M1$ is a
curve point and $M_2$ is a 128 bit integer, the total payload is 144 bytes. All other frames exchanged during this protocol are control frames and contain no payload.

4.3.2 Protocol With Authentication

The IBE protocol with authentication contains two phases: the initial authentication and subsequent authentications. The initial authentication includes the encrypted channel setup from the protocol without authentication, as well as 4 additional frames. These frames include payloads of the user ID, temporary password hash, temporary password, and the IBE client private key, as shown in figure 5. The most significant payload in this set is the IBE private key, which is an 128 byte curve point. All other payloads will contain only a small number of bytes.

The authenticated key setup for subsequent connections requires a total of 7 frames to be exchanged, as shown in figure 6. Of these 7, only 4 contain payloads: the 2 key setup frames and the 2 challenge-response frames. The key setup frames each contain a 128 byte elliptic curve points. The size of the challenge-response payload can vary, but needs to be at least 16 bytes in keeping with the 128 bit security level. All other frames contain only the control field.

Whether using the protocol with or without authentication, the exchanges described above only take place once per connection. After the session key is established, all following messages transferred via WPA2-CCMP , which incurs 12 bytes of overhead per frame over unsecured 802.11 wireless Ethernet.

4.4 Computation Overhead

Due to the small amount of communication required, computation will likely be the limiting factor on the speed of the IBE protocol. Access points will likely have limited computation power, although they do have a persistent power source. Mobile
devices may have limited computing power and often do not have a persistent power source. Elliptic curve and IBE computations have been shown to perform efficiently enough for devices with very limited computing power [24, 27]. In order to analyze and justify the efficiency of this IBE protocol, the basic computations required for each phase of the protocol are described. In addition, benchmark tests are conducted to estimate the timings of basic ECC and IBE operations. For the overall summary of computation overhead, only elliptic curve and IBE operations are considered, since these operations will easily dominate the computation requirements.

4.4.1 Benchmarks

In order to estimate the time requirements of various computations used throughout this protocol, benchmark tests were conducted using IBE pairing code adapted from [4, 5]. This code is written in C++, with some inline assembly instructions included. The benchmark tests were conducted on a 32-bit Intel Core2 Duo processor at 3.00 GHz. The code was built and run under Windows 7 with Microsoft Visual C++ 2008. Though the processor used is dual core, the benchmarking program was forced to run in a single thread, so no parallelism is used. The elliptic curve and pairing algorithm benchmarked here is the same curve over $F_{3^{509}}$ and $\eta T$ pairing used throughout the development of this work. In order to achieve accurate timings, each basic operation was iterated multiple times and the average computation time was calculated from these iterations. Though the original source code from [5] was only used to estimate the calculation time of the $\eta T$ pairing, it was adapted for this work to time other primitive operations involved in the overall IBE protocol. Table III shows the results of this benchmark.

The processor clock of 3.0 GHz is likely higher than most mobile devices that use wireless networks. In order to obtain timings for a slower processor speed, which
Table III. IBE Computation Benchmarks at 3.0 GHz

<table>
<thead>
<tr>
<th>Operation</th>
<th>Iterations</th>
<th>Average Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC Addition</td>
<td>100000</td>
<td>$2.34 \times 10^{-5}$</td>
</tr>
<tr>
<td>EC Point Triple</td>
<td>100000</td>
<td>$1.60 \times 10^{-6}$</td>
</tr>
<tr>
<td>Multiplication over $F_{3509 \times 6}$</td>
<td>10000</td>
<td>$2.81 \times 10^{-5}$</td>
</tr>
<tr>
<td>Cube over $F_{3509 \times 6}$</td>
<td>100000</td>
<td>$2.34 \times 10^{-6}$</td>
</tr>
<tr>
<td>$\eta T$ Pairing</td>
<td>1000</td>
<td>$6.879 \times 10^{-3}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Iterations</th>
<th>Average Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplication over $F_{3509 \times 6}$</td>
<td>10000</td>
<td>$4.165 \times 10^{-4}$</td>
</tr>
<tr>
<td>Cube over $F_{3509 \times 6}$</td>
<td>100000</td>
<td>$3.603 \times 10^{-5}$</td>
</tr>
<tr>
<td>$\eta T$ Pairing</td>
<td>1000</td>
<td>$1.07297 \times 10^{-1}$</td>
</tr>
</tbody>
</table>

is more representative of the limited-resource devices that should be accommodated by the IBE protocol, the CPU scaling feature of the benchmarking computer was used. With the help of this feature, another set of benchmarks was conducted using the lowest configurable processor frequency of 656 MHz. The results of the same benchmarks conducted at this processor speed are shown in table IV.

Beside the $\eta T$ pairing, elliptic curve scalar multiplication and exponentiation over $F_{3509}$ are the two other operations that play a large role the computation required for this IBE protocol. Using the benchmarks of the basic operations, the upper bounds for these two operations can be determined. The traditional method for efficiently computing elliptic curve scalar multiplication is the binary method [14]. In order
to multiply $xP$, where $x$ is a scalar number, the binary representation if $x$ is used. The result is initialized to 0 and the binary representation of $x$ is read from left to right. If the digit of the binary multiplier is 1, $P$ is added to the result. If the digit is 0, nothing is added to the result. When moving from one digit to the next, the result is doubled. This provides a simple and efficient method for calculating $xP$, assuming an efficient point doubling function is provided. An efficient method for point tripling is given for the particular curve used in this work, as can be seen from the benchmark results. Because of this, a similar ternary method can be used for efficiently calculating $xP$. For this method, the base 3 representation of $x$ is used. When the current digit of the ternary representation is 1, $P$ is added to the result. When the digit is 2, $2P$ is added to the result. When moving from one digit to the next, the result is tripled. To avoid extra multiplications, $2P$ should be precomputed.

For the elliptic curve scalar multiplications that take place in this IBE protocol, all scalar numbers are 128 bits long. In base 3, the scalars require at most 81 digits. This means that a scalar multiplication will require at most 80 point triples and 81 point additions, assuming that $2P$ is pre-computed. As shown in the benchmarks, an efficient function for cubing over $F_{3^{509}}$ is also given. Therefore, a similar method can be used to perform exponentiation in this field. Since this IBE protocol requires an exponentiation by a 128 bit number, at most 80 cubes and 81 multiplications are required, assuming the square of the base element is pre-computed. Table V shows the resulting upper bounds for elliptic curve scalar multiplication and exponentiation over $F_{3^{509}}$. It should also be noted that some speed can be gained in both calculations by using the sliding window method as described in [14]. However, the use of this method requires additional memory to store precomputed values.
Table V. Upper Bounds for Computing EC scalar multiplication and exponentiation over $\mathbb{F}_{3^{509\times 6}}$

<table>
<thead>
<tr>
<th>Operation</th>
<th>Time (3.0 GHz)</th>
<th>Time (656 MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC Scalar Multiplication</td>
<td>$2.023 \times 10^{-3}$ sec</td>
<td>$3.258 \times 10^{-2}$ sec</td>
</tr>
<tr>
<td>Exponentiation over $\mathbb{F}_{3^{509\times 6}}$</td>
<td>$2.463 \times 10^{-3}$ sec</td>
<td>$3.662 \times 10^{-2}$ sec</td>
</tr>
</tbody>
</table>

4.4.2 Parameter Generation

Before the IBE protocol is operational, the access point must generate the IBE public parameters and the master secret for the network. These parameters must also be regenerated when a system administrator determines that a key or parameter change needs to be made. The time needed to generate these parameters is generally not a concern since the AP has a persistent power source, and parameter generation is only done occasionally. However, it is shown that parameter generation can still be achieved quite fast in comparison with other common public key systems. The following list shows the basic calculations that must be done during the generation of IBE parameters:

1. Generate random 128 bit master secret $s$
2. Generate temporary random scalar $r$
3. Calculate $P = rG$ where $G$ is the generator point on the chosen curve
4. Calculate $Q = sP$
5. Calculate $g = e(P, P)$
6. Calculate $a = H_1($AP MAC address$)$
7. Calculate AP private key $A = \frac{1}{s+a}P$
It can be seen that the dominant operations in this process are items 3, 4, 5, and 7, which require a total of 3 elliptic curve scalar multiplications (3,4,7) and 1 pairing computation (5).

4.4.3 Protocol Without Authentication

When functioning in the unauthenticated mode, the IBE protocol designed here has the advantage that the client does not have to calculate a pairing. This is important to save as much energy as possible for resource-limited devices. The client must carry out the following basic operations in order to generate a session key in this mode:

1. Generate random 128 bit \( w \) and \( t \)

2. Calculate public key of AP, \( a = H1(AP \text{ MAC address}) \)

3. Calculate \((Q + aP)\)

4. Calculate \(C1 = w(Q + aP)\)

5. Calculate \(g^w\)

6. Calculate \(H2(g^w)\)

7. Calculate \(C2 = t \oplus H2(g^w)\)

The computationally significant operations in this process are items 3, 4, and 5. These steps require a total of 2 elliptic curve scalar multiplications, 1 elliptic curve addition, and 1 exponentiation over \( F_{3509 \times 6} \). After the first connection, \( a \) and \((Q + aP)\) can be cached for future connections. Therefore, subsequent connections only require 1 elliptic curve multiplication and 1 exponentiation. The total time required for these
operations is less than a single pairing calculation. As for the access point, 1 pairing calculation and 1 128-bit xor operation must be calculated.

4.4.4 Protocol With Authentication

The computation requirements for the protocol with authentication are different between the initial connection and the subsequent connections. For the initial connection, all computations from the protocol without authentication must be done by both the client and the AP. Beyond this, the client is not required to perform any more elliptic curve or IBE operations. The AP must perform one more elliptic curve scalar multiplication in order to compute the private key for the new client.

The subsequent connections with authentication require the most computation overhead of any phase of the protocol. This is due to the fact that IBE methods are used to authenticate both the AP and the client. The client must perform the following basic operations during this phase:

1. Generate 128 bit random \( p \)
2. Calculate \( p(Q + aP) \)
3. Calculate pairing, \( e(C_i, r(Q + c_iP)) \)
4. Calculate \( e(C_i, r(Q + c_iP))^p \)
5. Calculate hash, \( H2(e(C_i, r(Q + c_iP))^p) \)
6. Perform AES encryption of challenge message

The dominant operations in this process are items 2, 3, 4. Here we assume that the client has already cached \((Q + aP)\), since the first-time connection must have already taken place. Therefore, this process involves a total of 1 elliptic curve scalar
multiplication, 1 pairing calculation, and 1 exponentiation over $F_{3^{509} \times 6}$. The access point must perform the following operations during this phase:

1. Generate 128 bit random $r$

2. Calculate public key of client, $c_i = H1(\text{client MAC address})$

3. Calculate $(Q + c_iP)$

4. Calculate $r(Q + c_iP)$

5. Calculate pairing, $e(A, p(Q + aP))$

6. Calculate $e(A, p(Q + aP))^r$

7. Calculate hash, $H2(e(A, p(Q + aP))^r)$

8. Perform AES encryption to verify challenge response

The dominant operations in this process are items 3, 4, 5, and 6. Here we assume that the access point does not have the public key of the client or the value $(Q + c_iP)$ cached, since there is no guarantee that the AP has calculated these values before this point. Therefore, this process involves 2 elliptic curve scalar multiplications, 1 elliptic curve addition, 1 pairing calculation and 1 exponentiation over $F_{3^{509} \times 6}$.

4.4.5 Summary

To summarize, the total estimated upper bounds for each phase of the protocol are calculated using the benchmark results and the basic operations outlined above. Tables VI and VII show these results for the client and the access point, respectively.

In order to provide a reference point for these time estimations, benchmarks were performed on the same machine for the commonly used RSA public key encryption
Table VI. Summary of client computation time bounds

<table>
<thead>
<tr>
<th>Protocol Phase</th>
<th>Basic Ops.</th>
<th>Time (3.0 GHz)</th>
<th>Time (656 MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unauthenticated Protocol</td>
<td>2 EC mult.</td>
<td>$6.532 \times 10^{-3}$ sec</td>
<td>$1.022 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 EC add.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 exp.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auth. Protocol (First-time)</td>
<td>2 EC mult.</td>
<td>$6.532 \times 10^{-3}$ sec</td>
<td>$1.022 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 EC add.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 exp.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auth. Protocol (Subsequent)</td>
<td>1 EC mult.</td>
<td>$1.136 \times 10^{-2}$ sec</td>
<td>$1.765 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 pairing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 exp.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table VII. Summary of AP computation time bounds

<table>
<thead>
<tr>
<th>Protocol Phase</th>
<th>Basic Ops.</th>
<th>Time (3.0 GHz)</th>
<th>Time (656 MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter Generation</td>
<td>3 EC mult.</td>
<td>$1.295 \times 10^{-2}$ sec</td>
<td>$2.050 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 pairing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unauthenticated Protocol</td>
<td>1 pairing</td>
<td>$6.879 \times 10^{-3}$ sec</td>
<td>$1.073 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td>Auth. Protocol (First-time)</td>
<td>1 pairing</td>
<td>$8.902 \times 10^{-3}$ sec</td>
<td>$1.399 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 EC mult.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auth. Protocol (Subsequent)</td>
<td>2 EC mult.</td>
<td>$1.341 \times 10^{-2}$ sec</td>
<td>$2.094 \times 10^{-1}$ sec</td>
</tr>
<tr>
<td></td>
<td>1 EC add.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 pairing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 exp.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

method [21]. These benchmarks were performed using a 3072 bit modulus, which provides approximately the same security strength as the parameters used for this IBE protocol. The code for these benchmarks was developed from scratch and utilizes the standard OpenSSL library. Encryption and decryption is performed using a plaintext that is a random 128 bit number to simulate a session key. It should also be noted that RSA encryption turns the 16 byte plaintext into a 384 byte ciphertext. The operations were iterated multiple times, and the average time per iteration is reported. Table VIII shows these results.

As shown by the results, the performance of the IBE protocol surpasses that of RSA in all areas except encryption. Furthermore, the RSA operations shown are only sufficient to share a session key. Further operations would have to take place in order
Table VIII. Summary of RSA computation times (3072 bit modulus)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Iterations</th>
<th>Avg. Time (3.0 GHz)</th>
<th>Avg. Time (656 MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Generation</td>
<td>60</td>
<td>1.842 sec</td>
<td>23.82 sec</td>
</tr>
<tr>
<td>Encryption</td>
<td>6000</td>
<td>$1.408 \times 10^{-4}$ sec</td>
<td>$2.259 \times 10^{-3}$ sec</td>
</tr>
<tr>
<td>Decryption</td>
<td>600</td>
<td>$8.738 \times 10^{-2}$ sec</td>
<td>1.296 sec</td>
</tr>
</tbody>
</table>

to provide authentication. On the other hand, the IBE protocol is able to provide authentication within the same operations of key generation. These results show promise that this IBE protocol will be able to computationally outperform WPA-EAP protocols, which rely on upper layer encryptions mechanisms such as RSA.
CHAPTER 5

CONCLUSION

Wireless networks are vulnerable by nature, and the growing use of these networks makes security increasingly important. Though the WPA wireless security protocol already exists and is widely used, improvements on this protocol can be made using more recent cryptographic techniques. With this in mind, this thesis provides the design and analysis of a new wireless security protocol using identity based encryption methods to provide privacy and authentication. This new protocol also provides a novel feature in which privacy can be achieved on networks where authentication is not desired. This may prove useful for the growing number of public networks in coffee shops, airports, hotels, and so on. For networks that do require authentication, this new IBE protocol provides improvements over current WPA-PSK and WPA-EAP authentication mechanisms. More specifically, this protocol does not require a master key to be shared among all clients of the network as in WPA-PSK. In regard to WPA-EAP, this protocol eliminates the need for a central authentication server. This simplifies the setup for small, single access point networks and removes the single point of failure created by the authentication server on larger networks. Furthermore, this protocol distributes authentication among the access points of a large network, removing a possible performance bottleneck and ensuring that clients can always authenticate over only 1 hop.

This protocol has also been developed with smaller, resource-limited mobile device in mind, since IBE techniques are widely researched for use in wireless sensor networks. The IBE methods in this protocol use relatively small parameters, in comparison with traditional public key systems, while still providing a high level of security. It can be seen that these IBE techniques are more lightweight and computa-
tionally efficient than the commonly used RSA methods. Furthermore, this protocol does not rely on upper layer network protocols, so frame transmission is kept to a minimum during key set up and authentication.

The protocol designed here also provides a 128 bit security level throughout key setup and authentication. The security level is also maintained during data communications by the use of the WPA-CCMP privacy standard. This level of security is recommended for use by the NIST through the year 2030 [19]. Higher or lower security levels can also be used if desired with minor adjustments to the protocol.

One interesting study to extend this work would be the consideration of different pairing algorithms over different types of elliptic curves. An example would be curves over prime-order fields, $F_p$. The authors of [3] show that curves can be constructed over a 256 bit prime order field with an embedding degree of 12. This work results in curves with 128 bit security and small base fields. The work in [4] shows that pairings can be calculated even faster in this field than the field that has been used in this thesis. The disadvantage of these curves, however, is that they are not supersingular and require the use of the Ate pairing algorithm [10]. This would in turn require a more complicated overall protocol. Benchmark results of many different security levels are also shown in [4]. If a lower level of security is acceptable for a particular network, the use of elliptic curves over smaller finite fields can further increase the speed of this protocol.
REFERENCES


[7] Brown, D. R. Sec 2: Recommended elliptic curve domain parameters, January


[20] SAKAI, R., AND KASAHARA, M. Id based cryptosystems with pairing on


