Wireless LAN Security

CHAN Pak To Patrick

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering

Supervised by

Prof. WEI Keh-Wei, Victor

©The Chinese University of Hong Kong
June 2005

The Chinese University of Hong Kong holds the copyright of this thesis. Any person(s) intending to use a part or whole of the materials in the thesis in a proposed publication must seek copyright release from the Dean of the Graduate School.

香港中文大學
大學
圖書館系統
18 DEC 2006
UNIVERSITY
LIBRARY SYSTEM
THE CHINESE UNIVERSITY OF HONG KONG

Abstract of thesis entitled:
Wireless LAN Security
Submitted by CHAN Pak To Patrick
for the degree of Master of Philosophy
at The Chinese University of Hong Kong in June 2005

In recent years, WLAN becomes a trend and Wi-Fi systems are widely adopted, from home networking to enterprise-based infrastructure networking. The importance of WLAN grows up with its popularity.

One major concern limiting the deployment of wireless LAN (WLAN) is security issues. WEP is the first standardized measure introduced to provide different information security services to WLANs. However, the fact that WEP has many design flaws has motivated the finalization of the amendatory standard IEEE 802.11i in 2004 in order to provide a real secure wireless communications environment.

In this thesis, we survey on security of WEP and IEEE 802.11i standard by mainly focusing the authentication methods provided by the two standards, as well as other related background materials. A new identity-based IEEE 802.11i authentication protocol is proposed for WLAN in infrastructure mode to operate in enterprises. The protocol is based on Au and Wei's identity-based identification scheme, as well as other public-key and symmetric-key cryptographic techniques. We show that the proposed protocol fulfills various security requirements such as mutual authentication and session key derivation, and delivers desired properties to be used in a WLAN such as providing fast reconnect and delegation, overcoming the weaknesses of the WEP authentication and other 802.11i authentication methods.

近年無線區域網絡已成為一種潮流，無論是家居網絡或是企業性的基礎結構網絡，Ｗｉ－Ｆｉ系統均得到廣泛的應用，普及化使無線網絡的重要性日漸提高。

大衆對無線區域網絡安全性的憂慮卻是一項阻礙無線網絡進駐的主要因素。ＷＥＰ是首項提供無線區域網絡訊息保安的標準，可是其設計上的問題，促使另一項標準ＩＥＥＥ８０２．１１ｉ最後定案於２００４年，以修正前作之錯誤，並預期會給予一個真正安全的無線通訊環境。

在本論文，我們會研究ＷＥＰ及ＩＥＥＥ８０２．１１ｉ標準，主要著眼於它們所提供的身份認証方法和有關的背景資料。另外就研究所得，本論文提議一個以身份為公開金鑰的新ＩＥＥＥ８０２．１１i認証協定方式，這協定適用於企業所運行的基礎結構模式無線區域網絡。這協定是根據歐和魏以身份為公開金鑰的認証系統，配合其他公開金鑰及對稱金鑰的密碼學技術設計而成。我們可以看到所提議的認証協定如何符合不同的無線網絡保安要求，例如雙向認証及建立階段金鑰；以及由它所提供的可取特色，例如快速重新連線和支援授權他人建立戶口，這些均能克服ＷＥＰ認証及其他ＩＥＥＥ８０２．１１ｉ認証方式所存在的各種缺點。

Acknowledgement

I would like to express my sincere gratitude to my supervisor Professor Victor K. Wei, who gave me the opportunity to work in the area of cryptography and information security, provided me a good environment for research, and assisted me in different aspects during my researching. These are crucial to my postgraduate studies and future development.

I would also like to thank Professor Joseph Liu for his support and helpful discussions during my studies. Also, working together with my colleges in the Information Security Laboratory is fantastic and I am thankful for their kind encouragement. They are Allen Au, Tony Chan, Sebastian Fleissner, Karyin Fung, Robert Leung, Tsz Hon Yuen and Patrick Tsang. They have provided a relaxing but encouraging atmosphere, as well as Winning Eleven and free meals to me which are just fantastic.

Last but not least, I am grateful to my family for their continuous support in every single way. A heartfelt thanks to my wife and my little baby, who provide me an enriching home and always give me continued love, care and mental support.

Contents

Abstract i

Acknowledgement iii

Contents iv

List of Figures vii

List of Tables viii

1 Introduction 1
- 1.1 Motivation 1
- 1.2 The Problems 3
- 1.3 My Contribution 4
- 1.4 Thesis Organization 5

2 Wireless LAN Security Model 6
- 2.1 Preliminary Definitions on WLAN 6
- 2.2 Security Model 7
  - 2.2.1 Security Attributes 7
  - 2.2.2 Security Threats in WLAN 8
  - 2.2.3 Attacks on Authentication Scheme 10
  - 2.2.4 Attacks on Keys 10
- 2.3 Desired Properties of WLAN Authentication 11
  - 2.3.1 Security Requirements of WLAN Authentication . . . 11
  - 2.3.2 Security Requirements of Session Keys 12
  - 2.3.3 Other Desired Properties of WLAN Authentication . . 12

3 Cryptography 14
- 3.1 Overview on Cryptography 14
- 3.2 Symmetric-key Encryption 15
  - 3.2.1 Data Encryption Standard (DES) 15
  - 3.2.2 Advanced Encryption Standard (AES) 15
  - 3.2.3 RC4 16

3.3 Public-key Cryptography 16
3.3.1 RSA Problem and Related Encryption Schemes 17
3.3.2 Discrete Logarithm Problem and Related Encryption Schemes 18
3.3.3 Elliptic Curve Cryptosystems 19
3.3.4 Digital Signature 19
3.4 Public Key Infrastructure 20
3.5 Hash Functions and Message Authentication Code 21
3.5.1 SHA-256 22
3.5.2 Message Authentication Code 22
3.6 Entity Authentication 23
3.6.1 ISO/IEC 9798-4 Three-pass Mutual 23
3.6.2 ISO/IEC 9798-4 One-pass Unilateral 24
3.7 Key Establishment 24
3.7.1 Diffie-Hellman Key Exchange 24
3.7.2 Station-to-Station Protocol 25
3.8 Identity-Based Cryptography 25
3.8.1 The Boneh-Franklin Encryption Scheme 26
3.8.2 Au and Wei's Identification Scheme and Signature Scheme 27

4 Basics of WLAN Security and WEP 29
4.1 Basics of WLAN Security 29
4.1.1 Overview on "Old" WLAN Security 29
4.1.2 Some Basic Security Measures 29
4.1.3 Virtual Private Network (VPN) 30
4.2 WEP 31
4.2.1 Overview on Wired Equivalent Privacy (WEP) 31
4.2.2 Security Analysis on WEP 33

5 IEEE 802.11i 38
5.1 Overview on IEEE 802.11i and RSN 38
5.2 IEEE 802.1X Access Control in IEEE 802.11i 39
5.2.1 Participants 39
5.2.2 Port-based Access Control 40
5.2.3 EAP and EAPOL 40
5.2.4 RADIUS 41
5.2.5 Authentication Message Exchange 41
5.2.6 Security Analysis 41
5.3 RSN Key Management 43
5.3.1 RSN Pairwise Key Hierarchy 43
5.3.2 RSN Group Key Hierarchy 43
5.3.3 Four-way Handshake and Group Key Handshake . . . 44
5.4 RSN Encryption and Data Integrity 45
5.4.1 TKIP 45

5.4.2 CCMP 46
5.5 Upper Layer Authentication Protocols 47
5.5.1 Overview on the Upper Layer Authentication 47
5.5.2 EAP-TLS 48
5.5.3 Other Popular ULA Protocols 50

6 Proposed IEEE 802.11i Authentication Scheme 52
6.1 Proposed Protocol 52
6.1.1 Overview 52
6.1.2 The AUTHENTICATE Protocol 56
6.1.3 The RECONNECT Protocol 59
6.1.4 Packet Format 61
6.1.5 Ciphersuites Negotiation 64
6.1.6 Delegation 64
6.1.7 Identity Privacy 68
6.2 Security Considerations 68
6.2.1 Security of the AUTHENTICATE protocol 68
6.2.2 Security of the RECONNECT protocol 69
6.2.3 Security of Key Derivation 70
6.2.4 EAP Security Claims and EAP Methods Requirements 72
6.3 Efficiency Analysis 76
6.3.1 Overview 76
6.3.2 Bandwidth Performance 76
6.3.3 Computation Speed 76

7 Conclusion 79
7.1 Summary 79
7.2 Future Work 80

Bibliography 82

List of Figures

5.1 EAP message flow 42
5.2 Four-way handshake message flow 44
5.3 EAP-TLS handshake 49

6.1 WLAN security architecture with proposed protocol implemented 55
6.2 Simplified view of the AUTHENTICATE protocol message flow 59
6.3 Simplified view of the RECONNECT protocol message flow . 61
6.4 Header structure of proposed protocol packets 62
6.5 Structure of each payload value 64
6.6 Message flow of the AUTHENTICATE protocol for a guest . 67

List of Tables

6.1 Comparison of packet payload size between the AUTHENTICATE protocol and EAP-TLS 77
6.2 Comparison of total magnitude of exponents between the AUTHENTICATE protocol and EAP-TLS 78

Chapter 1

Introduction

1.1 Motivation

Wireless communication is essential in the modern world. When you listen to radio, watch TV or make a call with your mobile phone, you benefits from the wireless communication technology. Wireless technology is capable of reaching virtually every location on the face of the earth. Hundreds of millions of people exchange information every day using pagers, mobile phones, portable digital assistants (PDAs) and other wireless communication products. At the same time, not less number of people are enjoying their favorite radio, TV or even 3G broadcast programmes. With tremendous success of wireless telephony and broadcast services, it is hardly surprising that wireless communication is widely applied to the realm of personal and business computing.

Demands on wireless network services rise continuously in these few years. Wireless networks provide great convenience to users. No longer bound by the harnesses of wired networks, an end user can forget about wire connection and directly enters a Local Area Network (LAN) through wireless connection after he enters the coverage of access points. Moreover, users can move around while staying connected so as to access online information and communicate with other people. Besides convenience, wireless networks also reduce wire running cost as well as the fire hazard brought by wires. These factors has raised people interest on wireless LAN (WLAN) in a great extent.

The advance on wireless technology and the drop in price has responded the demands on WLAN services positively. The very beginning official WLAN specifications - IEEE 802.11 only provides 2 Mbps transmission speed. Now we hardly find 802.11 products in computer shops just because they are too slow. New standards IEEE 802.11b, 802.11a and 802.11g provides 11, 54, 54 Mbps transmission speed respectively and they fulfill broadband Internet access environment in many developed cities. Future

standard 802.11n, with the multi-input-multi-output technique, will further push the maximum bandwidth to more than 100 Mbps. On the other hand, the prices of WLAN products, such as access points, WLAN adapter cards and WLAN routers, has decreased to very competitive values. As a result, installation cost of an enterprise WLAN has also dropped. Due to the above reasons, together with the popularity of mobile laptop and PDAs, WLAN networking has become practical in both home networks and enterprise networks and thus it becomes much more popular. Nowadays, WLANs are widely installed at homes, in schools and companies. Adopting WLAN services is a tread.

Due to the importance and popularity of WLANs, we must look at another side of the coin - security. If you are the receiver, the broadcast capability in wireless communication may be very attractive on the first glance. However, this feature sometimes can be a major disadvantage. This is because anyone can easily sniff or even modify the data transmitted in air. You may consider an analogy that you have to send a mail to one of your friends. You type the mail by computer and put the mail into the mailbox of your friend. Physical boundary of wired networks is analogous to the lock of the mailbox. If the mailbox does not have a lock, everyone can access the mail without difficulties. You cannot make sure that the content of the mail is not exposed. Moreover, your friend cannot ensure the mail in his mailbox is really sent by you and without being edited. Undoubtedly, measures should be taken so as to maintain security of the mail, as well as data transmitted in WLANs.

Security is a major concern in designing a WLAN due to the physical nature of wireless communication. To secure a WLAN, the solutions are encryption and authentication. In brief, encryption ensures that only the person knows the secret can obtain the original data. Authentication ensures that the entity communicated is a "good guy" or a "good device" (entity authentication) and the message received is a "good message" (message authentication) from a good entity. In particular, entity authentication can be used to restrict access to WLANs to users with privilege, in order to control the access to sensitive or licensed resources and data available over the network, to avoid the extra cost of providing network resources to unauthorized users, or to prevent outsiders to launch attacks to the wired network directly.

In order to provide security, IEEE 802.11 standard includes a protocol called Wired Equivalent Privacy (WEP). It aims to provide encryption and authentication to WLANs. However, a number of researches discovered that WEP is in fact not secure enough in different aspects [12, 6, 26, 60]. To provide real security as well as better scalability and flexibility, a new generation of security standard for WLANs - IEEE 802.11i was introduced. The new standard specifies the whole security architecture and protocols or framework on various security aspects including encryption and authentica-

tion. It is expected that WLAN products compatible with the new standard will gradually replace the old ones.

1.2 The Problems

Because of the popularity of computers and the Internet, computer and Internet security is widely concerned. Nowadays, the Internet offers a convenient way to access various services. Some of the services require tough security configuration, for example, online shopping, E-banking and private E-mail. Internet security is widely researched, and new technologies protecting our computers are invented continuously. However, while the priest climbs a post, the devil climbs ten. Viruses, hacking, phishing attack and much more are threatening our life. In order to secure our valuable Internet transactions, computer data and privacy, security is always worth studying.

As discussed before, wireless technology becomes more important in today networking and thus WLAN security is what we must pay attention on. In most cases, a WLAN, from a home network to a corporate WLAN involves at least one access point. This kind of WLANs is said to be run in infrastructure mode and its security even more significant. IEEE 802.11i is the latest formal technical WLAN security standard. The draft of the IEEE 802.11i was ratified on last year and thus it is a new standard with many spaces to expose and analyze. While specifying robust key management, message authentication and encryption schemes, IEEE 802.11i allows flexible implementation on user authentication process. Constructing a new entity authentication protocol provides flexibility for deployment of WLANs. One can make the appropriate choice that will work best for a specific WLAN, according to the properties of different protocols.

Flexibility on user authentication process allows many existing authentication protocols to be applied to WLANs following 802.11i security architecture. Examples of the protocols include TLS and Kerberos, and they are evaluated in following chapters. A number of new protocols are also proposed such as LEAP and PEAP. All these protocols have different characteristics and weaknesses. For instance, while symmetric-key based approach suffers from secret key management and distribution problem in large enterprises, a common public-key based authentication protocol TLS has limitation that public key infrastructure (PKI) is required and its efficiency is quite unsatisfactory. Designing an authentication protocol overcoming most weaknesses in existing protocols, and fitting the WLAN environment and IEEE 802.11i standard is a beneficial task.

Identity-based (IDB) cryptography was proposed by Shamir [57] in 1984. In brief, it is a subset of public key cryptography, but unlike typical one, it changes the nature of obtaining public keys by constructing a one-to-one mapping between identities and public keys. It avoids the need of PKI and

simplifies key management because users can use E-mail addresses or network access identifier (NAI) as their public keys. Therefore it is interesting to apply IDB cryptography to WLAN security. Lee et al. firstly mentioned the possibility to apply IDB cryptography in WLAN authentication process [39]. However the protocol proposed by Lee et al. is not secure at all in the WLAN environment because it is unable to prevent replay attack and does not derive session keys. As a result, to develop a brand new secure IDB authentication scheme is of great benefit to WLAN security.

1.3 My Contribution

This thesis focuses on IEEE 802.11 WLAN or Wi-Fi system security. It describes basics of WLAN security and compares the old WLAN security standard WEP with the new IEEE 802.11i standard. It also evaluates various existing and proposed 802.11i authentication protocol. We focus on user authentication rather than message authentication and thus unless explicitly specified, "authentication" appearing in the following chapters means entity authentication.

We have designed a secure user authentication protocol under extensible authentication protocol (EAP) specification, which IEEE 802.11i authentication process takes the advantage of. The proposed protocol suits in IEEE 802.11i compatible WLAN running in infrastructure mode and has the following properties:

- Our solution is identity-based, based on Au and Wei's IDB identification scheme. Users can gain WLANs access with their NAIs without involvement in PKI. This overcomes the weaknesses of using PKI (as discussed in Section 3.4) since IDB cryptography grealy reduces the need for and reliance on public key certificates and certification authorities.

- Our protocol basically consists of two parts. The first one is an authentication protocol executed in the first join of a client, for authenticating the client to the server and deriving session keys. Another is a lightweight protocol for fast reconnect, based on session information created during the first join. The fast reconnect feature is useful in applications requiring seamless connections to networks.

- Additional features are supported by the proposed scheme. For example, delegation mechanism allows local users with appropriate right to offer temporary WLAN access to guests. Identity privacy hides the identities of the users accessing the network.

The goal of the scheme is to provide a secure and efficient entity authentication mechanism to WLANs in large enterprises, universities and other

organizations. The security and performance of the scheme are analyzed in details.

1.4 Thesis Organization

The rest of this thesis is organized as follows:

- Chapter 2 gives an overview on IEEE 802.11 WLAN. Moreover, it provides a short introduction to security in general and includes terminology and taxonomy of different kinds of attacks and threats appearing in WLANs. More importantly, it describes several desired properties of WLAN authentication which the proposed protocol follows.
- Chapter 3 introduces the concept of cryptography. It gives an overview of what are currently possible in the field. In specific, identity base cryptography is covered and its advantages are discussed.
- Chapter 4 points out basic techniques to secure a WLAN. Moreover, details of the WEP protocol and its security flaws are studied. Weaknesses related to the authentication process in 802.11 are emphasized. Shortcomings of different security approaches are discussed so as to show the need of new WLAN security standard.
- Chapter 5 describes the new WLAN security standard IEEE 802.11i. It evaluates several EAP protocols applied to the 802.11i authentication framework and shows deficiency of each of them so that we can know the spaces for improvement.
- Chapter 6 describes the proposed upper layer authentication protocol applied to IEEE 802.11i. The details include the message exchange, the packet format and the mechanisms of addition features. Various security properties of the protocol are discussed according to the requirements of WLAN environment including the EAP security claims in RFC3748 [2]. Performance of the protocol is also evaluated.
- Chapter 7 offers a summary of this thesis and concludes on the future work on the proposed scheme.

□ **End of chapter.**

Chapter 2

Wireless LAN Security Model

2.1 Preliminary Definitions on WLAN

A few preliminary definitions on WLAN must be clarified before proceeding with the later-on sections:

- A local area network (LAN) is a local computer network covering a local area, like a home, an office, an organization or an enterprise.
- A wireless LAN (WLAN) is a type of LAN while data are transmitted in air through electromagnetic wave, instead of wire or optical fiber. Throughout the thesis, the term "WLAN" in fact refers to the wireless LAN following IEEE 802.11 specification [46]
- IEEE 802.11 is a formal standard by IEEE Computer Society defining the physical (PHY) layer and media access control (MAC) layer of WLANs. IEEE 802.11 has several extensions on PHY, such as task group a, b and g.
- Wi-Fi (Wireless Fidelity) originally is an informal name for the IEEE 802.11 WLAN standard. It gives the Wi-Fi Alliance, an industry consortium of 802.11 vendors, its name. A Wi-Fi system now is referred to a WLAN following the industrial standard and passing compatibility tests from the Wi-Fi Alliance.

IEEE 802.11 defines two modes of operation:

- In ad-hoc mode, mobile wireless stations (i.e. client devices) communicate with other stations directly. It is also referred as peer-to-peer mode or an Independent Basic Service Set (IBSS).

- In infrastructure mode, mobile stations communicate with each other by first going through an access point (AP). An AP also acts as a bridge to a wired network, giving its clients access to resources on a larger LAN or the Internet through a gateway.

The proposed protocol is operated on WLANs in infrastructure mode. In order to join a WLAN in infrastructure mode, a client station is mapped to an AP so that other stations on the wired and wireless networks have a means to contact the client station. This mapping is called association. An end station can only be associated with one AP at a time. Association is a three-state process and the three states are:

1. unauthenticated and unassociated
2. authenticated and unassociated
3. authenticated and associated

Only associated stations can access the network. A client station first identifies and AP from the AP's Beacon frame or from response from the AP after the client sends a Probe frame. Then authentication which is described in Section 4.2.1 is executed. Only authenticated stations can associate to the AP in legitimate ways.

2.2 Security Model

2.2.1 Security Attributes

To be able to classify the different security needs of the applications of WLANs, the following attributes needs to be considered.

Confidentiality Confidentiality is the process of keeping the information sent unreadable to unauthorized readers. One way to reach confidentiality is the use of encryption, which is a process of combining a piece of data (plaintext) and a key to produce random-looking output (ciphertext).

Authentication The term authentication is used at different levels in security protocols. The first one is entity authentication and the second is message authentication. The former is also called identification, and it refers to the process to prove a person whom you want to communicate with really has the identity he claims. The latter is the process to prove a message is really sent by the sender you want to communicate with and it is not altered. Throughout our thesis, the term authentication represents entity authentication, unless explicitly specified.

Authorization Authorization is the process to decide whether to grant access to a party whose identity is proved after authentication.

Access control Access control is a much more general way of talking about controlling access to a resource. Access is granted depending on specific rules and access control lists.

Accounting Accounting is the process to measure the resource a user consumes during access after authorization. Often, the combination of authentication, authorization and accounting is referred as AAA.

Availability Availability means whether an application, a service, a client station is available and not blocked by attackers.

Integrity Integrity is the ability of the secure system to guarantee that the received message is in fact the real one that has not been tampered with.

Non-repudiation Non-Repudiation means the ability not to be able to deny sending a message, signing a signature and participating in an authentication process.

Keys are used in many cryptographic algorithms, such as entity authentication, digital signature, encryption and keyed message authentication code. Different kinds of keys are described as follows.

Private key Private key should be kept secret in order to secure the outputs of a cryptographic algorithm against any adversary.

Public key In public-key cryptography, in contrast to the secret private key, a public key is published publicly for others to carry out appropriate "inverse" algorithms such as encryption and signing , corresponding to those controlled by the private key, for example, decryption and verification.

Session key While private and public key are used in long term, a session key is used for encryption and data integrity for a short communication session. Often, a session key is generated by using key establishment algorithm.

2.2.2 Security Threats in WLAN

The main security issue with WLANs is that WLANs intentionally radiate data in air. Even worse, the radiation covers an area that exceed the limits of the area the organization physically controls in many cases. For instance, IEEE 802.11b radio waves at 2.4GHz easily penetrate building walls and are receivable from public area. While setting up rogue devices is much

more difficult in wired LAN, if an attacker wants to carry out attacks to a WLAN in infrastructure mode, he can simply set up a hidden station or access point inside the organization or outside the organization but inside coverage of legitimate- wireless devices. The latter is indeed a piece of cake for any professional adversary.

Inserting bad devices into a WLAN is only preparation of other more dangerous attacks. After so, an adversary can carry out attacks in different approaches, classified into four broad categories: snooping, modification, masquerading and denial of service (DoS).

Snooping Snooping, or eavesdropping, is a kind of passive attack. It involves gathering information from the data transmitted in air. The information is used in compromising the system or as preliminary for attacks in other categories.

Modification An attacker may modify the contents of packets, insert packets to and delete from a WLAN. For modification and deletion, an attacker has to achieve using a physical way, such as sending interfering radio signals. To modify a packet, the attacker has to act as a man-in-the-middle. He either does it on the fly, or store, modify and then send out the modified message.

Masquerading Masquerading is the term used when an attacking WLAN device impersonates a valid device. The attacker attempts to fool the target network into validating it as an authorized device in order to gain access rights.

DoS While the above three extend extra privilege to the attacker, a DoS attack causes damage to target by preventing operation of the network. DoS can be carried in physical level (PHY layer) and higher level (MAC layer, network layer or other upper layers).

By carrying out such attacks, the attacker may achieve the following purposes:

- To obtain free network resources such as Internet access
- To capture sensitive or classified information
- To insert information into an authentic transaction, without the knowledge of legitimate users
- To gain access to the corporate wired LAN through access of WLAN
- To block out legitimate users from accessing the network

2.2.3 Attacks on Authentication Scheme

The ultimate goal for an adversary to attack an entity authentication scheme is, if not to carry out a DoS attack, to impersonate a legitimate party of a network (prover) to the party checking the validity of the prover (verifier). An adversary can do the following specific attacks in order to achieve his goal.

Replay attack An attacker replays the message from a single previous protocol execution, on the same or a different verifier.

Interleaving attack An attacker selectively combines information from one or more previous or simultaneously ongoing protocol executions, including possible origination of one or more protocol executions by an adversary itself.

Reflection attack An attacker sends back the received information from an ongoing protocol execution back to the originator.

Forced delay An attacker relays a message at some later point of time after he intercepted it.

Chosen-text attack An attacker makes use of the challenge-response mechanism of an authentication protocol in an attempt to extract information about the long-term key of a legitimate user. This is because the attacker can choose the challenge during the protocol execution.

2.2.4 Attacks on Keys

If an adversary compromises a session key, confidentiality and data integrity of the corresponding session no longer exist. In order to obtain a session key, an adversary may carry out the following types of attack:

Brute force attack An attacker tries every possible key until he finds a match.

Dictionary attack An attacker guesses elements used to construct the key (e.g. letters and numbers if the key is a human-readable password) so as to reduce the key entropy. Then the attacker uses a huges database containing all the likely keys in order to find a match.

Algorithmic attacks An attacker breaks the encryption algorithm by using information from the algorithm construction and ciphertexts, and possibly the corresponding plaintexts and other session keys. Besides the encryption algorithm, the attacker can also attempt to break the key exchange algorithm by using attack methods similar to those on authentication schemes.

2.3 Desired Properties of WLAN Authentication

2.3.1 Security Requirements of WLAN Authentication

After studying possible attacks from an adversary, we summarize the security requirements of an authentication scheme for WLANs in the following

Mutual authentication A mutual authentication is a bidirectional authentication between two parties in a WLAN. If the WLAN is run in infrastructure mode, one of the parties will be a client station, while another will be a back-end authentication server. As setting up rogue devices in a WLAN is not a complex task, it is a must to validate the identity of another side of communication. Lack of mutual authentication allows an attacker to act as a man-in-the-middle between two parties and gather private information from their communication [44]. Authentication in each direction must be secure against impersonation by any adversary, including those carry out replay, interleaving and forced delay attacks.

Dictionary attack resistance If human-readable password or passphrase is used during the authentication, it should be protected against dictionary attack.

Derivation of strong session keys Authentication alone is usually useless. Derivation of session keys allows the effect of authentication to be extended to the data communication session and thus prevents session hijacking [44]. Confidentiality and data integrity are protected by the session key and this allows the long-term private keys to be kept away from adversary.

Key authentication If an authentication scheme involves derivation of session key, then it either provides implicit key authentication or explicit key authentication in both directions. Mutual implicit key authentication means that both parties are assured that no other entity besides them can possibly ascertain the value of the secret session key. Mutual explicit key authentication means that both parties are assured that the opposite sides has actually computed the key.

Resistance on key-compromise impersonation If long-term private key of an entity is compromised, an adversary can only impersonates that entity, but not the others. This can prevent attack from insiders.

Identity privacy Identity privacy involves hiding of WLAN client's identity. Privacy is one of the human rights which desire to protect. In addition, identity privacy prevents an adversary from linking two authentication executions and obtaining useful information from the relationship of them.

Limited damage from DoS attacks A successful DoS attack should only lead to short blocking of a service. No long term state attributes, such as session keys, are affected by the DoS.

2.3.2 Security Requirements of Session Keys

As key derivation is an important requirement for WLAN authentication scheme, the session keys derived must have appropriate security. Security attributes of session keys are listed as follows:

Key strength Entropy of the session key, namely key length, should be large enough so as to defend against brute force attack and dictionary attack. For session keys, a key establishment algorithm should depends on strong factors in order to generate keys with appropriate effective key strength.

Known-key security Session key in each authentication execution should be unique and independently to others. Compromise of one session does not result in compromise of others from the same or a different client. This reduces damage of a successful attack.

Forward secrecy If long-term private keys of one or more entities are compromised, the security of previously established session keys are not affected. This restricts the scope of damage.

Unknown key-share resilience An entity cannot be coerced into sharing a session key with another entity without the former's knowledge. This prevents the entity from sending sensitive data to an attacker without awareness.

Key control Neither entity is able to force the session key to a preselected value. This prevent either party from gaining benefits from forcing another to use a weak key.

2.3.3 Other Desired Properties of WLAN Authentication

Besides security, a WLAN authentication may provide other desired properties in order to provide convenience to users. Some examples are shown in the following.

Fast reconnect In a WLAN, a client can roam from one access point to another. In order to maintain his connection to the network, he has to reconnect and execute authentication again. As some applications for WLAN, such as Voice over IP (VoIP), require seamless connections to the network, a fast reconnect mechanism is desirable.

Delegation In real life, guests of an organization may have to access the WLAN. In order to reduce the burden for the network administrator to manage temporary guest accounts, mechanism of delegating the right to allow guests to access the WLAN to other WLAN clients is a useful feature.

Simple deployment A simple deployment of a WLAN system reduces the cost of installment of wireless devices and operation.

□ **End of chapter.**

Chapter 3

Cryptography

3.1 Overview on Cryptography

To understand what cryptography is, we may consider a simple cryptosystem. Suppose there are two parties, Alice and Bob. Alice wants to send a message to Bob without an opponent, Oscar, being able to understand the actual message. The message have to travel in an insecure channel, such as a computer network. The original message that Alice wants to send is called the plaintext. Alice shifts each character of the plaintext a number of steps, with wraparound on the alphabet in order to obtain the ciphertext. For example, the plaintext "abc" would be encrypted as the ciphertext "def". In this example, Alice uses the key $K = 3, K \in \mathbb{Z}_{26}$, which indicates number of shifting. By the use of such shift cipher, Oscar cannot obtain the plaintext directly.

The previously shown example is of course not suitable for any real security because Oscar can carry out a brute force attack by trying all possible K to retrieve the key. However, this naive encryption scheme indeed shows some basic ideas on cryptography. Menezes et al. [42] defined cryptography as "study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, entity authentication and data origin authentication". The fundamental goal of cryptography is to address these aspects in both theory and practice by using adequate mathematical techniques in order to defend against malicious activities.

In real life, in order to provide adequate security, secure algorithms and mathematical assumptions are necessary. In the following sections, cryptographic algorithms, protocols and standards on different security context are introduced so that basic components of WLAN security are displayed.

3.2 Symmetric-key Encryption

Symmetrical cryptography is the use of cryptographic solutions which make use of a shared secret (secret key) that both the sender and the receiver need to know in advance. A symmetric encryption algorithm can be described as

$$E_K(x) \rightarrow y$$

$$D_K(y) \rightarrow x$$

where E is the encryption function, D the decryption function, K the secret key, x the plaintext, and y the ciphertext.

This section presents various common ciphers for symmetric encryption algorithms. DES and AES are block ciphers, which encrypt data in blocks, namely portions of plaintexts in a fixed length. RC4 is a stream cipher, which encrypts one symbol of a plaintext each time.

3.2.1 Data Encryption Standard (DES)

DES is a cipher selected as an official Federal Information Processing Standard (FIPS) for the United States in 1976, and which has subsequently enjoyed widespread use internationally. Details of the DES algorithms are covered in [45].

DES has a block size of 64 bits while the secret key length is 56 bits. The overall algorithm structure consists of 16 identical stages of processing, termed rounds, together with an initial and final permutation, respectively termed IP and FP which are inverses of each other. Before the main rounds, the block is divided into two 32-bit halves and processed alternately. The processed two halves are finally rejoined before FP.

In each round, the right hand side of the block is input to the Feistel (F) function with that round's 48-bit subkey, derived from the secret key. The result of the function is XORed with the left hand side of the block. This combination is the new right hand side of the data block while the new left hand side is just the previous right hand side. The F function provides non-linear transpositions and permutations on the input data block.

The major security concern on DES is its relatively short key length. 56-bit key are easy to break using brute force attack by today's measure. Therefore, a new block cipher AES was introduced as a replacement on DES.

3.2.2 Advanced Encryption Standard (AES)

AES [18], also known as Rijndael, is a block cipher adopted by U.S. National Institute of Standards and Technolgoy (NIST) as an encryption standard in 2001. AES has a block size of 128 bits and a key size of 128, 192 or 256 bits, which is much more secure against brute force attack.

AES operates on a 4×4 array of bytes, termed the state. If the key size is 128 bits, expansion of key is done at first. Similar to DES, AES encryption consists of rounds. Number of rounds to execute depends on the key size. For example, 128-bit keys require 10 rounds. Each round consists of four main parts which is shown below.

1. SubBytes - a non-linear substitution step where each byte is replaced with another according to a lookup table
2. ShiftRows - a transposition step where each row of the state is shifted cyclically a certain number of steps
3. MixColumns - a mixing operation which operates on the columns of the state, combing the four bytes in each column using a linear transformation
4. AddRoundKey - each byte of the state is combined with the round key which is derived from the secret key

3.2.3 RC4

RC4 (or ARCFOUR [32]), designed in 1997, is the most widely-used software stream cipher and is used in popular protocols such as Secure Sockets Layer (SSL) and WEP. Key size of RC4 can be 64 or 128 bits while the encryption operations are byte-oriented.

To carry out encryption, RC4 first generates a pseudorandom stream of bits called keystream, using the pseudorandom generation algorithm (PRGA), by making use of a secret internal state consisting of a permutation of all 256 possible bytes and two 8-bit index pointers. The permutation is initialized with the secret key, using the key-scheduling algorithm (KSA). Once the keystream is computed, it is XORed with the plaintext to obtain the resulted ciphertext.

In 2001, a cryptanalysis on the KSA of RC4 was discovered by Fluhrer, Mantin and Shamir [26]. The statistics for the first few bytes of output keystream are strongly non-random, leaking information about the key. This caused a scramble for WEP and led to the use of AES instead in IEEE 802.11i.

3.3 Public-key Cryptography

The main problem of using the previously mentioned symmetric-key cryptographic systems is that all participants must know the secret key to be able to communicate efficiently. In 1976, Diffie and Hellman came up with the idea of public-key systems [21]. The next year, the idea becomes practical due to the invention of RSA cryptosystem by Rivest, Shamir and Adleman [55]. There are other public-key systems that are based on the same

ideas but exploit different mathematical problems to achieve the resulting systems. The following sections discuss several common public-key cryptosystems based on different mathematical assumptions. The areas of the cryptosystems include encryption and digital signature schemes.

The basic idea behind the public-key cryptography is the one-way function, trapdoor one-way function and the fact that the key is a pair of keys. The key pair consists of a public key which is distributed publicly, and a private key which should be kept secret. Roughly speaking, the one-way function is a mathematical function which is easy to compute in one direction, but believed to be difficult to compute in the opposite direction. The trapdoor one-way function is very similar to the one-way function except that with special "trapdoor" information, the computation in the opposite direction becomes easy.

Although usually having a slower speed and requiring a longer key length for the same level of security, public-key cryptography has some advantages over symmetric-key cryptography. It has better key management as a user only needs to store its own private key, but not all keys shared with other users. Also, it does not have to create secure channels to all other users in order to distribute the shared secret keys.

3.3.1 RSA Problem and Related Encryption Schemes

The RSA cryptosystem is based on the trapdoor one-way function $f : \mathbb{Z}_N \to \mathbb{Z}_N$:

$$f(x) = x^e \mod N$$

where N is a product of two distinct large primes, p and q. The trapdoor information is the factorization of N.

The RSA problem is to find a value d such that

$$f^{-1}(x) \equiv [f(x)]^d \equiv x \pmod N \qquad \text{or}$$

$$ed \equiv 1 \pmod{\phi(N)}$$

where ϕ is the Euler-phi function.

By the assumption that solving RSA problem is hard, an encryption scheme was constructed. Suppose a user Alice wishes to allow Bob to send her a private message M over an insecure channel. She first generates her key pair as follows and distributes her public key:

1. Choose two distinct and independent large primes p and q randomly and compute $N = pq$.

2. Choose an integer $1 < e < \phi(N)$ which is coprime to $\phi(N)$ where $\phi(N) = (p-1)(q-1)$

3. Computer d such that $ed \equiv 1 \pmod{\phi(N)}$.

4. Public key: (e, N); private key: (d, N)

After Bob obtains the public key, he turns the plaintext M into a number $m < N$, using some previously agreed-upon reversible protocol known as a padding scheme. Then he computes the ciphertext

$$c = m^e \mod N$$

and sends c to Alice. Alice decrypts c to obtain m by computing

$$m = c^d \mod N$$

and then gets M by reversing the padding scheme.

The best known attacks on RSA depend on solving the problem of factoring very large numbers. Since "short-cut" methods on calculation of factoring exist, key length of RSA cryptosystems is required to be longer than that of symmetric-key cryptosystmes so as to provide the same security level. The most widely used key length (size of N) nowadays is 1024 bits.

3.3.2 Discrete Logarithm Problem and Related Encryption Schemes

Modular exponentiation is a one-way function. Computing the corresponding inverse, namely the discrete logarithm (DL) problem is considered a hard problem. The DL problem concerns about solving a from

$$\alpha^a \equiv \beta \pmod p$$

given a prime p, generator $\alpha \in \mathbb{Z}_p$, and $\beta \in \mathbb{Z}_p^*$. The notation of $\mathbb{Z}_n$ represents the group of integers from 0 to $n-1$. The starred form is the multiplicative group of $\mathbb{Z}_n$, which contains all elements not dividing n. More formally,

$$\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n \mid \gcd(a, n) = 1\}$$

. If n is a prime, $\mathbb{Z}_n^* = \mathbb{Z}_{n-1}$.

One public-key cryptosystem based on the DL problem is ElGamal encryption scheme [23]. Suppose a user Alice wishes to allow Bob to send her a private message M over an insecure channel. She first generates her key pair and distributes her public key. The private key is (a, p) while the public key is (α, β, p). Then Bob selects a secret random number $k \in \mathbb{Z}_{p-1}$ and carries out encryption on the plaintext m to obtain the ciphertext (y_1, y_2) as follows:

$$y_1 = \alpha^k \mod p$$

$$y_2 = x\beta^k \mod p$$

On Alice's side, to decrypt the ciphertext, she computes

$$m = y_2 y_1^{-a} \mod p$$

3.3.3 Elliptic Curve Cryptosystems

Elliptic curve cryptosystems were first proposed independently by Miller [43] and Koblitz [35] in the mid-1980s. They are analogs of existing public-key cryptosystems in which modular arithmetic is replaced by operations defined over elliptic curves, such as point addition and multiplication.

The set of points on a elliptic curve in $\mathbb{Z}_p$ forms a group, analogous to group $\mathbb{Z}_p$. For example, the elliptic curve $y^2 = x^3 + ax + b$ over $\mathbb{Z}_p$ where p is prime and $p > 3$ is the set of solution $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$ to the congruence

$$y^2 = x^3 + ax + b \pmod p$$

The constants a and b are chosen such that $4a^3 + 27b^2 \not\equiv 0 \pmod p$. There is also a special point $\mathcal{O}$ called the point at infinity.

The security of the elliptic curve cryptosystems depends on the hard problem: Given two points G and Y on an elliptic curve such that $Y = kG$, find the integer k. This problem is commonly referred to as the elliptic curve discrete logarithm problem. For systems based on modular DL problem, groups from elliptic curve can be used instead of groups of $\mathbb{Z}_p$ so as to achieve a shorter key length because no good "short-cut" method on computing elliptic curve DL problem has been found.

3.3.4 Digital Signature

Digital signatures are a method of authenticating digital information analogous to ordinary physical signatures on paper. Often, digital signature schemes rely on public-key cryptography. They provide non-reputation and message authentication properties.

Suppose Alice sends a message to Bob and wants to be able to prove it comes from her. She sends a message to Bob together with a digital signature generated using her private key. On receipt, Bob can check whether the message really comes from Alice by running a verification algorithm on the signature together with the message and Alice's public key. If it passes, the message is really from Alice.

The RSA digital signature scheme [38] is presented here as an example. The secret key of Alice is (d, N) and the public key (e, N) as defined before. To sign a message m, Alice computes

$$s = m^d \mod N$$

to obtain the signature s. Bob can verify the signature by checking whether

$$m = s^e \mod N$$

3.4 Public Key Infrastructure

Public-key cryptography solves one of the biggest problems with key management by allowing the public key to be available to everybody. This cuts down the key management overhead enormously. However, this approach still has a problem that an attacker can fake a public key and say that the key belongs to someone else. So a mechanism is required to verify the validity of the public key. The solution is public-key infrastructure (PKI).

PKI is a comprehensive cryptography framework offering confidentiality, data integrity, authentication and non-repudiation, by using public-key cryptography. The mandatory components of the PKI include the following:

User' key pair The key pair consists of public and private keys of a user.

Digital certificate The certificate contains public key and other user information signed by a certificate authority. The IETF (Internet Engineering Task Force) standard for this certificate is named X.509 [30].

Certificate Authority (CA) The CA is trusted by the public to verify the public keys of others. It is responsible for managing a name space of unique user names, authorizing entities, generating key pairs and creating certificates.

A certificate can bind the user's identity with his public key. Other user can check the validity of the certificate by verifying the issuer's digital signature inside the certificate, following a certificate chain until a trusted CA is reached. X.509 standard states the fields a certificate holds and what they mean. All X.509 certificates have the following data:

Version The version of the X.509 applying to the certificate.

Subject name The name of the entity whose public key the certificate identifies.

Public key The public key the entity being named.

Issuer name The X.500 [15] name of the entity that signed the certificate, normally a CA.

Issuer signature The digital signature signed by the issuer.

Serial number A number for the issuer to distinguish the certificate from others.

Validity period The period the certificate is valid.

PKI provides a systematic means to manage keys. However, PKI has the following major shortcomings:

1. Revocation of a certificate is carried out when the certificate expires, the private key is compromised or the user information is changed. Certificates should be publicly accessible and ubiquitous so that other people can obtain any new or revoked certificates. Usually certificate directories, namely repositories, are used for this purpose. However, in some situations, for instance a user does not have Internet access, it is difficult to obtain certificates of other users.

2. A user may choose to send his certificate during the communication to others on demand instead of using the above approach. Nevertheless, the transmission of the certificate not only occupies bandwidth, but also requires verification of CA's signature in real-time. This lowers the efficiency of cryptosystems.

3. In order to allow users to check for revocation of certificates, a certificate revocation list (CRL) is maintained by the authority. Nonetheless, a fine-grained mechanism for receiving and checking the CRL profile has not yet been developed. Also, accessing the CRL requires extra communication and processing.

The first two problems motivate the development of identity-based cryptography, which is covered in Section 3.8.

3.5 Hash Functions and Message Authentication Code

A hash function is a mathematical function that maps a large domain into a smaller range. While an input of the function can be of arbitrary length, the output, called a hash-value or a hash, is of some fixed length. Hash functions are used in many cryptographic areas, and the functions have different properties in different applications. Examples include digital signature, data integrity and identification, For instance, instead of signing the whole document, a user may sign on the hashed document in order to fasten the speed.

One of the major properties of a hash function is being one-way. Although a hash-value is not unique for all inputs, the corresponding inputs should be hard to find from a given hash. Another important property is collision resistance. It should be hard to find two inputs of free choice which give the same hash.

Common hash functions include Message Digest Algorithm version 4 and 5 (MD4 [53], MD5 [54]) and Secure Hash Algorithm and its revised forms (SHA, SHA-1, SHA-256, etc. [22]). MD4, MD5 and SHA were broken (collisions are found by algorithms faster than brute force attack) and cryptographers suggest to use newer version of SHA such as SHA-256.

3.5.1 SHA-256

SHA-256 is the successor of SHA and SHA-1. SHA-0 and SHA-1 produce a 160-bit hash from a message with a maximum size of 2^{64} bits, and is based on principles similar to those used in MD4 and MD5. SHA-256 instead outputs a 256-bit hash which is more secure against brute force attack. Also, no known compromise on SHA-256 has been found so far. A brief introduction on SHA-256 algorithm is shown as follows:

1. Pre-processing: given the original input bitstring, append a single "1"-bit and then enough "0"-bits to get a bit length that is a multiple of 512 minus 64 bits. There last 64 bits are filled with the binary representation of the original message length modulus 2^{64}.

2. Formatted message is broken into 512-bit chunks.

3. Each chunk is process in turn by applying a number of rounds and steps, after it is broken into 32-bit words. Chaining variables are defined in order to "chain" the outputs after processing of each chunk.

4. Before the first chunk is processed, the chaining variables are initialized with specified constant. Different bitwise operations are applied among the chaining variables and and the words. The output values are assigned back to the chaining variables at the end of each round of of word processing.

5. After all words of a chunk are processed, a chunk hash stored by 8 working variables is produced depending on the values of chaining variables and the chunk hash of previous round. The overall result is obtained through the concatenation of the resulted working variables.

3.5.2 Message Authentication Code

A message authentication code (MAC) is a short piece of information used to authenticate a message. In other words, the MAC provides message authentication. A MAC algorithm accepts as input a secret key as well as the message and generates a valid MAC from hash functions or from block cipher algorithms.

In the case that a hash function is used, the MAC is called a keyed-hash message authentication code, or HMAC [37]. Any iterative cryptographic hash function, such as SHA-256, may be used in the caluclation of an HMAC. The cryptographic strength of the HMAC depends upon the cryptographic strength of the underlying hash function and on the size and quality of the key. The output length is the same as the underlying hash function. HMAC is defined as

$$\mathrm{HMAC}_K(m) = h((K \oplus opad) \| h((K \oplus ipad) \| m))$$

, where h is an hash function, K is a secret key padded to the block size of the hash function (512 bits for SHA-256), *ipad* and *opad* are 64 repetitions of bytes 0x36 and 0x5c respectively, represented in hexadecimal.

3.6 Entity Authentication

Entity authentication, or identification, is the process where one entity (verifier) is assured of the identity of another entity (prover) involved in a protocol by the evidence presented by the prover, and that the prover has actually participated. The authentication process should be secure against impersonation of valid parties from any adversary.

Security of an cryptographic identification scheme usually depends on something known to the prover, for instance, a password, a secret or private key. Knowledge of such information is demonstrated in a challenge-response mechanism. The challenge is typically selected randomly by the verifier. The prover has to response on the challenge message using his secret knowledge. However, this approach might reveal some partial information about the prover's secret because an adversarial verifier might strategically select challenges to obtain responses providing such information. To address these concerns, the concept of zero-knowledge (ZK) was introduced. By adding a commitment sent from the prover to the verifier, a adequate ZK identification protocol reveals no information about the secret because the response is based on both the commitment and the challenge, rather than only the challenge.

Entity authentication can be achieved by symmetric-key or public-key technique. The following describes two symmetric-key identification schemes and Section 3.8.2 describes an identity-based public-key identification scheme.

3.6.1 ISO/IEC 9798-4 Three-pass Mutual

ISO/IEC 9748-4 defines an entity authentication scheme [31] which provides mutual authentication through a three-pass challenge-response mechanism based on HMAC. The protocol is described in the following:

$$A \leftarrow B : r_B \tag{1}$$

$$A \rightarrow B : r_A, \mathrm{HMAC}_K(r_A, r_B, B) \tag{2}$$

$$A \leftarrow B : \mathrm{HMAC}_K(r_B, r_A, A) \tag{3}$$

r_A and r_B are random number generated by A and B respectively. K is the secret key shared by both A and B. Upon reception of (2) and (3), B and A respectively checks the validity of the HMACs.

3.6.2 ISO/IEC 9798-4 One-pass Unilateral

ISO/IEC 9798-4 also defines another timestamp-based unilateral authentication scheme:

$$A \rightarrow B : t_A, \mathrm{HMAC}_k(t_A, B)$$

where t_A is the timestamp obtained from A's local clock. Upon receiving the timestamped message, B obtains the current time from its own clock and subtracts the timestamp received. The received message is valid provided that the difference in time is within the acceptance window calculated from the maximum message transit and processing time.

3.7 Key Establishment

A key establishment process generates a shared secret key between two or more parties for subsequent cryptographic use though their communication. In most cases, the key established is a session key. There are two categories on key establishment, key transport and key agreement (key exchange). A key transport protocol allows an entity to generate a secret key and securely transferred to another, while in a key agreement protocol, secret key generation is based on information contributed by both parties. Usually, key authentication property is added to the key transport or key agreement protocols. Otherwise, an adversary can carry out man-in-the-middle attack and obtain the session key shared by both ends of the protocol execution.

When Alice wants to transfer a session key to another entity Bob by some means of key transport, she may simply encrypt the key with their long-term shared secret key or with Bob's public key. Then implicit key authentication can be assured because only Bob can decrypt the ciphertext and obtain the key. The problem of key transport is that if the long-term secret key or the public key is compromised, all the session keys are known. In other words, typical key transport protocols do not offer forward secrecy.

3.7.1 Diffie-Hellman Key Exchange

Using key agreement can distribute the generation of the key to both sides. A very basic key agreement protocol is Diffie-Hellman(DH) key exchange protocol [21] shown as follows.

$$A \rightarrow B : g^a \mod p$$
$$A \leftarrow B : g^b \mod p$$

a and b are random number generated by A and B respectively. p is a prime and g is a generator in $\mathbb{Z}_p$. The security of the protocol is based on the hardness on solving the Diffie-Hellman problem: Given a cyclic group $\mathbb{Z}_p$ where p is a prime, a generator g of $\mathbb{Z}_p$, and elements $g^a, g^b \in \mathbb{Z}_p$, find g^{ab}.

DH key exchange is vulnerable to man-in-the-middle attack. In this attack, an opponent Oscar intercepts Alice's public value (g^a) and sends her own public value (g^c). When Bob transmits his public value (g^b), Oscar substitudes it with his own and sends it to Alice. Oscar and Alice thus agree on one shared key (g^{ac}), and Oscar and Bob agree on another shared key (g^{bc}). This vulnerability is present because DH key exchange does not authenticate the participants. Though its vulnerability, DH key exchange motivates other secure key agreement protocols based on it.

3.7.2 Station-to-Station Protocol

Station-to-Station (STS) protocol [20] is a variant of DH key exchange protocol. It adds key authentication to the protocol by adding encryption, signature or HMAC. The protocol using encryption and signature is as follows:

$$
\begin{aligned}
&A \rightarrow B : g^a \mod p \\
&A \leftarrow B : g^b \mod p, E_k(S_B(g^b, g^a)) \\
&A \rightarrow B : E_k(S_A(g^a, g^b))
\end{aligned}
$$

where $k = g^{ab} \mod p$, S_A and S_B are signing algorithm using A's and B's private key respectively and E_k is an encryption algorithm. Both A and B have to validate the ciphertexts and signatures.

3.8 Identity-Based Cryptography

The concept of identity-based (IDB) cryptography was proposed in 1984 by Shamir [57]. Identity-based cryptography is a kind of public-key cryptography, but it uses human-readable users' identifier information such as email, IP address or network access identifier (NAI) instead of a random numerical value as the public key.

The use of IDB cryptography significantly reduces the cost of PKI. It does not involve certificates, which is either stored online requiring network access, or costs extra communication. It also solves some extent of the certificate revocation problem by the use of identities appended with a standardized valid period as the public key, for example, "$identity \| year$". Then the other users do not have to obtain a new public key after the private key is renewed.

There are some additional attractive features in IDB cryptography. It is simple to create a group identity. For instance, for a group of people is responsible on a duty, they can use the *duty* as the group identity. Moreover, delegation right can be appended to the identity for all users who is authorized to create delegated guest accounts for guests requiring temporary cryptographic services.

IDB cryptography, however, has its own shortcomings. Because human-readable identity strings are used as public keys, typos are also public keys. Also all practical IDB encryption scheme uses bilinear pairing arithmetic, which requires much more computations than simple modular exponentiation [8]. Third, so far all identity-based systems are escrow systems as a client's private key can be calculated by the PKG. The key-escrow property forces clients to have greater trust on the PKG than the CA in conventional PKI because all private keys can be computed by an attacker who compromises the master key of a PKG. Last, unlike conventional PKI, many IDB cryptosystems are not based on widely accepted standards and incompatible to existing standard systems.

Each of the IDB cryptosystems has a private key generator (PKG) which is responsible to generate private keys to and trusted by all users. The PKG carries two algorithms: *Setup*, *Extract*.

Setup The PKG creates its secret master key and system parameters. The system parameters are given to all the interested parties by some means and remains as a constant system parameter for a long period.

Extract A user authenticates himself to the PKG and obtains a private key associated with his identity.

For IDB signature and identification schemes, there are many satisfactory solutions, such as [24, 25]. However, there are only a few number of practical IDB encryption schemes. Boneh and Franklin proposed an IDB encryption scheme based on bilinear pairings techniques [11] while Cocks' proposal is based on quadratic residues [17]. Boneh and Franklin's proposal is more efficient than Cocks' and its settings motivate many other designs of IDB cryptographic schemes. In the following sections, Boneh and Franklin's encryption scheme, and Au and Wei identification scheme are introduced. The latter is a essential component of our proposed authentication scheme.

3.8.1 The Boneh-Franklin Encryption Scheme

The Boneh-Franklin scheme [11] utilizes the bilinear pairing. The admissible bilinear pairing $\hat{e}$ is defined over two groups of the same prime-order q denoted by G and F. Typically, additive notation is used to described the operation in G while multiplicative notation for the operation in F. In practice, the group G is implemented using group of points on certain elliptic curves, and the group F is implemented using a subgroup of the multiplicative group of a finite field. $\hat{e} : G \times G \rightarrow F$, has the following properties.

Bilinear $\hat{e}(aR_1, bR_2) = \hat{e}(R_1, R_2)^{ab}$, where $R_1, R_2 \in G$ and $a, b \in \mathbb{Z}_q^*$.

Non-degenerate $\hat{e}$ send all pairs of points in $G \times G$ to the identity in F. In other words, if R is a generator of G then $\hat{e}(R, R)$ is a generator of F.

Computable For all $R_1, R_2 \in G$, the map $\hat{e}(R_1, R_2)$ is efficiently computable.

Known suitable pairings only include Weil and Tate pairings one elliptic curves.

Security of the encryption scheme is based on the bilinear Diffie-Hellman (BDH) assumption: Given $(G, q, \hat{e}, P, aP, bP, cP)$ where a, b and c are chosen at random from $\mathbb{Z}_q*$, computing $\hat{e}(P, P)^{abc}$ is computationally intractable. The scheme provides semantic security [11].

Here is the description of the scheme in details:

Setup Given a security parameter k, the PKG generates groups G and F of k-bit prime order q together with a bilinear pairing $\hat{e} : G \times G \rightarrow F$; picks cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow G^*$ and $H_2 : F \rightarrow \{0,1\}^l$, where l denotes the length of a plaintext; and picks a random master key $s \in \mathbb{Z}_p^*$ to compute $P_{pub} = sP$. The public system parameters are $(G, F, \hat{e}, q, l, P, P_{pub}, H_1, H_2)$.

Extract Given a string $ID \in \{0,1\}^*$, s and the system parameters, the PKG returns $d_{ID} = sH_1(ID) \in G^*$.

Encrypt Given a plaintext $M \in \{0,1\}^l$, Bob computes $u = rP$ and $v = H_2(\hat{e}(H_1(ID), P_{pub})^r) \oplus M$, where r is chosen at random from $\mathbb{Z}_q^*$. The resulting cipertext $C = (u, v)$ is sent to Alice.

Decrypt Alice decrypts C by computing $M = v \oplus H_2(\hat{e}(d_{ID}, u))$.

3.8.2 Au and Wei's Identification Scheme and Signature Scheme

Au and Wei's identification scheme [7] is built on Paillier setting [48], which does not involve the use of bilinear pairings. Security of the scheme is based on the assumption that RSA$[N, N]$ problem, an instance of RSA problem, with the modulus and the public exponent both equal to N, is hard. The scheme is secure against impersonation attack by passive, active and concurrent adversary [9].

The identification scheme is a commitment-challenge-response protocol. Its works as follows:

Setup (MK_g) The PKG generates two safe prime p and q, and computes $N = pq$. It generates g of order in multiple of N. Also it chooses two cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{QR}_N$, where $\mathbb{QR}_N$ denotes the quadratic residues of N, and $H_2 : \mathbb{Z}_{N^2} \rightarrow \{0,1\}^l$, where l is a security parameter. The master secret is (p, q). The public system parameters are (N, g, H_1, H_2).

Extract (UK_g) Given an identity I, the PKG computes $Q = H_1(I)$, and $(x, y) \in (\mathbb{Z}_N \times \mathbb{QR}_N)$ such that $g^x y^N \equiv Q \pmod{N^2}$. (x, y) is the private key corresponding to the public key I.

Interactive protocol ($\bar{P}$) Suppose Alice wants to authenticate herself to Bob, she executes the protocol as follows:

$A \rightarrow B : t = H_2(g^r u^N \mod N^2)$, where r and u are random number
$A \leftarrow B : c$, a random challenge
$A \rightarrow B : z = (z_1, z_2) = (r - cx, uy^{-c}) \in (\mathbb{Z} \times \mathbb{Z}_N^*)$

Verification ($\bar{V}$) Bob verifies Alice by checking $t = H_2(Q^c g^{z_1} z_2^N \mod N^2)$.

Besides constructing an identity-based identification scheme, Au and Wei also suggested the IDB signature and IDB ring signature derived from their identification scheme, as well as the conversion to apply Paillier setting to Cocks IDB encryption scheme so that the setting can be used in various cryptographic mechanisms. The signature generation and the verification of IDB signature scheme are described as follows:

Signature generation For a message m, the signer with identity I computes:

$t = H_2(g^r u^N \mod N^2)$, where r and u are random number
$c = H_1(m)$
$z = (z_1, z_2) = (r - cx, uy^{-c}) \in (\mathbb{Z} \times \mathbb{Z}_N^*)$

The signature is (t, z).

Verification After receiving m and (t, z), the verifier checks that

$$t = H_2(Q^c g^{z_1} z_2^N \mod N^2)$$

where $c = H_1(m)$ and $Q = H_1(I)$

□ **End of chapter.**

Chapter 4

Basics of WLAN Security and WEP

4.1 Basics of WLAN Security

4.1.1 Overview on "Old" WLAN Security

Due to the threads explored in Chapter 2, a WLAN should employ different means of security. In brief, the traffic in a WLAN should be kept away from eavesdropping, modifying and accessing by any intruders. IEEE 802.11 specification contains several security features, including the use of Service Set Identifier (SSID) and Wired Equivalent Privacy (WEP). However, vulnerabilities are found in each of the measures.

The following sections cover various security measures on WLAN and their security analysis. It is summarized that without applying the new formal standard IEEE 802.11i, the entity authentication of IEEE 802.11 WLAN is not secure enough to provide mutual authentication, key derivation which in turn protects the confidentiality, and other security requirements.

4.1.2 Some Basic Security Measures

The following includes some basic security measures applied in a WLAN, providing minimal security.

Physical measures

Although the physical nature of a WLAN is far less secure than that of a wired LAN, some physical measures can still be applied in order to provide minimal security. Firstly, by using suitable antennas in access points, the reception area can be limited and thus an attacker needs more afford to enter the area. For example, a directional Yagi patch antenna limits the radiation

to a fixed angle in a direction; and a patch antenna limits the radiation to a circular area in a direction.

Secondly, a suitable Effective Isotropic Radiated Power (EIRP), which is determined by the transmitter power and antenna gain, should be chosen in order to limit the reception area.

Service Set Identifier (SSID)

Defined in IEEE 802.11, the SSID is a construct which allows logical partition of WLANs. The SSID is a network name that identifies the area covered by one or more APs. In a commonly used mode, anAP periodically broadcasts its SSID in a beacon, allowing stations wishing to associate with the AP to choose the AP based on its SSID.

In another mode of operation (Closed Network Access Control), the SSID is not broadcast and thus stations wishing to associate with the AP must already have its SSID configured to be the same as that of the AP. In this case, a client station must be manually configured with the appropriate SSID to gain access to the WLAN. However, the SSIS is still included in some management frames in IEEE 802.11 and always sent in the clear. An attacker can still sniff the SSID by just waiting until someone associates to the WLAN. In addition, many WLAN administrators make the attack even easier by using the vendor's default SSID, which are pretty well known.

MAC address authentication

MAC address authentication is not specified in the 802.11 standard, but many vendors support it. The authentication verifies the client's MAC address against a locally configured list (Access Control List, ACL for short) of allowed addresses or against an external authentication server, reducing the likelihood of unauthorized devices accessing the network.

Vulnerabilities appear in MAC address authentication because the MAC addresses are sent in the clear as required by the 802.11 standard. As a result, an attacker can subvert the MAC authentication process by spoofing a valid MAC address by using network interface cards (NICs) supporting this feature.

4.1.3 Virtual Private Network (VPN)

A VPN is a private communication network usually used within an organization, communicating over a public network. VPN message traffic is carried on public networking infrastructure, such as the Internet, using standard protocols, such as TCP/IP.

Secure VPNs use cryptographic tunneling protocols to provide the necessary confidentiality, sender authentication and message integrity so that

users can communicate securely over unsecured networks. The most popular VPN system is based on IPsec.

To utilize a VPN, stations connecting to APs in a WLAN are considered untrusted and kept away from the cooperate LAN by firewalls (VPN gateways). In order to access the LAN, a user must first create a VPN tunnel by authenticating himself to the VPN authentication server. Then the VPN tunneling protocol provides encryption and data integrity during communication between the wireless stations and the LAN.

While the VPN was the most secure measure in WLAN before the introduction of IEEE 802.11i and still has similar security level compared with 802.11i, it suffers from performance inefficiency. All wireless stations accessing public network have to communication through VPN gateways, which become the bottlenecks. A typical VPN gateway can achieve 30-50 mbits/sec throughput. So about 8 wireless stations can overload a VPN gateway. A high system installation cost is required if a large number of users is involved.

4.2 WEP

4.2.1 Overview on Wired Equivalent Privacy (WEP)

As stated by IEEE, WEP is designed to protect users of a WLAN from casual eavesdropping and was selected to meet the following criteria:

- Reasonably strong encryption - It relies on the difficulty of recovering the secret key through a brute force attack. The difficultly grows with key length.
- Self-synchronizing - Each packet contains the information required to decrypt it and it does not have to deal with lost packets.
- Computationally efficient - It can be reasonably implemented in software.
- Exportable - The key length is limited leading to a greater possibility of export beyond U.S. borders.
- Optional - It is an option not required in a WLAN.

WEP aims to provide confidentiality by encryption and data integrity by integrity check value (ICV). It is also used in IEEE 802.11 entity authentication. WEP protects links of a network, rather than provides end-to-end protection like what a VPN does. The following describes the algorithm of each cryptographic service.

IEEE 802.11 authentication

Before a wireless station associates with an AP and gains access to a WLAN, it must perform authentication. Two types of entity authentication are defined in IEEE 802.11: open system and shared key.

Open system authentication protocol simply consists of an authentication request message containing the station identity and an authentication response message containing the authentication result from the AP. The "authentication" protocol in fact allows the station to notify the access point rather than have a secure authentication. On success, both the station and the access point are considered to be mutually authenticated.

Shared key authentication is obtained through the unidirectional challenge-response mechanism. It makes use of the WEP encryption. The client station should know the WEP secret key in advance in order to answer the challenge by encrypting it. The message flow of the protocol is as follows:

$$\begin{aligned}
&\text{STA} \rightarrow \text{AP} : \text{authentication request}\\
&\text{STA} \leftarrow \text{AP} : \text{128 bytes challenge text}\\
&\text{STA} \rightarrow \text{AP} : \text{encrypted challenge text}\\
&\text{STA} \leftarrow \text{AP} : \text{authentication response}
\end{aligned}$$

WEP encryption and decryption

WEP encryption algorithm is constructed from a RC4 symmetric-key stream cipher. A WEP secret key is 40 or 104 bits long, and shared by both communicating parties. The algorithm operates as follows:

1. On the transmitting station, which may be a client station or a AP, the 40-bit secret key is concatenated with a 24-bit Initialization Vector (IV) to produce a seed for input into the WEP pseudorandom generator (PRNG), which is essentially the RC4 cipher excluding the last XOR operation.

2. A long plaintext is fragmented through fragmentation. Each fragment is known as a plaintext Protocol Data Unit (PDU).

3. The seed is passed into the PRGA to produce a keystream of pseudorandom octets.

4. The 32-bit Integrity Check Value (ICV) of the plaintext PDU is obtained through the CRC-32 (32-bit cyclic redundancy checksun) integrity algorithm and appended to the plaintext PDU.

5. The resulted value is XORed with the keystream to produce the ciphertext PDU appended with encrypted ICV.

6. The actual data sent in the data frame is (IV || ciphertext PDU || encrypted ICV).

The decryption algorithm is just the reverse of the encrytpion. As IV is sent in clear, the receiver is able obtain the keystream corresponding to the IV and the WEP secret key. After decryption, the recovered plaintext is verified by comparison between the recovered ICV and the ICV obtained from the CRC-32 algorithm on the recovered plaintext.

4.2.2 Security Analysis on WEP

Although WEP incorporates several mechanisms to help secure wireless traffic, many attacks have surfaced over time, demonstrating that the design goals were not achieved and WEP is unable to provide adequate security. The following sections analyze different attacks on WEP and show its failure on authentication, confidentiality and data integrity.

Attacks on unidirectional authentication

When shared key authentication is used, only the client station is authenticated to the AP. As a result, it suffers from man-in-the-middle attack. When an attacker receives a authentication request message, it forwards the message to the AP. He then receives the challenge from the AP, sends the challenge to the legitimate station and obtains the valid response from the station. He finally completes the process by returning the valid authentication response to the AP and creates another authentication instance to the station.

In this way, the attacker is able to act as a AP and a client station and associate to the legitimate station and AP respectively. Although the attacker cannot obtain the WEP secret key, he can control all network traffic from the client station and carry out other more dangerous attacks in a simple manner.

Authentication spoofing

In the first glance, the shared key authentication seems to be much more secure than the open authentication. However, this is not the truth. The challenge and response messages provide an eavesdropper with useful information that can compromise a network. This attack is known as authentication spoofing [6].

By listening to the shared key authentication handshake, the eavesdropper obtains the initial unencrypted challenge message (m) and the IV from the AP and the encrypted message (c) from the client station. Then he can obtain the keystream k corresponding to that IV from a known plaintext attack as follows:

$$k = (m \| \text{ICV}) \oplus c$$

Since the AP will not check whether the IV is reused, the eavesdropper can use the same keystream and IV to answer the challenge in another authentication session. In this way, an attacker can join the network without knowing the secret key.

This attack does not allow the attacker to communicate because data frames are encrypted with WEP. However, he can inject a packet by encrypting the packet by the keystream obtained and reusing its corresponding IV, without the knowledge on the secret key. Also, some attacks compromising the WEP secret key and described in the following requires plaintext-ciphertext pairs. The shared key authentication thus provides a convenient way for the attacker to obtain such pairs and carry out other more threatening attack.

Packet modification

Borisov et al. [12] discovered a weakness on the ICV which threaten the data integrity. The fact that CRC-32 algorithm is a linear function and independent to the WEP secret key and IV can compromise data integrity. The ICV, which is a CRC checksum, is designed to detect random errors, but not malicious or intentional modifications. An attacker can easily modify both the actual data and the corresponding ICV in order to pass inspection, making a successful message forgery.

Let m and m' be the original message and modified message respectively, c and c' the original ciphertext and modified ciphertext, $\Delta = m \oplus m'$ the different between m and m', k the keystream corresponding to a particular IV reused in modified packet. The following shown the process of modification mathematically.

$$
\begin{aligned}
c' &= k \oplus (m' \| \mathrm{CRC}(m')) \\
c' &= k \oplus ((m \oplus \Delta) \| \mathrm{CRC}(m \oplus \Delta)) \\
c' &= k \oplus ((m \oplus \Delta) \| (\mathrm{CRC}(m) \oplus \mathrm{CRC}(\Delta))) \\
c' &= k \oplus ((m \| \mathrm{CRC}(m)) \oplus (\Delta \| \mathrm{CRC}(\Delta))) \\
c' &= c \oplus (\Delta \| \mathrm{CRC}(\Delta))
\end{aligned}
$$

The consequences of this attack include the following:

1. If an attacker knows a portion of a plaintext, he can modify that portion and the corresponding checksum to obtain a valid ciphertext.

2. An attacker can guess the IP address of a packet, modify it, as well as the IP checksum, and obtain the recovered plaintext at the modified address. This is known as IP redirection attack. This attack is based on the fact that an IP address is at fixed field encapsulated in IEEE 802.11 MAC frames. If an IP checksum is known, an attacker can

simply carry out arithmetic as the IP checksum is also linear. But if the IP checksum is not known, he will have to guess it.

3. An attacker can carry out TCP reaction attack. By careful modification of an intercepted packet [12], he can know whether the TCP checksum is valid by checking whether an ACK is received.

   - The modified ciphertext is obtained by $c' = c \oplus \Delta$, where Δ is the bit positions to flip.
   - The attacker chooses Δ by picking i arbitrarily, and setting positions i and $i + 16$ of Δ to 1 and the rest to 0.
   - The attacker obtains one bit of information about the packet by the property that if the XORed result of the original plaintext at position i and $i + 16$ is 1, the TCP checksum will be valid.

   The attacker can repeat many times to discover the entire packet.

RC4 key schedule attack

WEP makes use of RC4 to generate pseudorandom keystream. Since 1994, researchers have identified a series of small flaws in RC4, none of which resulted in a practical attack. However, Fluhrer, Mantin and Shamir [26] finally found an algorithmic attack on RC4 key scheduling algorithm (also known as Fluhrer, Mantin, Shamir attack) resulting in recovery of the secret key. The recovery is in linear time with respect to the key size, and this means attack on 104-bit WEP is only slightly more difficult than that on 40-bit WEP. The attack is practical on WEP because of the use of IV.

Fluhrer, Mantin and Shamir revealed that if the first A bytes of the secret key s is known (initially $A = 0$), and the following criteria are fulfilled, an attacker can recover the $A + 1$-th byte of the key with an accuracy of 5%.

- The attacker gets a series of about 60 different packets encrypted with RC4 keys started with the 3 bytes $(A + 3, 0xFF, X)$ where X is any byte value and different in each packet. These keys are consider as weak keys.
- The attacker knows the first byte of the keystream used to encrypt the packets collected. This is equivalent to knowing the first byte of each of the plaintext.

With such class of keys, the first byte of the keystream is determined by the equation: $S[S[1] + S[S[1]]] = S[A + 3]$ where S is the S-box used in the implementation of RC4, computed according to the RC4 key. Then, the attacker can reverse the KSA to find the $A + 1$-th key byte from the known information. The process can be iterated in order to recover the whole secret key byte by byte.

This attack is practical for retrieving WEP secret key because first 3 bytes of the key used in RC4 are in fact the IV. Any WEP encrypted packet is started with the IV which is sent in clear. Moreover, the first byte of the plaintext can also be found as in IP data-networking environment, the first byte of the vast majority of packets is $0xAA$, a value in IP packet header. An attacker can make use of this information determine the first byte of the WEP secret key. Consequently, the attacker knows the first four bytes of the RC4 key and thus can find the second byte of the WEP key via the same way. By collecting enough packets, he can iterate the process to retrieve the whole key.

This attack has already been implemented on software like AirSnort and WEPCrack. Stubblefield et al. [59] carried out experiment with typical equipments (PIII/500 MHz laptop with Linux) and found that between 5 to 6 million packets are needed in order to recover a 104-bit key in an unoptimized situation. Recovering this quantity of packets depends on the network load and can range from less than one hour in a moderately used network to several hours in a lightly used network. This practical work has shown that no expensive hardware or software is necessary in order to break WEP.

IV collisions

The authentication spoofing attack is a specific instance of a more general attack in which confidentiality of any transmission can be compromised [12]. The confidentiality is at risk when any two plaintexts are encrypted with the same keystream. Because the keystream depends on a combination of the secret key and an IV, and because the secret key is constant, an adversary can determine that two messages are encrypted with the same keystream simply by comparing the IV sent unencrypted.

Given two ciphertexts produced with the same keystream k, XORing the two ciphertexts together removes the pseudorandom stream generated by RC4 and produces the XOR of the two corresponding plaintexts. An active adversary from a wired station can send a chosen plaintext over the wireless network and observe the encrypted message to obtain the keystream of a particular IV. For an adversary not able to access the wired LAN, he can utilize the expected distribution of the plaintexts to predict the contents since much of network traffic contents is predictable. A more aggressive attacker can inject packets to the WLAN, observe responses from upper layer protocols and gain information on a plaintext from the responses. TCP reaction attack introduced before is one of the examples. Another is an inductive chosen plaintext attack based on this technique was introduced by Arbaugh [5]. This attack involves inductively constructing and sending specially formatted packet, and observing the response of DHCP protocol.

Such attacks would not be possible if IVs were non-repeating. However,

with a 24-bit IV, at most 2^{24} possible values exist. According to [60], IV collision occurs with 50% probability after 4823 encrypted frames, and 99% after 12430 encrypted frames. In high-traffic environments, IVs are guaranteed to repeat in a matter of hours. Even worse, many vendors choose to reinitialize the IV to zero every time the AP or client station is restarted. This practice means the IV is likely to be a low-value number that was recently used, resulting in even more collisions.

Although such attack does not reveal the WEP secret key, an attacker can completely compromise the system using a dictionary attack. Once he successfully decrypts ciphertexts with different IV, he can build a table that lists the keystream corresponding to each IV. This table isapproximately 23.5 GB, well within range of today's computer storage. Since IV space is fixed and the attack does not involve the secret key, WEP is "unsafe at any key size" [60].

Brute force attack

WEP allows the use of 40-bit keys. Keys of such short length are easy to break. About 200 days are needed to obtain a 40-bit key by a PIII/500 MHz laptop via brute force attack. Even worse, some vendors allow users to enter the key in ASCII characters. This further reduces the entropy of the key and time to crack is just about 35 seconds.

□ **End of chapter.**

Chapter 5

IEEE 802.11i

5.1 Overview on IEEE 802.11i and RSN

The flaws of WEP are introduced in previous sections and we can comment that WEP does not provide sufficient security in WLANs. In order to deal with the security concerns on WLANs, the IEEE 802.11i task group was formed and worked on specification providing more rigorous security mechanisms.

Before completion of IEEE 802.11i standard which is also known as WPA2, a transitional industrial standard which is called Wi-Fi Protected Access (WPA) was defined by Wi-Fi Alliance. WPA can be considered as a subset of 802.11i standard while providing some extensions. IEEE 802.11i defines a new type of wireless network called robust security network (RSN). RSN aims to achieve the same goals as WEP-based network, but in a actually secure way.

There are several components consisting RSN. They include a new access control and authentication framework based on IEEE 802.1X, improved encryption and data integrity algorithms - TKIP and AES-CCMP, and key management algorithms. RSN also includes support of independent basic service sets (IBSSs), also known as ad hoc wireless networks, and preauthentication mechanism which minimizes latency.

In RSN, a client has to carry out RSN association (RSNA) through the 802.1X authentication before it can communicate with the network. The steps for the association under extended service set (ESS) without using a preshared key is as follows:

1. The client identifies the AP as RSNA-capable from the AP's Beacon or Probe Response frames.

2. The client optionally invoke open system authentication.

3. The client and the AP negotiate ciphersuites used for encryption and data integrity, in data communication after RSNA.

4. The client and the AP use IEEE 802.1X to authenticate.

5. The client and the AP establish temporal keys by executing the key management algorithms. The temporal keys are used for encryption and data integrity to protect the data link.

In the following sections, major components of IEEE 802.11i standard are covered. The description is mainly on security of infrastructure mode.

5.2 IEEE 802.1X Access Control in IEEE 802.11i

IEEE 802.1X [47] is an IEEE standard for port-based network access control. The 802.1X framework provides authentication to devices attached to a LAN port, establishing a point-to-point connection or preventing access from that port if authentication fails. It takes the advantage of Extensible Authentication Protocol (EAP) [2].

802.1X enhances the enterprise security model by providing the following improvements over the ordinary 802.11 authentication.

- It provides support for a centralized security management model, and allows good scalability in terms of the number of APs.
- The keys used in data communication are unique to each station so that traffic on any single key is significantly reduced.
- It allows the members of the network to generate keys dynamically and does not require a network administrator for configuration of pre-shared keys.
- It provides open support for strong upper layer authentication, and allows flexibility on the choice of EAP methods.

In the following, we will look at 802.1X standard working in WLANs under infrastructure mode, or ESS.

5.2.1 Participants

There are three kinds of participants in the 802.1X access control process in a WLAN.

Supplicant The end of the link that responds to an authenticator. It is the client requesting to join the network by presenting its credentials and following a proper protocol.

Authenticator The end of the link initiating EAP authentication. In a WLAN in ESS , it is an AP that must verify the identity of a supplicant before granting the network access.

Authentication server (AS) The AS provides an authentication service to an authenticator. Typically, it runs the Remote Authentication Dial-In User Service (RADIUS) supporting EAP and handling authentication requests relayed by authenticators from supplicants. It may maintain a list of valid users and their credentials to validate authentication requests.

5.2.2 Port-based Access Control

As mentioned before, 802.1X grants per-port access to clients requesting access to network resources. A port is a logical entity representing a physical connection to a network. There are two types of ports, namely an uncontrolled port and a controlled port. Uncontrolled ports allow communication between devices on a LAN without having to make an access control decision, while controlled ports are entry points to the LAN resources protected by the authenticator against unauthenticated supplicants.

To carry out RSN association, a supplicant and an authentication server exchange upper layer authentication protocol messages and carry out four-way handshake via the uncontrolled port. Before a successful association, the controlled port is blocked for all non-IEEE 802.1X packets. When the supplicant is authenticated by the authentication server and passes the four-way handshake to generate key hierarchies, the authenticator unlocks the controlled port for the supplicant to allow it to access the wired LAN and communicate with other wireless stations via the authenticator.

5.2.3 EAP and EAPOL

The authentication exchange used in 802.1X takes place over EAP. EAP is a protocol designed for use in transporting authentication messages. Before 802.1X became prominent in WLAN authentication, EAP was mainly used to authenticate dial-up users. In order for EAP messages to be transported on a LAN, they need to be encapsulated. 802.1X defines EAP over LAN, or EAPOL for this purpose. EAPOL is applied to encapsulate EAP messages between a supplicant and an authenticator during upper layer authentication, as well as the four-way handshake.

EAP has four message types: Request, Respond, Success and Failure. EAP can encapsulate messages of upper layer authentication protocols in its Request and Respond messages. The encapsulated protocol is called an EAP method. Request messages are sent from an authentication server while Response messages are from a supplicant. The authentication server uses the Success or Failure message to notify the supplicant and the authenticator whether the authentication was successful. EAPOL further adds four message types: Start, Key, Packet, Logoff. When a supplicant first connects to a LAN, it broadcasts the Start message. The Key message is used for

information exchange by the key management algorithms. The Packet message encapsulates the actual EAP messages. Logoff message indicates that the supplicant wishes to be disconnected from the network.

5.2.4 RADIUS

RADIUS is an Authentication, Authorization and Accounting (AAA) protocol [52, 50] for applications such as network access. In 802.1X, RADIUS protocol is often used to encapsulated the EAP messages exchanged between the authenticator and the authentication server. A RADIUS authentication server exchanges authentication information with supplicants, authorizes access to legitimated supplicants and records the client session information for billing and other statistical purposes. Also it is usually used in authentication between the authenticator and the AS after a supplicant is authenticated by the AS, as well as the transmission of PMK from the AS to the authenticator.

To carry EAP over RADIUS, extensions to RADIUS were defined [51]. The EAP-Start message was introduced for an authenticator to notify the RADIUS server when an execution of EAP authentication is started.

5.2.5 Authentication Message Exchange

The authentication message exchange between a supplicant and an authentication server, through an authenticator is encapsulated by EAP, which is further encapsulated by other protocols as introduced before. Figure 5.1 shows an EAP message flow. The AP is not aware of the authentication process in details. Instead it is responsible to relay the messages exchanged by the supplicant and the AS and only cares about the AS's decision whether to grant the client access to the network, through the listening on the EAP-Success or EAP-failure message.

At the beginning of an authentication process, EAP-Request and Response messages of type Identity are exchanged. The messages allow the authenticator and the AS to know the identity of the supplicant in order to distinguish different authentication sessions. Following the messages of type Identity, EAP-Request and Response messages encapsulating upper layer authentication protocols are exchanged in pairs, allowing the transmission of actual authentication data. IEEE 802.11i does not define a standardized upper layer authentication protocol used in 802.1X in order to allow flexibility.

5.2.6 Security Analysis

In [44], three problems on 802.1X access control are demonstrated. The first one is that some EAP methods, such as EAP-MD5 [10], offer only unidirectional authentication and man-in-the-middle attack is possible in such

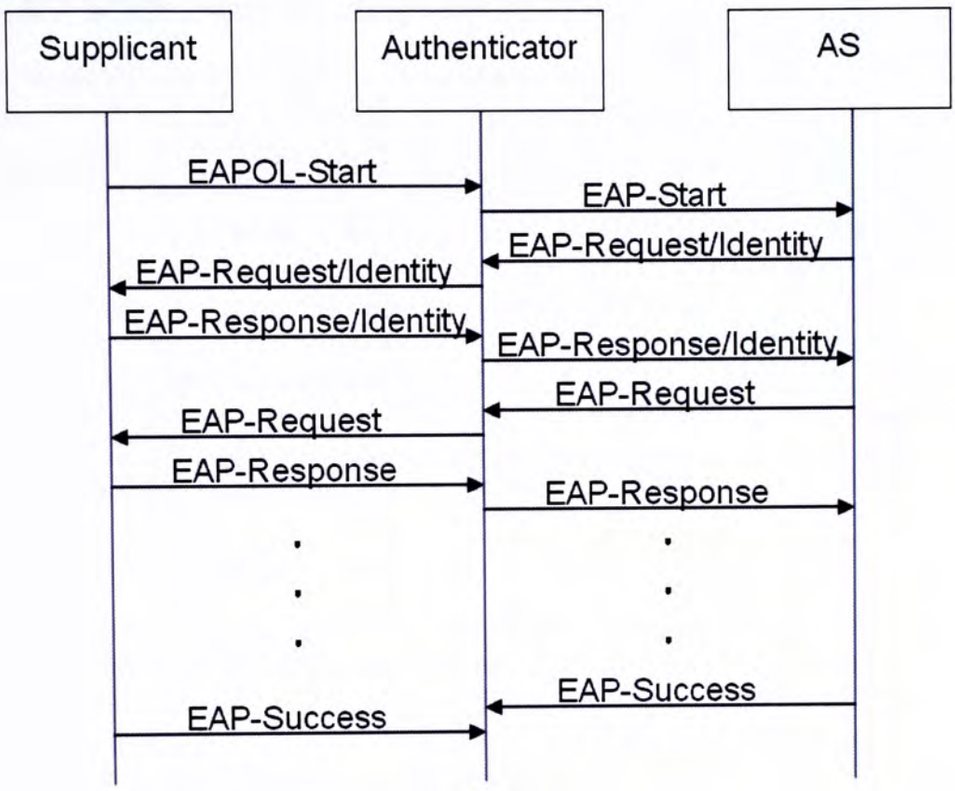

Figure 5.1: EAP message flow for successful authentication. If the access to the network were denied, the EAP-Failure messages would replace the EAP-Success messages in the figure.

schemes, allowing an attacker to impersonate an authenticated party. The second problem is session hijacking. Since the 802.11 MAC layer management frames are unprotected, an attacker can forge a Deassociation frame to the authenticator just after the authentication execution and pretend the authenticator in later-on communication. The last one is denial-of-service (DoS). The attack is brought by the possibility of attacks to forge the EAP-Failure message and other MAC layer frames.

The first problem can be prevented by explicitly requiring all ULA protocols to provide mutual authentication in order to minimize the risk from rogue APs. No effective measures can be taken to resist the success of the second attack. However, if the ULA protocol provides key establishment, the second problem is just converted to another DoS because the adversary cannot obtain the derived session key to compromise the confidentiality and data integrity. For the last one, DoS attack is always possible in different ways in real life. What we can do is to minimize the effect of it. For example, protocols should limit the time affected by a successful attack.

5.3 RSN Key Management

Unlike WEP, which uses the same key for message protection and authentication, IEEE 802.11i separate the authentication process and message protection. IEEE 802.11i allows for automatically generated per-user, per-session keys, known as pairwise keys. In addition, these pairwise keys can be regenerated periodically to increase security, through a process known as rekeying. Besides the pairwise keys, group keys can also be created by the authenticator for multicast and broadcast traffic.

Besides mutually authenticating both the supplicant and the authenticator, an upper layer authentication protocol in 802.11i is required to establish a Pairwise Master Key (PMK), required to be 256 bits, that a supplicant and an authenticator can use for message protection, in the case that any pre-shared key mechanism is not available.

5.3.1 RSN Pairwise Key Hierarchy

The PMK is at the top of RSN key hierarchy for unicast communication. It is transferred from the authentication server to the authenticator after a successful authentication through MS-MPPE-Recv-key attribute of RADIUS protocol if RADIUS is used. Then both the supplicant and the authenticator can generate the pairwise transient key (PTK) based on the PMK and nonces obtained through the four-way handshake, using the pseudorandom function defined in 802.11i standard. The PTK can be divided into four 128-bit keys:

- Data Encryption key - to protect confidentiality of data communication
- Data Integrity key - to protect integrity of data communication
- EAPOL-Key Encryption key - to protect confidentiality during the four-way handshake
- EAPOL-Key Integrity key - to protect integrity during the four-way handshake

If AES-CCMP is used, the Data Encryption key and Data Integrity key are integrated to form the Data Encryption/Integrity key.

5.3.2 RSN Group Key Hierarchy

Due to the fact that 802.11 communication also supports multicast and broadcast messages, 802.11i also standardizes a process for generating group keys which ensure secure communication for broadcast messaging.

The group keys are formed from a 256-bit cryptographic-quality random number, known as the Group Master key (GMK), chosen by the authenticator. Then the GMK is used to determine the Group Temporal key (GTK), consisting of 128-bit Group Encryption key and 128-bit Group Integrity key, which are for confidentiality and integrity for multicast and broadcast communication respectively. Again if AES-CCMP is used, a single Group Encryption/Integrity key is generated instead.

5.3.3 Four-way Handshake and Group Key Handshake

PMK can be generated in two ways. The first is to obtain from a shared key while another is to be established through an upper layer authentication protocol. The PMK obtained is then used to generate PTK via the four-way handshake shown in Figure 5.2. The handshake also deliver the GTK from the authenticator to the supplicant.

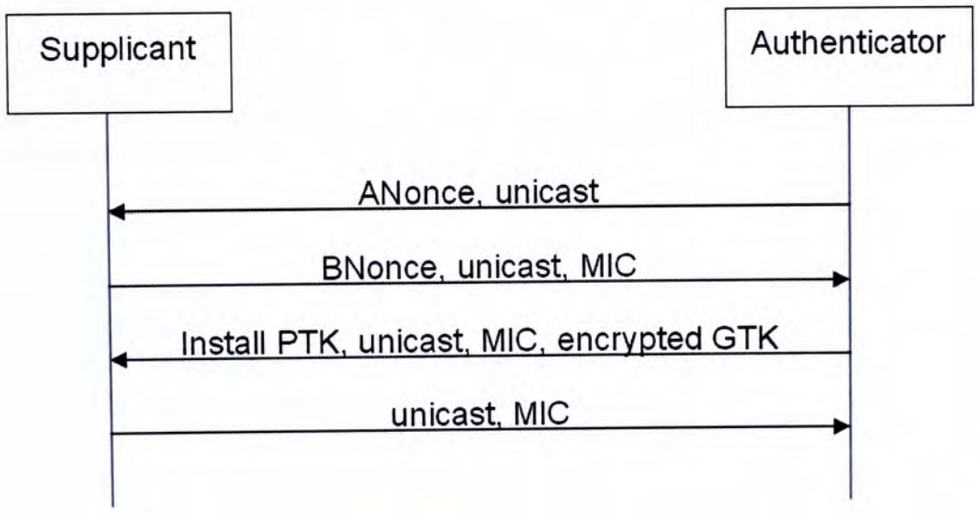

Figure 5.2: Four-way handshake message flow after successful authentication. ANonce and SNonce are random number generated by the authenticator and the supplicant respectively. "Unicast" is used to specify the handshake is the Four-way handshake rather than a group key delivery. MIC is the message integrity code (considered as a kind of HMAC) of previous messages utilizing EAPOL-Key Integrity key. The encryption of GTK uses the derived EAPOL-Key Encryption key.

The handshake messages are encapsulated in EAPOL-Key frames. When a successful handshake completes, the authenticator confirms the existence of the PMK at the supplicant and the supplicant obtains the GTK. The handshake ensures that the temporal keys are fresh and synchronizes the installation of the temporal keys.

If the authenticator later changes the GTK, the supplicant can obtain

the new key through the group key handshake. The handshake incudes two messages. The authenticator sends the encrypted new GTK with the MIC. The supplicant responds with the valid MIC.

After the four-way handshake, the last step before the setup of the RSN association is to open the controlled port by the authenticator. The authenticator also have to authenticate to the authentication server by some means, typically through RADIUS protocol.

Recently, an analysis reveals the possibility of a DoS attack on the four-way handshake [29]. The attack involves forging initial messages from the authenticator to the supplicant to produce inconsistent keys in both parties.

5.4 RSN Encryption and Data Integrity

IEEE 802.11i provides two improved symmetric-key encryption algorithm to replace WEP which was broken. They are Temporal Key Integrity Protocol (TKIP) and Advanced Encryption Standard-Counter mode-CBC MAC Protocol (AES-CCMP). In RSN, WEP encryption is no longer used. However, backward compatibility can be provided by the Transition Security Network, which uses WEP along with CCMP or TKIP. 802.11i defines both TKIP and CCMP while WPA only employs TKIP.

5.4.1 TKIP

TKIP was designed to address all the known attacks and deficiencies in the WEP algorithm while still maintaining backward compatibility with legacy hardware. It was designed to be made available as a firmware or software upgrade to existing hardware so that users would be able to upgrade their level of security without replacing existing equipment or purchasing new hardware. It provides an upgrade path by offering an additional protocol or a wrapper around WEP.

TKIP utilizes the key management scheme of 802.11i in order to bound the probability of successful forgery and the amount of information that an attacker can learn about a particular key. TKIP is comprised of the following elements:

- A message integrity code (MIC) provides a keyed cryptographic checksum using the source and destination MAC addresses and the plaintext data of the 802.11 frame (or MAC service data unit (MSDU)), in order to protects against message forgery.

- A per-packet key mixing mechanism of the IV is used to change the encryption key for every MSDU to prevent RC4 key schedule attack.

- A 64-bit IV and an 48-bit TKIP sequence counter (TSC) to minimize the impact from IV collisions and replay attacks. MPDUs received out

of order are dropped by the receiver.

TKIP uses an extended 48-bit TSC extends the life of the temporal key and eliminates the need to re-key the temporal key during a single association. Since the TSC is updated with each packet, 2^{48} packets can be exchanged using a single temporal key before key reuse would occur. Under steady, heavy traffic conditions, it would take approximately 100 years for key reuse to occur. The TSC is constructed from the first and second byte from the original WEP IV and the 4 bytes provided in the extended IV.

In the TKIP encryption, the temporal Data Encryption key, transmitter address, and TSC are combined in a two-phase key mixing function to generate a per packet key to be used to seed the WEP PRNG. The per packet key is 128 bits long and is split into a 104-bit RC4 key and a 24-bit IV for presentation to the WEP engine.

The 8-bit MIC is added to the packet residing before the CRC. It is calculated over the source and destination MAC addresses and the MSDU plaintext using the MIC function, Michael, seeded by the Data Integrity key and the TSC, where Michael is a one-way cryptographic hash function. This makes it much more difficult for an attacker to successfully alter packets. If necessary, the MSDU is fragmented and a unique MIC is appended to each of them.

The decryption process is the reverse process of the encryption where additional checking processes are applied. They include to discard packets with TSC smaller than that in previous packets and verification of the MIC value. If the MIC values do not match, the recovered MSDU is discarded and countermeasures such as rekeying and alerting are executed.

5.4.2 CCMP

In addition to TKIP, 802.11i defines a new encryption method based on AES, requiring update on hardware. Similar to other block ciphers, AES can be used in various operation modes. The mode that has been chosen for 802.11i is the counter mode with CBC-MAC. The counter mode encryption offers data confidentiality while the CBC-MAC delivers message authentication. CCMP is recommended to used in the RSN as it offers a better security than TKIP.

When the AES cipher is used in counter mode, it does not directly encrypt a plaintext. Instead, it encrypts an arbitrary value called counter preload which increments from a seed value, and then XORs the encrypted counter preload with each plaintext block.

In CCMP, the 128-bit Data Encryption/Integrity key is used in the AES cipher. The counter preload is formed from a 48-bit IV called the packet number (PN), a flag value, data from the frame header (such as the source MAC address), and a counter value which is initialized to 1. The plaintext

before the encryption is fragmented to produce a number of frames. The counter value increments for each plaintext block (128 bits) of a frame.

For CBC-MAC, CBC stands for cipher block chaining while MAC stands for message authentication code. In CCMP, a seed is formed by a flag value, the PN, and other data pulled from the header of the frame. The seed is fed into an AES cipher block and encrypted by the Data Encryption/Integrity key, and its output is XORed with specific elements from the frame header, which is then fed into the next AES block. This process continues over the remainder of the frame header and the actual plaintext data. The first 64 bits of the resulted MAC value are extracted as the final MIC output.

The resulted encrypted message consists of the frame header, the PN, the encrypted data blocks and the MIC in specified order. The decryption process is essentially the reverse of the encryption process plus the verification of the MIC.

5.5 Upper Layer Authentication Protocols

5.5.1 Overview on the Upper Layer Authentication

The IEEE 802.11i standard does not specify an upper layer authentication (ULA) protocol over EAP used in 802.1X authentication process. This is because an ULA protocol operates at higher layers of the OSI network layer model (usually in TCP/IP layer) and are thus outside the scope of the 802.11 standard.

There are a number of popular ULA protocols in use today, primarily in enterprise environment, namely network running in infrastructure mode. As described before, the protocols are encapsulated by EAP. Some mandatory requirements on the ULA protocol are specified in RFC4017 [58]:

- Generation of a 256-bit PMK. The PMK must have effective key strength of at least 128 bits.
- Mutual authentication between the supplicant and the authentication server.
- Shared state equivalence. After a successful completion of an authentication execution, the supplicant and the authentication server must share the same state attributes, such as the session key, identities of both the supplicant and the server, etc.
- Resistance to dictionary attacks.
- Protection against man-in-the-middle attack.
- Protected negotiation on the ciphersuits used in the EAP conversation.

Also, recommended requirements, including support on fragmentation and end-user identity hiding, and optional features, including channel binding and fast reconnect are specified.

Some of the more popular authentication protocols include: the extensible authentication protocol with transport layer security (EAP-TLS) [3], the protected extensible authentication protocol (PEAP) [4], Kerberos [36] and the lightweight extensible authentication protocol (LEAP).

5.5.2 EAP-TLS

EAP-TLS is an EAP authentication method used in 802.11i and WPA environment well-supported among wireless vendors. It is a public-key method based on the use of PKI and certificates. Original TLS protocol [19] includes ciphersuite negotiation, authentication with key exchange and symmetric-key encryption. EAP-TLS uses the first two to carry out secure authentication in 802.1X.

EAP-TLS support various ciphersuites on public-key encryption and hash functions. Figure 5.3 shows the handshake between a supplicant and an authentication server if RSA public-key encryption is used.

The messages are explained in the following:

Client hello - the list of supported ciphersuites and a 28-byte random number c.

Server hello - the chosen ciphersuites and a 28-byte random number s.

Server cert - the certificate of the authentication server.

Client cert request - if the AS does not have a local cache on the supplicant, it will request the certificate of the supplicant.

Server done - an empty message.

Client cert - the certificate of the supplicant.

Client key exchange - a 48 byte pre-master secret p encrypted by RSA encryption using server's public key obtained in the certificate.

Client verify - signature of previous handshake message sent or received, signed by the supplicant's private key.

Change cipher spec - indication about the change of the ciphersuite, from RSA to a symmetric-key algorithm.

TLS-Finished - hash of previous handshake messages.

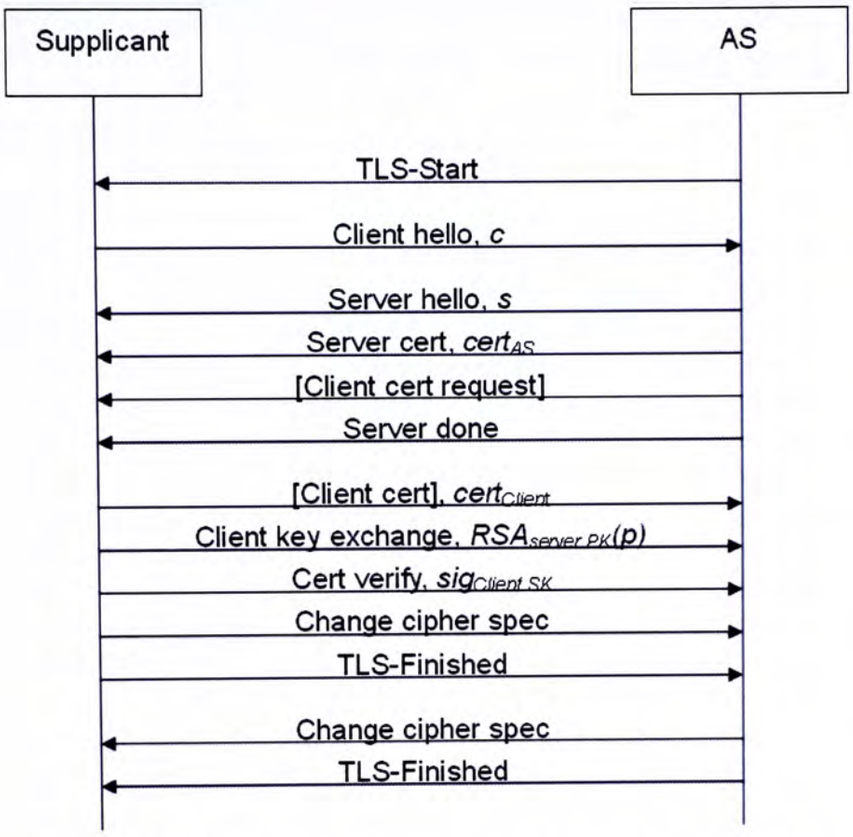

Figure 5.3: EAP-TLS handshake for successful authentication. Messages enclosed by "[ ]" are optional.

EAP-TLS supports mutual authentication through the use of certificates. If RSA is used, the supplicant is authenticated by the signature signed with its private key, while the authentication server is authenticated by the ability to decrypt the encrypted pre-master secret and create valid MACs using the information of the recovered secret. EAP-TLS also support static Diffie-Hellman key exchange, where both parties are authenticated by the the ability to create valid MACs using the agreed key.

EAP-TLS resists most known attacked, including replay and man-in-the-middle attacks. It has good extensibility due to the flexibility on choosing ciphersuites. It can derive a per-session key between the authentication server and the supplicant, through the RSA-based key transport or DH key exchange. TLS is well understood and well tested and is considered as a secure protocol.

There are some disadvantages on EAP-TLS, though. It requires the deployment of PKI. It suffers from bandwidth inefficiency because of the transmission of certificates. Verification on the validity of the certificates requires computation load in real-time. Last but not least, if RSA-based

key transport is used, compromise of the server's private key allow attackers to obtain all session keys. In other words, it violates forwary secrecy.

5.5.3 Other Popular ULA Protocols

Kerberos

Kerberos is an symmetric-key based authentication protocol designed for IP networks. Many places that use wired LAN already use Kerberos for authenticating its clients for services, such as access to an email server. In the Kerberos model, every service requires a ticket and a session key. The tickets and the session keys are issued by the Kerberos authentication server and its ticket granting server (TGS). Each ticket contains information, including server's identity, client's identity, client's IP address, valid period of the ticket, and the session key, so that the server controlling the service can verify the identity of the possessor of the ticket.

When Kerberos is applied to RSN, the authenticator (access point) acts as a Kerberos Proxy relaying messages from the supplicant to the Kerberos AS and the TGS. When an supplicant requests for network access, the AS issues a Ticket Granting Ticket (TGT) and a corresponding session key, encrypted with the secret key shared between the supplicant and the AS, to the supplicant. The supplicant makes use of the TGT and the session key to obtain a network access service ticket and the corresponding session key from the TGS for accessing the network. The supplicant then presents the service ticket in order to obtain network access.

Kerberos is well-tested as it has been invented for a long time. There are two major shortcoming on the system. It is a symmetric-key protocol and thus distribution of secret keys requires secure channels. Another problem is that the encrypted session keys make the system vulnerable to dictionary attack.

PEAP

The development of PEAP is motivated by the fact that the EAP-Response/Identity and EAP-Success/Failure messages are unprotected. An attacker can learn the identity of the user attempting to connect. Also he can spoof the Success and Failure messages.

PEAP solves these problems by carrying out authentication in two phases, In the first phase, EAP-TLS is used in a conventional way to establish a secure connection except that only the server is authenticated in this phase. In the second phase, the secure connection established is used for another complete EAP negotiation in which full authentication is performed.

The weakness of PEAP is that number of handshake message is double of that in unprotected EAP-TLS. This means that the extent of inefficiency is also doubled.

LEAP

LEAP, developed by Cisco, is the first commercial use of IEEE 802.1X for WLANs. It uses WEP for the message protection during the authentication. It provides mutual authentication through separate challenge-response mechanism in both directions. The challenge message is a random number while the response is the encrypted challenge. A session key is derived through an unpublished proprietary mechanism using the information from the challenge messages.

LEAP is symmetric-key based suffering key management problem and the encrypted challenge messages allow any attacker to carry out offline dictionary attack.

EAP-SRP

The Secure Remote Protocol (SRP) [61] is a type of symmetric-key protocols based on the use of passwords. Its key derivation is based on Diffie-Hellman key exchange, and the user password is hidden in a exponent of the DH key exchange parameter. It provides mutual authentication through the explicit key authentication. If both the supplicant and the AS agree on the same session key, they are mutually authenticated.

While the use of password can somehow simplify the key entry process at client side, secure channel to update the password is still required. Entropy of passwords is also much smaller than that of numerical keys.

□ **End of chapter.**

Chapter 6

Proposed IEEE 802.11i Authentication Scheme

6.1 Proposed Protocol

6.1.1 Overview

The proposed entity authentication scheme is a secure EAP method designed for authentication phase of the IEEE 802.1X framework using identity-based cryptography, avoiding involvement of PKI. It makes use of both symmetric-key and public-key cryptography to solve the key management problem in symmetric-key only systems. The scheme basically consists of two parts: the AUTHENTICATE and the RECONNECT protocols. The former is run when a supplicant joins the network in the first time, or want to creates a new communication session, in case the old session key is expired or the supplicant switch to another authenticator connected to a different authentication server. The latter allows the supplicant to resume the previous session and derive a new session key.

The AUTHENTICATE protocol makes use of Au and Wei.'s identification scheme (Section 3.8.2) in both directions of authentication, i.e. from supplicant to AS and from AS to supplicant. By inserting Diffie-Hellman (DH) key exchange parameters into the challenge messages, the protocol allows both parties to derive a session key (PMK), which is used in IEEE 802.11i key hierarchy for pairwise communication, and session information including a session secret and a timestamp for the RECONNECT protocol.

The RECONNECT protocol makes use of session information created in the AUTHENTICATE protocol and stored in both supplicant and authentication server to provide mutual authentication through symmetric-key cryptography. New session key is derived from the session information, as well as fresh nonces exchanged. It is a lightweight protocol providing fast reconnect characteristic.

The proposed scheme overcomes several weaknesses of identity-based

cryptography. The introduction of the RECONNECT protocol and avoidance of bilinear pairing techniques in the AUTHENTICATE protocol increases the efficiency. The scheme makes use of many cryptographic standards, such as those published in RFC, so that compatibility is not a big problem. Lastly, the key derivation process in the scheme provides forward secrecy so that compromising the PKG's master key does not allow an adversary to break the encryption after the authentication offered by the proposed scheme.

Terminology

The following describes various variables, functions and standards used in the proposed scheme. They will be referenced frequently in later sections.

N - RSA modulus [38] selected by the PKG, typically 1024-bit

g - Public generator of subgroup of quadratic residue of N^2, denoted as $\mathbb{QR}_{N^2}$, typically $N+1$ (1024-bit) selected by the PKG

p - Prime number selected by the authentication server, typically 3072-bit

h - Generator of subgroup of $\mathbb{Z}_p$ selected by the authentication server, typically of value 2

ID_c, ID_a - Identities of the supplicant and the authentication server respectively. Typically, the identities should follow the RFC2486 network access identifier (NAI) [1] grammar. Special strings may be appended for special purposes.

$< x_c, y_c >$, $< x_a, y_a >$ - Private keys of the supplicant and the authentication server respectively, where $x_c, x_a \in \mathbb{Z}_N$ and $y_c, y_a \in \mathbb{QR}_N^*$ are typically 1024-bit integers

r_c, u_c - Random integers in $\mathbb{Z}_N$ generated by the supplicant, typically 1024-bit

r_a, u_a - Random integers in $\mathbb{Z}_N$ generated by the server, typically 1024-bit

a, b - Random 256-bit integers in $\mathbb{Z}_p$ generated by the server and the supplicant respectively

u, v - Random 256-bit nonce integers generated by the server and the supplicant respectively

w - Timestamp generated by the AS in date format defined in RFC3339 [34]

w' - Old timestamp received in the last authentication execution, and stored in the supplicant

$T_w, T_{w'}$ - Reception time of w and w' respectively

D - Random 16-bit device ID generated by the supplicant for distinguishing different devices of the same client

K - Master Session Key (MSK), or simply session key derived in the authentication process by the both parties

K' - Session secret derived via the AUTHENTICATE protocol by the both parties

$H(X)$ - One-way hashing function computed over message X

$H_Y(X)$ - One-way hashing function computed over message with output in $\mathbb{Z}_Y$

$\tilde{H}_Y(X)$ - One-way hashing function computed over message X with output in $\mathbb{QR}_{Y^2}$

$\mathbf{PRF}(X, Y, Z, len)$ - Pseudorandom Function defined in IEEE 802.11i standard, computed using secret X, identifier Y, and seed Z, with len-bit output

$\mathbf{HMAC}_X(Y)$ - RFC2104 Keyed Message Authentication Code [37] computed over message Y with symmetric key X. HMAC with more than one input are computed by concatenating the specified values in the specified order.

Architecture

The infrastructure network in which our protocol applies follows the RSN architecture described in Section 5.1. Supplicant (client) asks authenticator (access point) for access of the network while authentication server makes the decision based on the conversation between the supplicant and the server forwarded by the authenticator. Besides carrying out authentication process, the server has to maintain storage of a list of identities authenticated and their corresponding session information. Supplicants and authentication server obtain private keys from the same PKG and thus a PKG should be maintained by the organization. Figure 6.1 shows an example of the WLAN security architecture.

Design principles

The design of the proposed scheme is relied on several properties of RSN.

- EAP-Request/Response messages are sent in pairs. Therefore, a protocol with odd number of rounds has overhead of one more empty message transmission.

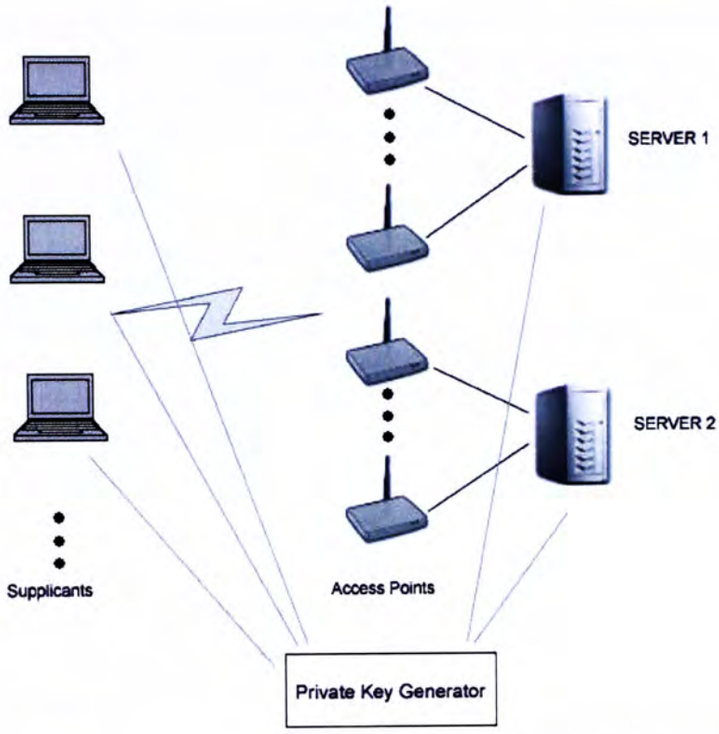

Figure 6.1: WLAN security architecture with proposed protocol implemented. A number of supplicants connect to an access point simultaneously while several access points connect to an authentication server through wired LAN. One enterprise may have more than one server.

- The AS in most cases maintains or connects to a database storing client information, such as an LDAP server, for purposes of authorization and accounting. This database can be used to maintain client's session attributes of each communication session.
- The backend AS is protected by high level of security against attackers from accessing through physical means or through the network. Directly attacking the server is not easier than attacking on network communication.

Preparation

When PKG is set up, it chooses N and g. Supplicants and authentication servers should have their identities (usually NAI) and corresponding private keys obtained from the same PKG before the authentication. The PKG extracts a private key $< x, y >$ of identity ID by computing $(x, y) \in (\mathbb{Z}_N \times \mathbb{QR}_N)$ such that $g^x y^N \equiv \tilde{H}_N(ID) \pmod{N^2}$. The identities of the server and the public parameters N and g defined by the PKG for extracting

private keys should be available to the public and the supplicants should be aware of the identities of legitimated servers in the organization.

The server optionally maintains information of legitimate supplicants and if so, the server should be aware of the revocation of keys of the supplicants. The server and the supplicant both have a system clock for timing. The time of them may not be synchronized but the counting of time should have a adequate accuracy. The clock built in the BIOS of any nowadays computer is good enough for the proposed scheme.

Lastly, the server should be able to store the session secret corresponding to all authenticated supplicants. It should keep track on the date of the creation. If the life of the secret, depending on its effective strength, exceeds a preset value, the server should remove such information.

Hash functions

The hash function H is selected by the authentication server. It is used to construct H_Y and $\tilde{H}$. Denoting l_Y the length of Y in bit, l_H the length of the output of $H(X)$, we construct $H_Y(X)$ as follows:

1. Recursively compute $A(i) = H(A(i-1) \parallel X)$ where $A(0) = H(X)$ and output a value $\tilde{A} = A(0) \parallel A(1) \parallel ... \parallel A(\lceil l_Y/l_H \rceil)$. Truncate $\tilde{A}$ to length l_Y to obtain A'.

2. Recursively compute $B(i) = H(B(i-1))$ where $B(0) = A'$ until $B(i)$ falls within the range $(0, 2^{l_Y} - (2^{l_Y} \mod Y) - 1)$. Denote such $B(i)$ be B'.

3. $H_Y(X) = B' \mod Y$.

To compute $\tilde{H}_Y(X)$, we have to make use of H_{Y^2} defined above as follows:

1. Recursively compute $F(i) = H_{Y^2}(F(i-1))$ where $F(0) = H_{Y^2}(X)$ until Jacobi Symbol of $F(i)$ is equal to 1.

2. Output such $F(i)$ as $\tilde{H}_Y(X)$

Since the computation of $\tilde{H}(X)$ requires relatively high computational load, the supplicant and the authentication server can decide to store the hash values of identities so as to increase performance.

6.1.2 The AUTHENTICATE Protocol

According to 802.1X framework described in Section 5.2, AUTHENTICATE protocol handshake and payload content transmitted in each step are shown below. Note that "#" is used in the EAP-type field as a new value of the field is allocated by Internet Assigned Numbers Authority (IANA) for

any new EAP protocol and so we are not able to determine the value now. Denote $[X]$ the value of the content field X received or sent, $\{Y\}$ the packet Y including header used to creating the HMAC, excluding the HMAC field itself.

- I1: Supplicant $\leftarrow$ AS: EAP-Request/Identity
- I2: Supplicant $\rightarrow$ AS: EAP-Response/Identity
  - I2.1: ID_c
- A1: Supplicant $\leftarrow$ AS: EAP-Request/EAP-type=#
  - A1.1: ID_a
  - A1.2: $H(g^{r_a}u_a^N \mod N^2)$
- A2: Supplicant $\rightarrow$ AS: EAP-Response/EAP-type=#
  - A2.1: $H(g^{r_c}u_c^N \mod N^2)$
  - A2.2: $h^a \mod p$
  - A2.3: D
- A3: Supplicant $\leftarrow$ AS: EAP-Request/EAP-type=#
  - A3.1: $r_a - H_N([\text{A2.2}])x_a$ ($\mod N$ if $g = N + 1$)
  - A3.2: $u_a y_a^{-[\text{A2.2}]} \mod N$
  - A3.3: $h^b \mod p$
  - A3.4: w
  - A3.5: $\text{HMAC}_{K'}(\{I2\}, \{A1\}, \{A2\}, \{A3\})$

  A4: Supplicant $\rightarrow$ AS: EAP-Response/EAP-type=#
  - A4.1: $r_c - H_N([\text{A3.3}])x_c$ ($\mod N$ if $g = N + 1$)
  - A4.2: $u_c y_c^{-[\text{A3.3}]} \mod N$
  - A4.3: $\text{HMAC}_{K'}(\{I2\}, \{A1\}, \{A2\}, \{A3\}, \{A4\})$

The Master Session Key (MSK), Extended Master Session Key (EMSK) and the session secret are derived by the supplicant and the authentication server after they receive A3 and A2 packet respectively. Both MSK and EMSK are 128 bits and the pairwise master key (PMK) used in RSN is formed by the MSK appended with the EMSK. They are computed as follows:

- $K' = \text{PRF}(h^{ab} \mod N, \text{"Session Secret"}, h^{ab} \mod N, 128)$
- $K = \text{PRF}(K', \text{"Master Session Key"}, h^{ab} \mod N, 128)$

- EMSK = PRF(K', "Extended Master Session Key", $h^{ab} \mod N, 128$)

Verification must be carried out by both supplicant and authentication server. If the verification fails, the verifier should send an EAP-Failure message and disconnect. Otherwise, an EAP-Success message is sent at the end of the authentication. The verification on each step of the way after the listed message is received is as follows:

- I2: Server
  The server optionally checks the identity of the supplicant against the legitimated supplicant list allowed to access the network if such list exists.

- A1: Supplicant
  The supplicant verifies that the received server identity ([A1.1]) is legitimated.

- A3: Supplicant
  The supplicant verifies that:
  $[\text{A1.2}] = H((\tilde{H}_N([\text{A1.1}]))^{H_N([\text{A2.2}])} g^{[\text{A3.1}]}([\text{A3.2}])^N \mod N^2)$
  Also the supplicant checks the validity of the HMAC value ([A3.5]).

- A4: Server
  The server verifies that:
  $[\text{A2.1}] = H((\tilde{H}_N([\text{I2.1}]))^{H_N([\text{A3.3}])} g^{[\text{A4.1}]}([\text{A4.2}])^N \mod N^2)$
  Also the server checks the validity of the HMAC value ([A4.3]).

Message I1 and I2 are EAP Request and Response identity messages required in any IEEE 802.11i authentication protocol. Message A1, A2, A3 contain commitment, challenge and response messages respectively of Au and Wei.'s identification scheme, to allow the supplicant to authenticate the server. Message A2, A3, A4 do the same job in reverse direction and thus two directions of authentication are overlapped. The challenge messages encapsulate DH parameters. DH key exchange is used to ensure the forward secrecy of the session key.

Message A3 contains a timestamp indicating the sent time of the message. When received, the timestamp is stored in the supplicant together with its reception time T_w. Message A3 and A4 contain HMAC of previous messages in order to provide integrity. Shared secret value obtained from DH key exchange is used to calculate the session key and session secret. The session key is used as the PMK which is required to be 256-bit. The session secret is stored by both parties. At the server, session secrets are distinguished by ID_c and D. A simplified view of the protocol is shown in Figure 6.2.

Supplicant		**Server**
Public: ID_c, g, N Secret: x_c, y_c		Public: ID_a, g, N Secret: x_a, y_a
		random r_a, u_a
	$\xleftarrow{ID_a, t_a = H(g^{r_a} u_a^N)}$	
check ID_a random r_c, u_c, b		
	$\xrightarrow{t_c = H(g^{r_c} u_c^N),\ d_a = h^b, D}$	
		random a $k = d_a^a$ timestamp w
	$\xleftarrow{z_a = r_a - H_N(d_a)x_a,\ \tilde{z_a} = u_a y_a^{-H_N(d_a)},\ w,\ d_c = h^a,\ \mathrm{HMAC}_k}$	
$k = d_c^b$ verify HMAC, $t_a = H(Q_a^{H_N(d_a)} g^{z_a} \tilde{z_a}^N)$ where $Q_a = \tilde{H}_N(ID_a)$		
	$\xrightarrow{z_c = r_c - H_N(d_c)x_c,\ \tilde{z_c} = u_c y_c^{-H_N(d_c)},\ \mathrm{HMAC}_k}$	
		verify HMAC, $t_c = H(Q_c^{H_N(d_c)} g^{z_c} \tilde{z_c}^N)$ where $Q_c = \tilde{H}_N(ID_c)$

Figure 6.2: Simplified view of the AUTHENTICATE protocol message flow. Moduli of the arithmetic are not shown. HMAC means the HMAC value of previous messages.

6.1.3 The RECONNECT Protocol

The RECONNECT protocol is a simple nonce-based authentication relied on the shared session secret K' and the stored timestamp w'. It can simplify the authentication process by reducing the handshake to only two moves. The message flow of the protocol is as follows:

- I1: Supplicant $\leftarrow$ AS: EAP-Request/Identity

- I2: Supplicant → AS: EAP-Response/Identity
  - I2.1: ID_c
- R1: Supplicant ← AS: EAP-Request/EAP-type=#
  - R1.1: ID_a
  - R1.2: u
  - R1.3: w
  - R1.4: $\text{HMAC}_{K'}\{I2\}, \{R1\})$
- R2: Supplicant → AS: EAP-Response/EAP-type=#
  - R2.1: v
  - R2.2: D
  - R2.2: $\text{HMAC}_{K'}(\{I2\}, \{R1\}, \{R2\})$

The new MSK and EMSK are derived by the supplicant and the authentication server based on the session secret K' after they receive R1 and R2 packet respectively. They are computed as follows:

- $K = \text{PRF}(K', \text{"Master Session Key"}, u\|v, 128)$
- $\text{EMSK} = \text{PRF}(K', \text{"Extended Master Session Key"}, u\|v, 128)$

Verification requirements of the RECONNECT protocol are shown below.

- I2: Server
  The server optionally checks the identity of the supplicant against the legitimate supplicant list.
- R1: Supplicant
  The supplicant verifies that the received server identity ([A1.1]) is legitimated and checks the validity of the HMAC value ([R1.4]) using the stored session secret. Also it checks whether the time difference between $(w - w')$ and $(T_w - T_{w'})$ lies within a reasonable range (the acceptance window), which is determined by the network latency.
- R2: Server
  The server checks the validity of the HMAC value ([R2.2]) using the stored session secret corresponding to the supplicant.

The server can find out whether the supplicant has an old session by searching its session secret. For supplicants with old session, it decides to run the RECONNECT protocol because the session secret has not expired.

The supplicant learns the type of the protocol executed in the first EAP-Request message. If the supplicant considers the old session insecure, it can explicitly trigger the execution of the RECONNECT protocol by responding an empty message to server. HMAC values created with old session secret on previous messages are used to prove the knowledge on the session secret since only valid parties know the old session secret. New session key are created based on the session secret, and the nonces exchanged. The new timestamp is stored in the supplicant, replacing the old one. A simplified view of the RECONNECT protocol is shown in Figure 6.3.

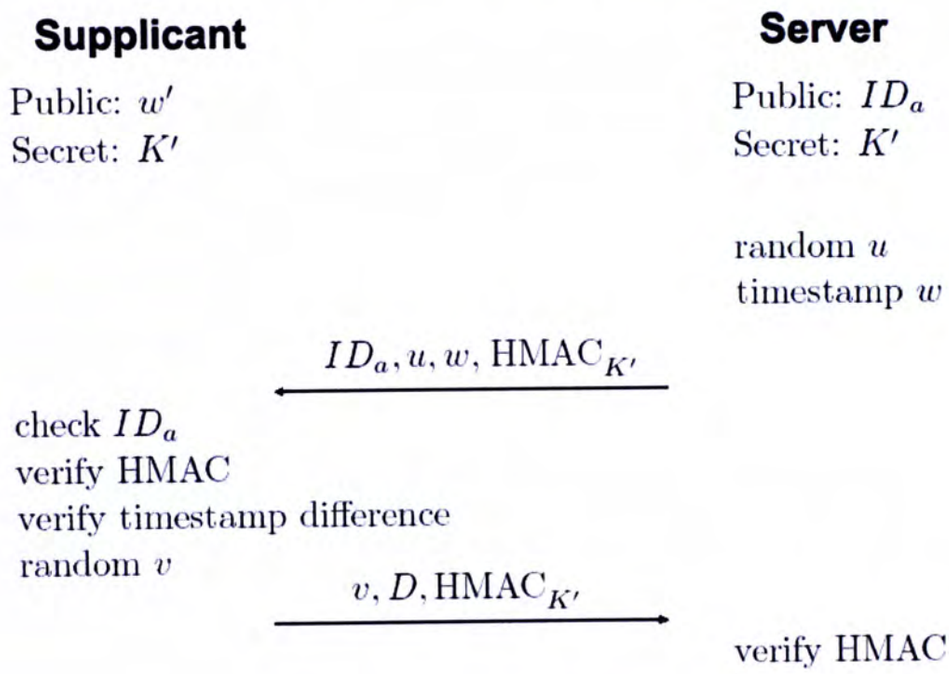

Figure 6.3: Simplified view of the RECONNECT protocol message flow. Exponential operations are in module p. HMAC means the HMAC value of previous messages

6.1.4 Packet Format

In this section, the packet format and content for the protocol messages are specified. Packets of the proposed protocol have the following structure (Figure 6.4):

Bit offset →

0								1								2								3							
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1

Code	Identifier	Length	
Type	Message Type	Flags	HMAC ID
DH Group	Hash ID	Payload ...	

Figure 6.4: Header structure of proposed protocol packets. Total length of the header is 9 bytes. The Code, Identifier, Length and Type fields are EAP header fields defined in RFC3748 [2].

Code

This field specifies the message type of EAP:
0x01 - EAP-Request
0x02 - EAP-Response
0x03 - EAP-Success
0x04 - EAP-Failure

Identifier

For matching Responses with Requests. The Identifier must be changed for each new Request message sent and must not be changed in retransmission of a given message. The Identifier in the Response message must match the corresponding Request message.

Length

For indicating the packet length including the header and payload.

Type

A new number of EAP type for this protocol is required as discussed before. EAP-Request/Identity and EAP-Response/Identity messages are of type 1.

Message Type

This field specifies the message type of the proposed protocol defined in Section 6.1.2 and 6.1.3 as follows:
0x01 - A1
0x02 - A2
0x03 - A3
0x04 - A4
0x05 - R1

0x06 - R2
0x07 - R3, empty EAP-Response message indicating that the supplicant refuses to execute the RECONNECT protocol

Flags

The flags field is broken up into 8 bits while each represents a binary flag. The field is defined as the logical OR or the following values:

- 0x01 - F-flag: The flag is set on all but the last fragment in a series of fragmented message packet.
- 0x02 - D-flag: The flag indicates that delegated guest access is enabled (see Section 6.1.6)
- 0x04 - P-flag: The flag indicates that pseudonym is allowed. (see Section 6.1.7)
- 0x08 to 0x80 - Reserved

HMAC ID

The field specifies the algorithm or the hash to implement the HMAC. Each bit represents one algorithm. Below is the recommended setting:

- 0x01 - HMAC-SHA-256 [22]

DH Group

The field specifies the Diffie-Hellman Group used in the AUTHENTICATE protocol. Each bit represents one group. For DH group, the following is recommended:

- 0x01 - 3072-bit MODP Group [33]

Hash ID

The field specifies the hash function used in the AUTHENTICATE protocol. Each bit represents one function. For the hash function to be secure, the following is recommended:

- 0x01 - SHA-256 [22]

Byte offset →

0	1
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4
len	Value ...

Figure 6.5: Structure of each payload value. The actual value is in variable length and the Len field specifies its length.

Payload

Payload field contains the actual handshake contents. The values and order of concatenation are defined in Section 6.1.2 and 6.1.3. For each value , a two-byte length field is appended at the beginning. Figure 6.5 shows the structure of each value.

All integer values are transmitted in network-byte order, with the most significant byte first. Strings, such as identities, are transmitted using 8-bit ASCII characters without null-termination. Binary data, such as hash outputs, are transmitted as-is.

6.1.5 Ciphersuites Negotiation

Ciphersuites negotiation is supported in the proposed scheme. This allows extensibility and flexibility on choosing different ciphersuites. Even if one ciphersuite is broken, a network administrator can change the security policy and disable the support on that ciphersuite.

The ciphersuites includes the DH group and the hash function used in the AUTHENTICATE protocol and the HMAC algorithm in both of the protocols. The AS indicates the supported ciphersuites in the first message (A1 or R1) by XORing different values representing the ciphersuites. The supplicant learns the supported ciphersuites, makes choices of its own and responses in the second message (A2 and R2) by setting the bit on each ciphersuite representing its choice. The ciphersuites specified at the second message should remain unchanged in successive messages.

6.1.6 Delegation

Delegation is an attractive feature which can lift the burden from the network administrator by allowing a valid client (the delegate) to authorized guests to access the WLAN. With identity-based cryptography, the right of delegation can be directly added to the identity, namely public key of an delegate. Meanwhile, the delegate can make a decision to authorize a guest by simply reading his identity.

Overview on the mechanism

To allow the authentication server to authenticate and authorize a guest, the guest must obtain a credential from the delegate and present it during the authentication process. The credential in our protocol is a digital signature from the delegate. The delegate signs on information including the guest's identity, public parameters of the PKG generating the private key to the guest, expiry date and guest type, using the identity-based signature scheme constructed from Au and Wei.'s identification scheme. Because of the similarities between Au and Wei.'s identification and signature scheme, software implementation can be shared in both schemes.

Preparation

Since in most cases, the guest and the delegate are in different organizations and their PKGs are different, enough information should be presented to the authentication server in order to compute valid messages during authentication. In order to get the credential from the delegate, the guest should have the following available:

ID_g - The guest's identity

N_g - RSA modulus [38] selected by the guest's PKG

g_g - Public generator of subgroup of $\mathbb{QR}_{N_g^2}$ selected by the guest's PKG

$< x_g, y_g >$ - Private key of the guest

If the guest does not have the above data, for example, his organization does not deploy the proposed system, the delegate has to act as a PKG to generate the private key for him.

For a local user to obtain the delegation right, he has to ask the local PKG to generate a private key corresponding to the public key $ID_d \parallel delegate = 1$ where ID_d is his identity. In another words, the delegation right is specified in the public key and the local user has to ask for the secret key to carry out delegation process. The PKG must ensure that users without delegation right do not obtain a private key corresponding to an identity containing "$delegate = 1$".

Obtaining Credential

The guest presents ID_g, N_g and g_g to the delegate who checks the validity of the data. As values of N_g and g_g depend on the organization the guest belongs to, it is the responsibility of the delegate to obtain the corresponding data from the organization. Then the delegate signs on the following data using the signature scheme described in Section 3.8.2, with the private key corresponding to $ID_d \parallel delegate = 1$.

1. g, N - Local PKG parameters
2. ID_g, g_g, N_g - Guest's parameters
3. T_{exp} - Expiry date of the guest access right, in date format defined in RFC2822 [49]
4. Y - Guest type, which may be used in authorization stage to restrict guest access

The guest obtains T_{exp}, Y and the signature sig from the delegate.

D-flag

The D-flag (Section 6.1.4) is reserved for delegation mechanism. D-flag set to 1 in A1 or R1 message means that the server supports delegated guest access. D-flag set to 1 in A2 or R2 message means that the supplicant is a guest. Therefore, for local user, D-flag in A2 and R2 messages is set to 0.

Authentication

The authentication process of the guest has the same bandwidth performance as that of a local user. D-flag is set in every message in order to notice the server that the supplicant is a guest. For AUTHENTICATE protocol, according the terminology defined in Section 6.1.1 and this section, the following modifications on messages are made for the guest:

- I2.1: The guest supplicant sends ID_d instead
- A2.1: The guest sends $H(g_g^{r_c} u_c^{N_g} \mod N_g^2)$, i.e. uses his own parameters
- A2.3 - A2.7: Extra fields - ID_g, N_g, g_g, T_{exp}, Y respectively
- A2.8: Signature obtained from the delegate, sig
- A4.1: The guest sends $r_c - H_{N_g}([\text{A3.3}])x_c$ ($\mod N$ if $g_g = N_g + 1$) instead
- A4.2: The guest sends $u_c y_c^{-[\text{A3.3}]} \mod N_g$ instead

Moreover, there is difference on verification process in the server while the key derivation remains unchanged:

- A3: Server
  The server verifies the validity of the signature ([A2.8]) on the values [A2.3]-[A2.7] received. To verifier the signature, it uses delegate_ || [I2.1] as the public key so that it ensures the delegate has the appropriate right. Also it checks whether the guest account has expired according to the expiry date in [A2.6].

- A4: Server
  The server verifies that:
  $[\text{A2.1}] = H((\tilde{H}_{[\text{A2.4}]}([\text{A2.3}]))^{H_{[A2.4]}([\text{A3.3}])}([\text{A2.5}])^{[\text{A4.1}]}([\text{A4.2}])^{[\text{A2.4}]} \mod ([\text{A2.4}])^2)$
  Also the server checks the validity of the HMAC value ([A4.3]).

Supplicant		**Server**
Public: $ID_d, g, N,$ $ID_g, g_g, N_g, T_{exp}, Y$ Secret: x_c, y_c		Public: ID_a, g, N Secret: x_a, y_a
		random r_a, u_a
	$\xleftarrow{ID_a, t_a = H(g^{r_a} u_a^N)}$	
check ID_a random r_c, u_c, b		
	$\xrightarrow{t_c = H(g_g^{r_c} u_c^{N_g}), d_a = h^b, D, ID_g, g_g, N_g, T_{exp}, Y, sig_{ID_c \| \text{delegate}=1}(g, N, ID_g, g_g, N_g, T_{exp}, Y)}$	
		verify sig random a timestamp w $k = d_a^a$
	$\xleftarrow{z_a = r_a - H_N(d_a)x_a, \tilde{z}_a = u_a y_a^{-H_N(d_a)}, w, d_c = h^a, \text{HMAC}_k}$	
$k = d_c^b$ verify HMAC, $t_a = H(Q_a^{H_N(d_a)} g^{z_a} \tilde{z}_a^{\,N})$ where $Q_a = \tilde{H}_N(ID_a)$		
	$\xrightarrow{z_c = r_c - H_{N_g}(d_c)x_g, \tilde{z}_c = u_c y_g^{-H_{N_g}(d_c)}, \text{HMAC}_k}$	
		verify HMAC, $t_c = H(Q_c^{H_{N_g}(d_c)} g_g^{z_c} \tilde{z}_c^{\,N_g})$ where $Q_c = \tilde{H}_{N_g}(ID_g)$

Figure 6.6: Message flow of the AUTHENTICATE protocol for a guest. Moduli of the arithmetic are not shown. HMAC means the HMAC value of previous messages. $sig_Y(X)$ means signature signed by public key Y on message X.

Figure 6.6 shows the message flow of the AUTHENTICATE protocol for a guest. After authenticating the guest, the server stores his identity and session secret in a guest list. For the RECONNECT protocol, the guest sends his real identity in I2 message. The server carries out the protocol executed by a guest just the same as that by a local user.

6.1.7 Identity Privacy

Our protocol supports identity privacy in the EAP layer. It is hard to prevent an eavesdropper to link two authentication processes by observing the Media Access Control (MAC) address, a hardware address that uniquely identifies each node of a network, since this requires change in MAC layer specification, IEEE 802.11 for example. However, hiding the identity in the EAP layer still provide some degree of anonymity. For instance, an eavesdropper is not able to determine whether a client requests for authentication through different computers. Therefore, identity privacy is still a desired property in WLAN.

Similar to EAP SRP-SHA1 [14], the identity privacy is protected by using pseudonym. The pseudonym is created after the supplicant authenticates the first time and therefore the supplicant can use the pseudonym in later-on authentication. But unlike EAP SRP-SHA1, our protocol does not require extra field in any message of authentication.

The authentication indicates that pseudonym is allowed by setting P-flag to 1. After the supplicant is authenticated, either through the AUTHENTICATE or the RECONNECT protocol, both supplicant and server can compute the pseudonym for next session as follows:

1. Compute $P = \text{RPF}(K, "Pseudonym", ID_c, 128)$.
2. Covert P to printable string by the algorithm described in Section 4.3.2.4 of RFC1421 [40].
3. if ID_c is a NAI, @*realm* is appended to the string obtained, where *realm* is the realm part of ID_c

Both supplicant and server stores the computed pseudonym if identity privacy is enabled. Next time when the supplicant requests for authentication, he presents "pseudonym_" || *pseudonym* in I2 message. The server recognizes the pseudonym and finds the actual identity string of the supplicant for the verification part of the authentication.

6.2 Security Considerations

6.2.1 Security of the AUTHENTICATE protocol

The security of the AUTHENTICATE protocol is based on the security of Au and Wei's identification scheme, which is proven to be secure against

impersonation under passive, active or concurrent attack given RSA[N,N] assumption holds in the Random Oracle Model, as described in Section 3.8.2. Therefore, the protocol prevents any adversary from impersonating a valid local user after obtaining transcripts of interactions between the local user and the authentication server, however the adversary has interacted with the server before the attempt. Securing concurrent attack allows the protocol defends any adversary who concurrently sets up a number of session with the authentication server. The server only has negligible possibility to accept the adversary's authentication attempt.

Because the protocol depends on the hardness of RSA problem, the security of it is more or less the same as any other authentication scheme based on the same problem, such as EAP-TLS. However, the criteria for our protocol to be secure, at the same time, also include that the hash function used in the protocol is able to act as the role of a random oracle. In other words, our protocol requires that the hash function to be one-way and collision-resistance. We recommend the use of SHA-256 function because of its longer output which is more secure under brute force attack.

Although key exchange information is embedded into the challenge message of the identification scheme, it does not affect the security of the scheme, given that the hash function used to calculate the actual challenge value is secure. This is because the output of the hash input is random and any adversary is not able to gain statistical information on the challenge value.

The protocol provides explicit key authentication. Since the protocol is secure against impersonation, only legitimate party can pass the authentication and establish the session key. Both parties are assured that another possesses the key by the HMAC because only a party with the key can construct it if the HMAC algorithm is secure.

The protocol is secure against key compromise impersonation. The private key generation stage of Au and Wei.'s identification scheme is basically the same as Paillier's one-way trapdoor permutation scheme [48], of which the security is based on the RSA[N,N] assumption. Therefore the compromise of long term private keys of one or more entities does not enable the adversary to obtain private keys of others, given that the RSA problem is hard. To impersonate the others without the corresponding private keys in the AUTHENTICATE protocol is equivalent to breaking Au and Wei.'s scheme.

Finally, we recommend the use of RSA modulus of length at least 1024 bits to provide similar security level as other commonly deployed RSA systems, such as EAP-TLS.

6.2.2 Security of the RECONNECT protocol

The RECONNECT protocol is based on exchange of random nonces while key confirmation using HMAC keyed by the session secret is added. The

HMAC allows implicit key authentication by confirming both parties have the same session secret, which is only shared between the supplicant and the authentication server. Only the supplicant and the server can construct a valid HMAC value given that the hash function used in the HMAC algorithm acts like a random oracle.

The HMAC values are used to authenticate the nonces and ensure that the values are sent by authenticated parties, which always generate fresh nonces. Any active or concurrent adversary cannot forge a valid HMAC value and thus a valid message. This allows the protocol to create a fresh session key in each run.

For the RECONNECT protocol, the authentication is based on shared session secrets which are independent for each supplicant. Therefore, compromise of long term private keys and session secrets of some entities again does not enable the adversary to impersonate others. In other words, the RECONNECT protocol is resistant to key-compromise impersonation.

The purpose to use a timestamp is to minimize the damage from replay attack. If the timestamp is not used, an adversary can replay the R1 message if the previous authentication was unsuccessful so that he can obtain a valid HMAC value keyed by the session secret while the session secret has not been modified. To achieve this, the adversary can just request for authentication by using the supplicant's identity and capture the R1 message. As a result, the supplicant considers the adversary a legitimate server. Since the adversary does not know the shared session secret, though he can authenticate to the supplicant, due to the implicit key authentication property of the protocol, he cannot obtain the session key which is protected by the pseudo-random function and the session secret, and used to determine other keys for data encryption and integrity. Therefore, the above problem is reduced to upper layer denial of service (DoS). The attacker cannot authenticate to the server since he cannot create a valid HMAC and so, he cannot set up a man-in-the-middle-attack. This kind of DoS is always possible because an attacker can always modify packets so the server does not authenticate the supplicant.

The use of timestamp can further minimize effect of the DoS attack. Since timestamps sent by the server change time by time and each of them is only valid for a very short period of time. This means even if an attacker captures the timestamp and the corresponding HMAC, he has to use it with a short range of time.

6.2.3 Security of Key Derivation

Session keys are derived using the PRF defined in IEEE 802.11i and fresh entropy supplied by both the supplicant and the authentication server through Diffie-Hellman key exchange in the AUTHENTICATE and through exchange of nonces in the RECONNECT protocol. To successfully recover

a session key, a passive adversary either solves the DH problem or inverts the HMAC algorithm used in the AUTHENTICATE protocol or compromising the session secret used in the RECONNECT protocol. Therefore the hash function adopted in the HMAC algorithm is required to be secure and a high level security should be applied to protect the server. The characteristics of the derived key, the reason to use DH key exchange and the security of the long-term private key are elaborated in the following subsections.

Key Strength

The PMK (MSK || EMSK) generated in our proposed protocol is 256 bits, which is the basic requirement of IEEE 802.11i. In the AUTHENTICATE protocol, the key generation is protected by DH key exchanged. In the RECONNECT protocol, it is protected by the session secret. It is believed that a 3072-bit MODP public-key scheme roughly has equivalent strength as 130-bit symmetric-key scheme [33]. RFC5326 [33] also state that the exponent used as the private DH parameter must have an entropy at least twice as large as the system strength. Consequently, 256-bit random DH exponents are used in order to allow the MSK and EMSK generated in the AUTHENTICATE protocol to contain 128-bit strength. On the other hand, the session secret, the same as the session key, has 128-bit strength. Therefore, the key generated in the RECONNECT protocol also has 128-bit strength. The PMK formed by MSK and EMSK thus has an effective key strength of 128 bits, which is the mandatory requirement of EAP authentication methods used in WLANs, stated in RFC4017 [58].

Known Key Security

By using DH key exchange and exchange of fresh nonces, unique secret session keys are generated in each authentication session. The DH parameters and the nonces are generated randomly and independent in each session. In addition, if the PRF and the HMAC is secure enough, the attacker knowing the session key of a client at some time cannot know previous or later-on session keys of him, because he cannot find the session secret from the HMAC. Therefore, knowledge of one session key does not allow deduction of the session key in another session.

Forward Secrecy

If long-term private keys of one or more of the entities obtained from the PKG are compromised, the secrecy of previously established session keys is not affected. This is because the private keys are only used in entity authentication of supplicants and authentication servers in the AUTHENTICATE protocol. The session key is derived from DH key exchange which is independently to the entity authentication, because unlike some other EAP

methods, such as RSA-based EAP-TLS, nonce values are not encrypted by the long-term private keys. Even if the secret of the PKG is corrupted, the previous sessions keys are still safe since only private keys of the entities can be computed from the PKG's secret.

Unknown Key-share resilience

To carry out unknown key-share attack, In the AUTHENTICATE protocol, an adversary has to replace any ephemeral DH key exchange value exchanged with his own one. The value, after being hashed, is used as a challenge that a server or a supplicant has to appropriately response. If the adversary modifies the value, verification will not succeed unless he can find out a collision on the hash output. In the RECONNECT protocol, the nonces are protected by the HMACs. It is infeasible for the adversary to transmit his own value while providing a valid HMAC if the hash function used in the HMAC algorithm is secure.

Key Control

Because both parties have an input into the session key, neither entity is able to force the full session key to be a preselected value. However, the receiver of the first ephemeral DH key exchange value or nonces exchanged (the server and the supplicant in AUTHENTICATE and RECONNECT protocol respectively) can set certain bits of the agreed session key by evaluating the result for different choices of the exponent of the reply. However, it does not appear possible for the party to set any substantial number of bits in a reasonable time frame.

6.2.4 EAP Security Claims and EAP Methods Requirements

This section describes our purposed protocol in terms of specific security terminology as required by RFC3748 [2], as well as additional EAP methods requirements in WLAN, mentioned in RFC4017 [58]. We demonstrate that our proposed protocol fulfills most requirements to be a secure EAP method. The weakness of our protocol is inability to provide EAP level DoS resistance, but one should notice that none of the EAP methods so far can defend such kind of attack.

Protected Ciphersuite Negotiation

In the first message from the server (A1, R1), the server specifies the ciphersuite in the packet header, including DH group, hash function and HMAC algorithm, as described in Section 6.1.4. The server is in total control of the ciphersuite supported, thus a client not supporting the specified ciphersuite will not be able to authenticate. The server can choose the list of

ciphersuites it considers secure, and therefore a centralized security policy on ciphersuites can be adopted.

The ciphersuite flags are located in the header and protected by the HMACs in A3, A4, R1 and R2 messages. The flags values in A3 and A4 must be the same as that in A2. Therefore, an attacker cannot modify the flag values in order to force the parties to choose weaker ciphersuites.

Mutual Authentication

The AUTHENTICATE protocol carries out Au and Wei.'s identification scheme in both directions and thus authenticate both the supplicant and the server. Consequently, the protocol achieves explicit mutual authentication. That means only valid parties can pass the verification of the authentication.

On the other hand, when RECONNECT protocol is used, only the client is explicitly authenticated. Since an adversary can continuously request for the R1 messages with different timestamp until the client attempts to connect to the WLAN, the server is not explicitly authenticated as the adversary can also pass the verification in supplicant's side. However, the server is instead implicitly authenticated to the supplicant because only the server containing the session secret, derived from the AUTHENTICATE protocol, which explicitly authenticates the server, is able to generate the required session keys. The supplicant can verify the server's authenticity during the four-way handshake.

Integrity Protection

The payload and headers in A1 and A2 messages of the proposed scheme are not explicitly covered by an integrity protection field. This is because no key is available to generate a HMAC. However, since HMAC values in A3 and A4 are computed according to the information including A1 and A2, the messages are also protected. All HMAC values in the RECONNECT protocol cover the header and the payload since the session secret is available before the execution of the protocol.

Replay Protection

The only messages capable of being replayed are A1 and R1 because other messages involve knowledge on previous messages. Impacts of replaying R1 message are explained in previous sections. Replaying A1 message, which is the commitment of the identification scheme, does not enable the adversary to response to the random challenge from the supplicant. Therefore in both cases, replaying theses messages provides an attacker with only a negligible advantage and only leads to DoS.

Confidentiality

Our protocol does not provide encryption, thus confidentiality to EAP messages. However, one should note that this does not affect the security of our scheme and in fact, confidentiality is not required in WLAN environment [58].

Key Derivation and Key Strength

Our scheme generates 128-bit MSK and EMSK which are combined to form a 256-bit PMK. The security on the keys are discussed in previous sections.

Dictionary Attack Resistance

Our scheme is resistant to dictionary attacks because no short password or key is used in created HMAC or in generation of keys by PRF. The keys (including session key and session secret) used in both algorithms are 128 bits and continuously change in each authentication execution. An attacker cannot obtain the keys in a reasonable amount of time.

Fast Reconnect

While a specific fast reconnection option is not included, execution of the RECONNECT protocol requires only minimal effort to authenticate a supplicant connected before. The protocol requires only string concatenation and HMAC calculation so as to identify both parties. It is a two-move protocol (EAP-Request/Idenity and EAP-Response/Identity are excluded as they are mandatory for all EAP methods in IEEE 802.1X), saving bandwidth and avoiding header overhead.

Session Independence

Session keys are independently generated for each session corresponding to fresh entropy. Generation of session keys in the RECONNECT protocol only depends on the session secret. We shows the forward secrecy and known key security of our protocol in previous sections. Even if an adversary obtains a session key, he cannot find out other session keys in previous or later-on sessions. Thus the damage is limited to the session compromised.

If an adversary compromises the session secret generated from the AUTHENTICATE protocol of a supplicant, he can compute later-on session keys. However since the session secret is only used in the stage of authentication, compromising the session secret is much harder than that on the session key. Therefore, compromise of the session key in one session does not allow an attacker to impersonate in another session. However, if the adversary only obtains the session key, he cannot find out the corresponding session secret due to the use of secure one-way PRF.

The session independence property facilitates the use of group identity. One user in the group cannot obtain the key generated by another user of the group. This maintains security of each user in the group.

Fragmentation

Fragmentation and reassembly are supported through the fragmentation flag in the header, although typically, size of any packet in our protocol is not larger than the EAP minimum transmission unit (MTU), which is assumed to be 1020 bytes.

Channel Binding

Our protocol does not explicitly include any channel binding. Channel Binding is only an optional feature suggested in RFC4017 [58] and the lack of it does not affect the overall security of our scheme.

Shared State Equivalence

After the supplicant and the authentication server successfully complete our EAP method, the shared EAP method state in both sides are equivalent. The EAP method state attributes include session key (MSK and EMSK), session secret, method version (obtained from Message Type field) and ciphersuite used. Both parties are able to distinguish this instance of the protocol from the other instances through the identity obtained from the communicated party.

Protection against Man-in-the-middle Attacks

Referring to the discussion of the integrity protection of our scheme, one should be aware that the ability of an adversary to modify messages does not allow him to get any advantage except to carry out DoS attack. A man-in-the-middle cannot either pass the verification or obtain the key derived in the attacked session in order to carry out further actions.

End-user Identity Hiding

End-user identity hiding is achieved by using pseudonym in our scheme as described in Section 6.1.7. The protection is limited to the peer-name portion of a NAI because the realm portion may be used for routing the EAP request of the supplicant to appropriate back-end authentication server.

6.3 Efficiency Analysis

6.3.1 Overview

Efficiency is another important concern for an EAP method, besides security. Only methods with good efficiency are practical to deploy in real life. Efficiency is involved in two areas: bandwidth and computation speed. As bandwidth in WLAN environment is limited, an authentication session should occupy as less bandwidth as possible. An EAP method with high computation speed allows a faster authentication process, especially for low-end devices such as PDAs. The following sections analyzes the efficiency of our EAP method. Comparisons are made between our scheme and EAP-TLS which is the most widely adopted method at this moment.

6.3.2 Bandwidth Performance

The bandwidth in an authentication process between a supplicant and a authentication server is mainly occupied by the response values of Au and Wei.'s identification scheme in messages A3, A4 and the DH values exchanged in A2, A3, R1 and R2. In order to provide sufficient security, size of the RSA modulus of the PKG and the prime for DH key exchange has to be large enough and thus modular arithmetic results are significant in size. This is in fact a drawback to use public-key cryptography.

Table 6.1 is a comparison of payload size between the AUTHENTICATE of our scheme and EAP-TLS, which also utilized public-key cryptography. It demonstrates that the AUTHENTICATE protocol occupies less bandwidth than EAP-TLS. The major reason is that our scheme does not require the use of certificate which has a large size. The RECONNECT protocol further reduces the total packet size to about 150 bytes.

In addition, the AUTHENTICATE protocol contains only 4 moves and the RECONNECT protocol contains only 2 moves while the EAP-TLS method requires 6 moves. Our scheme, therefore, has less overhead than EAP-TLS on packet header and connection setup.

6.3.3 Computation Speed

In the proposed protocol, the computation of modular exponentiation dominates in the computation time. Computation of all messages sent by supplicant or server, as well as the verification, requires such kind of arithmetic. Fortunately, many algorithm has been developed so as to increase the speed of the computation, such as those surveyed in [28].

Table 6.2 roughly shows the difference in computation of modular exponentiation between the AUTHENTICATION Protocol and EAP-TLS. If square-and-multiply method is used to calculation the exponentiation, the

AUTHENTICATE		EAP-TLS	
A1	32 + size of ID_c	TLS-Start	0
A2	32 + 384 + 20	Client Hello	28
A3	128 + 128 + 2 + 384 + 32	Server Hello	28
		Server Cert	$\sim$ 1500
A4	128 + 128 + 32	Client Key Exchange	384
		Finished (Client)	32
		Finished (Server)	32
Total:	$\sim$ 1500	Total:	$\sim$ 2100

Table 6.1: Comparison of packet payload size (in bytes) between the AUTHENTICATE protocol and EAP-TLS using 3072-bit RSA key exchange. The comparison is done in the way that both schemes have similar security level. For the data of EAP-TLS, only significant and mandatory payload fields are considered. Typical server's certificate is larger than 1500 bytes and client's certificate is not included. For our scheme, the data is based on the use of recommended ciphersuite.

results show that EAP-TLS approximately requires half the number of modular multiplication of our protocol. It is expected that the speed of the AUTHENTICATE protocol is slower slower than that of EAP-TLS if the computations are executed in real-time. However, pre-computation can be applied in computing the commitments of the identification scheme and DH key exchange values so that the speed of our protocol is reduced to just lower than that of EAP-TLS. The RECONNECT protocol is much faster than EAP-TLS since it only involves hash evaluation and does not use any modular arithmetic. In summary, our scheme is competitive with EAP-TLS in view of computation speed.

On the other hand, the lack of the use of bilinear mapping, which is always applied in other identity-based authentication and key exchange schemes ([56, 41, 16] for example) allows our EAP method to have a much higher speed of computation than those schemes. Experiment results on comparison of RSA operations and bilinear mapping operations are shown in [8].

□ **End of chapter.**

AUTHENTICATE		EAP-TLS	
A1	2048 [0]		
A2	2304 [0]		
Key generation at server	256		
A3	1280 [1024]		
Key generation at client	256	Server cert verification	1024
Server verification	3072	Secret encryption	3072
A4	1024		
Client verification	3072	Client cert verification	1024
		Secret decryption	3072
Total:	13312 [8704]	Total:	8192

Table 6.2: A Rough comparison of total magnitude of exponents (in bits) between the AUTHENTICATE protocol and EAP-TLS using 3072-bit RSA key exchange. If pre-computation is used, the values in the AUTHENTICATE protocol is reduced to that in [ ]. The comparison is done in the way that both schemes have similar security level. For our scheme, the data is based on the use of recommended ciphersuite.

Chapter 7

Conclusion

7.1 Summary

This thesis has described a new upper layer authentication protocol running over the Extensible Authentication Protocol. It is suitable to deploy in the IEEE 802.1X framework, fulfilling the requirements on entity authentication and key establishment of the new IEEE 802.11 wireless LAN based security standard IEEE 802.11i. The new EAP method is identity-based and avoids the use of PKI and certificates.

Because of enhanced mobility and productivity, WLANs are experiencing a period of substantial growth in the marketplace, as described in Chapter 1. The physical aspect of data transmission in a WLAN is based on electromagnetic waves rather than conventional wires or optical fibers. This raise various security threats on confidentiality, integrity of data transmitted and authentication process which allows the WLAN to prove the identities of clients. In enterprise environment, where a WLAN usually operates in infrastructure mode, data is even more sensitive and thus a higher security level is demanded. The use of an authentication server to provide secure authentication executions is an crucial step to protect the network. Another essential step is to provide encryption and data integrity to the sensitive communication by using keys. Of course, adversaries will not forgo the chance to obtain benefits from compromising the network and communication, and therefore, they carry out different kinds of attacks on the authentication process and the keys. All of these background information are covered in Chapter 2.

To secure WLANs, measures are specified in standards and implemented by vendors. Chapter 4 introduces conventional approaches to secure WLANs. Two significant methods are the use of VPN and WEP. VPN affects negatively the bandwidth and computational performance. WEP, despite being a part of IEEE 802.11 standard, is threatened by different attacks. The consequences range from successful message forgery, plaintext recovery to secret

key recovery. The entity authentication via shared key authentication which is based on WEP is totally useless when facing a replay attack. The Achilles heel is that the relationship between authentication and confidentiality in WEP allows an attacker to break one, get both.

In order to remedy the weaknesses in WEP, a new security standard IEEE 802.11i was released. Besides encryption and data integrity algorithms, it provides authentication and session key derivation through the use of IEEE 802.1X and upper layer authentication protocols such as EAP-TLS, as described in Chapter 5. Different ULA protocols can be adopted in different environments. For example, if backward compatibility is important, LEAP can be used. If shared secret key is available, Kerberos is a good choice due to its efficiency. While EAP-TLS is the most popular ULA protocol, being a public key based protocol, it requires the use of PKI and certificates, suffering from performance drawback.

When choosing an ULA for the RSN, while the public-key based approach is good for large enterprise due to its better key management, we have to solve the problems on bandwidth efficiency and computation speed because of the clients' need on frequent reconnect. Chapter 6 purposes an EAP method consisting of two protocols - the AUTHENTICATE and the RECONNECT protocol. They are secure against different kinds of attacks mentioned in Chapter 2. Using identity-based cryptography in AUTHENTICATE can provide a better bandwidth efficiency and the symmetric-key based RECONNECT protocol fastens the computation speed as well. The basic cryptographic concepts and components used in the proposed authentication scheme, such as key establishment, entity authentication, hash functions, and Au and Wei's identity based identification scheme are introduced in Chapter 3.

7.2 Future Work

Possible future work lays on two areas. The first one is to implementing the proposed scheme, and to experiment on the performance of our schemes and other EAP methods and compare the results. The implementation involves three components - a client side software, a RADIUS server supporting the scheme and a PKG. The performance analysis in this thesis is performed by estimation and approximation and thus may not reflect the actual situation because practical implementation involves other factors affecting the performance, such as upper layer header overhead and the choice on arithmetic algorithms.

Another area worth exploring is the use of identity-based PKI [13] and hierarchical identity-based settings in the authentication, such as the one described in [27]. One disadvantage on the system settings of the proposed scheme is that, in a very large network, the private key generator would

have a quite burdensome job. The delegation feature can reduce some work of the PKG, but it does not provide a robust system organization and thus is only suitable for creating guest accounts. The use of identity-based PKI, on the one hand provides a systematic and standardized architecture to the system, and on the other hand avoids the use of certificates, overcoming the problems in conventional PKI.

□ **End of chapter.**

Bibliography

[1] B. Aboba and M. Beadles. The Network Access Identifier. RFC 2486, IETF, Jan. 1999.

[2] B. Aboba, L. Blunk, J. Vollbrecht, J. Carson, and H. Levkowetz. Extensible Authentication Protocol (EAP). RFC 3748, IETF, June 2004.

[3] B. Aboba and D. Simon. PPP EAP TLS Authentication Protocol. RFC 2716, IETF, Oct. 1999.

[4] H. Andersson, S. Josefsson, and e. a. RSA Security. Protected EAP Protocol (PEAP). Internet draft draft-josefsson-pppext-eap-tls-eap-0.5.txt, IETF, Sept. 2002.

[5] W. A. Arbaugh. An inductive chosen plaintext attack against WEP/WEP2. doc IEEE802.11-01/230, IEEE, May 2001.

[6] W. A. Arbaugh, N. Shankar, and Y. J. Wan. Your 802.11 wireless network has no clothes. Mar. 2001.

[7] M. H. Au and V. K. wei Wei. Several identity-based identifications from pailliar system. 2005.

[8] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 354–368. Springer-Verlag, 2002.

[9] M. Bellare, C. Namprempre, and G. Neven. Security proofs for identity-based identification and signature schemes. In *Proc. EUROCRYPT 2004*, volume 3072, pages 268–286. Springer-Verlag, 2004.

[10] L. Blunk and J. Vollbrecht. PPP Extensible Authentication Protocol (EAP). RFC 2284, IETF, Mar. 1998.

[11] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Proc. of CRYPTO 2001, LNCS 2139*, pages 213–229. Springer-Verlag, 2001.

[12] N. Borisov, I. Goldberg, and D. Wagner. Intercepting mobile communications: The insecurity of 802.11. In *Proceedings of MOBICOM 2001*, 2001.

[13] J. Callas. Identity-based encryption with conventional public-key infrastructure. In *Proc. of PKI '05*, Apr. 2005.

[14] J. Carlson, B. Aboba, and H. Haverinen. EAP SRP-SHA1 Authentication Protocol. Internet Draft draft-ietf-pppext-eap-srp-03.txt, IETF, July 2001.

[15] CCITT/ISO. X.500, The Directory - overview of concepts, models and services. ISO9594 2459, CCITT/ISO.

[16] L. Chen and C. Kudla. Identity based authenticated key agreement protocols from pairings. Technical Report HPL-2003-25, HP Labs, Feb. 2003.

[17] C. Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding - Institute of Mathematics and Its Applications International Conference on Cryptography and Coding - Proceedings of IMA*, pages 360–363. Springer-Verlag, 2001.

[18] J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Springer-Verlag, 2002.

[19] T. Dierks and C. Allen. The TLS Protocol, version 1.0. RFC 2246, IETF, Jan. 1999.

[20] W. Diffie. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, pages 107 – 125, Mar. 1992.

[21] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22, 1976.

[22] U. DoC/NIST. Secure Hash Standard (shs). FIPS 180, U.S. DoC/NIST, Aug. 2002.

[23] T. ElGamal. A public-key cryptosystem and a signature scheme based on discret logarithms. In *CRYPTO 84*. Springer-Verlag, 1985.

[24] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, pages 77–94, 1988.

[25] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Proc. of CRYPTO' 86, LNCS 263*, pages 186–194. Springer-Verlag, 1987.

[26] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. In *8th Annual Workshop on Selected Areas in Cryptography*, Aug. 2001.

[27] C. Gentry and A. Silverberg. Hierachical ID-based cryptography. In *Proc. of ASIACRYPT 2002, LNCS2501*, pages 548–566, 2002.

[28] D. M. Gordon. A survey of fast exponentiation methods. *J. Algorithms*, 27(1):129–146, 1998.

[29] C. He and J. C. Mitchell. Analysis on the 802.11i 4-way handshake. In *Proc. of WiSe '04*, Oct. 2004.

[30] R. Housley, W. Ford, W. Polk, and D. Solo. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. RFC 2459, IETF, Jan. 1999.

[31] ISO. Information technology - Security techniques - Entity authentication - Part 4: Mechanisms using a cryptographic check function. IEC 9798-4, ISO, Dec. 1999.

[32] K. Kaukonen and R. Thayer. A Stream Cipher Encryption Algorithm "Arcfour". Internet-draft draft-kaukonen-cipher-arcfour-03.txt, IETF, July 1999.

[33] T. Kivinen and M. Kojo. More Modular Exponential (MODP) Diffie-Hellman Groups for Internet Key Exchange (IKE). RFC 3526, IETF, June 2003.

[34] G. Klyne and C. Neoman. Date and Time on the Internet: Timestamps. RFC 3339, IETF, July 2002.

[35] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48, 1987.

[36] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5). RFC 1510, IETF, Sept. 1993.

[37] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. RFC 2104, IETF, Feb. 1997.

[38] R. Labortories. PKCS #1: RSA Encryption Standard, version 1.5. (1), Nov. 1993.

[39] B.-G. Lee, D.-H. Choi, H.-G. Kim, S.-W. Sohn, and K.-H. Park. Mobile IP and WLAN with AAA authentication protocol using identity-based cryptography. In *10th International Conference on Telecommunications ICT*, volume 1, pages 597 – 603, 2003.

[40] J. Linn. Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures. RFC 1421, IETF, Feb. 1993.

[41] N. McCullagh and P. S. Barreto. A new two-party identity-based authenticated key agreement. In *Proc. CT-RSA 2005*, volume 3376, page 262. Springer-Verlag, Feb. 2005.

[42] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[43] V. Miller. Use of elliptic curves in cryptography. In *CRYPTO 85.* Springer-Verlag, 1986.

[44] A. Mishra and W. A. Arbaugh. An initial security analysis of the IEEE 802.1X standard, Feb. 2002. urlhttp://www.cs.umd.edu/ waa/1x.pdf.

[45] N. B. of Standards. Data Encryption Standard. FIPS 46, National Bureau of Standards, U.S. Department of Commerce, Jan. 1977.

[46] L. S. C. of the IEEE Computer Society. Wireless lan medium access control (mac) and physical layer (phy) specifications. IEEE std 802.11, IEEE, 1999.

[47] L. S. C. of the IEEE Computer Society. Port based network access control. IEEE std 802.1X, IEEE, 2001.

[48] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. EUROCRYPT 99*, pages 223–238. Springer-Verlag, 1999. Lecture Notes in Computer Science No. 196.

[49] P. Resnick. Internet Message Format. RFC 2822, IETF, Apr. 2001.

[50] C. Rigney. Radius Accounting. RFC 2866, IETF, June 2000.

[51] C. Rigney, W. Willats, and P. Calhoun. Radius Extentsions. RFC 2869, IETF, June 2000.

[52] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial in User Service (RADIUS). RFC 2865, IETF, June 2000.

[53] R. Rivest. The MD4 Message-Digest Algorithm. RFC 1320, IETF, Apr. 1992.

[54] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, IETF, Apr. 1992.

[55] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM (2)*, 21, 1978.

[56] E.-K. Ryu, E.-J. Yoon, and K.-Y. Yoo. An efficient ID-based authenticated key agreement protocol from pairings. In *Proc. NETWORKING 2004*, volume 3042, pages 1458–1463. Springer-Verlag, Apr. 2004.

[57] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53. Springer-Verlag, 1985.

[58] D. Stanley, J. Walker, and B. Aboba. Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs. RFC 4017, IETF, Mar. 2005.

[59] A. Stubblefield, J. Ioannidis, and A. D. Rubin. Using the Fluhrer, Mantin, and Shamir attack to break WEP. Technical Report TD4ZCPZZ, Revision 2, ATT Labs, Aug. 2002.

[60] J. R. Walker. Unsafe at any key size; an analysis of the WEP encapsulation, IEEE document 802.11-00/362. Oct. 2000.

[61] T. Wu. The secure remote password protocol. In *Proc. of the 1998 Internet Society Symposium on Network and Distributed Systems Security*, pages 97–111, 1998.

CUHK Libraries
004280589